

EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT


```
0001 0 %TITLE 'EDT$CHMPARSE - change mode parse and execute'
0002 0 MODULE EDT$CHMPARSE ( ! Change mode parse and execute
0003 0 IDENT = 'V04-000' ! File: CHMPARSE.BLI Edit: JBS1023
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 This module parses and executes a change mode command string.
0037 1
0038 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0039 1
0040 1 AUTHOR: Bob Kushlis, CREATION DATE: Unknown
0041 1
0042 1 MODIFIED BY:
0043 1
0044 1 1-001 - Original. DJS 04-Feb-1981. This module was created by
0045 1 extracting the routine EDT$SCHM_PAREXE from module CHANGE.BLI.
0046 1 1-002 - Regularize headers. JBS 03-Mar-1981
0047 1 1-003 - Use new message codes. JBS 04-Aug-1981
0048 1 1-004 - Add the bell verb. STS 11-Aug-1981
0049 1 1-005 - Add the date verb. STS 31-Aug-1981
0050 1 1-006 - Add the setting of default verb. STS 21-Sep-1981
0051 1 1-007 - Add desel and tgsel. STS 23-Sep-1981
0052 1 1-008 - Add ssel verb. STS 24-Sep-1981
0053 1 1-009 - Add a return value for journal file ended. JBS 02-Oct-1981
0054 1 1-010 - Revise direction setting for tadj. SMB 22-Oct-1981
0055 1 1-011 - Give an error message on an invalid repeat count. JBS 10-Mar-1982
0056 1 1-012 - Add NOOVERLAY symbol. JBS 04-Apr-1982
0057 1 1-013 - Set a flag if control C actually aborts something. JBS 24-May-1982
```

: R

:

EDT\$CHMPARSE
V04-000

EDT\$CHMPARSE - change mode parse and execute

N 5
16-Sep-1984 00:06:35
14-Sep-1984 12:22:41

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

```
: 58      0058 1 1-014 - Change alphabetic test. JBS 19-Jul-1982
: 59      0059 1 1-015 - Add the xlate verb. STS 13-Aug-1982
: 60      0060 1 1-016 - Add the CLSS verb. STS 26-Aug-1982
: 61      0061 1 1-017 - Change to using an alphabetically ordered table. STS 21-Sep-1982
: 62      0062 1 1-018 - Move the table into the keyword routine, to reduce the program size
: 63      0063 1 on the PDP-11. JBS 29-Sep-1982
: 64      0064 1 1-019 - Allow lower case commands. JBS 01-Oct-1982
: 65      0065 1 1-020 - Check the length of the command. STS 08-Oct-1982
: 66      0066 1 1-021 - Add conditional for VT220 support. JBS 11-Feb-1983
: 67      0067 1 1-022 - Add conditional for WPS support. JBS 13-Feb-1983
: 68      0068 1 1-023 - Correct a typo. JBS 22-Feb-1983
: 69      0069 1 --
: 70      0070 1
```

EDT
V04

SR
L
M
C

EDT\$CHMPARSE
V04-000

EDT\$CHMPARSE - change mode parse and execute
Declarations

B 6
16-Sep-1984 00:06:35
14-Sep-1984 12:22:41

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

```

: 72 0071 1 %SBTTL 'Declarations'
: 73 0072 1
: 74 0073 1 : TABLE OF CONTENTS:
: 75 0074 1 :
: 76 0075 1
: 77 0076 1 REQUIRE 'EDTSRC:TRAROUNAM';
: 78 0515 1
: 79 0516 1 FORWARD ROUTINE
: 80 0517 1 EDT$$CHM_PAREXE; ! Parse and execute a change mode command string
: 81 0518 1
: 82 0519 1 :
: 83 0520 1 : INCLUDE FILES:
: 84 0521 1 :
: 85 0522 1
: 86 0523 1 REQUIRE 'EDTSRC:EDTREQ';
: 87 0658 1
: 88 0659 1 LIBRARY 'EDTSRC:SUPPORTS';
: 89 0660 1
: 90 0661 1 :
: 91 0662 1 : MACROS:
: 92 0663 1 :
: 93 0664 1 : NONE
: 94 0665 1 :
: 95 0666 1 : EQUATED SYMBOLS:
: 96 0667 1 :
: 97 0668 1
: 98 0669 1 GLOBAL BIND
: 99 0670 1 ROUTINE
: 100 0671 1 EDT$$CHM_PAREXE_NOOVERLAY_REF = EDT$$CHM_PAREXE; ! Alt ref point
: 101 0672 1
: 102 0673 1 :
: 103 0674 1 : OWN STORAGE:
: 104 0675 1 :
: 105 0676 1 : NONE
: 106 0677 1 :
: 107 0678 1 : EXTERNAL REFERENCES:
: 108 0679 1 :
: 109 0680 1 : In the routine
```

```
111 0681 1 %SBTTL 'EDT$$SCHM_PAREXE - change mode parse and execute'
112 0682 1
113 0683 1 GLOBAL ROUTINE EDT$$SCHM_PAREXE (          ! Change mode parse and execute
114 0684 1 EXECUTE                                ! 1 = execute
115 0685 1 ) =
116 0686 1
117 0687 1 ++
118 0688 1 FUNCTIONAL DESCRIPTION:
119 0689 1
120 0690 1 This routine parses and executes a change mode command string.
121 0691 1
122 0692 1 FORMAL PARAMETERS:
123 0693 1
124 0694 1 EXECUTE - 1 if we should execute the command, 0 to parse only.
125 0695 1
126 0696 1 IMPLICIT INPUTS:
127 0697 1
128 0698 1 EDT$$G_DIR_MOD
129 0699 1 EDT$$A_CMD_END
130 0700 1 EDT$$G_EXI
131 0701 1 EDT$$A_CMD_BUF
132 0702 1 EDT$$G_DFLT_VERB
133 0703 1
134 0704 1 IMPLICIT OUTPUTS:
135 0705 1
136 0706 1 EDT$$G_DIR
137 0707 1 EDT$$A_ALT_BUF
138 0708 1 EDT$$G_TADJ
139 0709 1 EDT$$G_CC_DONE
140 0710 1
141 0711 1 RETURNED VALUE:
142 0712 1
143 0713 1 0 - Command failed or control C
144 0714 1 1 - Command succeeded
145 0715 1 2 - Journal file ended
146 0716 1
147 0717 1 SIDE EFFECTS:
148 0718 1
149 0719 1 Many
150 0720 1
151 0721 1 --
152 0722 1
153 0723 2 BEGIN
154 0724 2
155 0725 2 EXTERNAL ROUTINE
156 0726 2 EDT$$EXE_CHMCMD2,          ! Execute the verbs which take an entity specification
157 0727 2 EDT$$EXE_CHMCMD1,          ! Execute the verbs which do not take an entity specification
158 0728 2 EDT$$MSG_BELL : NOVALUE,    ! Output a message to the terminal with a warning bell
159 0729 2 EDT$$CHK_CC,                ! Check to see if a CTRL/C has been typed
160 0730 2 EDT$$GET_BUF : NOVALUE,    ! Look for a buffer spec for the cut, paste, and append commands
161 0731 2 EDT$$CHK_CNT,              ! Look for a repeat/entity count
162 0732 2 EDT$$CHK_DIR,              ! Determine the current direction
163 0733 2 EDT$$SCAN_SEASTR : NOVALUE, ! Scan over the search string when used as an entity
164 0734 2 EDT$$FND_SUBSTR : NOVALUE, ! Isolate the search and replace strings for the SUBSTITUTE command
165 0735 2 EDT$$SCAN_INSSTR : NOVALUE, ! Scan over the string of characters to be inserted
166 0736 2 EDT$$KEY_WORD,             ! Compare the command buffer contents to a table of keywords
167 0737 2 EDT$$PARENT,              ! Collect and execute a parenthesized string of commands
```



```
168 0738 2      EDT$$CNV_UPC;          ! Convert string to upper case
169 0739 2
170 0740 2      EXTERNAL
171 0741 2      EDT$$G_SEA_LEN,          ! length of string
172 0742 2      EDT$$G_DIR,             ! The current direction.
173 0743 2      EDT$$G_DIR_MOD,         ! The directional mode.
174 0744 2      EDT$$A_CMD_END,         ! End of command pointer
175 0745 2      EDT$$G_EXI,             ! Change mode has been exited.
176 0746 2      EDT$$A_CMD_BUF,         ! Command string pointer
177 0747 2      EDT$$A_ALT_BUF : REF TBCB_BLOCK, ! Alternate buffer used for cut/paste.
178 0748 2      EDT$$G_TADJ,           ! Count for tabs adjust.
179 0749 2
180 L 0750 2      %IF SUPPORT_WPS
181 0751 2      %THEN
182 0752 2      EDT$$G_DFLT_VERB,       ! Default verb
183 0753 2      %FI
184 0754 2
185 L 0755 2      %IF SUPPORT_VT220
186 0756 2      %THEN
187 0757 2      EDT$$B_CHAR_INFO : BLOCKVECTOR [256, 1, BYTE], ! Information about characters
188 0758 2      %FI
189 0759 2
190 0760 2      EDT$$G_CC_DONE;         ! Set to 1 if control C actually aborted something
191 0761 2
192 0762 2      MESSAGES ((INVENT, INVSUBCOM, NUMVALILL));
193 0763 2
194 0764 2      LOCAL
195 0765 2      REPT_COUNT,
196 0766 2      EXPLICIT_REPEAT,
197 0767 2      VERB,
198 0768 2      ENTITY,
199 0769 2      ENTITY_COUNT,
200 0770 2      EXPLICIT_ENTITY_COUNT,
201 0771 2      OPERAND,
202 0772 2      DIR_SAME,
203 0773 2      TADJ_DIR,
204 0774 2      SUCCEED;
205 0775 2
206 0776 2      !+
207 0777 2      !- Loop until we hit the end of the command buffer or a closing parenthesis.
208 0778 2      !-
209 0779 2
210 0780 2      WHILE CH$PTR_GTR (.EDT$$A_CMD_END, .EDT$$A_CMD_BUF) DO
211 0781 2      BEGIN
212 0782 2
213 0783 2      IF EDT$$CHK_CC ( )
214 0784 2      THEN
215 0785 2      BEGIN
216 0786 2      EDT$$G_CC_DONE = 1;
217 0787 2      RETURN (0);
218 0788 2      END;
219 0789 2
220 0790 2      !+
221 0791 2      !- Skip blanks at beginning of command.
222 0792 2      !-
223 0793 2
224 0794 2      WHILE (CH$RCHAR (.EDT$$A_CMD_BUF) EQL %C' ') DO
```



```
225 0795 3          EDT$$A_CMD_BUF = CH$PLUS (.EDT$$A_CMD_BUF, 1);
226 0796 3
227 0797 3
228 0798 3      + Look for closing paren.
229 0799 3      -
230 0800 3
231 0801 3          IF (CH$RCHAR (.EDT$$A_CMD_BUF) EQL %C')') THEN EXITLOOP;
232 0802 3
233 0803 3      +
234 0804 3      Set default direction, zero the alternate buffer and zero the
235 0805 3      direction change indicator.
236 0806 3      -
237 0807 3          EDT$$G_DIR = .EDT$$G_DIR_MOD;
238 0808 3          EDT$$A_ALT_BUF = 0;
239 0809 3      +
240 0810 3      Check for explicit direction specified.
241 0811 3      -
242 0812 3          DIR_SAME = EDT$$CHK_DIR ();
243 0813 3      +
244 0814 3      Look for a count, and remember whether one was seen.
245 0815 3      Give an error message and bail out if the count is invalid.
246 0816 3      (An invalid count means a count greater than 32767.)
247 0817 3      -
248 0818 3          REPT_COUNT = 0;
249 0819 3          EXPLICIT_REPEAT = EDT$$CHK_CNT (REPT_COUNT);
250 0820 3
251 0821 4          IF (.EXPLICIT_REPEAT EQL 2)
252 0822 3          THEN
253 0823 4              BEGIN
254 0824 4                  EDT$$MSG_BELL (EDT$_NUMVALILL);
255 0825 4                  RETURN (0);
256 0826 3              END;
257 0827 3
258 0828 3      +
259 0829 3      Look for the loop.
260 0830 3      -
261 0831 3
262 0832 4          IF (CH$RCHAR (.EDT$$A_CMD_BUF) EQL %C'(')
263 0833 3          THEN
264 0834 4              BEGIN
265 0835 4
266 0836 4                  IF ((.REPT_COUNT EQL 0) AND (.EXPLICIT_REPEAT EQL 0)) THEN REPT_COUNT = 1;
267 0837 4
268 0838 4                  SUCCEED = EDT$$PARENT (.REPT_COUNT, .EXECUTE);
269 0839 4
270 0840 4                  IF (.SUCCEED NEQ 1) THEN RETURN (.SUCCEED);
271 0841 4
272 0842 4              END
273 0843 3          ELSE
274 0844 4              BEGIN
275 0845 4      +
276 0846 4      And now, find the verb of the command.
277 0847 4      -
278 0848 4                  EDT$$KEY_WORD (1, VERB);
279 0849 4
280 L 0850 4 %IF SUPPORT_WPS
281 0851 4 %THEN
```

```
282 0852 4
283 0853 4
284 0854 4 IF (.VERB EQL 0) THEN VERB = .EDT$$G_DFLT_VERB;
285 0855 4 %FI
286 0856 4
287 0857 4 !+
288 0858 4 If the command is tabs adjust, then the [+!-] preceding the repeat count is
289 0859 4 used to determine adjustment of the entities right or left. The global
290 0860 4 direction (ADV or BACK) is not affected by this tab adjust direction.
291 0861 4 !-
292 0862 4
293 0863 5 IF (.VERB EQL VERB_K_TADJ)
294 0864 4 THEN
295 0865 5 BEGIN
296 0866 5 TADJ_DIR = 1;
297 0867 5 !+
298 0868 5 If the adjustment direction has explicitly been set by '-', then tab
299 0869 5 adjust to the left.
300 0870 5 !-
301 0871 5
302 0872 5 IF ((.EDT$$G_DIR EQL DIR_BACKWARD) AND (.DIR_SAME EQL 0)) THEN TADJ_DIR = -1;
303 0873 5
304 0874 5 EDT$$G_DIR = .EDT$$G_DIR_MOD;
305 0875 4 END;
306 0876 4
307 0877 4 !+
308 0878 4 This hack prevents the entities SEN and SR from being assumed
309 0879 4 to be a SUBSTITUTE command.
310 0880 4 !-
311 0881 4
312 0882 5 IF (.VERB EQL VERB_K_SUBS)
313 0883 4 THEN
314 0884 4
315 L 0885 4 %IF SUPPORT_VT220
316 0886 4 %THEN
317 0887 4
318 0888 5 IF (.EDT$$B_CHAR_INFO [CH$RCHAR (.EDT$$A_CMD_BUF), 0, 0, 2, 0] NEQ 0)
319 0889 4 THEN
320 U 0890 4 %ELSE
321 UU 0891 4
322 UU 0892 4 IF (((CH$RCHAR (.EDT$$A_CMD_BUF) GEQ %C'A') AND (CH$RCHAR (.EDT$$A_CMD_BUF) LEQ %C'Z'))
323 UU 0893 4 OR ((CH$RCHAR (.EDT$$A_CMD_BUF) GEQ %C'a') AND (CH$RCHAR (.EDT$$A_CMD_BUF) LEQ %C'z'))
324 UU 0894 4 ))
325 U 0895 4 THEN
326 0896 4 %FI
327 0897 4
328 0898 5 BEGIN
329 0899 5
330 L 0900 5 %IF SUPPORT_WPS
331 0901 5 %THEN
332 0902 5 VERB = .EDT$$G_DFLT_VERB;
333 U 0903 5 %ELSE
334 U 0904 5 VERB = 0;
335 0905 5 %FI
336 0906 5
337 0907 5 EDT$$A_CMD_BUF = CH$PLUS (.EDT$$A_CMD_BUF, -1);
338 0908 4 END;
```



```
339 0909 4
340 0910 4
341 0911 4 + Check to see if it is truly a verb which uses an entity specification.
342 0912 4 -
343 0913 4
344 0914 5 IF (.VERB LSS VERB_K_SEL)
345 0915 4 THEN
346 0916 5 BEGIN
347 0917 5
348 0918 5 + And once again, look for the explicit direction.
349 0919 5 -
350 0920 5 DIR_SAME = EDT$$CHK_DIR ();
351 0921 5
352 0922 5 + Set the entity count to the default of 1, then see if there
353 0923 5 was a count specified.
354 0924 5 -
355 0925 5 ENTITY_COUNT = 1;
356 0926 5 EXPLICIT_ENTITY_COUNT = EDT$$CHK_CNT (ENTITY_COUNT);
357 0927 5
358 0928 6 IF (.EXPLICIT_ENTITY_COUNT EQL 2)
359 0929 5 THEN
360 0930 6 BEGIN
361 0931 6 EDT$$MSG_BELL (EDT$_NUMVALILL);
362 0932 6 RETURN (0);
363 0933 5 END;
364 0934 5
365 0935 5 +
366 0936 5 + And try for a direction once again.
367 0937 5 -
368 0938 5 DIR_SAME = EDT$$CHK_DIR ();
369 0939 5
370 0940 5 + Now get the entity.
371 0941 5 -
372 0942 5
373 0943 6 IF CH$PTR_GTR (.EDT$$A_CMD_END, .EDT$$A_CMD_BUF)
374 0944 5 THEN
375 0945 5
376 0946 5 IF (CH$RCHAR (.EDT$$A_CMD_BUF) EQL '"') OR !
377 0947 5 (CH$RCHAR (.EDT$$A_CMD_BUF) EQL ' ') OR !
378 0948 6 (CH$RCHAR (.EDT$$A_CMD_BUF) EQL 0)
379 0949 5 THEN
380 0950 6 BEGIN
381 0951 6 ENTITY = ENT_K_QUOTE;
382 0952 6 EDT$$A_CMD_BUF = .EDT$$A_CMD_BUF + 1;
383 0953 6 END
384 0954 5 ELSE
385 0955 5 EDT$$KEY_WORD (2, ENTITY)
386 0956 5
387 0957 5 ELSE
388 0958 5 ENTITY = 0;
389 0959 5
390 0960 5 +
391 0961 5 + If entity was not found, something is wrong.
392 0962 5 -
393 0963 5
394 0964 6 IF (.ENTITY EQL 0)
395 0965 5 THEN
```

```
396 0966 5 | +
397 0967 5 | If verb was defaulted to move, assume it was wrong, otherwise
398 0968 5 | it must be the entity which is wrong.
399 0969 5 | -
400 0970 6 |
401 0971 6 | BEGIN
402 0972 6 |
403 0973 6 | IF (.VERB NEQ 0) THEN EDT$$MSG_BELL (EDT$_INVENT) ELSE EDT$$MSG_BELL (EDT$_INVSUBCOM);
404 0974 6 |
405 0975 5 | RETURN (0);
406 0976 5 | END;
407 0977 5 | +
408 0978 5 | If the command is tabs adjust, then process it specially.
409 0979 5 | The repeat count and (+ or -) directional indicator are used
410 0980 5 | to determine the adjustment and the entity count and direction
411 0981 5 | to select the text.
412 0982 5 | -
413 0983 5 |
414 0984 6 | IF (.VERB EQL VERB_K_TADJ)
415 0985 5 | THEN
416 0986 6 | BEGIN
417 0987 6 |
418 0988 6 | IF (.EXPLICIT_REPEAT EQL 0) THEN REPT_COUNT = 1;
419 0989 6 |
420 0990 6 | +
421 0991 6 | TADJ_DIR = 1 to adjust right
422 0992 6 | = -1 to adjust left
423 0993 6 | -
424 0994 6 | EDT$$G_TADJ = .REPT_COUNT*.TADJ_DIR;
425 0995 6 | EXPLICIT_REPEAT = 0;
426 0996 5 | END;
427 0997 5 |
428 0998 5 | +
429 0999 5 | If there was a repeat count as well as an entity count, multiply
430 1000 5 | them out to get the actual number of entities, and only execute
431 1001 5 | the command once.
432 1002 5 | -
433 1003 5 |
434 1004 5 | IF .EXPLICIT_REPEAT THEN ENTITY_COUNT = .REPT_COUNT*.ENTITY_COUNT;
435 1005 5 |
436 1006 5 | REPT_COUNT = 1;
437 1007 5 | +
438 1008 5 | If entity is quoted, get the string.
439 1009 5 | -
440 1010 5 |
441 1011 5 | IF (.ENTITY EQL ENT_K_QUOTE) THEN EDT$$SCAN_SEASTR ();
442 1012 5 |
443 1013 5 | +
444 1014 5 | Get the buffer spec if command is cut or append.
445 1015 5 | -
446 1016 5 |
447 1017 5 | IF (.VERB GEQ VERB_K_CUT) THEN EDT$$GET_BUF ();
448 1018 5 |
449 1019 5 | +
450 1020 5 | And execute it.
451 1021 5 | -
452 1022 5 |
```



```

453      1023 5      IF .EXECUTE
454      1024 5      THEN
455      1025 6      BEGIN
456      1026 6      SUCCEED = EDT$$EXE_CHMCMD2 (.ENTITY, .ENTITY_COUNT, .VERB);
457      1027 6
458      1028 6      IF (.SUCCEED NEQ 1) THEN RETURN (.SUCCEED);
459      1029 6
460      1030 5      END;
461      1031 5
462      1032 5      END
463      1033 4      ELSE
464      1034 5      BEGIN
465      1035 5      !+
466      1036 5      Handle the parsing of the 'funny' verbs, i.e. those which have
467      1037 5      variable length strings as a part.
468      1038 5      !-
469      1039 5      OPERAND = .EDT$$A_CMD_BUF;
470      1040 5
471      1041 5      CASE .VERB FROM VERB_K_SUBS TO VERB_K_CC OF
472      1042 5      SET
473      1043 5      [VERB_K_INSERT] :
474      1044 5      EDT$$SCAN_INSSTR ();
475      1045 5
476      1046 5      [VERB_K_XLATE] :
477      1047 5
478      1048 5      IF CH$PTR_NEQ (.EDT$$A_CMD_BUF, .EDT$$A_CMD_END)
479      1049 6      THEN
480      1050 5      EDT$$SCAN_INSSTR ()
481      1051 5      ELSE
482      1052 5      EDT$$G_SEA_LEN = 0;
483      1053 5
484      1054 5      [VERB_K_SUBS] :
485      1055 5      EDT$$FND_SUBSTR ();
486      1056 5
487      1057 5      [VERB_K_CC] :
488      1058 5      BEGIN
489      1059 6      EDT$$CNV_UPC (.EDT$$A_CMD_BUF, 1);
490      1060 6      EDT$$A_CMD_BUF = CH$PLUS (.EDT$$A_CMD_BUF, 1);
491      1061 6      END;
492      1062 5
493      1063 5      [VERB_K_PASTE] :
494      1064 5      EDT$$GET_BUF ();
495      1065 5
496      1066 5      [OUTRANGE] :
497      1067 5      0;
498      1068 5      TES;
499      1069 5
500      1070 5      !+
501      1071 5      Now, execute the command, and terminate the loop if it fails.
502      1072 5      !-
503      1073 5
504      1074 5      IF .EXECUTE
505      1075 5      THEN
506      1076 6      BEGIN
507      1077 6      SUCCEED = EDT$$EXE_CHMCMD1 (.VERB, .REPT_COUNT, .OPERAND, .EXPLICIT_REPEAT);
508      1078 6
509      1079 6
```

EDT\$CHMPARSE
V04-000

EDT\$CHMPARSE - change mode parse and execute
EDT\$\$SCHM_PAREXE - change mode parse and execut

J 6
16-Sep-1984 00:06:35
14-Sep-1984 12:22:41

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

```
: 510      1080  6      IF (.SUCCEED NEQ 1) THEN RETURN (.SUCCEED);
: 511      1081  6
: 512      1082  5
: 513      1083  5      END;
: 514      1084  4      END;
: 515      1085  4      END;
: 516      1086  3      IF .EDT$$G_EXI THEN RETURN (1);
: 517      1087  3      END;
: 518      1088  3      RETURN (1);
: 519      1089  3      END;
: 520      1090  2
: 521      1091  2
: 522      1092  2
: 523      1093  1      END;
```

! of routine EDT\$\$SCHM_PAREXE

.TITLE EDT\$CHMPARSE EDT\$CHMPARSE - change mode parse and execute

.IDENT \V04-000\

.EXTRN EDT\$\$EXE_CHMCMD2
.EXTRN EDT\$\$EXE_CHMCMD1
.EXTRN EDT\$\$MSG_BELL, EDT\$\$CHK_CC
.EXTRN EDT\$\$GET_BUF, EDT\$\$CHK_CNT
.EXTRN EDT\$\$CHK_DIR, EDT\$\$SCAN_SEASTR
.EXTRN EDT\$\$FND_SUBSTR
.EXTRN EDT\$\$SCAN_INSSTR
.EXTRN EDT\$\$KEY_WORD, EDT\$\$PARENT
.EXTRN EDT\$\$CNV_UPC, EDT\$\$G_SEA_LEN
.EXTRN EDT\$\$G_DIR, EDT\$\$G_DIR_MOD
.EXTRN EDT\$\$A_CMD_END, EDT\$\$G_EXI
.EXTRN EDT\$\$A_CMD_BUF, EDT\$\$A_ALT_BUF
.EXTRN EDT\$\$G_TADJ, EDT\$\$G_DFCT_VERB
.EXTRN EDT\$\$B_CHAR_INFO
.EXTRN EDT\$\$G_CC_DONE, EDT\$_INVENT
.EXTRN EDT\$_INVSUBCOM, EDT\$_NUMVALILL

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

.ENTRY EDT\$\$SCHM_PAREXE, Save R2,R3,R4,R5,R6,R7,R8,-; 0683
R9,R10,RT1
MOVAB EDT\$\$G_DIR, R11
MOVAB EDT\$\$A_CMD_BUF, R10
SUBL2 #16, SP
CMPL EDT\$\$A_CMD_END, EDT\$\$A_CMD_BUF 0780
BGTRU 3\$
BRW 39\$
CALLS #0, EDT\$\$CHK_CC 0783
BLBC R0, 4\$
MOVL #1, EDT\$\$G_CC_DONE 0786
BRW 40\$ 0787
MOVL EDT\$\$A_CMD_BUF, R0 0794
CMPB (R0), #32
BNEQ 5\$
INCL EDT\$\$A_CMD_BUF 0795
BRB 4\$
MOVL EDT\$\$A_CMD_BUF, R0 0801

OFFC 00000

```
5B 00000000G 00 9E 00002  
5A 00000000G 00 9E 00009  
5E 10 C2 00010  
6A 00000000G 00 D1 00013 1$:  
03 1A 0001A  
0224 31 0001C 2$:  
00 00 FB 0001F 3$:  
00 01 D0 00029  
0214 31 00030  
50 6A D0 00033 4$:  
20 60 91 00036  
04 12 00039  
6A D6 0003B  
F4 11 0003D  
50 6A D0 0003F 5$:
```


29		60	91	00042	CMPB	(R0), #41		
		D5	13	00045	BEQL	2\$		
6B	00000000G	00	D0	00047	MOVL	EDT\$\$G_DIR_MOD, EDT\$\$G_DIR	0807	
	00000000G	00	D4	0004E	CLRL	EDT\$\$A_ALT_BUF	0808	
00000000G	00	00	FB	00054	CALLS	#0, EDT\$\$CHK_DIR	0812	
	58	50	D0	0005B	MOVL	R0, DIR_SAME		
		6E	D4	0005E	CLRL	REPT_COUNT	0818	
		5E	DD	00060	PUSHL	SP	0819	
00000000G	00	01	FB	00062	CALLS	#1, EDT\$\$CHK_CNT		
	55	50	D0	00069	MOVL	R0, EXPLICIT_REPEAT		
	02	55	D1	0006C	CMPL	EXPLICIT_REPEAT, #2	0821	
		03	12	0006F	BNEQ	6\$		
		00A3	31	00071	BRW	14\$		
	50	6A	D0	00074	MOVL	EDT\$\$A_CMD_BUF, R0	0832	
	28	60	91	00077	CMPB	(R0), #40		
		1B	12	0007A	BNEQ	8\$		
		6E	D5	0007C	TSTL	REPT_COUNT	0836	
		07	12	0007E	BNEQ	7\$		
		55	D5	00080	TSTL	EXPLICIT_REPEAT		
		03	12	00082	BNEQ	7\$		
	6E	01	D0	00084	MOVL	#1, REPT_COUNT		
		04	AC	DD	00087	PUSHL	EXECUTE	0838
		04	AE	DD	0008A	PUSHL	REPT_COUNT	
00000000G	00	02	FB	0008D	CALLS	#2, EDT\$\$PARENT		
		0196	31	00094	BRW	37\$		
		04	AE	9F	00097	PUSHAB	VERB	0848
		01	DD	0009A	PUSHL	#1		
00000000G	00	02	FB	0009C	CALLS	#2, EDT\$\$KEY_WORD		
		04	AE	D5	000A3	TSTL	VERB	0853
		08	12	000A6	BNEQ	9\$		
04	AE	00000000G	00	D0	000A8	MOVL	EDT\$\$G_DFLT_VERB, VERB	
	08	04	AE	D1	000B0	CMPL	VERB, #8	0863
		15	12	000B4	BNEQ	11\$		
	56	01	D0	000B6	MOVL	#1, TADJ_DIR	0866	
		6B	D5	000B9	TSTL	EDT\$\$G_DIR	0872	
		07	12	000BB	BNEQ	10\$		
		58	D5	000BD	TSTL	DIR_SAME		
		03	12	000BF	BNEQ	10\$		
	56	01	CE	000C1	MNEGL	#1, TADJ_DIR		
6B	00000000G	00	D0	000C4	MOVL	EDT\$\$G_DIR_MOD, EDT\$\$G_DIR	0874	
	0C	04	AE	D1	000CB	CMPL	VERB, #12	0882
		1A	12	000CF	BNEQ	12\$		
	50	6A	D0	000D1	MOVL	EDT\$\$A_CMD_BUF, R0	0888	
	50	60	9A	000D4	MOVZBL	(R0), R0		
03	00000000G	00	40	93	000D7	BITB	EDT\$\$B_CHAR_INFO[R0], #3	
		0A	13	000DF	BEQL	12\$		
04	AE	00000000G	00	D0	000E1	MOVL	EDT\$\$G_DFLT_VERB, VERB	0902
		6A	D7	000E9	DECL	EDT\$\$A_CMD_BUF	0907	
	53	04	AE	D0	000EB	MOVL	VERB, R3	0914
	0B	53	D1	000EF	CMPL	R3, #11		
		03	19	000F2	BLSS	13\$		
		00D3	31	000F4	BRW	28\$		
00000000G	00	00	FB	000F7	CALLS	#0, EDT\$\$CHK_DIR	0920	
	58	50	D0	000FE	MOVL	R0, DIR_SAME		
	08	01	D0	00101	MOVL	#1, ENTITY_COUNT	0925	
		08	AE	9F	00105	PUSHAB	ENTITY_COUNT	0926
00000000G	00	01	FB	00108	CALLS	#1, EDT\$\$CHK_CNT		

59	50	D0	0010F	MOVL	R0, EXPLICIT_ENTITY_COUNT	
02	59	D1	00112	CMPL	EXPLICIT_ENTITY_COUNT, #2	0928
	08	12	00115	BNEQ	15\$	
	00000000G	8F	DD 00117	PUSHL	#EDT\$_NUMVALILL	0931
		55	11 0011D	BRB	21\$	
00000000G	00	00	FB 0011F	CALLS	#0, EDT\$\$CHK_DIR	0938
	58	50	D0 00126	MOVL	R0, DIR_SAME	
	50	6A	D0 00129	MOVL	EDT\$\$A_CMD_BUF, R0	0943
	50	00	D1 0012C	CMPL	EDT\$\$A_CMD_END, R0	
		24	1B 00133	BLEQU	18\$	
	22	60	91 00135	CMPB	(R0), #34	0946
		09	13 00138	BEQL	16\$	
	27	60	91 0013A	CMPB	(R0), #39	0947
		04	13 0013D	BEQL	16\$	
		60	95 0013F	TSTB	(R0)	0948
		08	12 00141	BNEQ	17\$	
OC	AE	29	D0 00143	MOVL	#41, ENTITY	0951
		6A	D6 00147	INCL	EDT\$\$A_CMD_BUF	0952
		11	11 00149	BRB	19\$	0946
		OC	AE 9F 0014B	PUSHAB	ENTITY	0955
		02	DD 0014E	PUSHL	#2	
00000000G	00	02	FB 00150	CALLS	#2, EDT\$\$KEY_WORD	
		03	11 00157	BRB	19\$	0946
		OC	AE D4 00159	CLRL	ENTITY	0958
	52	OC	AE D0 0015C	MOVL	ENTITY, R2	0964
		1C	12 00160	BNEQ	22\$	
		53	D5 00162	TSTL	R3	0972
		08	13 00164	BEQL	20\$	
	00000000G	8F	DD 00166	PUSHL	#EDT\$_INVENT	
		06	11 0016C	BRB	21\$	
	00000000G	8F	DD 0016E	PUSHL	#EDT\$_INVSUBCOM	
00000000G	00	01	FB 00174	CALLS	#1, EDT\$\$MSG_BELL	
		OC	9 31 0017B	BRW	40\$	0974
	08	53	D1 0017E	CMPL	R3, #8	0984
		11	12 00181	BNEQ	24\$	
		55	D5 00183	TSTL	EXPLICIT_REPEAT	0988
		03	12 00185	BNEQ	23\$	
	6E	01	D0 00187	MOVL	#1, REPT_COUNT	
00000000G	00	56	C5 0018A	MULL3	TADJ_DIR, REPT_COUNT, EDT\$\$G_TADJ	0994
	6E	55	D4 00192	CLRL	EXPLICIT_REPEAT	0995
	04	55	E9 00194	BLBC	EXPLICIT_REPEAT, 25\$	1004
	08	6E	C4 00197	MULL2	REPT_COUNT, ENTITY_COUNT	
	6E	01	D0 0019B	MOVL	#1, REPT_COUNT	1006
	29	52	D1 0019E	CMPL	R2, #41	1011
		07	12 001A1	BNEQ	26\$	
00000000G	00	00	FB 001A3	CALLS	#0, EDT\$\$SCAN_SEASTR	
	09	53	D1 001AA	CMPL	R3, #9	1017
		07	19 001AD	BLSS	27\$	
00000000G	00	00	FB 001AF	CALLS	#0, EDT\$\$GET_BUF	
	7F	04	AC E9 001B6	BLBC	EXECUTE, 38\$	1023
		53	DD 001BA	PUSHL	R3	1026
		OC	AE DD 001BC	PUSHL	ENTITY_COUNT	
		52	DD 001BF	PUSHL	R2	
00000000G	00	03	FB 001C1	CALLS	#3, EDT\$\$EXE_CHMCMD2	
		63	11 001C8	BRB	37\$	
	52	6A	D0 001CA	MOVL	EDT\$\$A_CMD_BUF, R2	1039
	57	52	D0 001CD	MOVL	R2, OPERAND	

000C	04 0015	0C 003E	53 0026 002F	CF 001D0 001D4 29\$: 001DC	CASEL .WORD	R3, #12, #4 33\$-29\$,- 35\$-29\$,- 31\$-29\$,- 30\$-29\$,- 34\$-29\$	1041
	00000000G	00	39 52	11 001DE D1 001E0 30\$:	BRB CMPL	36\$ R2, EDT\$\$A_CMD_END	1049
	00000000G	00	09 00	13 001E7 FB 001E9 31\$:	BEQL CALLS	32\$ #0, EDT\$\$SCAN_INSSTR	1051
		00000000G	27 00	11 001F0 D4 001F2 32\$:	BRB CLRL	36\$ EDT\$\$G_SEA_LEN	1053
	00000000G	00	1F 00	11 001F8 FB 001FA 33\$:	BRB CALLS	36\$ #0, EDT\$\$FND_SUBSTR	1049 1056
			16 01	11 00201 DD 00203 34\$:	BRB PUSHL	36\$ #1	1060
	00000000G	00	52 02	DD 00205 FB 00207	PUSHL CALLS	R2 #2, EDT\$\$CNV_UPC	
			6A 07	D6 0020E 11 00210	INCL BRB	EDT\$\$A_CMD_BUF 36\$	1061 1041
	00000000G	00	00	FB 00212 35\$:	CALLS	#0, EDT\$\$GET_BUF	1065
		1C	04	AC E9 00219 36\$:	BLBC PUSHL	EXECUTE, 38\$- EXPLICIT_REPEAT	1075 1078
			55 57	DD 0021D DD 0021F	PUSHL PUSHL	OPERAND	
			08	AE DD 00221	PUSHL	REPT_COUNT	
	00000000G	00	53	DD 00224	PUSHL	R3	
		54	04	FB 00226	CALLS	#4, EDT\$\$EXE_CHMCMD1	
		01	50	D0 0022D 37\$:	MOVL	R0, SUCCEED	
			54	D1 00230	CMPL	SUCCEED, #1	1080
			04	13 00233	BEQL	38\$	
		50	54	D0 00235	MOVL	SUCCEED, R0	
			04	00238	RET		
	03 00000000G	00	00	E8 00239 38\$:	BLBS	EDT\$\$G_EXI, 39\$	1088
			FD	D0 31 00240	BRW	1\$	
		50	01	D0 00243 39\$:	MOVL	#1, R0	1092
			04	00246	RET		
			50	D4 00247 40\$:	CLRL	R0	1093
			04	00249	RET		

; Routine Size: 586 bytes, Routine Base: _EDT\$CODE + 0000

: 524 1094 1
: 525 1095 1 !<BLF/PAGE>

: 527
: 528
: 529

1096 1 END
1097 1
1098 0 ELUDOM

! of module EDT\$CHMPARSE

EDT\$\$CHM_PAREXE_NOOVERLAY_REF==
EDT\$\$CHM_PAREXE

PSECT SUMMARY

Name	Bytes	Attributes
_EDT\$CODE	586	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	37	9	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	2	100	5	00:00.1

COMMAND QUALIFIERS

:
:
:

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

: Size: 586 code + 0 data bytes
: Run Time: 00:27.3
: Elapsed Time: 00:30.9
: Lines/CPU Min: 2415
: Lexemes/CPU-Min: 7421
: Memory Used: 202 pages
: Compilation Complete

0132 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

