

```

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
PASPAS
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	RRRRRRRR	EEEEEEEEEE	AAAAAA	SSSSSSSS	TTTTTTTTTT	11	
PPPPPPPP	AAAAAA	SSSSSSSS	RRRRRRRR	EEEEEEEEEE	AAAAAA	SSSSSSSS	TTTTTTTTTT	11	
PP	AA	SS	RR	EE	AA	SS	TT	1111	
PP	AA	SS	RR	EE	AA	SS	TT	1111	
PP	AA	SS	RR	EE	AA	SS	TT	11	
PP	AA	SS	RR	EE	AA	SS	TT	11	
PPPPPPPP	AA	SSSSSS	RRRRRRRR	EEEEEEEEEE	AA	SSSSSS	TT	11	
PPPPPPPP	AA	SSSSSS	RRRRRRRR	EEEEEEEEEE	AA	SSSSSS	TT	11	
PP	AAAAAAAAAA	SS	RR	EE	AAAAAAAAAA	SS	TT	11	
PP	AAAAAAAAAA	SS	RR	EE	AAAAAAAAAA	SS	TT	11	
PP	AA	SS	RR	EE	AA	SS	TT	11	
PP	AA	SS	RR	EE	AA	SS	TT	11	
PP	AA	SSSSSSSS	RR	EE	AA	SSSSSSSS	TT	111111	
PP	AA	SSSSSSSS	RR	EE	AA	SSSSSSSS	TT	111111	

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


```
1 0001 0 MODULE PASS$READ_STRING1 ( %TITLE 'Read a fixed-length string - V1 semantics'
2 0002 0 IDENT = '1-001' ! File: PASREAST1.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 a procedure which reads a fixed-length string
36 0036 1 from a textfile using the semantics of VAX-11 Pascal V1.
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';           ! Externals, linkages, 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     PASSREAD_STRING1: NOVALUE,       ! Read string
: 61      0123 1     LOCAL_HANDLER;                  ! Local condition handler
: 62      0124 1
: 63      0125 1
: 64      0126 1 MACROS:
: 65      0127 1
: 66      0128 1     NONE
: 67      0129 1
: 68      0130 1 EQUATED SYMBOLS:
: 69      0131 1
: 70      0132 1     NONE
: 71      0133 1
: 72      0134 1 FIELDS:
: 73      0135 1
: 74      0136 1     NONE
: 75      0137 1
: 76      0138 1 OWN STORAGE:
: 77      0139 1
: 78      0140 1     NONE
: 79      0141 1

```



```

81 0142 1 %SBTTL 'PASSREAD_STRING1 - READ a string with V1 semantics'
82 0143 1 GLOBAL ROUTINE PASSREAD_STRING1 ( READ a string
83 0144 1 STRING: REF VECTOR [, BYTE], String to read into
84 0145 1 PFV: REF $PASSPFV FILE_VARIABLE, File variable
85 0146 1 STRING_LENGTH: WORD, Length of string
86 0147 1 ERROR Error unwind address
87 0148 1 ) : NOVALUE =
88 0149 1
89 0150 1 ++
90 0151 1 FUNCTIONAL DESCRIPTION:
91 0152 1
92 0153 1 This procedure reads a fixed-length string from the specified textfile.
93 0154 1 If the string length is less than the number of characters remaining
94 0155 1 in the current line, only STRING_LENGTH characters will be read,
95 0156 1 otherwise the string will be padded with blanks.
96 0157 1
97 0158 1 This procedure will be referenced by the compiled code if the
98 0159 1 module has been compiled with the /OLD_VERSION qualifier so as to
99 0160 1 comply with the semantics of VAX-11 Pascal V1. The only difference
100 0161 1 between this procedure and PASSREAD_STRING is that in this procedure,
101 0162 1 if the file is currently at end-of-line, a GET is done before
102 0163 1 reading characters.
103 0164 1
104 0165 1 This procedure uses only public "single-dollar" interfaces to the
105 0166 1 rest of the Pascal Run-time Library. This is so that this module
106 0167 1 may be excluded from the shareable image PASRTL.EXE
107 0168 1
108 0169 1 CALLING SEQUENCE:
109 0170 1
110 0171 1 STRING.wt.r = PASSREAD_STRING1 (PFV.mr.r, STRING_LENGTH.rwu.v
111 0172 1 [, ERROR.ja.r])
112 0173 1
113 0174 1 FORMAL PARAMETERS:
114 0175 1
115 0176 1 PFV - The Pascal File Variable (PFV) passed by reference.
116 0177 1 The structure of the PFV is defined in PASPFV.REQ.
117 0178 1
118 0179 1 STRING_LENGTH - The length of the string to read.
119 0180 1
120 0181 1 ERROR - Optional. If specified, the address to unwind to
121 0182 1 in case of an error.
122 0183 1
123 0184 1 IMPLICIT INPUTS:
124 0185 1
125 0186 1 NONE
126 0187 1
127 0188 1 IMPLICIT OUTPUTS:
128 0189 1
129 0190 1 NONE
130 0191 1
131 0192 1 ROUTINE VALUE:
132 0193 1
133 0194 1 STRING - The string read, returned as a function value by
134 0195 1 having the string address passed as the first
135 0196 1 procedure parameter, in accordance with the
136 0197 1 VAX Procedure Calling Standard.
137 0198 1

```

```

138 0199 1 | If an error occurs and is continued by a user handler,
139 0200 1 | the result returned is a blank string.
140 0201 1 |
141 0202 1 | SIDE EFFECTS:
142 0203 1 |
143 0204 1 |     If the file is the standard file INPUT or OUTPUT, it is implicitly opened.
144 0205 1 |
145 0206 1 | SIGNALLED ERRORS:
146 0207 1 |
147 0208 1 |
148 0209 1 | --
149 0210 1 |
150 0211 2 | BEGIN
151 0212 2 |
152 0213 2 | LOCAL
153 0214 2 |     CHARS_REMAINING, | Number of characters remaining in line
154 0215 2 |     FCB: REF $PASSFCB_CONTROL_BLOCK, | File Control block
155 0216 2 |     PFV_ADDR: VOLATILE, | Enable argument
156 0217 2 |     UNWIND_ACT: VOLATILE, | Enable argument
157 0218 2 |     ERROR_ADDR: VOLATILE; | Enable argument
158 0219 2 |
159 0220 2 | BUILTIN
160 0221 2 |     ACTUALCOUNT; | Count of arguments
161 0222 2 |
162 0223 2 | ENABLE
163 0224 2 |     LOCAL_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR); | Enable error handler
164 0225 2 |
165 0226 2 | +
166 0227 2 | | Get ERROR parameter, if present.
167 0228 2 | -
168 0229 2 |
169 0230 2 | IF ACTUALCOUNT () GEQU 4
170 0231 2 | THEN
171 0232 2 |     ERROR_ADDR = .ERROR; | Set unwind address
172 0233 2 |
173 0234 2 | PFV_ADDR = PFV [PFV$R_PFV]; | Set PFV address
174 0235 2 |
175 0236 2 | +
176 0237 2 | | The V2 semantics for read of a string say that if you are at EOLN,
177 0238 2 | | then an empty string is returned. In other words, read of a string
178 0239 2 | | does not read past EOLNs. The V1 semantics, however, say that if
179 0240 2 | | you are at EOLN, you first do a GET to go to the next line. So,
180 0241 2 | | find out if we are at EOLN, and if so, do a GET.
181 0242 2 | -
182 0243 2 |
183 0244 2 | IF PAS$EOLN2 (PFV [PFV$R_PFV])
184 0245 2 | THEN
185 0246 2 |     PAS$GET (PFV [PFV$R_PFV]);
186 0247 2 |
187 0248 2 | +
188 0249 2 | | Now call the V2 read string routine.
189 0250 2 | -
190 0251 2 |
191 0252 2 | PASSREAD_STRING (STRING [0], PFV [PFV$R_PFV], .STRING_LENGTH);
192 0253 2 |
193 0254 2 | RETURN;
194 0255 2 |

```


; 195

0256 1

END;

! End of routine PASSREAD_STRING1

				.TITLE	PASSREAD_STRING1 Read a fixed-length string - V1 semantics	
				.IDENT	\1-001\	
				.EXTRN	PASSREAD_STRING1	
				.EXTRN	PASSEOLN2, PASSGET	
				.EXTRN	PASSREAD_STRING	
				.PSECT	_PASSCODE, NOWRT, SHR, PIC, 2	
				.ENTRY	PASSREAD_STRING1, Save nothing	: 0143
				SUBL2	#8, SP	
				CLRL	ERROR_ADDR	: 0211
				CLRQ	UNWIND_ACT	
				MOVAL	3\$, (FP)	
				CMPB	(AP), #4	: 0230
				BLSSU	1\$	
				MOVL	ERROR, ERROR_ADDR	: 0232
				MOVL	PFV, PFV_ADDR	: 0234
				PUSHL	PFV	: 0244
				CALLS	#1, PASSEOLN2	
				BLBC	R0, 2\$	
				PUSHL	PFV	: 0246
				CALLS	#1, PASSGET	
				MOVZWL	STRING_LENGTH, -(SP)	: 0252
				MOVQ	STRING, -(SP)	
				CALLS	#3, PASSREAD_STRING	
				RET		: 0256
				.WORD	Save nothing	: 0211
				MOVL	8(AP), R0	
				MOVL	4(R0), R0	
				PUSHAB	ERROR_ADDR	
				PUSHAB	UNWIND_ACT	
				PUSHAB	PFV_ADDR	
				PUSHL	#3	
				PUSHL	SP	
				MOVQ	4(AP), -(SP)	
				CALLS	#3, LOCAL_HANDLER	
				RET		

; Routine Size: 101 bytes, Routine Base: _PASSCODE + 0000

; 196

0257 1

; 197

0258 1 !<BLF/PAGE>

```

199 0259 1 %SBTTL 'LOCAL_HANDLER - Local handler'
200 0260 1 ROUTINE LOCAL_HANDLER (
201 0261 1     SIGNAL_ARGS: REF BLOCK [, BYTE],      ! Signal arguments array
202 0262 1     MECH_ARGS: REF BLOCK [, BYTE],      ! Mechanism arguments array
203 0263 1     ENABLE_ARGS: REF VECTOR [, LONG]    ! Enable arguments array
204 0264 1 ) =
205 0265 1
206 0266 1 !++
207 0267 1 FUNCTIONAL DESCRIPTION:
208 0268 1
209 0269 1     This is the condition handler enabled by PASSREAD_STRING1.
210 0270 1     If the current signal is a Pascal error on the file our establisher
211 0271 1     was called with, we unwind to the caller of PASSREAD_STRING1
212 0272 1     with R0 being the status code of the error.
213 0273 1
214 0274 1
215 0275 1 CALLING SEQUENCE:
216 0276 1
217 0277 1     status.wlc.v = STATUS_HANDLER (SIGNAL_ARGS.rl.ra, MECH_ARGS.rl.ra,
218 0278 1                                   ENABLE_ARGS.rl.ra)
219 0279 1
220 0280 1 FORMAL PARAMETERS:
221 0281 1
222 0282 1     SIGNAL_ARGS    - The signal argument list.
223 0283 1
224 0284 1     MECH_ARGS      - The mechanism argument list.
225 0285 1
226 0286 1     ENABLE_ARGS    - An array with the following
227 0287 1                     format:
228 0288 1
229 0289 1
230 0290 1
231 0291 1
232 0292 1
233 0293 1
234 0294 1
235 0295 1
236 0296 1
237 0297 1
238 0298 1
239 0299 1
240 0300 1
241 0301 1
242 0302 1
243 0303 1
244 0304 1
245 0305 1
246 0306 1
247 0307 1
248 0308 1
249 0309 1
250 0310 1
251 0311 1
252 0312 1
253 0313 1
254 0314 1
255 0315 1

```

ENB_COUNT is the count of following enable arguments.
The count is always at least 2.

ENB_PFV_ADDR - If non-zero, the address of a longword
containing the PFV our establisher is operating on.

ENB_UNWIND_ACT - Specifies the action
to take on an unwind. The values are:
PASSK_UNWIND_NOP - Do nothing
PASSK_UNWIND_UNLOCK - Unlock PFV

ENB_ERROR_ADDR - Ignored here.

IMPLICIT INPUTS:

The signaller's PFV placed as the first FA0 argument in the primary
signalled message.


```

256 0316 1 |
257 0317 1 | IMPLICIT OUTPUTS:
258 0318 1 |
259 0319 1 |     NONE
260 0320 1 |
261 0321 1 | ROUTINE VALUE:
262 0322 1 |
263 0323 1 |     SS$_RESIGNAL
264 0324 1 |
265 0325 1 | SIDE EFFECTS:
266 0326 1 |
267 0327 1 |     May cause an unwind.
268 0328 1 |
269 0329 1 | --
270 0330 1 |
271 0331 2 | BEGIN
272 0332 2 |
273 0333 2 | LITERAL
274 0334 2 |     ENB_COUNT = 0,           ! Count of enable arguments
275 0335 2 |     ENB_PFV_ADDR = 1,       ! Address of address of PFV
276 0336 2 |     ENB_UNWIND_ACT = 2,     ! Address of unwind action
277 0337 2 |     ENB_ERROR_ADDR = 3;     ! Address of address of unwind PC
278 0338 2 |
279 0339 2 | !+
280 0340 2 | ! Determine if this is an unwind. If so, resignal.
281 0341 2 | ! Otherwise, see if we should cause an unwind.
282 0342 2 | !-
283 0343 2 |
284 0344 2 | IF .SIGNAL_ARGS [CHF$_SIG_NAME] EQLU SS$_UNWIND
285 0345 2 | THEN
286 0346 2 |     RETURN SS$_RESIGNAL;
287 0347 2 |
288 0348 2 | IF ..ENABLE_ARGS [ENB_ERROR_ADDR] NEQ 0      ! Error:=Continue specified?
289 0349 2 | THEN
290 0350 2 |     BEGIN
291 0351 2 |         LOCAL
292 0352 2 |             COND_NAME: BLOCK [4, BYTE], ! Primary condition name
293 0353 2 |             COND_CODE;                  ! Sequence number of error
294 0354 2 |
295 0355 2 |         !+
296 0356 2 |         ! Get primary condition name.
297 0357 2 |         !-
298 0358 2 |
299 0359 2 |         COND_NAME = .SIGNAL_ARGS [CHF$_SIG_NAME];
300 0360 2 |
301 0361 2 |         !+
302 0362 2 |         ! Is this a PASS$ error? If not, resignal.
303 0363 2 |         !-
304 0364 2 |
305 0365 2 |         IF .COND_NAME [STSSV_FAC_NO] NEQU PASS$_FACILITY
306 0366 2 |         THEN
307 0367 2 |             RETURN SS$_RESIGNAL;
308 0368 2 |
309 0369 2 |         !+
310 0370 2 |         ! See if the error message is one which is "trapped"
311 0371 2 |         ! by ERROR:=CONTINUE. This is done by comparing the
312 0372 2 |

```

```

313      0373      3      ! message number against a select range.
314      0374      3      !
315      0375      3      !
316      0376      3      COND_CODE = .COND_NAME [ST$V CODE];      ! Get error number
317      0377      3      IF .COND_CODE GEQD PASS$K_MSGCONTLO AND      ! Lowest number
318      0378      3      .COND_CODE LEQU PASS$K_MSGCONTHI      ! Highest number
319      0379      3      THEN
320      0380      4      BEGIN
321      0381      4      !
322      0382      4      !+ See if the PFVs match. The signaller's PFV is the
323      0383      4      ! first FA0 parameter in the primary message.
324      0384      4      !-
325      0385      4      !
326      0386      4      !
327      0387      4      IF .SIGNAL_ARGS [12,0,32,0] EQLA ..ENABLE_ARGS [ENB_PFV_ADDR]
328      0388      4      THEN
329      0389      4      !+
330      0390      4      ! We want to unwind to the PC specified in the enable argument
331      0391      4      ! error address.
332      0392      4      !-
333      0393      4      !
334      0394      5      BEGIN
335      0395      6      IF NOT $UNWIND (NEWPC=..ENABLE_ARGS [ENB_ERROR_ADDR])
336      0396      5      THEN
337      0397      5      SIGNAL_STOP (PASS$_BUGCHECK,1,BUG_UNWINDFAIL);
338      0398      4      END;
339      0399      3      END;
340      0400      2      END;
341      0401      2      RETURN SS$_RESIGNAL;      ! Resignal error
342      0402      2
343      0403      2
344      0404      1      END;      ! End of routine LOCAL_HANDLER

```

```

.EXTRN PASS$_FACILITY, PASS$K_MSGCONTLO
.EXTRN PASS$K_MSGCONTHI
.EXTRN SY$UNWIND, PASS$_BUGCHECK

```

0004 00000 LOCAL_HANDLER:

		51	04	AC	D0	00002	.WORD	Save R2	0260
		8F	04	A1	D1	00006	MOVL	SIGNAL_ARGS, R1	0344
	00000920			52	13	0000E	CMPL	4(R1), #2336	
		50	0C	AC	D0	00010	BEQL	1\$	
			0C	B0	D5	00014	MOVL	ENABLE_ARGS, R0	0348
				49	13	00017	TSTL	@12(R0)	
		52	04	A1	D0	00019	BEQL	1\$	
00G	52	0C		10	ED	0001D	MOVL	4(R1), COND_NAME	0360
				3E	12	00022	CMPL	#16, #12, COND_NAME, S^PASS\$_FACILITY	0366
				03	EF	00024	BNEQ	1\$	
52	52	0C		03	EF	00024	EXTZV	#3, #12, COND_NAME, COND_CODE	0376
	00000000G	8F		52	D1	00029	CMPL	COND_CODE, #PASS\$K_MSGCONTLO	0377
				30	1F	00030	BLSSU	1\$	
	00000000G	8F		52	D1	00032	CMPL	COND_CODE, #PASS\$K_MSGCONTHI	0378
				27	1A	00039	BGTRU	1\$	
	04	B0	0C	A1	D1	0003B	CMPL	12(R1), @4(R0)	0387
				20	12	00040	BNEQ	1\$	

		0C	B0	DD	00042		PUSHL	@12(R0)
			7E	D4	00045		CLRL	-(SP)
00000000G	00		02	FB	00047		CALLS	#2, SYSSUNWIND
	11		50	E8	0004E		BLBS	R0, 1\$
			03	DD	00051		PUSHL	#3
			01	DD	00053		PUSHL	#1
		00000000G	8F	DD	00055		PUSHL	#PASS\$ BUGCHECK
00000000G	00		03	FB	00058		CALLS	#3, LIB\$STOP
	50	0918	8F	3C	00062	1\$:	MOVZWL	#2328, R0
				04	00067		RET	

```
; Routine Size: 104 bytes,    Routine Base: _PASSCODE + 0065
```

```

: 345      0405 1
: 346      0406 1 !<BLF/PAGE>

```

PASSREAD_STRING Read a fixed-length string - V1 semantics
1-001 LOCAL_HANDLER - Local handler

M 12
16-Sep-1984 02:02:20 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:51:50 [PASRTL.SRC]PASREAST1.B32;1

Page 10
(5)

: 348 0407 1 END
: 349 0408 1
: 350 0409 0 ELUDOM

! End of module PASSREAD_STRING1

.EXTRN LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
_PASS\$CODE	205 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	7	0	581	00:01.0
\$255\$DUA28:[PASRTL.OBJ]PASLIB.L32;1	427	87	20	33	00:00.4

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LISS\$:PASREAST1/OBJ=OBJ\$:PASREAST1 MSRC\$:PASREAST1/UPDATE=(ENH\$:PASREAST1)

: Size: 205 code + 0 data bytes
: Run Time: 00:06.6
: Elapsed Time: 00:23.5
: Lines/CPU Min: 3740
: Lexemes/CPU-Min: 9804
: Memory Used: 70 pages
: Compilation Complete

0296 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

