

```

EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD TTTTTTTTTTTTTTTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEEEEEEEEEEEEE DDD DDD TTT
EEEEEEEEEEEEEE DDD DDD TTT
EEEEEEEEEEEEEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD TTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD TTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD TTT

```

EXE

Mod

ED 1

ED

ED

ED

ED

ED1

ED

ED

ED
EDED
EDSYS
LIB

L18

★★

```

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 IIIIIIII      SSSSSSSS
LLLLLLLLLLLL IIIIIIII      SSSSSSSS

```



```
0001 0 %TITLE 'EDT$PRGETTOK - scan a token'
0002 0 MODULE EDT$PRGETTOK (
0003 0 IDENT = 'V04-000'
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1 *****
0008 1 *
0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0011 1 * ALL RIGHTS RESERVED.
0012 1 *
0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0018 1 * TRANSFERRED.
0019 1 *
0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0022 1 * CORPORATION.
0023 1 *
0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0026 1 *
0027 1 *
0028 1 *****
0029 1
0030 1
0031 1 ++
0032 1 FACILITY: EDT -- The DEC Standard Editor
0033 1
0034 1 ABSTRACT:
0035 1
0036 1 Scan the next parse token.
0037 1
0038 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0039 1
0040 1 AUTHOR: Bob Kushlis, CREATION DATE: December 12, 1978
0041 1
0042 1 MODIFIED BY:
0043 1
0044 1 1-001 - Original. DJS 25-Feb-1981. This module was created by
0045 1 extracting routine EDT$SPA_SCANTOK from module PARSER.
0046 1 1-002 - Regularize headers. JBS 12-Mar-1981
0047 1 1-003 - Suppress quoted strings if requested. JBS 26-Aug-1981
0048 1 1-004 - Change index on line numbers for 15 instead of 10 digits. SMB 18-Jan-1982
0049 1 1-005 - Accept quoted keys. STS 07-Apr-1982
0050 1 1-006 - Delete reference to edt$gpa_val. STS 09-Apr-1982
0051 1 1-007 - Change numeric test. JBS 19-Jul-1982
0052 1 1-008 - Modify to use new 48 bit arith macors. STS 01-Oct-1982
0053 1 1-009 - Don't strip diacritical marks from letters. JBS 13-Dec-1982
0054 1 1-010 - Adjust EDT$SA_CMD_END when shortening a string to eliminate a doubled quote. JBS 18-Jan-1983
0055 1 1-011 - Add VT220 support conditional. JBS 11-Feb-1983
0056 1 1-012 - Correct an error in a comment. JBS 17-Jun-1983
0057 1 --
```

EDT\$PRGETTOK
V04-000

EDT\$PRGETTOK - scan a token

: 58

0058 1

I 15
16-Sep-1984 01:17:40
14-Sep-1984 12:24:09

VAX-11 BLISS-32 V4.0-742
DISK\$VMMASTER:[EDT.SRC]PRGETTOK.BLI;1 Page 2 (1)

ED
VO

EDT\$PRGETTOK
V04-000

EDT\$PRGETTOK - scan a token
Declarations

J 15
16-Sep-1984 01:17:40
14-Sep-1984 12:24:09

VAX-11 Bliss-32 V4.0-742
DISK\$VMMASTER:[EDT.SRC]PRGETTOK.BLI;1
Page 3
(2)

```

: 60      0059 1 %SBTTL 'Declarations'
: 61      0060 1
: 62      0061 1 | TABLE OF CONTENTS:
: 63      0062 1 |
: 64      0063 1 |
: 65      0064 1 REQUIRE 'EDT$SRC:TRAROUNAM';
: 66      0503 1
: 67      0504 1 FORWARD ROUTINE
: 68      0505 1     EDT$SPA_SCANTOK : NOVALUE;
: 69      0506 1
: 70      0507 1 |
: 71      0508 1 | INCLUDE FILES:
: 72      0509 1 |
: 73      0510 1 |
: 74      0511 1 REQUIRE 'EDT$SRC:EDTREQ';
: 75      0646 1
: 76      0647 1 REQUIRE 'EDT$SRC:PARLITS';
: 77      0931 1
: 78      0932 1 LIBRARY 'EDT$SRC:SUPPORTS';
: 79      0933 1
: 80      0934 1 |
: 81      0935 1 | MACROS:
: 82      0936 1 |
: 83      0937 1 |     NONE
: 84      0938 1 |
: 85      0939 1 | EQUATED SYMBOLS:
: 86      0940 1 |
: 87      0941 1 |     NONE
: 88      0942 1 |
: 89      0943 1 | OWN STORAGE:
: 90      0944 1 |
: 91      0945 1 |     NONE
: 92      0946 1 |
: 93      0947 1 | EXTERNAL REFERENCES:
: 94      0948 1 |
: 95      0949 1 |     In the routine
```

ED
VO

```

97 0950 1 %SBTTL 'EDT$$PA_SCANTOK - scan the next token'
98 0951 1
99 0952 1 GLOBAL ROUTINE EDT$$PA_SCANTOK ! Scan the next token
100 0953 1 : NOVALUE =
101 0954 1
102 0955 1 ++
103 0956 1 FUNCTIONAL DESCRIPTION:
104 0957 1
105 0958 1 This routine isolates the next token in the command line, setting the variables
106 0959 1 EDT$$G_PA_TOKCLASS, EDT$$A_PA_CURTOK and EDT$$G_PA_CURTOKLEN.
107 0960 1 EDT$$C_PA_CH contains the first character following the last token, so
108 0961 1 while it is blank, we keep getting the next character. The previous token
109 0962 1 address and length are stored in EDT$$A_PA_PRVTOK and
110 0963 1 EDT$$G_PA_PRVTOKLEN. If the token is numeric then
111 0964 1 EDT$$L_PA_NUMVAL will get the numeric value of the token.
112 0965 1
113 0966 1 FORMAL PARAMETERS:
114 0967 1
115 0968 1 NONE
116 0969 1
117 0970 1 IMPLICIT INPUTS:
118 0971 1
119 0972 1 EDT$$A_CMD_BUF
120 0973 1 EDT$$A_CMD_END
121 0974 1 EDT$$C_PA_CH
122 0975 1 EDT$$L_LNO_ZERO
123 0976 1 EDT$$L_LNO
124 0977 1 EDT$$G_PA_NOQUO
125 0978 1
126 0979 1 IMPLICIT OUTPUTS:
127 0980 1
128 0981 1 EDT$$A_PA_PRVTOK
129 0982 1 EDT$$G_PA_PRVTOKLEN
130 0983 1 EDT$$A_PA_CURTOK
131 0984 1 EDT$$G_PA_CURTOKLEN
132 0985 1 EDT$$L_PA_NUMVAL
133 0986 1 EDT$$G_PA_PCEN
134 0987 1 EDT$$G_PA_TOKCLASS
135 0988 1 EDT$$A_CMD_END
136 0989 1
137 0990 1 ROUTINE VALUE:
138 0991 1
139 0992 1 NONE
140 0993 1
141 0994 1 SIDE EFFECTS:
142 0995 1
143 0996 1 NONE
144 0997 1
145 0998 1 --
146 0999 1
147 1000 2 BEGIN
148 1001 2
149 1002 2 EXTERNAL ROUTINE
150 1003 2 EDT$$PA_GETCH : NOVALUE; ! Get the next character from the input line
151 1004 2
152 1005 2 EXTERNAL
153 1006 2 EDT$$G_DEFKEY,
```



```
154      1007      2      EDT$SA_CMD_BUF,      | Pointer into command buffer.
155      1008      2      EDT$SA_CMD_END,      | Pointer to end of current command.
156      1009      2      EDT$SC_PA_CH,      | the currently being processed character
157      1010      2      EDT$SA_PA_CURTOK,      | start of the current token
158      1011      2      EDT$SG_PA_CURTOKLEN,      | length of current token
159      1012      2      EDT$SL_PA_NUMVAL : LIN_BLOCK,      | the value of a numeric literal
160      1013      2      EDT$SG_PA_PCEN,      | Did the keyword contain a percent?
161      1014      2      EDT$SA_PA_PRVTOK,      | Previous token address
162      1015      2      EDT$SG_PA_PRVTOKLEN,      | Previous token length
163      1016      2      EDT$SG_PA_TOKCLASS,      | class of current token
164      1017      2      EDT$SG_PA_NOQUO,      | 1 = don't accept ' and " as starting a string
165      1018      2
166      L 1019      2      %IF SUPPORT_VT220
167      1020      2      %THEN
168      1021      2      EDT$SB_CHAR_INFO : BLOCKVECTOR [256, 1, BYTE], ! Information about characters
169      1022      2      %FI
170      1023      2
171      1024      2      EDT$SL_LNO_ZERO : LN_BLOCK,
172      1025      2      EDT$SL_LNOO : LNOVECTOR [14];
173      1026      2
174      1027      2      !+
175      1028      2      ! Save off the address and length of the last token.
176      1029      2      !-
177      1030      2      EDT$SA_PA_PRVTOK = .EDT$SA_PA_CURTOK;
178      1031      2      EDT$SG_PA_PRVTOKLEN = .EDT$SG_PA_CURTOKLEN;
179      1032      2      !+
180      1033      2      ! First scan until the first non-blank character.
181      1034      2      !-
182      1035      2
183      1036      2      UNTIL ((.EDT$SC_PA_CH NEQ %C' ') AND (.EDT$SC_PA_CH NEQ ASC_K_TAB)) DO
184      1037      2      EDT$SPA_GETCH ?);
185      1038      2
186      1039      2      EDT$SA_PA_CURTOK = .EDT$SA_CMD_BUF - 1;
187      1040      2      EDT$SG_PA_TOKCLASS = CL_SPECIAL;
188      1041      2      EDT$SG_PA_PCEN = 0;
189      1042      2
190      1043      2      SELECTONE .EDT$SC_PA_CH OF
191      1044      2      SET
192      1045      2
193      1046      2      [%C'0' TO %C'9'] :
194      1047      2      BEGIN
195      1048      2
196      1049      2      LOCAL
197      1050      2      DIGIT : LN_BLOCK;
198      1051      2
199      1052      2      EDT$SG_PA_TOKCLASS = CL_NUMBER;
200      1053      2      !+
201      1054      2      ! Accumulate the number as though it might a line number. This means that we
202      1055      2      ! must do 48-bit arithmetic.
203      1056      2      !-
204      1057      2      MOVELINE (EDT$SL_LNO_ZERO, EDT$SL_PA_NUMVAL);
205      1058      2
206      1059      2      DO
207      1060      2      BEGIN
208      1061      2      MULTLINE (EDT$SL_LNOO [1], EDT$SL_PA_NUMVAL, EDT$SL_PA_NUMVAL);
209      1062      2      BUILDLINE (.EDT$SC_PA_CH - %C'0', DIGIT);
210      1063      2      ADDLINE (DIGIT, EDT$SC_PA_NUMVAL);
```

```

211      1064 4      EDT$PA_GETCH ();
212      1065 4      END
213      1066 4      UNTIL (
214      1067 4
215      L 1068 4 %IF SUPPORT_VT220
216      1069 4 %THEN
217      1070 5      (.EDT$B_CHAR_INFO [.EDT$C_PA_CH, 0, 0, 8, 0] NEQ %X'F0')
218      U 1071 5 %ELSE
219      U 1072 5      ((.EDT$C_PA_CH LSS %C'0') OR (.EDT$C_PA_CH GTR %C'9'))
220      1073 5 %FI
221      1074 5
222      1075 5      );
223      1076 5      ! Not a digit
224      1077 5
225      1078 2      END;
226      1079 2      [%C'A' TO %C'Z', %C'X', %X'CO' TO %X'CF', %X'D1' TO %X'DD'] :
227      1080 2      BEGIN
228      1081 2      EDT$G_PA_TOKCLASS = CL_NAME;
229      1082 3      !+
230      1083 3      !- Accept an optional percent sign.
231      1084 3      !-
232      1085 3
233      1086 4      IF (.EDT$C_PA_CH EQL %C'X')
234      1087 3      THEN
235      1088 4      BEGIN
236      1089 4      EDT$G_PA_PCEN = 1;
237      1090 4      EDT$A_PA_CURTOK = .EDT$A_CMD_BUF;
238      1091 4      EDT$PA_GETCH ();
239      1092 3      END;
240      1093 3
241      1094 3      !+
242      1095 3      !- Accept alphabetic characters and underscores.
243      1096 3      !-
244      1097 3
245      1098 3      DO
246      1099 4      BEGIN
247      1100 4      CH$WCHAR (.EDT$C_PA_CH, CH$PLUS (.EDT$A_CMD_BUF, -1));
248      1101 4      EDT$PA_GETCH ();
249      1102 4      END
250      1103 4      UNTIL (
251      1104 4
252      L 1105 4 %IF SUPPORT_VT220
253      1106 4 %THEN
254      1107 5      (.EDT$B_CHAR_INFO [.EDT$C_PA_CH, 0, 0, 2, 0] EQL 0) ! Not alphabetic
255      1108 5 %ELSE
256      U 1109 5      ( NOT (((.EDT$C_PA_CH GEQ %C'A') AND (.EDT$C_PA_CH LEQ %C'Z')) OR ((.EDT$C_PA_CH GEQ %C'a
257      U 1110 5      AND (.EDT$C_PA_CH LEQ %C'z'))))
258      1111 5 %FI
259      1112 5
260      1113 4      AND (.EDT$C_PA_CH NEQ %C'_'))
261      1114 4
262      1115 2      END;
263      1116 2
264      1117 2      [%C''''', %C'''''] :
265      1118 3      ! Start of a quoted string
266      1119 3      BEGIN
267      1120 3      LOCAL
```



```
268      1121 3      QUOTE;
269      1122 3
270      1123 3
271      1124 3      + Only scan the string if we are permitted to do so. In the SUBSTITUTE command we may not
272      1125 3      scan a string since SUBSTITUTE has its own, very special, syntax for its two strings.
273      1126 3      -
274      1127 3
275      1128 4      IF (.EDT$G_PA_NOQUO)
276      1129 3      THEN
277      1130 4      BEGIN
278      1131 4      EDT$PA_GETCH ();
279      1132 4      END
280      1133 3      ELSE
281      1134 3
282      1135 3      IF .EDT$G_DEFKEY
283      1136 3      THEN
284      1137 4      BEGIN
285      1138 4      EDT$G_PA_TOKCLASS = CL NAME;
286      1139 4      QUOTE = CH$RCHAR (.EDT$A_PA_CURTOK);
287      1140 4      EDT$A_PA_CURTOK = .EDT$A_PA_CURTOK + 1;
288      1141 4
289      1142 4      DO
290      1143 5      BEGIN
291      1144 5
292      1145 5      LOCAL
293      1146 5      CHAR;
294      1147 5
295      1148 5      IF (.EDT$A_CMD_BUF EQLA .EDT$A_CMD_END) THEN EXITLOOP;
296      1149 5
297      1150 5      EDT$PA_GETCH ();
298      1151 5      CHAR = CH$RCHAR (.EDT$A_PA_CURTOK);
299      1152 5
300      L 1153 5      %IF SUPPORT_VT220
301      1154 5      %THEN
302      1155 5
303      1156 5      IF .EDT$B_CHAR_INFO [.CHAR, 0, 0, 1, 0]      ! Lower case
304      1157 5      THEN
305      1158 5      %ELSE
306      1159 5
307      1160 5      IF ((.CHAR GEQ %C'a') AND (.CHAR LEQ %C'z'))
308      1161 5      THEN
309      1162 5      %FI
310      1163 5
311      1164 6      BEGIN
312      1165 6      CHAR = .CHAR - %C'a' + %C'A';
313      1166 6      CH$WCHAR (.CHAR, .EDT$A_PA_CURTOK);
314      1167 6      END;
315      1168 5
316      1169 5      END
317      1170 4      UNTIL (.EDT$C_PA_CH EQL .QUOTE);
318      1171 4
319      1172 4      EDT$PA_GETCH ();
320      1173 4      EDT$G_PA_CURTOKLEN = CH$DIFF (.EDT$A_CMD_BUF, .EDT$A_PA_CURTOK) - 2;
321      1174 4      RETURN;
322      1175 4      END
323      1176 3      ELSE
324      1177 4      BEGIN
```

EDT\$PRGETTOK
V04-000

EDT\$PRGETTOK - scan a token
EDT\$SPA_SCANTOK - scan the next token

B 16
16-Sep-1984 01:17:40
14-Sep-1984 12:24:09

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]PRGETTOK.BLI;1 (3) Page 8

```

: 325      1178  4      EDT$G_PA_TOKCLASS = CL STRING;
: 326      1179  4      QUOTE = CH$RCHAR (.EDT$A_PA_CURTOK);
: 327      1180  5      BEGIN
: 328      1181  5
: 329      1182  5      WHILE CH$PTR_NEQ (.EDT$A_CMD_BUF, .EDT$A_CMD_END) DO
: 330      1183  6      BEGIN
: 331      1184  6
: 332      1185  7      IF (CH$RCHAR (.EDT$A_CMD_BUF) EQL .QUOTE)
: 333      1186  6      THEN
: 334      1187  7      BEGIN
: 335      1188  7
: 336      1189  7      IF (CH$RCHAR (CH$PTR (.EDT$A_CMD_BUF, 1)) NEQ .QUOTE) THEN EXITLOOP;
: 337      1190  7
: 338      1191  7      EDT$COPY_MEM (CH$DIFF (.EDT$A_CMD_END, .EDT$A_CMD_BUF),
: 339      1192  7      CH$PTR (.EDT$A_CMD_BUF, 1), .EDT$A_CMD_BUF);
: 340      1193  7      EDT$A_CMD_END = CH$PLUS (.EDT$A_CMD_END, -1);
: 341      1194  6      END;
: 342      1195  6
: 343      1196  6      EDT$A_CMD_BUF = CH$PLUS (.EDT$A_CMD_BUF, 1);
: 344      1197  5      END;
: 345      1198  5
: 346      1199  5      EDT$G_PA_CURTOKLEN = CH$DIFF (.EDT$A_CMD_BUF, .EDT$A_PA_CURTOK) - 1;
: 347      1200  5      EDT$PA_GETCH ();
: 348      1201  4      END;
: 349      1202  4      EDT$PA_GETCH ();
: 350      1203  4      RETURN;
: 351      1204  3      END;
: 352      1205  3
: 353      1206  2      END;
: 354      1207  2
: 355      1208  2      [%C'!'] :
: 356      1209  2      ;
: 357      1210  2
: 358      1211  2      [OTHERWISE] :
: 359      1212  2      EDT$PA_GETCH ();
: 360      1213  2      TES;
: 361      1214  2
: 362      1215  2      EDT$G_PA_CURTOKLEN = CH$DIFF (.EDT$A_CMD_BUF, .EDT$A_PA_CURTOK) - 1;
: 363      1216  1      END;
                                ! of routine EDT$PA_SCANTOK
```

.TITLE EDT\$PRGETTOK EDT\$PRGETTOK - scan a token
.IDENT \V04-000\

.EXTRN EDT\$PA_GETCH, EDT\$G_DEFKEY
.EXTRN EDT\$A_CMD_BUF, EDT\$A_CMD_END
.EXTRN EDT\$C_PA_CH, EDT\$A_PA_CURTOK
.EXTRN EDT\$G_PA_CURTOKLEN
.EXTRN EDT\$SL_PA_NUMVAL
.EXTRN EDT\$G_PA_PCEN
.EXTRN EDT\$A_PA_PRVTOK
.EXTRN EDT\$G_PA_PRVTOKLEN
.EXTRN EDT\$G_PA_TOKCLASS
.EXTRN EDT\$G_PA_NOQUO
.EXTRN EDT\$B_CHAR_INFO
.EXTRN EDT\$SL_LNO_ZERO
.EXTRN EDT\$SL_LNOO

				.PSECT	_EDT\$CODE,NOWRT, SHR, PIC,2	
				.ENTRY	EDT\$PA_SCANTOK, Save R2,R3,R4,R5,R6,R7,R8,-;	0952
			OFFC 00000		R9,R10,R11	
	5B	00000000G	00 9E 00002	MOVAB	EDT\$PA_CURTOK, R11	
	5A	00000000G	00 9E 00009	MOVAB	EDT\$PA_CMD_BUF, R10	
	59	00000000G	00 9E 00010	MOVAB	EDT\$PA_NUMVAL, R9	
	58	00000000G	00 9E 00017	MOVAB	EDT\$PA_GETCH, R8	
	5E		18 C2 0001E	SUBL2	#24, SP	
00000000G	00		6B D0 00021	MOVL	EDT\$PA_CURTOK, EDT\$PA_PRVTOK	1030
00000000G	00	00000000G	00 D0 00028	MOVL	EDT\$G_PA_CURTOKLEN, EDT\$G_PA_PRVTOKLEN	1031
	50	00000000G	00 D0 00033 1\$:	MOVL	EDT\$C_PA_CH, R0	1036
	20		50 D1 0003A	CMPL	R0, #32	
			05 13 0003D	BEQL	2\$	
	09		50 D1 0003F	CMPL	R0, #9	
			05 12 00042	BNEQ	3\$	
	68		00 FB 00044 2\$:	CALLS	#0, EDT\$PA_GETCH	1037
			EA 11 00047	BRB	1\$	
	57		6A D0 00049 3\$:	MOVL	EDT\$PA_CMD_BUF, R7	1039
	6B	FF	A7 9E 0004C	MOVAB	-1(R7), EDT\$PA_CURTOK	
00000000G	00		02 D0 00050	MOVL	#2, EDT\$G_PA_TOKCLASS	1040
		00000000G	00 D4 00057	CLRL	EDT\$G_PA_PCEN	1041
	56	00000000G	00 D0 0005D	MOVL	EDT\$C_PA_CH, R6	1043
	30		56 D1 00064	CMPL	R6, #48	1046
			79 19 00067	BLSS	7\$	
	39		56 D1 00069	CMPL	R6, #57	
			74 14 0006C	BGTR	7\$	
00000000G	00		01 D0 0006E	MOVL	#1, EDT\$G_PA_TOKCLASS	1052
69 00000000G	00		06 28 00075	MOVC3	#6, EDT\$SL_LNO_ZERO, EDT\$SL_PA_NUMVAL	1057
	52	00000000G	00 D0 0007D	MOVL	EDT\$C_PA_CH, R2	1062
	08	AE	69 D0 00084 4\$:	MOVL	EDT\$SL_PA_NUMVAL, M2	1061
	0C	AE	A9 3C 00088	MOVZWL	EDT\$SL_PA_NUMVAL+4, M2+4	
		04	6E 7C 0008D	CLRQ	P	
	50		10 D0 0008F	MOVL	#16, I	
6E	6E		01 79 00092 5\$:	ASHQ	#1, P, P	
09 00000000G	00		50 E1 00096	BBC	I, M1, 6\$	
	6E	08	AE C0 0009E	ADDL2	M2, P	
	04	AE	AE D8 000A2	ADWC	M2, P	
	E8		50 F4 000A7 6\$:	SOBGEQ	I, 5\$	
	69		6E D0 000AA	MOVL	P, EDT\$SL_PA_NUMVAL	
	04	A9	AE B0 000AD	MOVW	P+4, EDT\$SL_PA_NUMVAL+4	
	10	AE	A2 9E 000B2	MOVAB	-48(R2), DIGIT	1062
		14	AE D4 000B7	CLRL	DIGIT+4	
	50	06	A9 B0 000BA	MOVW	UPPER_WORD, SAVE	1063
	69	10	AE C0 000BE	ADDL2	DIGIT, EDT\$SL_PA_NUMVAL	
	04	A9	AE D8 000C2	ADWC	DIGIT, EDT\$SL_PA_NUMVAL+4	
	06	A9	50 B0 000C7	MOVW	SAVE, UPPER_WORD	
	68		00 FB 000CB	CALLS	#0, EDT\$PA_GETCH	1064
	52	00000000G	00 D0 000CE	MOVL	EDT\$C_PA_CH, R2	1070
F0	8F	00000000G	00 42 91 000D5	CMPL	EDT\$B_CHAR_INFO[R2], #240	
			A4 13 000DE	BEQL	4\$	
			7E 11 000E0	BRB	13\$	
	25		56 D1 000E2 7\$:	CMPL	R6, #37	1043
			36 13 000E5	BEQL	10\$	1079
00000041	8F		56 D1 000E7	CMPL	R6, #65	
			09 19 000EE	BLSS	8\$	

0000005A	8F	56	D1	000F0	CMPL	R6, #90	:
		24	15	000F7	BLEQ	10\$	:
000000C0	8F	56	D1	000F9	CMPL	R6, #192	:
		09	19	00100	BLSS	9\$	:
000000CF	8F	56	D1	00102	CMPL	R6, #207	:
		12	15	00109	BLEQ	10\$	:
000000D1	8F	56	D1	0010B	CMPL	R6, #209	:
		4F	19	00112	BLSS	14\$	:
000000DD	8F	56	D1	00114	CMPL	R6, #221	:
		46	14	0011B	BGTR	14\$	:
	00000000G	00	D4	0011D	CLRL	EDT\$\$G_PA_TOKCLASS	1081
	25	56	D1	00123	CMPL	R6, #37	1086
		0D	12	00126	BNEQ	11\$	:
00000000G	00	01	D0	00128	MOVL	#1, EDT\$\$G_PA_PCEN	1089
	6B	57	D0	0012F	MOVL	R7, EDT\$\$A_PA_CURTOK	1090
	68	00	FB	00132	CALLS	#0, EDT\$\$PA_GETCH	1091
	52	00	D0	00135	MOVL	EDT\$\$C_PA_CH, R2	1100
	50	6A	D0	0013C	MOVL	EDT\$\$A_CMD_BUF, R0	:
FF	A0	52	90	0013F	MOVB	R2, -1(R0)	:
	68	00	FB	00143	CALLS	#0, EDT\$\$PA_GETCH	1101
	52	00	D0	00146	MOVL	EDT\$\$C_PA_CH, R2	1107
	03	00	D0	0014D	BITB	EDT\$\$B_CHAR_INFO[R2], #3	:
		E5	12	00155	BNEQ	12\$	:
0000005F	8F	52	D1	00157	CMPL	R2, #95	1113
		DC	13	0015E	BEQL	12\$	:
		00BB	31	00160	BRW	26\$	1098
	22	56	D1	00163	CMPL	R6, #34	1117
		08	13	00166	BEQL	15\$	:
	27	56	D1	00168	CMPL	R6, #39	:
		03	13	0016B	BEQL	15\$	:
		00A6	31	0016D	BRW	24\$	:
	03	00	E9	00170	BLBC	EDT\$\$G_PA_NOQUO, 16\$	1128
		00A1	31	00177	BRW	25\$	:
	57	6B	D0	0017A	MOVL	EDT\$\$A_PA_CURTOK, R7	1139
	45	00	E9	0017D	BLBC	EDT\$\$G_DEFKEY, 20\$	1135
		00	D4	00184	CLRL	EDT\$\$G_PA_TOKCLASS	1138
	56	67	9A	0018A	MOVZBL	(R7), QUOTE	1139
		6B	D6	0018D	INCL	EDT\$\$A_PA_CURTOK	1140
00000000G	00	6A	D1	0018F	CMPL	EDT\$\$A_CMD_BUF, EDT\$\$A_CMD_END	1148
		21	13	00196	BEQL	19\$	:
	68	00	FB	00198	CALLS	#0, EDT\$\$PA_GETCH	1150
	51	6B	D0	0019B	MOVL	EDT\$\$A_PA_CURTOK, R1	1151
	50	61	9A	0019E	MOVZBL	(R1), CHAR	:
06	00000000G	00	E1	001A1	BBC	#0, EDT\$\$B_CHAR_INFO[CHAR], 18\$	1156
		20	C2	001AA	SUBL2	#32, CHAR	1165
		50	90	001AD	MOVB	CHAR, (R1)	1166
	56	00	D1	001B0	CMPL	EDT\$\$C_PA_CH, QUOTE	1170
		D6	12	001B7	BNEQ	17\$	:
	68	00	FB	001B9	CALLS	#0, EDT\$\$PA_GETCH	1172
50	6A	6B	C3	001BC	SUBL3	EDT\$\$A_PA_CURTOK, EDT\$\$A_CMD_BUF, R0	1173
	00000000G	00	9E	001C0	MOVAB	-2(R0), EDT\$\$G_PA_CURTOK[EN	:
		04	00	001C8	RET		1177
	00000000G	00	D0	001C9	MOVL	#3, EDT\$\$G_PA_TOKCLASS	1178
		67	9A	001D0	MOVZBL	(R7), QUOTE	1179
	50	6A	D0	001D3	MOVL	EDT\$\$A_CMD_BUF, R0	1182
	51	00	D0	001D6	MOVL	EDT\$\$A_CMD_END, R1	:
	51	50	D1	001DD	CMPL	R0, R1	:

EDT\$PRGETTOK
V04-000

EDT\$PRGETTOK - scan a token
EDT\$\$PA_SCANTOK - scan the next token

E 16
16-Sep-1984 01:17:40
14-Sep-1984 12:24:09

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]PRGETTOK.BLI;1
Page 11
(3)

56	60	08	21	13	001E0	BEQL	23\$	:	1185
			00	ED	001E2	CMPZV	#0, #8, (R0), QUOTE	:	
56	01	A0	16	12	001E7	BNEQ	22\$	:	1189
			00	ED	001E9	CMPZV	#0, #8, 1(R0), QUOTE	:	
			12	12	001EF	BNEQ	23\$	:	1192
		51	50	C2	001F1	SUBL2	R0, R1	:	
	60	01	51	28	001F4	MOVCL	R1, 1(R0), (R0)	:	
			00	D7	001F9	DECL	EDT\$\$A_CMD_END	:	1193
			6A	D6	001FF	INCL	EDT\$\$A_CMD_BUF	:	1196
			D0	11	00201	BRB	21\$	:	1182
	50		57	C3	00203	SUBL3	R7, EDT\$\$A_CMD_BUF, R0	:	1199
		6A	A0	9E	00207	MOVAB	-1(R0), EDT\$\$G_PA_CURTOKLEN	:	
	00000000G	00	00	FB	0020F	CALLS	#0, EDT\$\$PA_GETCH	:	1200
		68	00	FB	00212	CALLS	#0, EDT\$\$PA_GETCH	:	1202
		68	00	FB	00212	CALLS	#0, EDT\$\$PA_GETCH	:	1177
			04	00215	RET			:	1208
		21	56	D1	00216	CMPL	R6, #33	:	
			03	13	00219	BEQL	26\$	:	
		68	00	FB	0021B	CALLS	#0, EDT\$\$PA_GETCH	:	1212
	50		6B	C3	0021E	SUBL3	EDT\$\$A_PA_CURTOK, EDT\$\$A_CMD_BUF, R0	:	1215
		6A	A0	9E	00222	MOVAB	-1(R0), EDT\$\$G_PA_CURTOKLEN	:	
	00000000G	00	04	0022A	RET			:	1216

; Routine Size: 555 bytes, Routine Base: _EDT\$CODE + 0000

: 364 1217 1
: 365 1218 1 !<BLF/PAGE>

EDT\$PRGETTOK
V04-000

EDT\$PRGETTOK - scan a token
EDT\$PA_SCANTOK - scan the next token

F 16
16-Sep-1984 01:17:40
14-Sep-1984 12:24:09

VAX-11 Bliss-32 V4.0-742
DISK\$VM\$MASTER:[EDT.SRC]PRGETTOK.BLI;1 (4)

: 367 1219 1 END
: 368 1220 1
: 369 1221 0 ELUDOM

! of module EDT\$PRGETTOK

PSECT SUMMARY

Name	Bytes	Attributes
_EDT\$CODE	555	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[EDT.SRC]EDT.L32;1	377	24	6	40	00:00.2
_\$255\$DUA28:[EDT.SRC]PSECTS.L32;1	2	1	50	7	00:00.1
_\$255\$DUA28:[EDT.SRC]SUPPORTS.L32;1	2	1	50	5	00:00.1

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACEBACK/LIS=LIS\$:PRGETTOK/OBJ=OBJ\$:PRGETTOK MSRC\$:PRGETTOK.BLI/UPDATE=(ENH\$:PRGETTOK)

: Size: 555 code + 0 data bytes
: Run Time: 00:27.0
: Elapsed Time: 00:33.7
: Lines/CPU Min: 2709
: Lexemes/CPU-Min: 11105
: Memory Used: 193 pages
: Compilation Complete

0137

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY