

```

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

PAS
PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS
PAS

PAS

PAS

PAS

PAS

PAC

PAS

PAS

PAS
PAS

PAS

PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS

```
PPPPPPPP      AAAAAA      SSSSSSSS  WW      WW  RRRRRRRR      IIIIII  HH      HH  EEEEEEEEEEE  XX      XX
PPPPPPPP      AAAAAA      SSSSSSSS  WW      WW  RRRRRRRR      IIIIII  HH      HH  EEEEEEEEEEE  XX      XX
PP      PP  AA      AA  SS      WW      WW  RR      RR      II      HH      HH  EE      XX      XX
PP      PP  AA      AA  SS      WW      WW  RR      RR      II      HH      HH  EE      XX      XX
PP      PP  AA      AA  SS      WW      WW  RR      RR      II      HH      HH  EE      XX      XX
PP      PP  AA      AA  SS      WW      WW  RR      RR      II      HH      HH  EE      XX      XX
PPPPPPPP      AA      AA      SSSSSS  WW      WW  RRRRRRRR      II      HH      HH  EEEEEEEEE  XX      XX
PPPPPPPP      AA      AA      SSSSSS  WW      WW  RRRRRRRR      II      HH      HH  EEEEEEEEE  XX      XX
PP      AAAAAAAAAA      SS  WW      WW  RR      RR      II      HH      HH  EE      XX      XX
PP      AAAAAAAAAA      SS  WW      WW  RR      RR      II      HH      HH  EE      XX      XX
PP      AA      AA      SS  WWW      WWW  RR      RR      II      HH      HH  EE      XX      XX
PP      AA      AA      SS  WWW      WWW  RR      RR      II      HH      HH  EE      XX      XX
PP      AA      AA      SSSSSSSS  WW      WW  RR      RR      IIIIII  HH      HH  EEEEEEEEEEE  XX      XX
PP      AA      AA      SSSSSSSS  WW      WW  RR      RR      IIIIII  HH      HH  EEEEEEEEEEE  XX      XX
                                                                ....
                                                                ....
                                                                ....
                                                                ....

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_HEX ( %TITLE 'Write a value in base 16'
2 0002 0 IDENT = '1-002' ! File: PASWRIHEX.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 procedures which write a value in
36 0036 1 base 16 to a textfile or string.
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 30-Jun-1982
46 0046 1 --
47 0047 1
```

PASSWRITE_HEX
1-002

Write a value in base 16
Declarations

K 16
16-Sep-1984 02:19:37
14-Sep-1984 12:52:05

VAX-11 Bliss-32 V4.0-742
[PASRTL.SRC]PASWRITE_HEX.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';
: 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_HEX: NOVALUE;
: 62      0124 1      PASSWRITEV_HEX: NOVALUE;
: 63      0125 1
: 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
: 81      0143 1
: 82      0144 1  !! If this is for a V2 system, redefine OTSS$CVT_L_TZ as PASS$CVT_L_TZ.
: 83      L 0145 1  %IF %VARIANT
: 84      U 0146 1  %THEN
: 85      U 0147 1  UNDECLARE
: 86      U 0148 1      OTSS$CVT_L_TZ;
: 87      U 0149 1  EXTERNAL ROUTINE
: 88      U 0150 1      PASS$CVT_L_TZ;
: 89      U 0151 1  BIND ROUTINE
: 90      U 0152 1      OTSS$CVT_L_TZ = PASS$CVT_L_TZ;
: 91      0153 1  %FI
```

! Externals, linkages, PSECTs, structures

! Write to textfile

! Write to string

PASSWRITE_HEX
1-002

Write a value in base 16

PASSWRITE_HEX - Write a value in hexadecimal to

L 16

16-Sep-1984 02:19:37

14-Sep-1984 12:52:05

VAX-11 Bliss-32 V4.0-742

[PASRTL.SRC]PASWRIHEX.B32;1

Page 3
(3)

```

: 93      0154 1 %SBTTL 'PASSWRITE_HEX - Write a value in hexadecimal to textfile'
: 94      0155 1 GLOBAL ROUTINE PASSWRITE_HEX (
: 95      0156 1     PFV: REF $PASSPFV_FILE_VARIABLE,      ! File variable
: 96      0157 1     NBITS,                                ! Size of value in bits
: 97      0158 1     VALUE,                                ! Address of value
: 98      0159 1     TOTAL_WIDTH: SIGNED,                  ! Total field width
: 99      0160 1     ERROR,                                ! Error unwind address
: 100     0161 1     MIN_DIGITS: SIGNED                     ! Minimum number of digits
: 101     0162 1     ): NOVALUE =
: 102     0163 1
: 103     0164 1 ++
: 104     0165 1 FUNCTIONAL DESCRIPTION:
: 105     0166 1
: 106     0167 1     This procedure writes a base 16 representation of a value to the
: 107     0168 1     specified textfile.
: 108     0169 1
: 109     0170 1 CALLING SEQUENCE:
: 110     0171 1
: 111     0172 1     CALL PASSWRITE_HEX (PFV.mr.r, NBITS.rl.v, VALUE.rz.r, TOTAL_WIDTH.rl.v
: 112     0173 1     [, [ERROR.j.r] [, MIN_DIGITS.rl.v]])
: 113     0174 1
: 114     0175 1 FORMAL PARAMETERS:
: 115     0176 1
: 116     0177 1     PFV                                     - The Pascal File Variable (PFV) passed by reference.
: 117     0178 1                                     The structure of the PFV is defined in PASPFV.REQ.
: 118     0179 1
: 119     0180 1     NBITS                                    - The size of VALUE in bits.
: 120     0181 1
: 121     0182 1     VALUE                                    - The address of the value to write.
: 122     0183 1
: 123     0184 1     TOTAL_WIDTH                             - Total field width.
: 124     0185 1
: 125     0186 1     ERROR                                    - Optional. Address to unwind to if an error occurs.
: 126     0187 1
: 127     0188 1     MIN_DIGITS                             - Optional. The minimum number of digits to appear
: 128     0189 1     in the result. Defaults to the minimum number of digits
: 129     0190 1     needed to represent every bit of the value.
: 130     0191 1
: 131     0192 1 IMPLICIT INPUTS:
: 132     0193 1
: 133     0194 1     NONE
: 134     0195 1
: 135     0196 1 IMPLICIT OUTPUTS:
: 136     0197 1
: 137     0198 1     NONE
: 138     0199 1
: 139     0200 1 ROUTINE VALUE:
: 140     0201 1
: 141     0202 1     NONE
: 142     0203 1
: 143     0204 1 SIDE EFFECTS:
: 144     0205 1
: 145     0206 1     If the file is the standard file INPUT or OUTPUT, it is implicitly opened.
: 146     0207 1
: 147     0208 1 SIGNALLED ERRORS:
: 148     0209 1
: 149     0210 1     NEGWIDDIG - negative field width or digits specification is not allowed
```

PASSWRITE_HEX
1-002

Write a value in base 16

PASSWRITE_HEX - Write a value in hexadecimal to

M 16

16-Sep-1984 02:19:37

14-Sep-1984 12:52:05

VAX-11 Bliss-32 V4.0-742

[PASRTL.SRC]PASWRIHEX.B32;1

Page 4
(3)

```

150 0211 1 | LINTOOLON - line too long
151 0212 1 |
152 0213 1 | --
153 0214 1 |
154 0215 2 BEGIN
155 0216 2
156 0217 2 LOCAL
157 0218 2 FCB: REF $PASSFCB_CONTROL_BLOCK, | File control block
158 0219 2 ACTUAL_DIGITS, | Number of digits actually used
159 0220 2 ACTUAL_NBITS, | Value size actually used
160 0221 2 DESCR: BLOCK [8, BYTE], | String descriptor
161 0222 2 PFV_ADDR: VOLATILE, | Enable argument
162 0223 2 UNWIND_ACT: VOLATILE, | Enable argument
163 0224 2 ERROR_ADDR: VOLATILE; | Enable argument
164 0225 2
165 0226 2 LITERAL
166 0227 2 M_SIZE_IN_BITS = %X'04'; | Flags argument for
167 0228 2 | convert routine
168 0229 2 BUILTIN
169 0230 2 ACTUALCOUNT;
170 0231 2
171 0232 2 ENABLE
172 0233 2 PASS$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR); | Enable error handler
173 0234 2
174 0235 2 !+
175 0236 2 | Get ERROR parameter, if present.
176 0237 2 | -
177 0238 2
178 0239 2 IF ACTUALCOUNT () GEQU 5
179 0240 2 THEN
180 0241 2 ERROR_ADDR = .ERROR; | Set unwind address
181 0242 2
182 0243 2 PFV_ADDR = PFV [PFV$R_PFV]; | Set PFV address
183 0244 2
184 0245 2 !+
185 0246 2 | Validate PFV and get PFV.
186 0247 2 | -
187 0248 2
188 0249 2 PASS$VALIDATE_PFV (PFV [PFV$R_PFV]; FCB);
189 0250 2
190 0251 2 !+
191 0252 2 | Set unwind action to unlock file.
192 0253 2 | -
193 0254 2
194 0255 2 UNWIND_ACT = PASS$K_UNWIND_UNLOCK;
195 0256 2
196 0257 2 !+
197 0258 2 | Do common initialization.
198 0259 2 | -
199 0260 2
200 0261 2 PASS$INIT_WRITE (PFV [PFV$R_PFV], FCB [FCB$R_FCB]; FCB);
201 0262 2
202 0263 2 !+
203 0264 2 | Set initial values for conversion.
204 0265 2 | -
205 0266 2
206 0267 2 ACTUAL_NBITS = .NBITS;
```

```

207 0268 2 ACTUAL_DIGITS = (.ACTUAL_NBITS+3)/4;
208 0269 2
209 0270 2
210 0271 2 !+ Create result string descriptor with actual width.
211 0272 2 !-
212 0273 2
213 0274 2 DESCR [DSC$B_CLASS] = DSC$K_CLASS_S;
214 0275 2 DESCR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
215 0276 2 DESCR [DSC$A_POINTER] = .FCB [FCB$A_RECORD_CUR];
216 0277 2 IF ACTUALCOUNT () GEQU 6
217 0278 2 THEN
218 0279 2 BEGIN
219 0280 2 ACTUAL_DIGITS = .MIN_DIGITS;
220 0281 2 IF .ACTUAL_DIGITS LSS 0
221 0282 2 THEN
222 0283 2 $PAS$IO_ERROR (PASS$_NEGWIDDIG,0);
223 0284 2 END;
224 0285 2
225 0286 2 IF .TOTAL_WIDTH LSS 0
226 0287 2 THEN
227 0288 2 $PAS$IO_ERROR (PASS$_NEGWIDDIG,0);
228 0289 2
229 0290 2 DESCR [DSC$W_LENGTH] = .TOTAL_WIDTH;
230 0291 2 !+
231 0292 2 ! Will TOTAL_WIDTH truncate the value? (Truncation occurs
232 0293 2 ! on the left.)
233 0294 2 !+
234 0295 2 IF .TOTAL_WIDTH LSSU (.ACTUAL_NBITS+3)/4
235 0296 2 THEN
236 0297 2 BEGIN
237 0298 2 !+
238 0299 2 ! Change value size to cause a truncated result.
239 0300 2 !-
240 0301 2 ACTUAL_NBITS = .TOTAL_WIDTH*4;
241 0302 2 END;
242 0303 2
243 0304 2 IF .ACTUAL_DIGITS GTR .TOTAL_WIDTH
244 0305 2 THEN
245 0306 2 ACTUAL_DIGITS = .TOTAL_WIDTH;
246 0307 2
247 0308 2 !+
248 0309 2 ! See if field will fit in record.
249 0310 2 !-
250 0311 2
251 0312 2 BEGIN
252 0313 2 LOCAL
253 0314 2 EXTRA; ! Extra characters past end of line
254 0315 2 EXTRA = (.FCB [FCB$A_RECORD_CUR] + .DESCR [DSC$W_LENGTH]) - .FCB [FCB$A_RECORD_END];
255 0316 2 IF .EXTRA GTR 0
256 0317 2 THEN
257 0318 2 $PAS$IO_ERROR (PASS$_LINTOOLON,1,.EXTRA);
258 0319 2 END;
259 0320 2
260 0321 2 !+
261 0322 2 ! Do the conversion. It can't fail.
262 0323 2 !-
263 0324 2

```

PASSWRITE_HEX
1-002

Write a value in base 16

PASSWRITE_HEX - Write a value in hexadecimal to

C 1

16-Sep-1984 02:19:37

14-Sep-1984 12:52:05

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

Page 6
(3)

```

: 264      0325 2      OTSS$CVT_L_TZ (.VALUE, DESCR, .ACTUAL_DIGITS, .ACTUAL_NBITS,
: 265      0326 2      M_SIZE_IN_BITS);
: 266      0327 2
: 267      0328 2      FCB [FCB$A_RECORD_CUR] = .FCB [FCB$A_RECORD_CUR] + .DESCR [DSC$W_LENGTH];
: 268      0329 2
: 269      0330 2
: 270      0331 2      !+ Call WRITE epilogue routine to move the last character written to the
: 271      0332 2      ! user's buffer and to unlock the file variable.
: 272      0333 2      !-
: 273      0334 2
: 274      0335 2      PASS$END_WRITE (PFV [PFV$R_PFV], FCB [FCB$R_FCB]);
: 275      0336 2
: 276      0337 2      RETURN;
: 277      0338 2
: 278      0339 1      END;
```

! End of routine PASSWRITE_HEX

.TITLE PASSWRITE_HEX Write a value in base 16
.IDENT \1-002\

.EXTRN PASSWRITE_HEX, PASSWRITEV_HEX
.EXTRN PASS\$IO_HANDLER
.EXTRN PASS\$VALIDATE_PFV
.EXTRN PASS\$INIT_WRITE
.EXTRN PASS\$SIGNAL, PASS\$NEGWIDDIG
.EXTRN PASS\$LINTOOLON
.EXTRN OTSS\$CVT_L_TZ, PASS\$END_WRITE

.PSECT _PASS\$CODE, NOWRT, SHR, PIC, 2

58	00000000G	00	9E	00002	.ENTRY PASSWRITE_HEX, Save R2,R3,R4,R5,R6,R7,R8	: 0155
5E		10	C2	00009	MOVAB PASS\$SIGNAL, R8	
		7E	D4	0000C	SUBL2 #16, SP	
	04	AE	7C	0000E	CLRL ERROR_ADDR	: 0215
6D	00A7	CF	DE	00011	CLRQ UNWIND_ACT	
05		6C	91	00016	MOVAL 8\$, (FP)	
		04	1F	00019	CMPB (AP), #5	: 0239
6E	14	AC	D0	0001B	BLSSU 1\$	
56	04	AC	D0	0001F	MOVL ERROR, ERROR_ADDR	: 0241
08	AE	56	D0	00023	MOVL PFV, R6	: 0243
	00000000G	00	16	00027	MOVL R6, PFV_ADDR	
04	AE	01	D0	0002D	JSB PASS\$VALIDATE_PFV	: 0249
	00000000G	00	16	00031	MOVL #1, UNWIND_ACT	: 0255
52	08	AC	D0	00037	JSB PASS\$INIT_WRITE	: 0261
54	03	A2	9E	0003B	MOVL NBITS, ACTUAL_NBITS	: 0267
54		04	C6	0003F	MOVAB 3(R2), R4	: 0268
55		54	D0	00042	DIVL2 #4, R4	
0E	AE	8F	B0	00045	MOVL R4, ACTUAL_DIGITS	
10	AE	A7	D0	0004B	MOVW #270, DESCR+2	: 0275
		6C	91	00050	MOVL -20(FCB), DESCR+4	: 0276
		06	1F	00053	CMPB (AP), #6	: 0277
55	18	AC	D0	00055	BLSSU 2\$	
		06	19	00059	MOVL MIN_DIGITS, ACTUAL_DIGITS	: 0280
53	10	AC	D0	0005B	BLSS 3\$	: 0281
		0A	18	0005F	MOVL TOTAL_WIDTH, R3	: 0286
		7E	D4	00061	BGEQ 4\$	
					CLRL -(SP)	: 0288

PASSWRITE_HEX
1-002

Write a value in base 16

PASSWRITE_HEX - Write a value in hexadecimal to

D 1

16-Sep-1984 02:19:37

14-Sep-1984 12:52:05

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

Page 7
(3)

	7E	00G	8F	9A	00063	MOVZBL	#PASSK_NEGWIDDIG, -(SP)	
	68		02	FB	00067	CALLS	#2, PASS\$SIGNAL	
				04	0006A	RET		
OC	AE		53	B0	0006B	4\$: MOVW	R3, DESCR	0290
	54		53	D1	0006F	CML	R3, R4	0295
			04	1E	00072	BGEQU	5\$	
52	53		02	78	00074	ASHL	#2, R3, ACTUAL_NBITS	0301
	53		55	D1	00078	5\$: CML	ACTUAL_DIGITS, R3	0304
			03	15	0007B	BLEQ	6\$	
	55		53	D0	0007D	MOVL	R3, ACTUAL_DIGITS	0306
	50	OC	AE	3C	00080	6\$: MOVZWL	DESCR, R0	0315
	50	EC	A7	C0	00084	ADDL2	-20(FCB), R0	
	50	FO	A7	C2	00088	SUBL2	-16(FCB), EXTRA	
			OC	15	0008C	BLEQ	7\$	0316
			50	DD	0008E	PUSHL	EXTRA	0318
			01	DD	00090	PUSHL	#1	
	7E	00G	8F	9A	00092	MOVZBL	#PASSK_LINTOOLON, -(SP)	
	68		03	FB	00096	CALLS	#3, PASS\$SIGNAL	
				04	00099	RET		
			04	DD	0009A	7\$: PUSHL	#4	0325
			52	DD	0009C	PUSHL	ACTUAL_NBITS	
			55	DD	0009E	PUSHL	ACTUAL_DIGITS	
		18	AE	9F	000A0	PUSHAB	DESCR	
		OC	AC	DD	000A3	PUSHL	VALUE	
00000000G	00		05	FB	000A6	CALLS	#5, OT\$SCVT_L_TZ	
	50	OC	AE	3C	000AD	MOVZWL	DESCR, R0	0328
EC	A7		50	C0	000B1	ADDL2	R0, -20(FCB)	
		00000000G	00	16	000B5	JSB	PASS\$END_WRITE	0335
				04	000BB	RET		0339
			0000	000BC	8\$: .WORD	Save nothing		0215
	50	08	AC	D0	000BE	MOVL	8(AP), R0	
	50	04	A0	D0	000C2	MOVL	4(R0), R0	
		EC	A0	9F	000C6	PUSHAB	ERROR_ADDR	
		FO	A0	9F	000C9	PUSHAB	UNWIND_ACT	
		F4	A0	9F	000CC	PUSHAB	PFV_ADDR	
			03	DD	000CF	PUSHL	#3	
			5E	DD	000D1	PUSHL	SP	
	7E	04	AC	7D	000D3	MOVQ	4(AP), -(SP)	
00000000G	00		03	FB	000D7	CALLS	#3, PASS\$IO_HANDLER	
			04	000DE	RET			

; Routine Size: 223 bytes, Routine Base: _PASSCODE + 0000

; 279 0340 1
; 280 0341 1 !<BLF/PAGE>

PASSWRITE_HEX
1-002

Write a value in base 16

PASSWRITEV_HEX - Write value in base 16 to string

E 1
16-Sep-1984 02:19:37
14-Sep-1984 12:52:05

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

Page 8
(4)

```

282 0342 1 %SBTTL 'PASSWRITEV_HEX - Write value in base 16 to string'
283 0343 1 GLOBAL ROUTINE PASSWRITEV_HEX (
284 0344 1     MAX_LENGTH: WORD,           ! Maximum length of string
285 0345 1     STRING_LINE: REF VECTOR [, WORD], ! String to write to
286 0346 1     NBITS,                   ! Number of bits in VALUE
287 0347 1     VALUE: REF VECTOR [, BYTE], ! Value to write
288 0348 1     TOTAL_WIDTH: SIGNED,      ! Total field width
289 0349 1     ERROR,                   ! Error unwind address
290 0350 1     MIN_DIGITS: SIGNED        ! Minimum number of digits
291 0351 1 ) : NOVALUE =
292 0352 1
293 0353 1 ++
294 0354 1 FUNCTIONAL DESCRIPTION:
295 0355 1
296 0356 1     This procedure writes a value in base 16 to the specified string.
297 0357 1
298 0358 1 CALLING SEQUENCE:
299 0359 1
300 0360 1     CALL PASSWRITEV_HEX (MAX_LENGTH.rw.v, STRING_LINE.wvt.r,
301 0361 1     NBITS.rl.v, VALUE.rz.v, TOTAL_WIDTH.rl.v
302 0362 1     [, [ERROR.].r] [, MIN_DIGITS.r[v]])
303 0363 1
304 0364 1 FORMAL PARAMETERS:
305 0365 1
306 0366 1     MAX_LENGTH      - The maximum length of STRING_LINE.
307 0367 1
308 0368 1     STRING_LINE    - A varying string to which the output will be appended.
309 0369 1
310 0370 1     NBITS          - Size of VALUE in bits.
311 0371 1
312 0372 1     VALUE          - The value to write.
313 0373 1
314 0374 1     TOTAL_WIDTH    - The width of the field to write.
315 0375 1
316 0376 1     ERROR          - Optional. If specified, the address to unwind to
317 0377 1                   in case of an error.
318 0378 1
319 0379 1     MIN_DIGITS     - Minimum number of digits to write. Defaults to
320 0380 1                   the minimum necessary to represent every bit of
321 0381 1                   the value.
322 0382 1
323 0383 1 IMPLICIT INPUTS:
324 0384 1
325 0385 1     NONE
326 0386 1
327 0387 1 IMPLICIT OUTPUTS:
328 0388 1
329 0389 1     NONE
330 0390 1
331 0391 1 ROUTINE VALUE:
332 0392 1
333 0393 1     NONE
334 0394 1
335 0395 1 SIDE EFFECTS:
336 0396 1
337 0397 1     NONE
338 0398 1
```

```

339 0399 1 | SIGNALLED ERRORS:
340 0400 1 |
341 0401 1 |     See PASSWRITE_HEX
342 0402 1 |
343 0403 1 |
344 0404 1 |
345 0405 2 | BEGIN
346 0406 2 |
347 0407 2 | LOCAL
348 0408 2 |     PFV: $PASSPFV_FILE_VARIABLE, | Pascal File Variable
349 0409 2 |     ARG_LIST: VECTOR [7, LONG], | Argument list
350 0410 2 |     PFV_ADDR: VOLATILE, | Enable argument
351 0411 2 |     UNWIND_ACT: VOLATILE, | Enable argument
352 0412 2 |     ERROR_ADDR: VOLATILE; | Enable argument
353 0413 2 |
354 0414 2 | BUILTIN
355 0415 2 |     ACTUALCOUNT; | Count of arguments
356 0416 2 |
357 0417 2 | ENABLE
358 0418 2 |     PASS$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR); | Enable error handler
359 0419 2 |
360 0420 2 | !+
361 0421 2 | ! Get ERROR parameter, if present.
362 0422 2 | !-
363 0423 2 |
364 0424 2 | IF ACTUALCOUNT () GEQU 6
365 0425 2 | THEN
366 0426 2 |     ERROR_ADDR = .ERROR; | Set unwind address
367 0427 2 |
368 0428 2 | PFV_ADDR = PFV [PFV$R_PFV]; | Set PFV address
369 0429 2 |
370 0430 2 | !+
371 0431 2 | ! Set up ARG_LIST.
372 0432 2 | !-
373 0433 2 |
374 0434 2 | ARG_LIST [0] = 4; | Four arguments
375 0435 2 | ARG_LIST [1] = PFV [PFV$R_PFV]; | PFV address
376 0436 2 | ARG_LIST [2] = .NBITS; | Number of bits
377 0437 2 | ARG_LIST [3] = VALUE [0]; | Value to write
378 0438 2 | ARG_LIST [4] = .TOTAL_WIDTH; | Field width
379 0439 2 | IF ACTUALCOUNT () GEQU 7
380 0440 2 | THEN
381 0441 3 | BEGIN
382 0442 3 |     ARG_LIST [0] = 6; | Add two more arguments
383 0443 3 |     ARG_LIST [5] = 0; | Error address
384 0444 3 |     ARG_LIST [6] = .MIN_DIGITS; | Minimum digits
385 0445 2 | END;
386 0446 2 |
387 0447 2 | !+
388 0448 2 | ! Call PASS$DO_WRITEV to do the work, giving it the address of
389 0449 2 | ! PASSWRITE_HEX to call.
390 0450 2 | !-
391 0451 2 |
392 0452 2 | PASS$DO_WRITEV (PFV [PFV$R_PFV], .MAX_LENGTH, STRING_LINE [0], ARG_LIST,
393 0453 2 |     PASSWRITE_HEX);
394 0454 2 |
395 0455 2 | RETURN;

```

```

: 396      0456 2
: 397      0457 1      END:

```

```
! End of routine PASSWRITEV_HEX
```

				007C 00000	.EXTRN	PASS\$DO_WRITEV	
	5E		34	C2 00002	.ENTRY	PASSWRITEV_HEX, Save R2,R3,R4,R5,R6	: 0343
			7E	D4 00005	SUBL2	#52, SP	: 0405
		04	AE	7C 00007	CLRL	ERROR_ADDR	: 0424
	6D	004F	CF	DE 0000A	CLRL	UNWIND_ACT	: 0426
	06		6C	91 0000F	MOVAL	3\$, (FP)	: 0428
			04	1F 00012	CMPB	(AP), #6	: 0434
	6E	18	AC	D0 00014	BLSSU	1\$	: 0435
08	AE	28	AE	9E 00018	MOVL	ERROR, ERROR_ADDR	: 0436
0C	AE		04	D0 0001D	MOVAB	PFV, PFV_ADDR	: 0438
10	AE	28	AE	9E 00021	MOVL	#4, ARG_LIST	: 0439
14	AE	0C	AC	7D 00026	MOVAB	PFV, ARG_LIST+4	: 0442
1C	AE	14	AC	D0 0002B	MOVQ	NBITS, ARG_LIST+8	: 0443
	07		6C	91 00030	MOVQ	TOTAL_WIDTH, ARG_LIST+16	: 0444
			0C	1F 00033	CMPB	(AP), #7	: 0452
0C	AE		06	D0 00035	BLSSU	2\$	: 0457
		20	AE	D4 00039	MOVL	#6, ARG_LIST	: 0405
24	AE	1C	AC	D0 0003C	CLRL	ARG_LIST+20	
	55	FEDC	CF	9E 00041	MOVL	MIN_DIGITS, ARG_LIST+24	
	54	0C	AE	9E 00046	MOVAB	PASSWRITE_HEX, R5	
	56	28	AE	9E 0004A	MOVAB	ARG_LIST, R4	
	53	08	AC	D0 0004E	MOVAB	PFV, R6	
	52	04	AC	3C 00052	MOVL	STRING_LINE, R3	
		00000000G	00	16 00056	MOVZWL	MAX_LENGTH, R2	
			04	0005C	JSB	PASS\$DO_WRITEV	
			0C	0005D	RET		
	50	08	AC	D0 0005F	.WORD	Save nothing	
	50	04	A0	D0 00063	MOVL	8(AP), R0	
		C8	A0	9F 00067	MOVL	4(R0), R0	
		CC	A0	9F 0006A	PUSHAB	ERROR_ADDR	
		D0	A0	9F 0006D	PUSHAB	UNWIND_ACT	
			03	DD 00070	PUSHAB	PFV_ADDR	
			5E	DD 00072	PUSHL	#3	
			04	7D 00074	PUSHL	SP	
00000000G	7E	04	AC	7D 00074	MOVQ	4(AP), -(SP)	
	00		03	FB 00078	CALLS	#3, PASS\$IO_HANDLER	
			04	0007F	RET		

```
; Routine Size: 128 bytes, Routine Base: _PASS$CODE + 00DF
```

```

: 398      0458 1
: 399      0459 1 !<BLF/PAGE>

```

PASSWRITE_HEX

1-002

Write a value in base 16

PASSWRITEV_HEX - Write value in base 16 to stri

H 1

16-Sep-1984 02:19:37

14-Sep-1984 12:52:05

VAX-11 Bliss-32 V4.0-742

[PASRTL.SRC]PASWRIHEX.B32;1

Page 11

(5)

: 401

: 402

: 403

0460 1 END

0461 1

0462 0 ELUDOM

! End of module PASSWRITE_HEX

:

:

:

:

PSECT SUMMARY

Name

Bytes

Attributes

_PASS\$CODE

351

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	6	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\$:PASWRIHEX/OBJ=OBJ\$:PASWRIHEX MSRC\$:PASWRIHEX/UPDATE=(ENH\$:PASWRIHEX

)

Size: 351 code + 0 data bytes

Run Time: 00:09.0

Elapsed Time: 00:25.8

Lines/CPU Min: 3076

Lexemes/CPU-Min: 15029

Memory Used: 103 pages

Compilation Complete

0297

AH-BT13A-SE
 VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

0298 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PASWIREH
LIS

PASWIREL
LIS

PASWIREG
LIS

PASWISTR
LIS

PATCH

PATDEF
MDL

PASWIREF
LIS

PATCH
MAP

SRMDEF
MDL

PASWIRFH
LIS

PASWIRAR
LIS

PASWIREL
LIS

PASWIRLS
LIS

PASWIRINT
LIS

PASWIRIOCT
LIS

PASWIRFG
LIS

BSTRUC
REQ

PASWIRFF
LIS

CHRKEY
REQ

PASWIRFD
LIS