


```

PPPPPPPP  RRRRRRRR  SSSSSSSS  EEEEEEEEE  MM      MM  RRRRRRRR  TTTTTTTTTT  NN      NN
PPPPPPPP  RRRRRRRR  SSSSSSSS  EEEEEEEEE  MM      MM  RRRRRRRR  TTTTTTTTTT  NN      NN
PP      PP  RR      RR  SS      SS      SS      SS      RR      RR  TT      TT      NN      NN
PP      PP  RR      RR  SS      SS      SS      SS      RR      RR  TT      TT      NN      NN
PP      PP  RR      RR  SS      SS      SS      SS      RR      RR  TT      TT      NN      NN
PPPPPPPP  RRRRRRRR  SSSSSSSS  EEEEEEEEE  MM      MM  RRRRRRRR  TT      TT      NN      NN
PPPPPPPP  RRRRRRRR  SSSSSSSS  EEEEEEEEE  MM      MM  RRRRRRRR  TT      TT      NN      NN
PP      RR  RR      SS      SS      SS      SS      RR      RR  TT      TT      NN      NN
PP      RR  RR      SS      SS      SS      SS      RR      RR  TT      TT      NN      NN
PP      RR      RR  SS      SS      SS      SS      RR      RR  TT      TT      NN      NN
PP      RR      RR  SS      SS      SS      SS      RR      RR  TT      TT      NN      NN
PP      RR      RR  SS      SS      SS      SS      RR      RR  TT      TT      NN      NN
PP      RR      RR  SS      SS      SS      SS      RR      RR  TT      TT      NN      NN

```

```

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 %TITLE 'EDT$PRSEMRTN - parser semantic actions'
2 0002 0 MODULE EDT$PRSEMRTN ( ! Parser semantic actions
3 0003 0 IDENT = 'V04-000' ! File: PRSEMRTN.BLI Edit: JBS1023
4 0004 0 ) =
5 0005 1 BEGIN
6 0006 1
7 0007 1 *****
8 0008 1 *
9 0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
10 0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
11 0011 1 * ALL RIGHTS RESERVED.
12 0012 1 *
13 0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
14 0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
15 0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
16 0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
17 0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
18 0018 1 * TRANSFERRED.
19 0019 1 *
20 0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
21 0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
22 0022 1 * CORPORATION.
23 0023 1 *
24 0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
25 0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1
31 0031 1 **
32 0032 1 FACILITY: EDT -- The DEC Standard Editor
33 0033 1
34 0034 1 ABSTRACT:
35 0035 1
36 0036 1 Parser semantic actions.
37 0037 1
38 0038 1 ENVIRONMENT: Runs at any access mode - AST reentrant
39 0039 1
40 0040 1 AUTHOR: Bob Kushlis, CREATION DATE: December 12, 1978
41 0041 1
42 0042 1 MODIFIED BY:
43 0043 1
44 0044 1 1-001 - Original. DJS 25-Feb-1981. This module was created by
45 0045 1 extracting routine SEM_ROUTINES from module PARSER.
46 0046 1 1-002 - Regularize headers. JBS 12-Mar-1981
47 0047 1 1-003 - Use the ASSERT macro. JBS 01-Jun-1981
48 0048 1 1-004 - Use the new message codes. JBS 05-Aug-1981
49 0049 1 1-005 - In a substitute command, don't allow the scanner to swallow
50 0050 1 a quoted string after the command, since SUBSTITUTE has its
51 0051 1 own syntax for its two strings. JBS 26-Aug-1981
52 0052 1 1-006 - Add PREV_RANGE, the back pointer for NEXT_RANGE. JBS 02-Nov-1981
53 0053 1 1-007 - Don't scan too far if the SUBSTITUTE command is ill-formed. JBS 28-Dec-1981
54 0054 1 1-008 - Make the NEXT command have the same fix from edit 1-005 as the SUBSTITUTE
55 0055 1 NEXT command. JBS 04-Jan-1982
56 0056 1 1-009 - Change index for line numbers from 10 digits to 15. SMB 18-Jan-1982
57 0057 1 1-010 - Add error checks for line numbers out of range. SMB 06-Feb-1982

```

```
58 0058 1 | 1-011 - Correct the file name scanner so it doesn't loop on an unquoted string. JBS 10-Feb-1982
59 0059 1 | 1-012 - Don't let a key number be larger than 21. JBS 10-Feb-1982
60 0060 1 | 1-013 - Add a missing dot in edit 1-011. JBS 13-Feb-1982
61 0061 1 | 1-014 - Fix bad range check from edit 1-011. SMB 15-Feb-1982
62 0062 1 | 1-015 - Change range check and error code (part of 1-011 problem). SMB 16-Feb-1982
63 0063 1 | 1-016 - Set define key flag so we can accept quoted key. STS 07-Apr-1982
64 0064 1 | 1-017 - Delete reference to edt$$g pa keyval. STS 09-Apr-1982
65 0065 1 | 1-018 - Make TAB COUNT signed. JBS 21-Apr-1982
66 0066 1 | 1-019 - Change alphanumeric test. JBS 19-Jul-1982
67 0067 1 | 1-020 - New implementation of defined keys. JBS 13-Aug-1982
68 0068 1 | 1-021 - modify to use new 48 bit arith macro. STS 01-Oct-1982
69 0069 1 | 1-022 - Modify to use new compare macro. STS 20-Oct-1982
70 0070 1 | 1-023 - Add VT220 support conditional. JBS 11-Feb-1983
71 0071 1 | --
72 0072 1 |
```

```
74 0073 1 %SBTTL 'Declarations'  
75 0074 1  
76 0075 1 : TABLE OF CONTENTS:  
77 0076 1 :  
78 0077 1  
79 0078 1 REQUIRE 'EDT$SRC:TRAROUNAM';  
80 0517 1  
81 0518 1 FORWARD ROUTINE  
82 0519 1 EDT$SPA_SEMRUT;  
83 0520 1  
84 0521 1 :  
85 0522 1 : INCLUDE FILES:  
86 0523 1 :  
87 0524 1  
88 0525 1 REQUIRE 'EDT$SRC:EDTREQ';  
89 0660 1  
90 0661 1 REQUIRE 'EDT$SRC:PARLITS';  
91 0945 1  
92 0946 1 LIBRARY 'EDT$SRC:KEYPADDEF';  
93 0947 1  
94 0948 1 LIBRARY 'EDT$SRC:SUPPORTS';  
95 0949 1  
96 0950 1 :  
97 0951 1 : MACROS:  
98 0952 1 :  
99 0953 1 : NONE  
100 0954 1 :  
101 0955 1 : EQUATED SYMBOLS:  
102 0956 1 :  
103 0957 1 : NONE  
104 0958 1 :  
105 0959 1 : OWN STORAGE:  
106 0960 1 :  
107 0961 1 : NONE  
108 0962 1 :  
109 0963 1 : EXTERNAL REFERENCES:  
110 0964 1 :  
111 0965 1 : In the routine
```

```

113 0966 1 %SBTTL 'EDT$$PA_SEMRUT - parser semantic actions'
114 0967 1
115 0968 1 GLOBAL ROUTINE EDT$$PA_SEMRUT (           ! Parser semantic actions
116 0969 1     WHICH,                               ! Action number to perform
117 0970 1     OPERAND                             ! Token
118 0971 1     ) =
119 0972 1
120 0973 1
121 0974 1 ++
122 0975 1 FUNCTIONAL DESCRIPTION:
123 0976 1     The semantic actions for the parser. Which specifies which of the
124 0977 1     actions to perform. Operand is the index of the token which matched
125 0978 1     if the semantic routine was called as a result of a select operator.
126 0979 1
127 0980 1 FORMAL PARAMETERS:
128 0981 1
129 0982 1     WHICH             Action number to perform
130 0983 1
131 0984 1     OPERAND           Token which matched
132 0985 1
133 0986 1 IMPLICIT INPUTS:
134 0987 1
135 0988 1     EDT$$A_CMD_END
136 0989 1     EDT$$C_PA_CH
137 0990 1     EDT$$G_PA_CURCMD
138 0991 1     EDT$$A_PA_CURTOK
139 0992 1     EDT$$G_PA_CURTOKLEN
140 0993 1     EDT$$L_PA_NUMVAL
141 0994 1     EDT$$G_PA_PCEN
142 0995 1     EDT$$A_PA_PRVTOK
143 0996 1     EDT$$G_PA_PRVTOKLEN
144 0997 1     EDT$$G_PA_SP
145 0998 1     EDT$$Z_PA_THRURNG
146 0999 1     EDT$$G_PA_TOKCLASS
147 1000 1     EDT$$L_LN00
148 1001 1     EDT$$G_TAB_SIZ
149 1002 1
150 1003 1 IMPLICIT OUTPUTS:
151 1004 1
152 1005 1     EDT$$G_PA_CURCMD
153 1006 1     EDT$$G_PA_ERRNO
154 1007 1     EDT$$Z_PA_CURRNG
155 1008 1     EDT$$Z_PA_BUFRNG
156 1009 1     EDT$$Z_PA_ANDLSTHD
157 1010 1     EDT$$A_CMD_BUF
158 1011 1     EDT$$G_PA_NOQUO
159 1012 1
160 1013 1 ROUTINE VALUE:
161 1014 1
162 1015 1     0 = failure, 1 = success
163 1016 1
164 1017 1 SIDE EFFECTS:
165 1018 1
166 1019 1     MANY
167 1020 1
168 1021 1 --
169 1022 1

```

```
170 1023 2 BEGIN
171 1024 2
172 1025 2 EXTERNAL ROUTINE
173 1026 2 EDT$$PA_SCANTOK : NOVALUE, ! Get the next token
174 1027 2 EDT$$PA_APPDIG,
175 1028 2 EDT$$PA_GETCH : NOVALUE, ! Get the next character from the input line
176 1029 2 EDT$$PA_CRERNGNOD, ! Create a range node
177 1030 2 EDT$$PA_NEW_NOD; ! Create a semantic node
178 1031 2
179 1032 2 EXTERNAL
180 1033 2 EDT$$L_MAX_LINES, ! maximum line number value
181 1034 2 EDT$$A_CMD_BUF, ! Pointer into command buffer.
182 1035 2 EDT$$A_CMD_END, ! Pointer to end of current command.
183 1036 2 EDT$$Z_PA_ANDLSTHD : REF NODE_BLOCK,
184 1037 2 EDT$$Z_PA_BUFRNG : REF NODE_BLOCK,
185 1038 2 EDT$$C_PA_CH, ! the currently being processed character
186 1039 2 EDT$$G_PA_CURCMD : REF NODE_BLOCK,
187 1040 2 EDT$$Z_PA_CURRNG : REF NODE_BLOCK, ! the current range node
188 1041 2 EDT$$A_PA_CURTOK, ! start of the current token
189 1042 2 EDT$$G_DEFKEY,
190 1043 2 EDT$$G_PA_CURTOKLEN, ! length of current token
191 1044 2 EDT$$G_PA_ERRNO, ! Error number of parsing error.
192 1045 2 EDT$$L_PA_NUMVAL : LN_BLOCK, ! the value of a numeric literal
193 1046 2 EDT$$G_PA_PCEN, ! Did the keyword contain a percent?
194 1047 2 EDT$$A_PA_PRVTOK, ! Previous token address
195 1048 2 EDT$$G_PA_PRVTOKLEN, ! Previous token length
196 1049 2 EDT$$G_PA_SP,
197 1050 2 EDT$$Z_PA_THRURNG : REF NODE_BLOCK, ! The currently being built thru type range
198 1051 2 EDT$$G_PA_TOKCLASS, ! class of current token
199 1052 2 EDT$$G_PA_NOQUO, ! Don't allow quoted strings in the scanner
200 1053 2 EDT$$L_LJ00 : LNOVECTOR [14],
201 1054 2
202 L 1055 2 %IF SUPPORT_VT220
203 1056 2 %THEN
204 1057 2 EDT$$B_CHAR_INFO : BLOCKVECTOR [256, 1, BYTE], ! Information about characters
205 1058 2 %FI
206 1059 2
207 1060 2 EDT$$G_TAB_SIZ; ! Current tab size, for error checking
208 1061 2
209 P 1062 2 MESSAGES ((INVBUFNAM, QUOSTREQ, NONALPNUM, SUBSTRNUL, UNRCOM, KEYNOTDEF, NUMVALREQ, INVPARFOR, INVVALSE
210 1063 2 ERRRANSPC, ERRCOMOPT, UNRCOMOPT, COLONREQ, MACKEYREQ, ENTMUSTBE, ASREQ, INVSTR, NUMVALILL));
211 1064 2 !
212 1065 2
213 1066 2 CASE .WHICH FROM 1 TO NUM_SEM OF
214 1067 2 SET
215 1068 2
216 1069 2 [INI_COM] : ! Initialize for a command
217 1070 2 BEGIN
218 1071 2 !+
219 1072 2 ! Make sure the last command turned off EDT$$G_PA_NOQUO , otherwise there may
220 1073 2 ! be subtle interactions of commands.
221 1074 2 !-
222 1075 2 ASSERT (.EDT$$G_PA_NOQUO EQL 0);
223 1076 2 EDT$$G_DEFKEY = 0;
224 1077 2
225 1078 2 IF (.EDT$$G_PA_CURCMD NEQ 0) THEN EDT$$G_PA_CURCMD [NEXT_COM] = .EDT$$G_PA_SP;
226 1079 2
```

```
227 1080      IF ((EDT$$G_PA_CURCMD = EDT$$PA_NEW_NOD (COM_NODE, .OPERAND)) EQL 0) THEN RETURN (0);
228 1081
229 1082
230 1083      + If this is the SUBSTITUTE or NEXT command, don't let the scanner take the next token as a quoted string.
231 1084      -
232 1085
233 1086      IF ((.OPERAND EQL 16) OR (.OPERAND EQL 19)) THEN EDT$$G_PA_NOQUO = 1;
234 1087
235 1088      END;
236 1089
237 1090      [INIRAN] :      ! Initialize for a range
238 1091      BEGIN
239 1092
240 1093      IF ((EDT$$Z_PA_CURRNG = EDT$$PA_NEW_NOD (RANGE_NODE, .OPERAND)) EQL 0) THEN RETURN (0);
241 1094
242 1095      IF (.EDT$$G_PA_TOKCLASS EQL CL_NUMBER)      !
243 1096      THEN
244 1097          MOVELINE (EDT$$L_PA_NUMVAL, EDT$$Z_PA_CURRNG [RAN_VAL]);
245 1098
246 1099      END;
247 1100
248 1101      [START_RANGE] :
249 1102      BEGIN
250 1103
251 1104      IF (.OPERAND NEQ 0)
252 1105      THEN
253 1106
254 1107          IF (EDT$$PA_SEMRUT (INIRAN, .OPERAND + NUM_SLR) EQL 0) THEN RETURN (0);
255 1108
256 1109      END;
257 1110
258 1111      [BUF_RAN] :
259 1112      BEGIN
260 1113          EDT$$G_PA_ERRNO = EDT$_INVBUFNAM;
261 1114
262 1115          IF ( NOT EDT$$PA_APPDIG ()) THEN RETURN (0);
263 1116
264 1117          IF (EDT$$PA_SEMRUT (INIRAN, RAN_BUFFER) EQL 0) THEN RETURN (0);
265 1118
266 1119          EDT$$Z_PA_CURRNG [BUF_NAME] = .EDT$$A_PA_CURTOK;
267 1120          EDT$$Z_PA_CURRNG [BUF_LEN] = .EDT$$G_PA_CURTOKLEN;
268 1121          EDT$$Z_PA_BUFRNG = .EDT$$Z_PA_CURRNG;
269 1122          EDT$$PA_SCANTOK ();
270 1123      END;
271 1124
272 1125      [APP_NUM] :      ! Append numerics to a name.
273 1126      EDT$$PA_APPDIG ();
274 1127
275 1128      [BUF_RAN2] :
276 1129      BEGIN
277 1130          EDT$$Z_PA_BUFRNG [RANGE1] = .EDT$$Z_PA_CURRNG;
278 1131          EDT$$Z_PA_CURRNG = .EDT$$Z_PA_BUFRNG;
279 1132      END;
280 1133
281 1134      [GETSTR] :
282 1135      BEGIN
283 1136          EDT$$Z_PA_CURRNG [RAN_VAL] = .EDT$$G_PA_PRVTOKLEN;
```



```
284      EDT$$Z_PA_CURRNG [STR_PNT] = .EDT$$A_PA_PRVTOK + 1;
285      1138
286      1139      IF (.EDT$$Z_PA_CURRNG [RAN_TYPE] EQL RAN_MINUS) THEN EDT$$Z_PA_CURRNG [RAN_TYPE] = RAN_MINSTR;
287      1140
288      1141      END;
289      1142
290      1143      [ALLRAN] :      ! ALL appended to a range
291      1144      BEGIN
292      1145
293      1146      LOCAL
294      1147      SUB_RAN : REF NODE_BLOCK;
295      1148
296      1149      SUB_RAN = .EDT$$Z_PA_CURRNG;      ! Save the first part of the range
297      1150
298      1151      IF ( NOT EDT$$PA_SEMRUT (INIRAN, RAN_ALL)) THEN RETURN (0);
299      1152
300      1153      !+
301      1154      !- Link the original range with the ALL clause.
302      1155
303      1156      EDT$$Z_PA_CURRNG [NEXT_RANGE] = .SUB_RAN;
304      1157      SUB_RAN [PREV_RANGE] = .EDT$$Z_PA_CURRNG;
305      1158
306      1159      IF (.EDT$$G_PA_TOKCLASS NEQ CL_STRING)
307      1160      THEN
308      1161      BEGIN
309      1162      EDT$$G_PA_ERRNO = EDT$_QUOSTRREQ;
310      1163      RETURN (0);
311      1164      END;
312      1165
313      1166      EDT$$Z_PA_CURRNG [RAN_VAL] = .EDT$$G_PA_CURTOKLEN;
314      1167      EDT$$Z_PA_CURRNG [STR_PNT] = .EDT$$A_PA_CURTOK + 1;
315      1168      EDT$$PA_SCANTOK ();
316      1169      END;
317      1170
318      1171      [RAN1] :
319      1172      EDT$$G_PA_CURCMD [RANGE1] = .EDT$$Z_PA_CURRNG;
320      1173
321      1174      [RAN2] :
322      1175      EDT$$G_PA_CURCMD [RANGE2] = .EDT$$Z_PA_CURRNG;
323      1176
324      1177      [PLUSRAN] :
325      1178
326      1179      IF (EDT$$PA_CRERNGNOD (RAN_PLUS) EQL 0) THEN RETURN (0);
327      1180
328      1181      [MINUSRAN] :
329      1182
330      1183      IF (EDT$$PA_CRERNGNOD (RAN_MINUS) EQL 0) THEN RETURN (0);
331      1184
332      1185      [FORRAN] :
333      1186
334      1187      IF (EDT$$PA_CRERNGNOD (RAN_FOR) EQL 0) THEN RETURN (0);
335      1188
336      1189      [RANNUM] :      ! value following FOR, +, ORIGINAL and -
337      1190      BEGIN
338      1191
339      1192      IF ((.EDT$$L_PA_NUMVAL [LN_MD] NEQ 0) OR (.EDT$$L_PA_NUMVAL [LN_HI] NEQ 0))
340      1193      THEN
```

```

341      1194  4          BEGIN
342      1195  4          EDT$$G_PA_ERRNO = EDT$_NUMVALILL;
343      1196  4          RETURN(0);
344      1197  3          END;
345      1198  3
346      1199  3          EDT$$Z_PA_CURRNG [RAN_VAL] = .EDT$$L_PA_NUMVAL [LN_LO];
347      1200  2          END;
348      1201  2
349      1202  2          [LINE NUM RANGE] :          ! Numeric range value
350      1203  2          MOVELINE (EDT$$L_PA_NUMVAL, EDT$$Z_PA_CURRNG [RAN_VAL]);
351      1204  2
352      1205  2          [LINE NUM] :          ! the line number
353      1206  2          BEGIN
354      1207  2
355      1208  2          LOCAL
356      1209  2          MULTIPLIER,
357      1210  2          DIGIT : LN_BLOCK;
358      1211  2
359      1212  2          !+
360      1213  2          ! If the line number coming in is greater than maximum allowed before
361      1214  2          ! multiplication by 10**5, then return error
362      1215  2          !-
363      1216  3
364      1217  4          IF (CMLNO (EDT$$L_PA_NUMVAL, EDT$$L_MAX_LINES) GTR 0)
365      1218  3          THEN
366      1219  4          BEGIN
367      1220  4          EDT$$G_PA_ERRNO = EDT$_NUMVALILL;
368      1221  4          RETURN(0);
369      1222  3          END;
370      1223  3
371      1224  3          MULTLINE (EDT$$L_LNOO [5], EDT$$L_PA_NUMVAL, EDT$$L_PA_NUMVAL);
372      1225  3
373      1226  4          IF (CH$RCHAR (.EDT$$A_PA_CURTOK) EQL %C'.')
374      1227  3          THEN
375      1228  4          BEGIN
376      1229  4          MULTIPLIER = 4;
377      1230  4
378      1231  4          %IF SUPPORT_VT220
379      1232  4          %THEN
380      1233  4
381      1234  4          WHILE (.EDT$$B_CHAR_INFO [.EDT$$C_PA_CH, 0, 0, 8, 0] EQL %X'F0') DO
382      1235  4          %ELSE
383      1236  4
384      1237  4          WHILE ((.EDT$$C_PA_CH GEQ %C'0') AND (.EDT$$C_PA_CH LEQ %C'9')) DO
385      1238  4          %FI
386      1239  4
387      1240  5          BEGIN
388      1241  5          BUILDLINE (.EDT$$C_PA_CH - %C'0', DIGIT);
389      1242  5
390      1243  6          IF (.MULTIPLIER GEQ 0)
391      1244  5          THEN
392      1245  6          BEGIN
393      1246  6          MULTLINE (EDT$$L_LNOO [.MULTIPLIER], DIGIT, DIGIT);
394      1247  6          ADDLINE (DIGIT, EDT$$L_PA_NUMVAL);
395      1248  5          END;
396      1249  5
397      1250  5          EDT$$PA_GETCH ();

```

```
398 1251 5          MULTIPLIER = .MULTIPLIER - 1;
399 1252 4          END;
400 1253 4
401 1254 4          EDT$$PA_SCANTOK ();
402 1255 4          END;
403 1256 4
404 1257 2          END;
405 1258 2
406 1259 2          [BIN_RANGE] :
407 1260 2          BEGIN
408 1261 2
409 1262 2          IF ((EDT$$Z_PA_THRURNG = EDT$$PA_NEW_NOD (RANGE_NODE, 0)) EQL 0) THEN RETURN (0);
410 1263 2
411 1264 2          EDT$$Z_PA_THRURNG [RANGE1] = .EDT$$Z_PA_CURRNG;
412 1265 2          END;
413 1266 2
414 1267 2          [THRU_RAN] :
415 1268 2          BEGIN
416 1269 2          EDT$$Z_PA_THRURNG [RAN_TYPE] = RAN_THRU;
417 1270 2          EDT$$Z_PA_THRURNG [RANGE2] = .EDT$$Z_PA_CURRNG;
418 1271 2          EDT$$Z_PA_CURRNG = .EDT$$Z_PA_THRURNG;
419 1272 2          END;
420 1273 2
421 1274 2          [AND_HEAD] :
422 1275 2          EDT$$Z_PA_ANDLSTHD = .EDT$$Z_PA_CURRNG;
423 1276 2
424 1277 2          [AND_NEXT] :
425 1278 2          BEGIN
426 1279 2          LOCAL
427 1280 2          RANGE : REF NODE_BLOCK;
428 1281 2          RANGE = .EDT$$Z_PA_ANDLSTHD;
429 1282 2
430 1283 2          RANGE = .EDT$$Z_PA_ANDLSTHD;
431 1284 2          !+
432 1285 2          !- Find the last range so we can put the new one on the end.
433 1286 2
434 1287 2
435 1288 2          WHILE (.RANGE [NEXT_RANGE] NEQA 0) DO
436 1289 2          RANGE = .RANGE [NEXT_RANGE];
437 1290 2
438 1291 2          RANGE [NEXT_RANGE] = .EDT$$Z_PA_CURRNG;
439 1292 2          EDT$$Z_PA_CURRNG [PREV_RANGE] = .RANGE;
440 1293 2          EDT$$Z_PA_CURRNG = .EDT$$Z_PA_ANDLSTHD;
441 1294 2          END;
442 1295 2
443 1296 2          [WHICHSUBS] :
444 1297 2          BEGIN
445 1298 2          ! Distinguish SUBSTITUTE from SUBSTITUTE NEXT
446 1299 2          IF (.OPERAND EQL 1) THEN EDT$$G_PA_CURCMD [COM_NUM] = COM_SUBS_NEXT;
447 1300 2
448 1301 2          !+
449 1302 2          !- Since we are in what seemed to have been a substitute command, the EDT$$G_PA_NOQUO
450 1303 2          flag must be set.
451 1304 2
452 1305 2          ASSERT (.EDT$$G_PA_NOQUO);
453 1306 2          END;
454 1307 2
```

```

455 1308 2      [STRINGS] :      ! Get the search and replace strings for SUBSTITUTE
456 1309      BEGIN
457 1310
458 1311      LOCAL
459 1312          STRNODE : REF NODE_BLOCK,
460 1313          CURSOR,
461 1314          QUOTE;
462 1315
463 1316
464 1317      + The EDT$G_PA_NOQUO flag had better be set, to keep the scanner from having
465 1318      + swallowed a quoted string. Consider the following case:
466 1319
467 1320          +SUBSTITUTE 'A'B'
468 1321
469 1322      We must use ' as the delimiter, but the scanner would absorb 'A' as a single (string)
470 1323      token unless the flag is set. We clear the flag here since we will not be calling
471 1324      the scanner again until after we have scanned out two strings.
472 1325
473 1326      ASSERT (.EDT$G_PA_NOQUO);
474 1327      EDT$G_PA_NOQUO = 0;
475 1328
476 1329      IF ((STRNODE = EDT$PA_NEW_NOD (STR_NODE, 0)) EQL 0) THEN RETURN (0);
477 1330
478 1331      EDT$G_PA_CURCMD [STR_PNT] = .STRNODE;
479 1332
480 1333      IF (.EDT$G_PA_TOKCLASS NEQ CL_SPECIAL)
481 1334      THEN
482 1335          BEGIN
483 1336              EDT$G_PA_ERRNO = EDT$_NONALPNUM;
484 1337              RETURN (0);
485 1338          END;
486 1339
487 1340      QUOTE = CHRCHAR (.EDT$PA_CURTOK);
488 1341      CURSOR = CH$PLUS (.EDT$PA_CURTOK, 1);
489 1342      STRNODE [SRCHADDR] = .CURSOR;
490 1343
491 1344      UNTIL ((CHRCHAR (.CURSOR) EQL .QUOTE) OR (.CURSOR GEQU .EDT$PA_CMD_END)) DO
492 1345          CURSOR = CH$PLUS (.CURSOR, 1);
493 1346
494 1347      STRNODE [SRCHLEN] = .CURSOR - .EDT$PA_CURTOK - 1;
495 1348      CURSOR = CH$PLUS (.CURSOR, 1);
496 1349
497 1350      IF (.CURSOR GTRU .EDT$PA_CMD_END)
498 1351      THEN
499 1352          BEGIN
500 1353              EDT$G_PA_ERRNO = EDT$_INVSTR;
501 1354              RETURN (0);
502 1355          END;
503 1356
504 1357      STRNODE [REPADDR] = .CURSOR;
505 1358
506 1359      UNTIL ((CHRCHAR (.CURSOR) EQL .QUOTE) OR (.CURSOR GEQU .EDT$PA_CMD_END)) DO
507 1360          CURSOR = CH$PLUS (.CURSOR, 1);
508 1361
509 1362      STRNODE [REPLEN] = .CURSOR - .STRNODE [REPADDR];
510 1363      EDT$PA_CMD_BUF = CH$PLUS (.CURSOR, 1);
511 1364      EDT$PA_GETCH ();

```

```

512 1365 3      EDT$$PA_SCANTOK ();
513 1366 3
514 1367 4      IF ((.STRNODE [REPLEN] EQL 0) AND (.STRNODE [SRCHLEN] EQL 0))
515 1368 3      THEN
516 1369 4      BEGIN
517 1370 4      EDT$$G_PA_ERRNO = EDT$_SUBSTRNUL;
518 1371 4      RETURN (0);
519 1372 3      END;
520 1373 3
521 1374 2      END;
522 1375 2
523 1376 2 [DEFAULT STRINGS] : ! We will be using the strings from the last SUB or SUB NEXT
524 1377 3 BEGIN
525 1378 3 ASSERT (.EDT$$G_PA_NOQUO);
526 1379 3 EDT$$G_PA_NOQUO = 0;
527 1380 2 END;
528 1381 2
529 1382 2 [FILSPEC] : ! Scan a file name
530 1383 3 BEGIN
531 1384 3
532 1385 3 LOCAL
533 1386 3 SCAN_DONE, ! 1 = file name scan complete
534 1387 3 CHAR, ! Current character being processed
535 1388 3 QUOTE_CHAR; ! 0 = not in a string, non-zero = right quote character
536 1389 3
537 1390 3 ASSERT ((%C'' NEQ 0) AND (%C'' NEQ 0));
538 1391 3 EDT$$G_PA_CURCMD [FILSPEC] = .EDT$$A_PA_CURTOK;
539 1392 3 EDT$$A_CMD_BUF = .EDT$$A_PA_CURTOK;
540 1393 3 SCAN_DONE = 0;
541 1394 3 QUOTE_CHAR = 0;
542 1395 3
543 1396 3 WHILE ( NOT .SCAN_DONE) DO
544 1397 3
545 1398 4 IF CH$PTR_GTR (.EDT$$A_CMD_BUF, .EDT$$A_CMD_END)
546 1399 3 THEN
547 1400 3 SCAN_DONE = 1
548 1401 3 ELSE
549 1402 4 BEGIN
550 1403 4 CHAR = CH$RCHAR_A (EDT$$A_CMD_BUF);
551 1404 4
552 1405 5 IF (.QUOTE_CHAR EQL 0)
553 1406 4 THEN
554 1407 4
555 1408 4 SELECTONE .CHAR OF
556 1409 4 SET
557 1410 4
558 1411 4 [%C' ', %C'/'] :
559 1412 4 SCAN_DONE = 1;
560 1413 4
561 1414 4 [%C''', %C'''''] :
562 1415 4 QUOTE_CHAR = .CHAR;
563 1416 4
564 1417 4 [OTHERWISE] :
565 1418 5 BEGIN
566 1419 5 0
567 1420 4 END;
568 1421 4 YES

```

```
569 1422 4  
570 1423 4  
571 1424 4  
572 1425 4  
573 1426 4  
574 1427 4  
575 1428 4  
576 1429 4  
577 1430 4  
578 1431 4  
579 1432 4  
580 1433 4  
581 1434 4  
582 1435 4  
583 1436 4  
584 1437 4  
585 1438 4  
586 1439 4  
587 1440 4  
588 1441 4  
589 1442 4  
590 1443 4  
591 1444 4  
592 1445 4  
593 1446 4  
594 1447 4  
595 1448 4  
596 1449 4  
597 1450 4  
598 1451 4  
599 1452 4  
600 1453 4  
601 1454 4  
602 1455 4  
603 1456 4  
604 1457 4  
605 1458 4  
606 1459 4  
607 1460 4  
608 1461 4  
609 1462 4  
610 1463 4  
611 1464 4  
612 1465 4  
613 1466 4  
614 1467 4  
615 1468 4  
616 1469 4  
617 1470 4  
618 1471 4  
619 1472 4  
620 1473 4  
621 1474 4  
622 1475 4  
623 1476 4  
624 1477 4  
625 1478 4
```

```
ELSE  
IF (.CHAR EQL .QUOTE_CHAR) THEN QUOTE_CHAR = 0;  
END;  
EDT$$C_PA_CH = .CHAR;  
EDT$$G_PA_CURCMD [FSPCLN] = .EDT$$A_CMD_BUF - .EDT$$G_PA_CURCMD [FILSPEC] - 1;  
EDT$$PA_SCANTOK ();  
END;  
[HELPSTR] :  
BEGIN  
EDT$$G_PA_CURCMD [FILSPEC] = .EDT$$A_PA_CURTOK;  
EDT$$A_CMD_BUF = .EDT$$A_PA_CURTOK;  
EDT$$PA_GETCH ();  
WHILE ((.EDT$$C_PA_CH NEQ %C'!') AND (.EDT$$C_PA_CH NEQ %C';')) DO  
EDT$$PA_GETCH ();  
EDT$$G_PA_CURCMD [FSPCLN] = .EDT$$A_CMD_BUF - .EDT$$G_PA_CURCMD [FILSPEC] - 1;  
EDT$$PA_SCANTOK ();  
END;  
[CHKALPHA] :  
IF ((.EDT$$G_PA_TOKCLASS EQL CL_NAME) AND (NOT .EDT$$G_PA_PCEN))  
THEN  
BEGIN  
EDT$$G_PA_ERRNO = EDT$UNRCOM;  
RETURN (0);  
END;  
[A_SWITCH] :  
BEGIN  
LOCAL  
SWITCH_NODE : REF NODE_BLOCK;  
IF (.EDT$$G_PA_CURCMD [SWITS] EQL 0)  
THEN  
BEGIN  
IF ((SWITCH_NODE = EDT$$PA_NEW_NOD (SW_NODE, 0)) EQL 0) THEN RETURN (0);  
EDT$$G_PA_CURCMD [SWITS] = .SWITCH_NODE;  
END  
ELSE  
SWITCH_NODE = .EDT$$G_PA_CURCMD [SWITS];  
IF ((.SWITCH_NODE [SW_BITS] AND (1^.OPERAND)) NEQ 0) THEN RETURN (0);  
SWITCH_NODE [SW_BITS] = (.SWITCH_NODE [SW_BITS] OR (1^.OPERAND));  
END;  
[SWITCH_1] :
```

```

626 1479 3 BEGIN
627 1480 3
628 1481 3 BIND
629 1482 3 SWITCH = .EDT$$G_PA_CURCMD [SWITS] : NODE_BLOCK;
630 1483 3
631 1484 3 MOVELINE (EDT$$L_PA_NUMVAL, SWITCH [SW_VAL1]);
632 1485 3 SWITCH [SEQ_VAL] = T;
633 1486 3 END;
634 1487 3
635 1488 3 [SWITCH 2] :
636 1489 3 BEGIN
637 1490 3
638 1491 3 BIND
639 1492 3 SWITCH = .EDT$$G_PA_CURCMD [SWITS] : NODE_BLOCK;
640 1493 3
641 1494 3 MOVELINE (EDT$$L_PA_NUMVAL, SWITCH [SW_VAL2]);
642 1495 3 END;
643 1496 3
644 1497 3 [SETTYPE] :
645 1498 3 EDT$$G_PA_CURCMD [SET_TYPE] = .OPERAND;
646 1499 3
647 1500 3 [SETVAL] :
648 1501 3 EDT$$G_PA_CURCMD [SET_VAL] = .OPERAND;
649 1502 3
650 1503 3 [SET_ARG] :
651 1504 3 BEGIN
652 1505 3
653 1506 4 IF ((.EDT$$L_PA_NUMVAL [LN_LO] GTRU 32767) OR !
654 1507 4 (.EDT$$L_PA_NUMVAL [LN_MD] NEQ 0) OR !
655 1508 4 (.EDT$$L_PA_NUMVAL [LN_HI] NEQ 0))
656 1509 3 THEN
657 1510 4 BEGIN
658 1511 4 EDT$$G_PA_ERRNO = EDT$_NUMVALILL;
659 1512 4 RETURN (0);
660 1513 3 END;
661 1514 3
662 1515 3 EDT$$G_PA_CURCMD [SET_VAL] = .EDT$$L_PA_NUMVAL [LN_LO];
663 1516 3 END;
664 1517 3
665 1518 3 [SET_ARG1] :
666 1519 3 BEGIN
667 1520 3
668 1521 4 IF ((.EDT$$L_PA_NUMVAL [LN_LO] GTRU 32767) OR !
669 1522 4 (.EDT$$L_PA_NUMVAL [LN_MD] NEQ 0) OR !
670 1523 4 (.EDT$$L_PA_NUMVAL [LN_HI] NEQ 0))
671 1524 3 THEN
672 1525 4 BEGIN
673 1526 4 EDT$$G_PA_ERRNO = EDT$_NUMVALILL;
674 1527 4 RETURN (0);
675 1528 3 END;
676 1529 3
677 1530 3 EDT$$G_PA_CURCMD [SET_VAL1] = .EDT$$L_PA_NUMVAL [LN_LO];
678 1531 3 END;
679 1532 3
680 1533 3 [DEF_KEY] : ! Start of key description
681 1534 3 BEGIN
682 1535 3 EDT$$G_DEFKEY = 1;

```

```

683 1536 2      END;
684 1537 2
685 1538 2      [KEY_NUM] :                ! Key number
686 1539 2      BEGIN
687 1540 3      EDT$$G_PA_CURCMD [KEY_VAL] = .EDT$$L_PA_NUMVAL [LN_LO] + K_KPAD_BASE;
688 1541 3
689 1542 4      IF ((.EDT$$L_PA_NUMVAL [LN_LO] GTRU 32767) OR      !
690 1543 4          (.EDT$$L_PA_NUMVAL [LN_MD] NEQ 0) OR      !
691 1544 4          (.EDT$$L_PA_NUMVAL [LN_HI] NEQ 0))
692 1545 3      THEN
693 1546 4          BEGIN
694 1547 4              EDT$$G_PA_ERRNO = EDT$_NUMVALILL;
695 1548 4              RETURN(0);
696 1549 3          END;
697 1550 3
698 1551 4      IF (.EDT$$L_PA_NUMVAL [LN_LO] GTR 21)
699 1552 3      THEN
700 1553 4          BEGIN
701 1554 4              EDT$$G_PA_ERRNO = EDT$_KEYNOTDEF;
702 1555 4              RETURN(0);
703 1556 3          END;
704 1557 3
705 1558 2      END;
706 1559 2
707 1560 2      [GOLD_KEY_NUM] :          ! GOLD key number
708 1561 3      BEGIN
709 1562 3      EDT$$G_PA_CURCMD [KEY_VAL] = .EDT$$L_PA_NUMVAL [LN_LO] + K_KPAD_BASE + K_GOLD_BASE;
710 1563 3
711 1564 4      IF ((.EDT$$L_PA_NUMVAL [LN_LO] GTRU 32767) OR      !
712 1565 4          (.EDT$$L_PA_NUMVAL [LN_MD] NEQ 0) OR      !
713 1566 4          (.EDT$$L_PA_NUMVAL [LN_HI] NEQ 0))
714 1567 3      THEN
715 1568 4          BEGIN
716 1569 4              EDT$$G_PA_ERRNO = EDT$_NUMVALILL;
717 1570 4              RETURN(0);
718 1571 3          END;
719 1572 3
720 1573 4      IF (.EDT$$L_PA_NUMVAL [LN_LO] GTR 21)
721 1574 3      THEN
722 1575 4          BEGIN
723 1576 4              EDT$$G_PA_ERRNO = EDT$_KEYNOTDEF;
724 1577 4              RETURN(0);
725 1578 3          END;
726 1579 3
727 1580 2      END;
728 1581 2
729 1582 2      [DEF GOLD_DEL] :
730 1583 3      BEGIN
731 1584 3      EDT$$G_PA_CURCMD [KEY_VAL] = ASC_K_DEL + K_GOLD_BASE;
732 1585 2      END;
733 1586 2
734 1587 2      [DEF DELETE] :
735 1588 3      BEGIN
736 1589 3      EDT$$G_PA_CURCMD [KEY_VAL] = ASC_K_DEL;
737 1590 2      END;
738 1591 2
739 1592 2      [DEF_CHAR] :
```



```

: 740      1593 3      BEGIN
: 741      1594 3      EDT$$G_PA_ERRNO = EDT$_KEYNOTDEF,
: 742      1595 3      RETURN(0);
: 743      1596 2      END;
: 744      1597 2
: 745      1598 2      [DEF GOLD_CHAR] :
: 746      1599 3      BEGIN
: 747      1600 3
: 748      1601 3      LOCAL
: 749      1602 3      CHAR;
: 750      1603 3
: 751      1604 3      CHAR = CH$RCHAR (.EDT$$A_PA_CURTOK);
: 752      1605 3      EDT$$G_PA_CURCMD [KEY_VAC] = .CHAR + K_GOLD_BASE;
: 753      1606 3
: 754      1607 4      IF ((.EDT$$G_PA_CURTOKLEN NEQ 1) OR          ! Other than one char in string
: 755      1608 4
: 756      L 1609 4 %IF SUPPORT_VT220
: 757      1610 4 %THEN
: 758      1611 4          (.EDT$$B_CHAR_INFO [.CHAR, 0, 0, 8, 0] EQL %X'F0') OR ! Digit
: 759      U 1612 4 %ELSE
: 760      U 1613 4          ((.CHAR GEQ %C'0') AND (.CHAR LEQ %C'9')) OR ! Digit
: 761      1614 4 %FI
: 762      1615 4
: 763      1616 4          (.CHAR LSS 32) OR ! C0 control char (must use CONTROL)
: 764      1617 4          (.CHAR GTR 255) OR ! Not a character
: 765      1618 4          ((.CHAR GEQ 128) AND (.CHAR LSS 128 + 32)) OR ! C1 control char
: 766      1619 4          (.CHAR EQL ASC_K_DEL)) ! DEL (must use DELETE)
: 767      1620 3      THEN
: 768      1621 4      BEGIN
: 769      1622 4      EDT$$G_PA_ERRNO = EDT$_KEYNOTDEF;
: 770      1623 4      RETURN(0);
: 771      1624 3      END;
: 772      1625 3
: 773      1626 3      EDT$$PA_SCANTOK ();
: 774      1627 2      END;
: 775      1628 2
: 776      1629 2      [GOLD CONT] :
: 777      1630 3      BEGIN
: 778      1631 3
: 779      1632 3      LOCAL
: 780      1633 3      CHAR;
: 781      1634 3
: 782      1635 3      CHAR = CH$RCHAR (.EDT$$A_PA_CURTOK) - 64;
: 783      1636 3      EDT$$G_PA_CURCMD [KEY_VAC] = .CHAR + K_GOLD_BASE;
: 784      1637 3
: 785      1638 4      IF ((.EDT$$G_PA_CURTOKLEN NEQ 1) OR          !
: 786      1639 4          (.CHAR LSS 0) OR !
: 787      1640 4          (.CHAR GTR 255) OR !
: 788      1641 4          ((.CHAR GEQ 32) AND (.CHAR LSS 128)) OR !
: 789      1642 4          (.CHAR GEQ 128 + 32)) !
: 790      1643 3      THEN
: 791      1644 4      BEGIN
: 792      1645 4      EDT$$G_PA_ERRNO = EDT$_KEYNOTDEF;
: 793      1646 4      RETURN(0);
: 794      1647 3      END;
: 795      1648 3
: 796      1649 3      EDT$$PA_SCANTOK ();
```

```
797 1650 2      END;
798 1651 2
799 1652 2      [CONT CHAR] :
800 1653 3      BEGIN
801 1654 3
802 1655 3      LOCAL
803 1656 3      CHAR;
804 1657 3
805 1658 3      CHAR = CHRCHAR (.EDT$$A_PA_CURTOK) - 64;
806 1659 3      EDT$$G_PA_CURCMD [KEY_VAL] = .CHAR;
807 1660 3
808 1661 4      IF ((.EDT$$G_PA_CURTOKLEN NEQ 1) OR      !
809 1662 4          (.CHAR LSS 0) OR      !
810 1663 4          (.CHAR GTR 255) OR      !
811 1664 4          ((.CHAR GEQ 32) AND (.CHAR LSS 128)) OR      !
812 1665 4          (.CHAR GEQ 128 + 32))
813 1666 3      THEN
814 1667 4      BEGIN
815 1668 4      EDT$$G_PA_ERRNO = EDT$_KEYNOTDEF;
816 1669 4      RETURN (0);
817 1670 3      END;
818 1671 3
819 1672 3      EDT$$PA_SCANTOK ();
820 1673 2      END;
821 1674 2
822 1675 2      [DEF FUN] :
823 1676 3      BEGIN
824 1677 3      EDT$$G_PA_CURCMD [KEY_VAL] = .EDT$$L_PA_NUMVAL [LN_LO] + K_FUN_BASE;
825 1678 3
826 1679 4      IF ((.EDT$$L_PA_NUMVAL [LN_LO] GTRU 32767) OR      !
827 1680 4          (.EDT$$L_PA_NUMVAL [LN_MD] NEQ 0) OR      !
828 1681 4          (.EDT$$L_PA_NUMVAL [LN_HI] NEQ 0))
829 1682 3      THEN
830 1683 4      BEGIN
831 1684 4      EDT$$G_PA_ERRNO = EDT$_NUMVALILL;
832 1685 4      RETURN (0);
833 1686 3      END;
834 1687 3
835 1688 4      IF (.EDT$$L_PA_NUMVAL [LN_LO] GTR K_MAX_FUN_VAL)
836 1689 3      THEN
837 1690 4      BEGIN
838 1691 4      EDT$$G_PA_ERRNO = EDT$_KEYNOTDEF;
839 1692 4      RETURN (0);
840 1693 3      END;
841 1694 3
842 1695 2      END;
843 1696 2
844 1697 2      [DEF GOLD_FUN] :
845 1698 3      BEGIN
846 1699 3      EDT$$G_PA_CURCMD [KEY_VAL] = .EDT$$L_PA_NUMVAL [LN_LO] + K_FUN_BASE + K_GOLD_BASE;
847 1700 3
848 1701 4      IF ((.EDT$$L_PA_NUMVAL [LN_LO] GTRU 32767) OR      !
849 1702 4          (.EDT$$L_PA_NUMVAL [LN_MD] NEQ 0) OR      !
850 1703 4          (.EDT$$L_PA_NUMVAL [LN_HI] NEQ 0))
851 1704 3      THEN
852 1705 4      BEGIN
853 1706 4      EDT$$G_PA_ERRNO = EDT$_NUMVALILL;
```

```

: 854 1707 4          RETURN (0);
: 855 1708 3          END;
: 856 1709 3
: 857 1710 4          IF (.EDT$$L_PA_NUMVAL [LN_LO] GTR K_MAX_FUN_VAL)
: 858 1711 3          THEN
: 859 1712 4              BEGIN
: 860 1713 4                  EDT$$G_PA_ERRNO = EDT$_KEYNOTDEF;
: 861 1714 4                  RETURN (0);
: 862 1715 3                  END;
: 863 1716 3
: 864 1717 2          END;
: 865 1718 2
: 866 1719 2          [AS_STRING] :
: 867 1720 3              BEGIN
: 868 1721 3                  EDT$$G_PA_CURCMD [AS_STR] = .EDT$$A_PA_PRVTOK + 1;
: 869 1722 3                  EDT$$G_PA_CURCMD [AS_LEN] = .EDT$$G_PA_PRVTOKLEN;
: 870 1723 2              END;
: 871 1724 2
: 872 1725 2          [INIT_SEQ] :
: 873 1726 3              BEGIN
: 874 1727 3
: 875 1728 3                  BIND
: 876 1729 3                      SWIT = .EDT$$G_PA_CURCMD [SWITS] : NODE_BLOCK;
: 877 1730 3
: 878 1731 3                      MOVELINE (EDT$$L_LNOO [5], SWIT [SW_VAL1]);
: 879 1732 3                      MOVELINE (EDT$$L_LNOO [5], SWIT [SW_VAL2]);
: 880 1733 3                      SWIT [SEQ_VAL] = 0;
: 881 1734 2              END;
: 882 1735 2
: 883 1736 2          [DEF_MAC] :
: 884 1737 3              BEGIN
: 885 1738 3                  EDT$$G_PA_CURCMD [RANGE1] = .EDT$$Z_PA_CURRNG;
: 886 1739 3                  EDT$$G_PA_CURCMD [COM_NUM] = COM_DEF_MAC;
: 887 1740 2              END;
: 888 1741 2
: 889 1742 2          [TABCOUNT] :
: 890 1743 3              BEGIN
: 891 1744 3
: 892 1745 3                  LOCAL
: 893 1746 3                      NEG;
: 894 1747 3
: 895 1748 3                      NEG = 0;
: 896 1749 3
: 897 1750 4                      IF (CH$RCHAR (.EDT$$A_PA_CURTOK) EQL %C'-')
: 898 1751 3                      THEN
: 899 1752 4                          BEGIN
: 900 1753 4                              NEG = .NEG + 1;
: 901 1754 4                              EDT$$PA_SCANTOK ();
: 902 1755 3                          END;
: 903 1756 3
: 904 1757 4                      IF (.EDT$$G_PA_TOKCLASS NEQ CL_NUMBER)
: 905 1758 3                      THEN
: 906 1759 4                          BEGIN
: 907 1760 4                              EDT$$G_PA_ERRNO = EDT$_NUMVALREQ;
: 908 1761 4                              RETURN (0);
: 909 1762 3                          END;
: 910 1763 3

```

: R

:

```

911 1764 4      IF ((.EDTSSL_PA_NUMVAL [LN_LO] GTRU 32767) OR      !
912 1765 4      (.EDTSSL_PA_NUMVAL [LN_MD] NEQ 0) OR      !
913 1766 4      (.EDTSSL_PA_NUMVAL [LN_HI] NEQ 0))
914 1767 3      THEN
915 1768 4      BEGIN
916 1769 4      EDT$G_PA_ERRNO = EDT$_NUMVALILL;
917 1770 4      RETURN(0);
918 1771 3      END;
919 1772 3
920 1773 4      IF ((.EDTSSL_PA_NUMVAL [LN_LO]*.EDT$G_TAB_SIZ) GTR 255)
921 1774 3      THEN
922 1775 4      BEGIN
923 1776 4      EDT$G_PA_ERRNO = EDT$_NUMVALILL;
924 1777 4      RETURN(0);
925 1778 3      END;
926 1779 3
927 1780 3      IF .NEG
928 1781 3      THEN
929 1782 3      EDT$G_PA_CURCMD [TAB_COUNT] = -.EDTSSL_PA_NUMVAL [LN_LO]
930 1783 3      ELSE
931 1784 3      EDT$G_PA_CURCMD [TAB_COUNT] = .EDTSSL_PA_NUMVAL [LN_LO];
932 1785 3
933 1786 3      EDT$PA_SCANTOK ();
934 1787 2      END;
935 1788 2
936 1789 2      [BAD_PARAM] :
937 1790 2      EDT$G_PA_ERRNO = EDT$_INVPARFOR;
938 1791 2
939 1792 2      [BAD_VALUE] :
940 1793 2      EDT$G_PA_ERRNO = EDT$_INVVALSET;
941 1794 2
942 1795 2      [REQ_NUM] :
943 1796 2      EDT$G_PA_ERRNO = EDT$_NUMVALREQ;
944 1797 2
945 1798 2      [REQ_STRING] :
946 1799 2      EDT$G_PA_FRRNO = EDT$_QUOSTRREQ;
947 1800 2
948 1801 2      [BAD_RANGE] :
949 1802 2      EDT$G_PA_ERRNO = EDT$_ERRRANSPC;
950 1803 2
951 1804 2      [BAD_OPTION] :
952 1805 2      EDT$G_PA_ERRNO = EDT$_ERRCOMOPT;
953 1806 2
954 1807 2      [UNREC_OPTION] :
955 1808 2      EDT$G_PA_ERRNO = EDT$_UNRCOMOPT;
956 1809 2
957 1810 2      [REQ_COLON] :
958 1811 2      EDT$G_PA_ERRNO = EDT$_COLONREQ;
959 1812 2
960 1813 2      [MACORKEY] :
961 1814 2      EDT$G_PA_ERRNO = EDT$_MACKEYREQ;
962 1815 2
963 1816 2      [ENTITY_ERR] :
964 1817 2      EDT$G_PA_ERRNO = EDT$_ENTMUSTBE;
965 1818 2
966 1819 2      [REQ_AS] :
967 1820 3      BEGIN

```

```

: 968      1821 3      EDT$$G_DEFKEY = 0;
: 969      1822 3      EDT$$G_PA_ERRNO = EDT$_ASREQ;
: 970      1823 3      END;
: 971      1824 3
: 972      1825 3      [NO_ACTION] :
: 973      1826 3      :
: 974      1827 3
: 975      1828 3      [OUTRANGE] :
: 976      1829 3      ASSERT (0);
: 977      1830 3      TES;
: 978      1831 3
: 979      1832 3      RETURN (1);
: 980      1833 1      END;

```

! don't accept quoted key anymore

! of routine EDT\$\$\$PA_SEMRUT

.TITLE EDT\$PRSEMRTN EDT\$PRSEMRTN - parser semantic actions

.IDENT \V04-000\

```

.EXTRN EDT$$$PA_SCANTOK
.EXTRN EDT$$$PA_APPDIG, EDT$$$PA_GETCH
.EXTRN EDT$$$PA_CRERNGNOD
.EXTRN EDT$$$PA_NEW_NOD
.EXTRN EDT$$L_MAX_LINES
.EXTRN EDT$$A_CMD_BUF, EDT$$A_CMD_END
.EXTRN EDT$$Z_PA_ANDLSTHD
.EXTRN EDT$$Z_PA_BUFRNG
.EXTRN EDT$$C_PA_CH, EDT$$G_PA_CURCMD
.EXTRN EDT$$Z_PA_CURRNG
.EXTRN EDT$$A_PA_CURTOK
.EXTRN EDT$$G_DEFKEY, EDT$$G_PA_CURTOKLEN
.EXTRN EDT$$G_PA_ERRNO
.EXTRN EDT$$L_PA_NUMVAL
.EXTRN EDT$$G_PA_PCEN
.EXTRN EDT$$A_PA_PRTOK
.EXTRN EDT$$G_PA_PRTOKLEN
.EXTRN EDT$$G_PA_SP, EDT$$Z_PA_THRURNG
.EXTRN EDT$$G_PA_TOKCLASS
.EXTRN EDT$$G_PA_NOQUO
.EXTRN EDT$$L_LN00, EDT$$B_CHAR_INFO
.EXTRN EDT$$G_TAB_SIZ, EDT$_INVBUFFNAM
.EXTRN EDT$_QOOSTRREQ, EDT$_NONALPNUM
.EXTRN EDT$_SUBSTRNUL, EDT$_UNRCOM
.EXTRN EDT$_KEYNOTDEF, EDT$_NUMVALREQ
.EXTRN EDT$_INVPARFOR, EDT$_INVVALSET
.EXTRN EDT$_ERRRANSPC, EDT$_ERRCOMOPT
.EXTRN EDT$_UNRCOMOPT, EDT$_COLONREQ
.EXTRN EDT$_MACKEYREQ, EDT$_ENTMUSTBE
.EXTRN EDT$_ASREQ, EDT$_INVSTR
.EXTRN EDT$_NUMVALILL, EDT$$INTER_ERR

```

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

```

.ENTRY EDT$$$PA_SEMRUT, Save R2,R3,R4,R5,R6,R7,R8,- ; 0968
      R9,R10,R11
MOVAB EDT$$A_PA_CURTOK, R11
MOVAB EDT$$Z_PA_CURRNG, R10

```

```

OFFC 00000
5B 00000000G 00 9E 00002
5A 00000000G 00 9E 00009

```


				02BE	31	000A5		BRW	154\$-1\$,-		
		00000000G	00	D5	000A8	2\$:		TSTL	152\$-1\$,-		1829
			07	13	000AE			BEQL	147\$-1\$,-		1075
	00000000G	00	FB	000B0				CALLS	122\$-1\$,-		
		50	D4	000B7	3\$:			CLRL	108\$-1\$,-		1076
			D0	000BD				MOVL	114\$-1\$,-		1078
			08	13	000C0			BEQL	104\$-1\$,-		
	10	A0	00000000G	00	D0	000C2		MOVL	93\$-1\$		
			08	AC	DD	000CA	4\$:	PUSHL	60\$		
				01	DD	000CD		PUSHL	EDTSSG_PA_NOQUO		
	00000000G	00	FB	000CF				CALLS	3\$		
		68	D0	000D6				MOVL	#0, EDT\$\$INTER_ERR		
			02	13	000D9			BEQL	EDTSSG_DEFKEY		1076
		10	AC	D1	000DB			BEQL	EDTSSG_PA_CURCMD, R0		1078
			06	13	000DF			BEQL	4\$		
		13	AC	D1	000E1			MOVL	EDTSSG_PA_SP, 16(R0)		
			77	12	000E5			PUSHL	OPERAND		1080
	00000000G	00	D0	000E7	5\$:			PUSHL	#1		
			0E	11	000EE			CALLS	#2, EDTSSPA_NEW_NOD		
			AC	DD	000F0	6\$:		MOVL	R0, EDTSSG_PA_CORCMD		
			02	DD	000F3			BEQL	9\$		
	00000000G	00	FB	000F5				BEQL	OPERAND, #16		1086
		6A	D0	000FC				BEQL	5\$		
			3C	13	000FF			BEQL	OPERAND, #19		
		01	D1	00101				BNEQ	11\$		
			7C	12	00108			MOVVL	#1, EDTSSG_PA_NOQUO		
			00FA	31	0010A			BRB	11\$		1066
			AC	D5	0010D	7\$:		PUSHL	OPERAND		1093
			7A	13	00110			PUSHL	#2		
	7E	08	AC	C1	00112			CALLS	#2, EDTSSPA_NEW_NOD		
			15	DD	00117			MOVL	R0, EDTSSZ_PA_CORRNG		
			02	FB	00119			BEQL	9\$		
			00C9	31	0011E			BEQL	EDTSSG_PA_TOKCLASS, #1		1095
			8F	D0	00121	8\$:		BNEQ	14\$		
			00	FB	00128			BRW	35\$		1097
			50	E9	0012F			TSTL	OPERAND		1104
			0D	DD	00132			BEQL	15\$		
			15	DD	00134			ADDL3	#7, OPERAND, -(SP)		1107
			02	FB	00136			PUSHL	#21		
			00C9	31	0011E			CALLS	#2, EDTSSPA_SFMRUT		
			8F	D0	00121	8\$:		BRW	28\$		
			00	FB	00128			MOVL	#EDT\$ INVBUFNAM, EDTSSG_PA_ERRNO		1113
			50	E9	0012F			CALLS	#0, EDTSSPA_APPDIG		1115
			0D	DD	00132			BLBC	R0, 17\$		
			15	DD	00134			PUSHL	#13		1117
			02	FB	00136			PUSHL	#21		
			50	D5	00138			CALLS	#2, EDTSSPA_SEMRUT		
			79	13	0013D	9\$:		TSTL	R0		
			6A	D0	0013F			BEQL	18\$		
			08	D0	00142			MOVL	EDTSSZ_PA_CURRNG, R0		1119
			0C	D0	00146			MOVL	EDTSSA_PA_CURTOK, 8(R0)		
			50	D0	0014E			MOVL	EDTSSG_PA_CURTOKLEN, 12(R0)		1120
			70	11	00155			MOVL	R0, EDTSSZ_PA_BUFRNG		1121
			00	FB	00157	10\$:		BRB	20\$		1122
			77	11	0015E	11\$:		CALLS	#0, EDTSSPA_APPDIG		1126
			50	D0	00160	12\$:		BRB	23\$		
			00	D0	00160			MOVL	EDTSSZ_PA_BUFRNG, R0		1130

	C4	A0		6A	D0	00167	MOVL	EDT\$\$Z_PA_CURRNG, 4(R0)			
				01B1	31	0016B	BRW	53\$	1131		
		50		6A	D0	0016E	13\$:	MOVL	EDT\$\$Z_PA_CURRNG, R0	1136	
	04	A0	00000000G	00	D0	00171	MOVL	EDT\$\$G_PA_PRTOKLEN, 4(R0)			
08	A0	00000000G		00	C1	00179	ADDL3	#1, EDT\$\$A_PA_PRTOK, 8(R0)	1137		
		0F		01	A0	91	00182	CMPB	1(R0), #15	1139	
				7D	12	00186	14\$:	BNEQ	34\$		
	01	A0		12	90	00188	MOVb	#18, 1(R0)		1066	
				77	11	0018C	15\$:	BRB	34\$		
		52		6A	D0	0018E	16\$:	MOVL	EDT\$\$Z_PA_CURRNG, SUB_RAN	1149	
				13	DD	00191	PUSHL	#19	1151		
				15	DD	00193	PUSHL	#21			
	FE66	CF		02	FB	00195	CALLS	#2, EDT\$\$PA_SEMRUT			
		51		50	E9	0019A	17\$:	BLBC	R0, 29\$		
		50		6A	D0	0019D	MOVL	EDT\$\$Z_PA_CURRNG, R0	1156		
	10	A0		52	D0	001A0	MOVL	SUB_RAN, T6(R0)			
	14	A2		50	D0	001A4	MOVL	R0, -20(SUB_RAN)	1157		
		03	00000000G	00	D1	001A8	CMPL	EDT\$\$G_PA_TOKCLASS, #3	1159		
				09	13	001AF	BEQL	19\$			
		69	00000000G	8F	D0	001B1	MOVL	#EDT\$_QUOSTRREQ, EDT\$\$G_PA_ERRNO	1162		
				34	11	001B8	18\$:	BRB	29\$	1163	
	08	A0	00000000G	00	D0	001BA	19\$:	MOVL	EDT\$\$G_PA_CURTOKLEN, 4(R0)	1166	
		6B		01	C1	001C2	ADDL3	#1, EDT\$\$A_PA_CURTOK, 8(R0)	1167		
				05D5	31	001C7	20\$:	BRW	141\$	1168	
		50		68	D0	001CA	21\$:	MOVL	EDT\$\$G_PA_CURCMD, R0	1172	
				013A	31	001CD	BRW	51\$			
		50		68	D0	001D0	22\$:	MOVL	EDT\$\$G_PA_CURCMD, R0	1175	
	08	A0		6A	D0	001D3	MOVL	EDT\$\$Z_PA_CURRNG, 8(R0)			
				36	11	001D7	23\$:	BRB	36\$		
				0E	DD	001D9	24\$:	PUSHL	#14	1179	
				06	11	001DB	BRB	27\$			
				0F	DD	001DD	25\$:	PUSHL	#15	1183	
				02	11	001DF	BRB	27\$			
				10	DD	001E1	26\$:	PUSHL	#16	1187	
		00000000G	00	01	FB	001E3	27\$:	CALLS	#1, EDT\$\$PA_CRERNGNOD		
				50	D5	001EA	28\$:	TSTL	R0		
				21	12	001EC	BNEQ	36\$			
				0622	31	001EE	29\$:	BRW	156\$		
				02	A7	B5	001F1	30\$:	TSTW	EDT\$\$L_PA_NUMVAL+2	1192
				03	12	001F4	BNEQ	31\$			
				04	A7	B5	001F6	TSTW	EDT\$\$L_PA_NUMVAL+4		
				03	13	001F9	31\$:	BEQL	33\$		
				0587	31	001F9	32\$:	BRW	137\$		
		50		6A	D0	001FE	33\$:	MOVL	EDT\$\$Z_PA_CURRNG, R0	1199	
	04	A0		67	3C	00201	MOVZWL	EDT\$\$L_PA_NUMVAL, 4(R0)			
				08	11	00205	34\$:	BRB	36\$	1066	
		50		6A	D0	00207	35\$:	MOVL	EDT\$\$Z_PA_CURRNG, R0	1203	
04	A0	67		06	28	0020A	MOVc3	#6, EDT\$\$E_PA_NUMVAL, 4(R0)			
				05FD	31	0020F	36\$:	BRW	155\$		
		52	04	A7	3C	00212	37\$:	MOVZWL	HIGH_1, R2	1217	
		50	00000000G	00	3C	00216	MOVZWL	HIGH_2, R0			
		50		52	D1	0021D	CMPL	R2, R0			
				11	1F	00220	BLSSU	38\$			
				1A	12	00222	BNEQ	40\$			
		51		67	D0	00224	MOVL	LOW_1, R1			
		50	00000000G	00	D0	00227	MOVL	LOW_2, R0			
		50		51	D1	0022E	CMPL	R1, -R0			

.....
L B L R

				05	1E	00231	BGEQU	39\$		
		50		01	CE	00233	MNEGL	#1, RO		
				09	11	00236	BRB	41\$		
				04	12	00238	BNEQ	40\$		
				50	D4	0023A	CLRL	RO		
				03	11	0023C	BRB	41\$		
		50		01	D0	0023E	MOVL	#1, RO		
				88	14	00241	BGTR	32\$		
	08	AE		67	D0	00243	MOVL	EDT\$SL_PA_NUMVAL, M2		1224
	0C	AE		52	D0	00247	MOVL	R2, M2+4		
				6E	7C	0024B	CLRQ	P		
		50		10	D0	0024D	MOVL	#16, I		
6E		6E		01	79	00250	ASHQ	#1, P, P		
09	00000000G	00		50	E1	00254	BBC	I, M1, 43\$		
		6E	08	AE	C0	0025C	ADDL2	M2, P		
		04	0C	AE	D8	00260	ADWC	M2, P		
		E8		50	F4	00265	SOBGEO	I, 42\$		
		67		6E	D0	00268	MOVL	P, EDT\$SL_PA_NUMVAL		
		04	04	AE	B0	0026B	MOVW	P+4, EDT\$SL_PA_NUMVAL+4		
		50		68	D0	00270	MOVL	EDT\$SA_PA_CORTOK, RO		1226
		2E		60	91	00273	CMPB	(RO), #46		
				97	12	00276	BNEQ	36\$		
		52		04	D0	00278	MOVL	#4, MULTIPLIER		1229
		50	00000000G	00	D0	0027B	MOVL	EDT\$SC_PA_CH, RO		1234
		F0	8F	00000000G	0040	91	CMPB	EDT\$SB_CHAR_INFO[RO], #240		
				03	13	0028B	BEQL	45\$		
				050F	31	0028D	BRW	141\$		
		10	AE	D0	A0	9E	MOVAB	-48(RO), DIGIT		1241
				14	AE	D4	CLRL	DIGIT+4		
				52	D5	00298	TSTL	MULTIPLIER		1243
				46	19	0029A	BLSS	48\$		
		50		06	C5	0029C	MULL3	#6, MULTIPLIER, RO		1246
		08	AE	10	AE	D0	MOVL	DIGIT, M2		
		0C	AE	14	AE	3C	MOVZWL	DIGIT+4, M2+4		
				6E	7C	002AA	CLRQ	P		
				10	D0	002AC	MOVL	#16, I		
6E		6E		01	79	002AF	ASHQ	#1, P, P		
09	00000000G	0040		51	E1	002B3	BBC	I, EDT\$SL_LNOO[RO], 47\$		
		6E	08	AE	C0	002BC	ADDL2	M2, P		
		04	0C	AE	D8	002C0	ADWC	M2, P		
		E7		51	F4	002C5	SOBGEO	I, 46\$		
		10	AE	6E	D0	002C8	MOVL	P, DIGIT		
		14	AE	04	AE	B0	MOVW	P+4, DIGIT+4		
		50		06	A7	B0	MOVW	UPPER_WORD, SAVE		1247
		67		10	AE	C0	ADDL2	DIGIT, EDT\$SL_PA_NUMVAL		
		04	A7	14	AE	D8	ADWC	DIGIT, EDT\$SL_PA_NUMVAL+4		
		06	A7	50	B0	002DE	MOVW	SAVE, UPPER_WORD		
	00000000G	00		00	FB	002E2	CALLS	#0, EDT\$SPA_GETCH		1250
				52	D7	002E9	DECL	MULTIPLIER		1251
				8E	11	002EB	BRB	44\$		1234
		7E		02	7D	002ED	MOVQ	#2, -(SP)		1262
	00000000G	00		02	FB	002F0	CALLS	#2, EDT\$SPA_NEW_NOD		
	00000000G	00		50	D0	002F7	MOVL	RO, EDT\$SZ_PA_THRURNG		
				03	12	002FE	BNEQ	50\$		
				0510	31	00300	BRW	156\$		
		50	00000000G	00	D0	00303	MOVL	EDT\$SZ_PA_THRURNG, RO		1264
		04	A0	6A	D0	0030A	MOVL	EDT\$SZ_PA_CURRNG, 4(RO)		

			5D	11	0030E		BRB	61\$	1066	
		00000000G	00	D0	00310	52\$:	MOVL	EDT\$\$Z_PA_THRURNG, R0	1269	
01	A0		11	90	00317		MOVB	#17, 1(RO)		
08	A0		6A	D0	0031B		MOVL	EDT\$\$Z_PA_CURRNG, 8(RO)	1270	
	6A		50	D0	0031F	53\$:	MOVL	R0, EDT\$\$Z_PA_CURRNG	1271	
			49	11	00322		BRB	61\$	1066	
00000000G	00		6A	D0	00324	54\$:	MOVL	EDT\$\$Z_PA_CURRNG, EDT\$\$Z_PA_ANDLSTHD	1275	
			40	11	0032B		BRB	61\$		
		00000000G	00	D0	0032D	55\$:	MOVL	EDT\$\$Z_PA_ANDLSTHD, R2	1283	
			52	D0	00334		MOVL	R2, RANGE		
			10	A0	05	00337	56\$:	TSTL	16(RANGE)	1288
			06	13	0033A		BEQL	57\$		
			50	10	0033C		MOVL	16(RANGE), RANGE	1289	
			F5	11	00340		BRB	56\$		
			6A	D0	00342	57\$:	MOVL	EDT\$\$Z_PA_CURRNG, R1	1291	
10	A0		51	D0	00345		MOVL	R1, 16(RANGE)		
14	A1		50	D0	00349		MOVL	RANGE, 20(R1)	1292	
	6A		52	D0	0034D		MOVL	R2, EDT\$\$Z_PA_CURRNG	1293	
			1B	11	00350		BRB	61\$	1066	
			01	08	00352	58\$:	CMPL	OPERAND, #1	1299	
			07	12	00356		BNEQ	59\$		
			50	D0	00358		MOVL	EDT\$\$G_PA_CURCMD, R0		
01	A0		13	90	0035B		MOVB	#19, 1(RO)		
00000000G	00	00000000G	00	E8	0035F	59\$:	BLBS	EDT\$\$G_PA_NOQUO, 61\$	1305	
			00	FB	00366	60\$:	CALLS	#0, EDT\$\$INTER_ERR		
			049F	31	0036D	61\$:	BRW	155\$	1066	
			07	E8	00370	62\$:	BLBS	EDT\$\$G_PA_NOQUO, 63\$	1326	
00000000G	00	00000000G	00	FB	00377		CALLS	#0, EDT\$\$INTER_ERR		
			00	D4	0037E	63\$:	CLRL	EDT\$\$G_PA_NOQUO	1327	
			7E	03	7D	00384		MOVQ	#3, -(SP)	1329
00000000G	00	00000000G	00	FB	00387		CALLS	#2, EDT\$\$PA_NEW_NOD		
			52	50	0038E		MOVL	R0, STRNODE		
			59	13	00391		BEQL	67\$		
			50	D0	00393		MOVL	EDT\$\$G_PA_CURCMD, R0	1331	
08	A0		52	D0	00396		MOVL	STRNODE, 8(RO)		
	02	00000000G	00	D1	0039A		CMPL	EDT\$\$G_PA_TOKCLASS, #2	1333	
			09	13	003A1		BEQL	64\$		
			69	D0	003A3		MOVL	#EDT\$_NONALPNUM, EDT\$\$G_PA_ERRNO	1336	
			40	11	003AA		BRB	67\$	1337	
			51	D0	003AC	64\$:	MOVL	EDT\$\$A_PA_CURTOK, R1	1340	
			53	9A	003AF		MOVZBL	(R1), QUOTE		
			50	01	9E	003B2		MOVAB	1(R1), CURSOR	1341
53	A2		50	D0	003B6		MOVL	CURSOR, 4(STRNODE)	1342	
60	08		00	E0	003BA	65\$:	CMPZV	#0, #8, (CURSOR), QUOTE	1344	
			0D	13	003BF		BEQL	66\$		
			00	D1	003C1		CMPL	CURSOR, EDT\$\$A_CMD_END		
		00000000G	04	1E	003C8		BGEQU	66\$		
			50	D6	003CA		INCL	CURSOR	1345	
			EC	11	003CC		BRB	65\$		
			51	C3	003CE	66\$:	SUBL3	R1, CURSOR, R1	1347	
51	A2		FF	9E	003D2		MOVAB	-1(R1), 8(STRNODE)		
			50	D6	003D7		INCL	CURSOR	1348	
			51	D0	003D9		MOVL	EDT\$\$A_CMD_END, R1	1350	
			51	D1	003E0		CMPL	CURSOR, R1		
			09	1B	003E3		BLEQU	68\$		
			69	D0	003E5		MOVL	#EDT\$_INVSTR, EDT\$\$G_PA_ERRNO	1353	
			41	11	003EC	67\$:	BRB	71\$	1354	

53	60	OC	A2	50	D0	003EE	68\$:	MOVL	CURSOR, 12(STRNODE)	1357	
			08	00	ED	003F2	69\$:	CMPZV	#0, #8, (CURSOR), QUOTE	1359	
				09	13	003F7		BEQL	70\$		
			51	50	D1	003F9		C MPL	CURSOR, R1		
				04	1E	003FC		BGEQU	70\$		
				50	D6	003FE		INCL	CURSOR	1360	
				FO	11	00400		BRB	69\$		
10	A2		50	OC	A2	C3	00402	70\$:	SUBL3	12(STRNODE), CURSOR, 16(STRNODE)	1362
		00000000G	00	01	A0	9E	00408		MOVAB	1(R0), EDT\$\$A_CMD_BUF	1363
		00000000G	00	00	FB	00410		CALLS	#0, EDT\$\$PA_GETCH	1364	
		00000000G	00	00	FB	00417		CALLS	#0, EDT\$\$PA_SCANTOK	1365	
				10	A2	D5	0041E		TSTL	16(STRNODE)	1367
				23	12	00421		BNEQ	74\$		
				08	A2	D5	00423		TSTL	8(STRNODE)	
				1E	12	00426		BNEQ	74\$		
			69	00000000G	8F	D0	00428		MOVL	#EDT\$_SUBSTRNUL, EDT\$\$G_PA_ERRNO	1370
				03E1	31	0042F	71\$:	BRW	156\$	1371	
			07	00000000G	00	EB	00432	72\$:	BLBS	EDT\$\$G_PA_NOQUO, 73\$	1378
		00000000G	00	00	FB	00439		CALLS	#0, EDT\$\$INTER_ERR		
				00000000G	00	D4	00440	73\$:	CLRL	EDT\$\$G_PA_NOQUO	1379
				03C6	31	00446	74\$:	BRW	155\$	1066	
			50		68	D0	00449	75\$:	MOVL	EDT\$\$G_PA_CURCMD, R0	1391
			51		68	D0	0044C		MOVL	EDT\$\$A_PA_CURTOK, R1	
		08	A0	00	51	D0	0044F		MOVL	R1, 8(R0)	
		00000000G	00	00	51	D0	00453		MOVL	R1, EDT\$\$A_CMD_BUF	1392
					53	D4	0045A		CLRL	SCAN_DONE	1393
					52	D4	0045C	76\$:	CLRL	QUOTE_CHAR	1394
			46		53	EB	0045E	77\$:	BLBS	SCAN_DONE, 82\$	1396
		00000000G	00	00000000G	00	D1	00461		C MPL	EDT\$\$A_CMD_BUF, EDT\$\$A_CMD_END	1398
					1E	1A	0046C		BGTRU	78\$	
			54	00000000G	00	D0	0046E		MOVL	EDT\$\$A_CMD_BUF, R4	1403
			51		64	9A	00475		MOVZBL	(R4), CHAR	
				00000000G	00	D6	00478		INCL	EDT\$\$A_CMD_BUF	
					52	D5	0047E		TSTL	QUOTE_CHAR	1405
					1E	12	00480		BNEQ	81\$	
			20		51	D1	00482		C MPL	CHAR, #32	1411
					05	13	00485		BEQL	78\$	
			2F		51	D1	00487		C MPL	CHAR, #47	
					05	12	0048A		BNEQ	79\$	
			53		01	D0	0048C	78\$:	MOVL	#1, SCAN_DONE	1412
					CD	11	0048F		BRB	77\$	
			22		51	D1	00491	79\$:	C MPL	CHAR, #34	1414
					05	13	00494		BEQL	80\$	
			27		51	D1	00496		C MPL	CHAR, #39	
					C3	12	00499		BNEQ	77\$	
			52		51	D0	0049B	80\$:	MOVL	CHAR, QUOTE_CHAR	1415
					BE	11	0049E		BRB	77\$	
			52		51	D1	004A0	81\$:	C MPL	CHAR, QUOTE_CHAR	1425
					B9	12	004A3		BNEQ	77\$	
					B5	11	004A5		BRB	76\$	
		00000000G	00		51	D0	004A7	82\$:	MOVL	CHAR, EDT\$\$C_PA_CH	1429
					2C	11	004AE		BRB	86\$	1430
			50		68	D0	004B0	83\$:	MOVL	EDT\$\$G_PA_CURCMD, R0	1436
			51		68	D0	004B3		MOVL	EDT\$\$A_PA_CURTOK, R1	
		08	A0	00	51	D0	004B6		MOVL	R1, 8(R0)	
		00000000G	00	00	51	D0	004BA		MOVL	R1, EDT\$\$A_CMD_BUF	1437
		00000000G	00	00	00	FB	004C1	84\$:	CALLS	#0, EDT\$\$PA_GETCH	1438

	50	00000000G	00	D0	004C8	HOVL	EDT\$\$C_PA_CH, R0	1440		
	21		50	D1	004CF	CMPL	R0, #33			
			05	13	004D2	BEQL	85\$			
	3B		50	D1	004D4	CMPL	R0, #59			
			E8	12	004D7	BNEQ	84\$			
	50		68	D0	004D9	85\$:	MOVL	EDT\$\$G_PA_CURCMD, R0	1443	
51	00000000G	00	08	A0	C3	004DC	86\$:	SUBL3	8(R0), EDT\$\$A_CMD_BUF, R1	
	OC	A0	FF	A1	9E	004E5		MOVAB	-1(R1), 12(R0)	
			02B2	31	004EA	BRW	141\$			1444
		00000000G	00	D5	004ED	87\$:	TSTL	EDT\$\$G_PA_TOKCLASS		1449
			77	12	004F3	BNEQ	96\$			
	70	00000000G	00	E8	004F5	BLBS	EDT\$\$G_PA_PCEN, 96\$			
	69	00000000G	8F	D0	004FC	MOVL	#EDT\$_ONRCOM, EDT\$\$G_PA_ERRNO			1452
			030D	31	00503	88\$:	BRW	156\$		1453
	50		68	D0	00506	89\$:	MOVL	EDT\$\$G_PA_CURCMD, R0		1462
			14	A0	D5	00509	TSTL	20(R0)		
			17	12	0050C	BNEQ	90\$			
	7E		04	7D	0050E	MOVQ	#4, -(SP)			1466
00000000G	00		02	FB	00511	CALLS	#2, EDT\$\$PA_NEW_NOD			
			50	D5	00518	TSTL	SWITCH_NODE			
			E7	13	0051A	BEQL	88\$			
	51		68	D0	0051C	MOVL	EDT\$\$G_PA_CURCMD, R1			1468
	14	A1	50	D0	0051F	MOVL	SWITCH_NODE, 20(R1)			
			04	11	00523	BRB	91\$			1462
	50		14	A0	D0	00525	90\$:	MOVL	20(R0), SWITCH_NODE	1471
51	01		08	AC	78	00529	91\$:	ASHL	OPERAND, #1, R1	1473
	51		04	A0	D3	0052E		BITL	4(SWITCH_NODE), R1	
			CF	12	00532	BNEQ	88\$			
	04	A0	51	C8	00534	BISL2	R1, 4(SWITCH_NODE)			1475
			76	11	00538	BRB	100\$			1066
	50		68	D0	0053A	92\$:	MOVL	EDT\$\$G_PA_CURCMD, R0		1482
	56		14	A0	D0	0053D		MOVL	20(R0), R5	
08	A6		06	28	00541	MOVC3	#6, EDT\$\$L_PA_NUMVAL, 8(R6)			1484
	01	A6	01	90	00546	MOVB	#1, 1(R6)			1485
			64	11	0054A	BRB	100\$			1066
	50		68	D0	0054C	93\$:	MOVL	EDT\$\$G_PA_CURCMD, R0		1492
	50		14	AC	D0	0054F		MOVL	20(R0), R0	
14	A0		06	28	00553	MOVC3	#6, EDT\$\$L_PA_NUMVAL, 20(R0)			1494
			56	11	00558	BRB	100\$			1066
	50		68	D0	0055A	94\$:	MOVL	EDT\$\$G_PA_CURCMD, R0		1498
	04	A0	08	AC	D0	0055D		MOVL	OPERAND, 4(R0)	
			4C	11	00562	BRB	100\$			
	50		68	D0	00564	95\$:	MOVL	EDT\$\$G_PA_CURCMD, R0		1501
	10	A0	08	AC	D0	00567		MOVL	OPERAND, 16(R0)	
			0080	31	0056C	96\$:	BRW	107\$		
	7FFF	51	67	3C	0056F	97\$:	MOVZWL	EDT\$\$L_PA_NUMVAL, R1		1506
		8F	51	B1	00572		CMPW	R1, #32767		
			56	1A	00577	BGTRU	103\$			
			02	A7	B5	00579	TSTW	EDT\$\$L_PA_NUMVAL+2		1507
			51	12	0057C	BNEQ	103\$			
			04	A7	B5	0057E	TSTW	EDT\$\$L_PA_NUMVAL+4		1508
			4C	12	00581	BNEQ	103\$			
	50		68	D0	00583	MOVL	EDT\$\$G_PA_CURCMD, R0			1515
	10	A0	51	D0	00586	MOVL	R1, 16(R0)			
			6D	11	0058A	BRB	109\$			1066
	7FFF	51	67	3C	0058C	98\$:	MOVZWL	EDT\$\$L_PA_NUMVAL, R1		1521
		8F	51	B1	0058F		CMPW	R1, #32767		

			39	1A	00594	BGTRU	103\$		
		02	A7	B5	00596	TSTW	EDT\$SL_PA_NUMVAL+2		1522
			34	12	00599	BNEQ	103\$		
		04	A7	B5	0059B	TSTW	EDT\$SL_PA_NUMVAL+4		1523
			2F	12	0059E	BNEQ	103\$		
	50		68	D0	005A0	MOVL	EDT\$SG_PA_CURCMD, RO		1530
	OC	A0	51	D0	005A3	MOVL	R1, 12(R0)		
			50	11	005A7	BRB	109\$		1066
	00000000G	00	01	D0	005A9	99\$: 100\$: 101\$: MOVL	#1, EDT\$SG_DEFKEY		1535
			47	11	005B0	BRB	109\$		1066
			68	D0	005B2	101\$: MOVZWL	EDT\$SG_PA_CURCMD, RO		1540
	50		67	3C	005B5	MOVZWL	EDT\$SL_PA_NUMVAL, R1		
	10	A0	012C	C1	9E	005B8	MOVAB	300(R1), T6(R0)	
	7FFF	8F		51	B1	005BE	102\$: CMPW	R1, #32767	1542
				0A	1A	005C3	BGTRU	103\$	
			02	A7	B5	005C5	TSTW	EDT\$SL_PA_NUMVAL+2	1543
				05	12	005C8	BNEQ	103\$	
			04	A7	B5	005CA	TSTW	EDT\$SL_PA_NUMVAL+4	1544
				11	13	005CD	BEQL	105\$	
			01B3	31	005CF	103\$: BRW	137\$		1547
	50		68	D0	005D2	104\$: MOVL	EDT\$SG_PA_CURCMD, RO		1562
	51		67	3C	005D5	MOVZWL	EDT\$SL_PA_NUMVAL, R1		
	10	A0	0320	C1	9E	005D8	MOVAB	800(R1), T6(R0)	
				DE	11	005DE	BRB	102\$	1564
				51	B1	005E0	105\$: CMPW	R1, #21	1573
			0107	31	005E3	BRW	127\$		
	50		68	D0	005E6	106\$: MOVL	EDT\$SG_PA_CURCMD, RO		1584
	10	A0	0273	8F	3C	005E9	MOVZWL	#627, T6(R0)	
				08	11	005EF	107\$: BRB	109\$	1066
	50		68	D0	005F1	108\$: MOVL	EDT\$SG_PA_CURCMD, RO		1589
	10	A0	7F	8F	9A	005F4	MOVZBL	#127, T6(R0)	
			0213	31	005F9	109\$: BRW	155\$		1066
	50		68	D0	005FC	110\$: MOVL	EDT\$SA_PA_CURTOK, RO		1604
	51		60	9A	005FF	MOVZBL	(RO), CHAR		
	50		68	D0	00602	MOVL	EDT\$SG_PA_CURCMD, RO		1605
	10	A0	01F4	C1	9E	00605	MOVAB	500(R1), T6(R0)	
		01	00000000G	00	D1	00608	CMPL	EDT\$SG_PA_CURTOKLEN, #1	1607
				51	12	00612	BNEQ	115\$	
	F0	8F	00000000G00	41	91	00614	CMPB	EDT\$SB_CHAR_INFO[CHAR], #240	1611
				27	13	0061D	BEQL	112\$	
				51	D1	0061F	CMPL	CHAR, #32	1616
	000000FF	8F		7B	19	00622	BLSS	118\$	
				51	D1	00624	CMPL	CHAR, #255	1617
	00000080	8F		7B	14	0062B	BGTR	119\$	
				51	D1	0062D	CMPL	CHAR, #128	1618
	000000A0	8F		09	19	00634	BLSS	111\$	
				51	D1	00636	CMPL	CHAR, #160	
	0000007F	8F		79	19	0063D	BLSS	121\$	
				51	D1	0063F	111\$: CMPL	CHAR, #127	1619
				70	13	00646	112\$: BEQL	121\$	
			0154	31	00648	113\$: BRW	141\$		
	50		68	D0	0064B	114\$: MOVL	EDT\$SA_PA_CURTOK, RO		1635
	51		60	9A	0064E	MOVZBL	(RO), CHAR		
	51		C0	A1	9E	00651	MOVAB	-64(R1), CHAR	
	50		68	D0	00655	MOVL	EDT\$SG_PA_CURCMD, RO		1636
	10	A0	01F4	C1	9E	00658	MOVAB	500(R1), T6(R0)	
		C1	00000000G	00	D1	0065E	CMPL	EDT\$SG_PA_CURTOKLEN, #1	1638

			51	12	00665	115\$:	BNEQ	121\$		
			51	D5	00667		TSTL	CHAR		1639
			4D	19	00669		BLSS	121\$		
	000000FF	8F	51	D1	0066B		CMPL	CHAR, #255		1640
			7B	14	00672		BGTR	128\$		
		20	51	D1	00674		CMPL	CHAR, #32		1641
			36	19	00677		BLSS	120\$		
	00000080	8F	51	D1	00679	116\$:	CMPL	CHAR, #128		
			6D	19	00680		BLSS	128\$		
			2B	11	00682		BRB	120\$		1642
		50	6B	D0	00684	117\$:	MOVL	EDT\$PA_CURTOK, R0		1658
			51	60	9A	00687	MOVZBL	(R0), CHAR		
			51	38	C2	0068A	SUBL2	#56, CHAR		
			50	68	D0	0068D	MOVL	EDT\$G_PA_CURCMD, R0		1659
	10	A0	71	7E	00690		MOVAQ	-(CHAR), T6(R0)		
		01	00	D1	00694		CMPL	EDT\$G_PA_CURTOKLEN, #1		1661
			52	12	0069B		BNEQ	128\$		
			51	D5	0069D		TSTL	CHAR		1662
	000000FF	8F	4E	19	0069F	118\$:	BLSS	128\$		1663
			51	D1	006A1		CMPL	CHAR, #255		
			45	14	006A8	119\$:	BGTR	128\$		
		20	51	D1	006AA		CMPL	CHAR, #32		1664
			CA	18	006AD		BGEQ	116\$		
	000000A0	8F	51	D1	006AF	120\$:	CMPL	CHAR, #160		1665
			90	19	006B6		BLSS	113\$		
			35	11	006B8	121\$:	BRB	128\$		1668
		50	68	D0	006BA	122\$:	MOVL	EDT\$G_PA_CURCMD, R0		1677
			51	67	3C	006BD	MOVZWL	EDT\$SL_PA_NUMVAL, R1		
	10	A0	0190	C1	9E	006C0	MOVAB	400(R1), T6(R0)		
		7FFF	8F	51	B1	006C6	123\$:	CMPW	R1, #32767	1679
			0A	1A	006CB		BGTRU	124\$		
			02	A7	B5	006CD	TSTW	EDT\$SL_PA_NUMVAL+2		1680
			05	12	006D0		BNEQ	124\$		
			04	A7	B5	006D2	TSTW	EDT\$SL_PA_NUMVAL+4		1681
			11	13	006D5		BEQL	126\$		
			00AB	31	006D7	124\$:	BRW	137\$		1684
		50	68	D0	006DA	125\$:	MOVL	EDT\$G_PA_CURCMD, R0		1699
			51	67	3C	006DD	MOVZWL	EDT\$SL_PA_NUMVAL, R1		
	10	A0	0384	C1	9E	006E0	MOVAB	900(R1), T6(R0)		
			DE	11	006E6		BRB	123\$		1701
	0063	8F	51	B1	006E8	126\$:	CMPW	R1, #99		1710
			49	1B	006ED	127\$:	BLEQU	132\$		
		69	00000000G	8F	D0	006EF	128\$:	MOVL	#EDT\$_KEYNOTDEF, EDT\$G_PA_ERRNO	1713
				66	11	006F6		BRB	135\$	1714
		50	68	D0	006F8	129\$:	MOVL	EDT\$G_PA_CURCMD, R0		1721
08	A0	00000000G	00	01	C1	006FB	ADDL3	#1, EDT\$PA_PRIVTOK, 8(R0)		
		0C	A0	00000000G	00	D0	00704	MOVL	EDT\$G_PA_PRIVTOKLEN, 12(R0)	1722
			2A	11	0070C		BRB	132\$		1066
		50	68	D0	0070E	130\$:	MOVL	EDT\$G_PA_CURCMD, R0		1729
		56	14	A0	D0	00711	MOVL	20(R0), R6		
08	A6	00000000G	00	06	28	00715	MOVC3	#6, EDT\$SL_LN00+30, 8(R6)		1731
14	A6	00000000G	00	06	28	0071E	MOVC3	#6, EDT\$SL_LN00+30, 20(R6)		1732
			01	A6	94	00727	CLRB	1(R6)		1733
			0082	31	0072A		BRW	143\$		1066
		50	68	D0	0072D	131\$:	MOVL	EDT\$G_PA_CURCMD, R0		1738
		04	A0	6A	D0	00730	MOVL	EDT\$Z_PA_CURRNG, 4(R0)		
		01	A0	18	90	00734	MOVW	#24, 1(R0)		1739

		0086	31	00738	132\$:	BRW	146\$		1066
		52	D4	0073B	133\$:	CLRL	NEG		1748
50		6B	D0	0073D		MOVL	EDT\$PA_CURTOK, R0		1750
2D		60	91	00740		CMPB	(R0), #45		
		09	12	00743		BNEQ	134\$		
		52	D6	00745		INCL	NEG		1753
0000000G	00	00	FB	00747		CALLS	#0, EDT\$SPA_SCANTOK		1754
	01	0000000G	00	D1	0074E	134\$:	C MPL	EDT\$G_PA_TOKCLASS, #1	1757
		09	13	00755		BEQL	136\$		
	69	0000000G	8F	D0	00757		MOVL	#EDT\$_NUMVALREQ, EDT\$G_PA_ERRNO	1760
		2C	11	0075E	135\$:	BRB	138\$		1761
	51		67	3C	00760	136\$:	MOVZWL	EDT\$SL_PA_NUMVAL, R1	1764
7FFF	8F		51	B1	00763		CMPW	R1, #32767	
			1B	1A	00768		BGTRU	137\$	
		02	A7	B5	0076A		TSTW	EDT\$SL_PA_NUMVAL+2	1765
			16	12	0076D		BNEQ	137\$	
		04	A7	B5	0076F		TSTW	EDT\$SL_PA_NUMVAL+4	1766
			11	12	00772		BNEQ	137\$	
50	51	0000000G	00	C5	00774		MULL3	EDT\$G_TAB_SIZ, R1, R0	1773
	8F		50	D1	0077C		C MPL	R0, #255	
			0A	15	00783		BLEQ	139\$	
	69	0000000G	8F	D0	00785	137\$:	MOVL	#EDT\$_NUMVALILL, EDT\$G_PA_ERRNO	1776
			0084	31	0078C	138\$:	BRW	156\$	1777
	50		68	D0	0078F	139\$:	MOVL	EDT\$G_PA_CURCMD, R0	1782
	06		52	E9	00792		BLBC	NEG, 140\$	1780
	08	A0	51	CE	00795		MNEGL	R1, 8(R0)	1782
			04	11	00799		BRB	141\$	
	08	A0	51	D0	0079B	140\$:	MOVL	R1, 8(R0)	1784
	0000000G	00	00	FB	0079F	141\$:	CALLS	#0, EDT\$SPA_SCANTOK	1786
			67	11	007A6		BRB	155\$	1066
	69	0000000G	8F	D0	007A8	142\$:	MOVL	#EDT\$_INVPARFOR, EDT\$G_PA_ERRNO	1790
			5E	11	007AF	143\$:	BRB	155\$	
	69	0000000G	8F	D0	007B1	144\$:	MOVL	#EDT\$_INVVALSET, EDT\$G_PA_ERRNO	1793
			55	11	007B8		BRB	155\$	
	69	0000000G	8F	D0	007BA	145\$:	MOVL	#EDT\$_NUMVALREQ, EDT\$G_PA_ERRNO	1796
			4C	11	007C1	146\$:	BRB	155\$	
	69	0000000G	8F	D0	007C3	147\$:	MOVL	#EDT\$_QUOSTRREQ, EDT\$G_PA_ERRNO	1799
			43	11	007CA		BRB	155\$	
	69	0000000G	8F	D0	007CC	148\$:	MOVL	#EDT\$_ERRRANSPC, EDT\$G_PA_ERRNO	1802
			3A	11	007D3		BRB	155\$	
	69	0000000G	8F	D0	007D5	149\$:	MOVL	#EDT\$_ERRCOMOPT, EDT\$G_PA_ERRNO	1805
			31	11	007DC		BRB	155\$	
	69	0000000G	8F	D0	007DE	150\$:	MOVL	#EDT\$_UNRCOMOPT, EDT\$G_PA_ERRNO	1808
			28	11	007E5		BRB	155\$	
	69	0000000G	8F	D0	007E7	151\$:	MOVL	#EDT\$_COLONREQ, EDT\$G_PA_ERRNO	1811
			1F	11	007EE		BRB	155\$	
	69	0000000G	8F	D0	007F0	152\$:	MOVL	#EDT\$_MACKEYREQ, EDT\$G_PA_ERRNO	1814
			16	11	007F7		BRB	155\$	
	69	0000000G	8F	D0	007F9	153\$:	MOVL	#EDT\$_ENTMUSTBE, EDT\$G_PA_ERRNO	1817
			0D	11	00800		BRB	155\$	
		0000000G	00	D4	00802	154\$:	CLRL	EDT\$G_DEFKEY	1821
	69	0000000G	8F	D0	00808		MOVL	#EDT\$_ASREQ, EDT\$G_PA_ERRNO	1822
	50		01	D0	0080F	155\$:	MOVL	#1, R0	1832
				04	00812		RET		
			50	D4	00815	156\$:	CLRL	R0	1833
				04	00815		RET		

EDT\$PRSEMRTN
V04-000

EDT\$PRSEMRTN - parser semantic actions
EDT\$SPA_SEMRUT - parser semantic actions

F 7
16-Sep-1984 01:23:05
14-Sep-1984 12:24:15

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

Page 31

: 984 1836 1 END
: 985 1837 1
: 986 1838 0 ELUDOM

: of module EDT\$PRSEMRTN

PSECT SUMMARY

Name Bytes Attributes
_EDT\$CODE 2070 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	82	21	40	00:00.2
_\$255\$DUA28:[EDT.SRC]PSECTS.L32;1	2	1	50	7	00:00.1
_\$255\$DUA28:[EDT.SRC]KEYPADDEF.L32;1	34	4	11	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\$:PRSEMRTN/OBJ=OBJ\$:PRSEMRTN MSRCS\$:PRSEMRTN.BLI/UPDATE=(ENH\$:PRSEMRTN)

: Size: 2070 code + 0 data bytes
: Run Time: 01:33.4
: Elapsed Time: 01:51.7
: Lines/CPU Min: 1180
: Lexemes/CPU-Min: 7943
: Memory Used: 563 pages
: Compilation Complete

0138

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

A grid of 10 columns and 12 rows of small, faded text screens, likely representing various system utilities or diagnostic tools. The screens contain various alphanumeric strings, some of which are legible and include the following labels:

- PRNUMRAN LIS
- PRPARCOMN LIS
- PRMACCAL LIS
- PRPARCOM LIS
- PRPARDRV LIS
- PRSEMRTN LIS
- REAJOUTEX LIS
- PSECTS LIS
- RANRPOS LIS
- RANNEXT LIS
- SAUDIT LIS
- SCRCHKREV LIS
- SCRCLURS LIS
- SCRDELETE LIS
- SCRRESOR LIS
- SCRCONCUR LIS
- SCRLOB LIS
- SCRRELINE LIS
- SCRFCURS LIS
- SCRFINO LIS

Each screen also contains various graphical elements such as histograms, bar charts, and lists of data points, though they are too small to read clearly.