

[illegible]

EXE

Mod

EDT

ED

ED
EDED
EDED
EDED
ED

ED

ED

ED

ED

ED
EDSYN
LBA

110

KK	KK	EEEEEEEEEE	YY	YY	FFFFFFFF	MM	MM	TTTTTTTTTT	SSSSSSSS	TTTTTTTTTT	RRRRRRRR	
KK	KK	EEEEEEEEEE	YY	YY	FFFFFFFF	MM	MM	TTTTTTTTTT	SSSSSSSS	TTTTTTTTTT	RRRRRRRR	
KK	KK	EE	YY	YY	FF	MMM	MMM	TT	SS	TT	RR	RR
KK	KK	EE	YY	YY	FF	MMM	MMM	TT	SS	TT	RR	RR
KK	KK	EE	YY	YY	FF	MM	MM	TT	SS	TT	RR	RR
KK	KK	EE	YY	YY	FF	MM	MM	TT	SS	TT	RR	RR
KKKKKK		EEEEEEEE	YY		FFFFFFFF	MM	MM	TT	SSSSSS	TT	RRRRRRRR	
KKKKKK		EEEEEEEE	YY		FFFFFFFF	MM	MM	TT	SSSSSS	TT	RRRRRRRR	
KK	KK	EE	YY		FF	MM	MM	TT		TT	RR	RR
KK	KK	EE	YY		FF	MM	MM	TT		TT	RR	RR
KK	KK	EE	YY		FF	MM	MM	TT		TT	RR	RR
KK	KK	EE	YY		FF	MM	MM	TT		TT	RR	RR
KK	KK	EE	YY		FF	MM	MM	TT		TT	RR	RR
KK	KK	EEEEEEEEEE	YY		FF	MM	MM	TT	SSSSSSSS	TT	RR	RR
KK	KK	EEEEEEEEEE	YY		FF	MM	MM	TT	SSSSSSSS	TT	RR	RR

```

LL          IIIIII          SSSSSSSS
LL          IIIIII          SSSSSSSS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SSSSSS
LL          II             SSSSSS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SS
LLLLLLLLLLLL IIIIII          SSSSSSSS
LLLLLLLLLLLL IIIIII          SSSSSSSS

```



```
0001 0 %TITLE 'EDT$KEYFMTSTR - translate a key string'
0002 0 MODULE EDT$KEYFMTSTR ( ! Translate a key string
0003 0 IDENT = 'V04-000' ! File: KEYFMTSTR.BLI Edit: JBS1026
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 Translate a key string.
0037 1
0038 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0039 1
0040 1 AUTHOR: Bob Kushlis, CREATION DATE: April 7, 1979
0041 1
0042 1 MODIFIED BY:
0043 1
0044 1 1-001 - Original. DJS 24-Feb-1981. This module was created by
0045 1 extracting routine COM_STRING from module KEYTRAN.
0046 1
0047 1 1-002. DJS 26-Feb-1981. Handling of prompt following ? in a key
0048 1 definition modified to allow double quotes as well as single
0049 1 quotes to enclose a prompt string.
0050 1 1-003 - Regularize headers. JBS 09-Mar-1981
0051 1 1-004 - Build in T_ADV, T_BACK and T_RESET. JBS 31-Mar-1981
0052 1 1-005 - Add return_value for end of journal file. JBS 02-Oct-1981
0053 1 1-006 - Add an additional terminator when prompt is ?*. STS 21-Oct-1981
0054 1 1-007 - If terminator of search string begins with adv or back then insert
0055 1 this and throw away rest of key definition. STS 16-Nov-1981
0056 1 1-008 - Revise the call to turn on autorepeat. JBS 30-Jan-1982
0057 1 1-009 - Remove an unused external reference. JBS 09-Jun-1982
```

58	0058	1	1-010	- Modify message output. SMB 30-Jun-1982
59	0059	1	1-011	- Add resetting search string and set direction forward to RESET.
60	0060	1		STS 02-Jul-1982
61	0061	1	1-012	- Only erase the last line on a reset. SMB 15-Jul-1982
62	0062	1	1-013	- Change the string match call. JBS 19-Jul-1982
63	0063	1	1-014	- Catch errors in prompt string. STS 03-Aug-1982
64	0064	1	1-015	- Position cursor after error message. STS 10-Aug-1982
65	0065	1	1-016	- New implementation of defined keys. JBS 13-Aug-1982
66	0066	1	1-017	- Fix a bug in recognizing ADV and BACK. JBS 16-Aug-1982
67	0067	1	1-018	- Make RESET not reset the search string. JBS for STS 26-Aug-1982
68	0068	1	1-019	- New screen update logic. JBS 13-Sep-1982
69	0069	1	1-020	- Remove external declaration of EDT\$FMT_LIT, not used. JBS 05-Oct-1982
70	0070	1	1-021	- Change the cursor positioning call, to obsolete a module. JBS 05-Oct-1982
71	0071	1	1-022	- Use partial matching for ADV and BACK. JBS 14-Dec-1982
72	0072	1	1-023	- Insert the whole text of a terminator to a prompt, if it starts with
73	0073	1		ADV or BACK. This is for compatibility with EDT version 2. JBS 15-Dec-1982
74	0074	1	1-024	- Don't position cursor if no position recorded, and treat a cancelled command the same
75	0075	1		as an undefined key. JBS 17-Jan-1983
76	0076	1	1-025	- Put WPS support under a conditional. JBS 10-Feb-1983
77	0077	1	1-026	- Allow unprompted input, for compatibility with EDT version 2. JBS 01-Apr-1983
78	0078	1	--	
79	0079	1		

EDTSKEYFMTSTR
V04-000

EDTSKEYFMTSTR - translate a key string
Declarations

G 8
16-Sep-1984 00:43:41
14-Sep-1984 12:23:22

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]KEYFMTSTR.BLI;1

Page 3
(2)

```

: 81      0080 1 %SBTTL 'Declarations'
: 82      0081 1
: 83      0082 1 | TABLE OF CONTENTS:
: 84      0083 1 |
: 85      0084 1
: 86      0085 1 REQUIRE 'EDTSRC:TRAROUNAM';
: 87      0524 1
: 88      0525 1 FORWARD ROUTINE
: 89      0526 1     EDT$STRN_KSTR;
: 90      0527 1
: 91      0528 1 |
: 92      0529 1 | INCLUDE FILES:
: 93      0530 1 |
: 94      0531 1
: 95      0532 1 REQUIRE 'EDTSRC:EDTREQ';
: 96      0667 1
: 97      0668 1 LIBRARY 'EDTSRC:KEYPADDEF';
: 98      0669 1
: 99      0670 1 LIBRARY 'EDTSRC:SUPPORTS';
100      0671 1
101      0672 1 |
102      0673 1 | MACROS:
103      0674 1 |
104      0675 1 |     NONE
105      0676 1 |
106      0677 1 | EQUATED SYMBOLS:
107      0678 1 |
108      0679 1 |     NONE
109      0680 1 |
110      0681 1 | OWN STORAGE:
111      0682 1 |
112      0683 1 |     NONE
113      0684 1 |
114      0685 1 | EXTERNAL REFERENCES:
: 115      0686 1 |
: 116      0687 1 |     In the routine
```

```
118 0688 1 %SBTTL 'EDT$$TRN_KSTR - translate a key string'
119 0689 1
120 0690 1 GLOBAL ROUTINE EDT$$TRN_KSTR (
121 0691 1     K,
122 0692 1     EXPAND,
123 0693 1     ECHO
124 0694 1 ) =
125 0695 1
126 0696 1 ++
127 0697 1 FUNCTIONAL DESCRIPTION:
128 0698 1
129 0699 1     This routine processes the translation string of a key placing the
130 0700 1     characters in the buffer.  If a '?' is seen, and the EXPAND parameter
131 0701 1     is 1, then the user is prompted for input which replaces the '?' and
132 0702 1     the prompt string in the buffer.
133 0703 1
134 0704 1 FORMAL PARAMETERS:
135 0705 1
136 0706 1     K                The number of the key to translate.
137 0707 1
138 0708 1     EXPAND           Flag indicating ? should be expanded.
139 0709 1
140 0710 1     ECHO             Flag indicating that the characters should be echoed.
141 0711 1
142 0712 1 IMPLICIT INPUTS:
143 0713 1
144 0714 1     EDT$$G_MESSAGE_LINE
145 0715 1     EDT$$G_CS_LNO
146 0716 1     EDT$$G_CUR_COL
147 0717 1     EDT$$A_SEL_BUF
148 0718 1     EDT$$T_CMD_BUF
149 0719 1     EDT$$A_CUR_BUF
150 0720 1     EDT$$G_TI_TYP
151 0721 1     EDT$$G_TIN_ECHOPOS
152 0722 1
153 0723 1 IMPLICIT OUTPUTS:
154 0724 1
155 0725 1     EDT$$A_CMD_BUF
156 0726 1     EDT$$A_CMD_END
157 0727 1     EDT$$G_TIN_ECHOPOS
158 0728 1     EDT$$G_TIN_ECHOFLG
159 0729 1
160 0730 1 ROUTINE VALUE:
161 0731 1
162 0732 1     0 = definition did not end with a period
163 0733 1     1 = key was null or translation was successful and definition ended with period
164 0734 1     or translation was cancelled
165 0735 1     2 = end of journal file
166 0736 1
167 0737 1 SIDE EFFECTS:
168 0738 1
169 0739 1     NONE
170 0740 1
171 0741 1 --
172 0742 1
173 0743 2 BEGIN
174 0744 2
```



```
175 0745 2 EXTERNAL ROUTINE
176 0746 2 EDT$SC_ERATOEOOL,
177 0747 2 EDT$MSG_BELL,
178 0748 2 EDT$OUT_MSG,
179 0749 2 EDT$SC_REVID,
180 0750 2 EDT$SC_NONREVID,
181 0751 2 EDT$OUT_FMTBUF,
182 0752 2 EDT$KPAD_INP,
183 0753 2 EDT$PUT_CMDCH : NOVALUE,
184 0754 2 EDT$SC_POSCSIF : NOVALUE,
185 0755 2 EDT$TI_ENBLAUTREP,
186 0756 2 EDT$TI_INPCH,
187 0757 2 EDT$TI_DELK,
188 0758 2 EDT$TST_KEYDEF,
189 0759 2 EDT$FIND_KEY;
190 0760 2
191 0761 2 EXTERNAL
192 0762 2 EDT$G_MESSAGE_LINE,
193 0763 2 EDT$G_CS_LNO,
194 0764 2 EDT$G_CUR_COL,
195 0765 2 EDT$A_SEL_BUF,
196 0766 2 EDT$T_CMD_BUF,
197 0767 2 EDT$A_CMD_BUF,
198 0768 2 EDT$A_CMD_END,
199 0769 2 EDT$A_CUR_BUF,
200 0770 2 EDT$G_TI_TYP,
201 0771 2 EDT$G_TIN_ECHOFLG,
202 0772 2 EDT$G_TIN_ECHOPOS,
203 0773 2 EDT$G_DIR_MOD,
204 0774 2
205 L 0775 2 %IF SUPPORT_WPS
206 0776 2 %THEN
207 0777 2 EDT$G_DFLT_VERB,
208 0778 2 %FI
209 0779 2
210 0780 2 EDT$G_SEA_STRLN;
211 0781 2
212 0782 2 LOCAL
213 0783 2 C,
214 0784 2 TERM,
215 0785 2 ALT_TERM,
216 0786 2 TRAN_POINT,
217 0787 2 TRAN_END,
218 0788 2 KEY_PTR : REF BLOCK [, BYTE] FIELD (KEY_DEF_FIELD);
219 0789 2
220 0790 2 !+
221 0791 2 !- Check for a key that is null.
222 0792 2 !-
223 0793 2
224 0794 2 IF ( NOT EDT$FIND_KEY (.K, KEY_PTR))
225 0795 2 THEN
226 0796 2 BEGIN
227 0797 2 EDT$A_CMD_END = CH$PTR (.EDT$A_CMD_BUF);
228 0798 2 RETURN (1);
229 0799 2 END;
230 0800 2
231 0801 2 !+
```

```
232 0802 2 ! Look for the string RESET, which is handled specially.
233 0803 2 !-
234 0804 2
235 0805 2 IF EDT$STST_KEYDEF (.K, UPLIT (BYTE ('RESET')), 5, 0)
236 0806 2 THEN
237 0807 2 BEGIN
238 0808 2 !+
239 0809 2 ! Reset was seen. Flush the buffer, and undo the select range.
240 0810 2 !-
241 0811 2 EDT$A_CMD_BUF = CH$PTR (EDT$ST_CMD_BUF);
242 0812 2 EDT$A_CMD_END = CH$PTR (EDT$ST_CMD_BUF);
243 0813 2
244 0814 2 IF (.ECHO)
245 0815 2 THEN
246 0816 2 BEGIN
247 0817 2 EDT$SSC_POSCSIF (.EDT$G_MESSAGE_LINE + 1, 0); ! Erase key definition
248 0818 2 EDT$SSC_ERATOEOI ();
249 0819 2 EDT$OUT_FMTBUF ();
250 0820 2 END;
251 0821 2
252 L 0822 2 %IF SUPPORT_WPS
253 0823 2 %THEN
254 0824 2 EDT$G_DFLT_VERB = VERB_K_MOVE; ! reset mode to move
255 0825 2 %FI
256 0826 2
257 0827 2 EDT$G_DIR_MOD = DIR_FORWARD; ! reset direction to forward
258 0828 2 EDT$A_SEL_BUF = 0; ! There is no more select buffer
259 0829 2 RETURN (1);
260 0830 2 END;
261 0831 2
262 0832 2 !+
263 0833 2 ! Get a pointer to the end of the translation string.
264 0834 2 !-
265 0835 2 TRAN_POINT = KEY_PTR [KEY_DEF_TEXT];
266 0836 2 TRAN_END = CH$PTR (.TRAN_POINT, .KEY_PTR [KEY_DEF_LEN]);
267 0837 2 TERM = 0;
268 0838 2 !+
269 0839 2 ! Loop through the string.
270 0840 2 !-
271 0841 2
272 0842 2 WHILE CH$PTR_LSS (.TRAN_POINT, .TRAN_END) DO
273 0843 2
274 0844 2 IF ((CH$RCHAR (.TRAN_POINT) EQL %C'?' ) AND .EXPAND)
275 0845 2 THEN
276 0846 2 !+
277 0847 2 ! Character is ? and expand is on.
278 0848 2 !-
279 0849 2 BEGIN
280 0850 2
281 0851 2 LOCAL
282 0852 2 QUOTE, ! Character following ?
283 0853 2 P_ADDR, ! Address of prompt.
284 0854 2 STATUS, ! for status of kpad_inp routine
285 0855 2 P_LEN; ! Length of prompt.
286 0856 2
287 0857 2 TRAN_POINT = CH$PLUS (.TRAN_POINT, 1);
288 0858 2 !+
```



```
289 0859 3 ! Look for a second ?, in which case it is literal.
290 0860 3 !-
291 0861 3
292 0862 4 IF (CH$RCHAR (.TRAN_POINT) EQL %C'?'')
293 0863 3 THEN
294 0864 3 EDT$SPUT_CMDCH (CH$RCHAR_A (TRAN_POINT), .ECHO)
295 0865 3 ELSE
296 0866 3 !+
297 0867 3 !- Expand the ?.
298 0868 3
299 0869 4 BEGIN
300 0870 4 P_LEN = 0;
301 0871 4 !+
302 0872 4 !- See if it's the special prompt which takes a CR.
303 0873 4
304 0874 4
305 0875 5 IF (CH$RCHAR (.TRAN_POINT) EQL %C'*')
306 0876 4 THEN
307 0877 5 BEGIN
308 0878 5 TRAN_POINT = CH$PLUS (.TRAN_POINT, 1); ! point to the next char
309 0879 5 ALT_TERM = ASC_K_CR; ! set up an alternative terminator of CR
310 0880 5 END
311 0881 4 ELSE
312 0882 4 ALT_TERM = ASC_K_ESC; ! otherwise we just stop at escapes
313 0883 4
314 0884 4 !+
315 0885 4 !- Look for a quote mark following to start the prompt.
316 0886 4
317 0887 4 QUOTE = CH$RCHAR (.TRAN_POINT);
318 0888 4
319 0889 5 IF ((.QUOTE EQL %C''''') OR (.QUOTE EQL %C'''''))
320 0890 4 THEN
321 0891 5 BEGIN
322 0892 5 !+
323 0893 5 !- Get the entire prompt string.
324 0894 5
325 0895 5 TRAN_POINT = CH$PLUS (.TRAN_POINT, 1);
326 0896 5 P_ADDR = .TRAN_POINT;
327 0897 5
328 0898 5 WHILE (CH$PTR_GTR (.TRAN_END, .TRAN_POINT) AND (CH$RCHAR_A (TRAN_POINT) NEQ .QUOTE)) DO
329 0899 5 P_LEN = .P_LEN + 1;
330 0900 5
331 0901 4 END;
332 0902 4
333 0903 4 !+
334 0904 4 !- Position the cursor and put out the prompt.
335 0905 4
336 0906 4 EDT$SOUT_MSG (.EDT$$MESSAGE_LINE, 0, .P_ADDR, .P_LEN);
337 0907 4 EDT$STI_ENBLAUTREP (1);
338 0908 4 EDT$SG_TIN_ECHOPOS = .EDT$SG_TIN_ECHOPOS + .P_LEN;
339 0909 4 EDT$SG_TIN_ECHOFLG = 1;
340 0910 4 !+
341 0911 4 !- Now, read up to the next escape character.
342 0912 4
343 0913 4
344 0914 4 IF (EDT$STI_INPCH (C) EQL 0) THEN RETURN (2);
345 0915 4
```

```
346      0916 4      STATUS = EDT$$KPAD_INP (.C, .ALT_TERM, TERM);
347      0917 4
348      0918 4      IF (.STATUS EQL 2) THEN RETURN (2);
349      0919 4
350      0920 5      IF (.STATUS EQL 0)
351      0921 4      THEN
352      0922 4      !+
353      0923 4      !- Command was canceled, flush the buffer and return 1.
354      0924 4      !-
355      0925 5      BEGIN
356      0926 5      EDT$$SC_NONREVID ();
357      0927 5
358      0928 5      IF (.EDT$$G_CS_LNO GEQ 0) THEN EDT$$SC_POSCSIF (.EDT$$G_CS_LNO, .EDT$$G_CUR_COL);
359      0929 5
360      0930 5      EDT$$A_CMD_BUF = CH$PTR (EDT$$T_CMD_BUF);
361      0931 5      EDT$$A_CMD_END = CH$PTR (EDT$$T_CMD_BUF);
362      0932 5      RETURN (1);
363      0933 4      END;
364      0934 4
365      0935 4      !+
366      0936 4      !- Do a positioning sequence to left column so that we know
367      0937 4      !- the command has been seen.
368      0938 4      !-
369      0939 4      EDT$$SC_POSCSIF (.EDT$$G_MESSAGE_LINE, 0);
370      0940 4      EDT$$OUT_FMTBUF ();
371      0941 4      END
372      0942 4
373      0943 3      END
374      0944 2      ELSE
375      0945 2      EDT$$PUT_CMDCH (CH$RCHAR_A (TRAN_POINT), .ECHO);
376      0946 2
377      0947 2      !+
378      0948 2      !- If the translation ended with a period, then remove it from the
379      0949 2      !- buffer and return a one, otherwise, return a 0.
380      0950 2      !-
381      0951 2
382      0952 3      IF (CH$RCHAR (CH$PTR (.EDT$$A_CMD_BUF, -1)) EQL %C'.')
383      0953 2      THEN
384      0954 2      BEGIN
385      0955 2      EDT$$A_CMD_BUF = CH$PTR (.EDT$$A_CMD_BUF, -1);
386      0956 2
387      0957 2      IF .ECHO THEN EDT$$TI_DELK (CH$RCHAR (.EDT$$A_CMD_BUF));
388      0958 2
389      0959 2      !+
390      0960 3      !- If the terminator of input was advance or backup, then throw it into the buffer
391      0961 3      !- so we can implement the KED feature of hitting advance or backup after a search
392      0962 3      !- string. For compatibility with EDT version 2, any key whose definition starts
393      0963 3      !- with ADV is considered an advance key, and any key whose definition starts with
394      0964 3      !- BACK is considered a backup key. The entire text of the definition is inserted.
395      0965 3      !-
396      0966 3
397      0967 4      IF (EDT$$TST_KEYDEF (.TERM, UPLIT (BYTE ('ADV')), 3, 1) OR      !
398      0968 4      EDT$$TST_KEYDEF (.TERM, UPLIT (BYTE ('BACK')), 4, 1))
399      0969 4      THEN
400      0970 4      EDT$$STRN_KSTR (.TERM, .EXPAND, .ECHO);
401      0971 4
402      0972 3      EDT$$A_CMD_END = CH$PTR (.EDT$$A_CMD_BUF);
```


EDT\$KEYFMTSTR
V04-000

EDT\$KEYFMTSTR - translate a key string
EDT\$\$TRN_KSTR - translate a key string

M 8
16-Sep-1984 00:43:41
14-Sep-1984 12:23:22

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]KEYFMTSTR.BLI;1

Page 9
(3)

```
; 403      0973  3      RETURN (1);
; 404      0974  2      END;
; 405      0975  2
; 406      0976  2      RETURN (0);
; 407      0977  1      END;
```

! of routine EDT\$\$TRN_KSTR

.TITLE EDT\$KEYFMTSTR EDT\$KEYFMTSTR - translate a key s
tring

.IDENT \V04-000\

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

```
54 45 53 45 52 00000 P.AAA: .ASCII \RESET\
                                00005 .BLKB 3
                                56 44 41 00008 P.AAB: .ASCII \ADV\
                                0000B .BLKB 1
48 43 41 42 0000C P.AAC: .ASCII \BACK\
```

```
.EXTRN EDT$$SC_ERATOEOI
.EXTRN EDT$$MSG_BELL, EDT$$OUT_MSG
.EXTRN EDT$$SC_REVID, EDT$$SC_NONREVID
.EXTRN EDT$$OUT_FMTBUF
.EXTRN EDT$$KPAD_INP, EDT$$PUT_CMDCH
.EXTRN EDT$$SC_POSCSIF
.EXTRN EDT$$TI_ENBLAUTREP
.EXTRN EDT$$TI_INPCH, EDT$$TI_DELK
.EXTRN EDT$$TST_KEYDEF
.EXTRN EDT$$FIND_KEY, EDT$$G_MESSAGE_LINE
.EXTRN EDT$$G_CS_LNO, EDT$$G_CUR_COL
.EXTRN EDT$$A_SEC_BUF, EDT$$T_CMD_BUF
.EXTRN EDT$$A_CMD_BUF, EDT$$A_CMD_END
.EXTRN EDT$$A_CUR_BUF, EDT$$G_TI_TYP
.EXTRN EDT$$G_TIN_ECHOFLG
.EXTRN EDT$$G_TIN_ECHOPOS
.EXTRN EDT$$G_DIR_MOD, EDT$$G_DFLT_VERB
.EXTRN EDT$$G_SEA_STRLEN
```

OFFC 00000

```
5B 00000000G 00 9E 00002
5A 00000000G 00 9E 00009
59 00000000G 00 9E 00010
58 00000000G 00 9E 00017
57 00000000G 00 9E 0001E
5E          0C C2 00025
          5E DD 00028
          04 AC DD 0002A
00000000G 00 02 FB 0002D
03          50 E8 00034
          018D 31 00037
7E          05 7D 0003A 1$:
          B0 AF 9F 0003D
          04 AC DD 00040
69          04 FB 00043
3B          50 E9 00046
67          68 9E 00049
```

```
.ENTRY EDT$$TRN_KSTR, Save R2,R3,R4,R5,R6,R7,R8,- ; 0690
R9,R10,R11
MOVAB EDT$$G_MESSAGE_LINE, R11
MOVAB EDT$$A_CMD_END, R10
MOVAB EDT$$TST_KEYDEF, R9
MOVAB EDT$$T_CMD_BUF, R8
MOVAB EDT$$A_CMD_BUF, R7
SUBL2 #12, SP
PUSHL SP
PUSHL K
CALLS #2, EDT$$FIND_KEY
BLBS R0, 1$
BRW 22$
MOVQ #5, -(SP)
PUSHAB P.AAA
PUSHL K
CALLS #4, EDT$$TST_KEYDEF
BLBC R0, 3$
MOVAB EDT$$T_CMD_BUF, EDT$$A_CMD_BUF ; 0811
```

0794

0805

EDT\$KEYFMTSTR
V04-000

EDT\$KEYFMTSTR - translate a key string
EDT\$STRN_KSTR - translate a key string

N 8
16-Sep-1984 00:43:41
14-Sep-1984 12:23:22

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]KEYFMTSTR.BLI;1

Page 10
(3)

7E	00000000G	00	6A	0C	68	9E	0004C	MOVAB	EDT\$ST_CMD_BUF, EDT\$A_CMD_END	0812
	00000000G	00	1B		AC	E9	0004F	BLBC	ECHO, 2\$	0814
	00000000G	00			7E	D4	00053	CLRL	-(SP)	0817
	00000000G	00	6B		01	C1	00055	ADDL3	#1, EDT\$G_MESSAGE_LINE, -(SP)	
		00	00		02	FB	00059	CALLS	#2, EDT\$SC_POSCSIF	0818
		00	00		00	FB	00060	CALLS	#0, EDT\$SC_ERATOEOI	0819
		00	00		00	FB	00067	CALLS	#0, EDT\$OUT_FMTBUF	0824
		00	00000000G		00	D4	0006E	CLRL	EDT\$G_DFLT_VERB	0827
		00	00		01	D0	00074	MOVL	#1, EDT\$G_DIR_MOD	0828
		00	00000000G		00	D4	0007B	CLRL	EDT\$A_SEL_BUF	0829
					0146	31	00081	BRW	23\$	0835
			50		6E	D0	00084	MOVL	KEY_PTR, R0	
			52	07	A0	9E	00087	MOVAB	7(R0), TRAN_POINT	
			55	06	A0	9A	0008B	MOVZBL	6(R0), TRAN_END	0836
			55		52	C0	0008F	ADDL2	TRAN_POINT, TRAN_END	
				08	AE	D4	00092	CLRL	TERM	0837
			55		52	D1	00095	CMPL	TRAN_POINT, TRAN_END	0842
					03	1F	00098	BLSSU	5\$	
					00E0	31	0009A	BRW	19\$	
			3F		62	91	0009D	CMPB	(TRAN_POINT), #63	0844
					03	13	000A0	BEQL	7\$	
					00C8	31	000A2	BRW	17\$	
			F9	08	AC	E9	000A5	BLBC	EXPAND, 6\$	
					52	D6	000A9	INCL	TRAN_POINT	0857
			3F		62	91	000AB	CMPB	(TRAN_POINT), #63	0862
					F2	13	000AE	BEQL	6\$	
					53	D4	000B0	CLRL	P_LEN	0870
			2A		62	91	000B2	CMPB	(TRAN_POINT), #42	0875
					07	12	000B5	BNEQ	8\$	
					52	D6	000B7	INCL	TRAN_POINT	0878
			56		0D	D0	000B9	MOVL	#13, ALT_TERM	0879
					03	11	000BC	BRB	9\$	0875
			56		1B	D0	000BE	MOVL	#27, ALT_TERM	0882
			50		62	9A	000C1	MOVZBL	(TRAN_POINT), QUOTE	0887
			27		50	D1	000C4	CMPL	QUOTE, #39	0889
					05	13	000C7	BEQL	10\$	
			22		50	D1	000C9	CMPL	QUOTE, #34	
					16	12	000CC	BNEQ	12\$	
					52	D6	000CE	INCL	TRAN_POINT	0895
			51		52	D0	000D0	MOVL	TRAN_POINT, P_ADDR	0896
			52		55	D1	000D3	CMPL	TRAN_END, TRAN_POINT	0898
					0C	1B	000D6	BLEQU	12\$	
			54		82	9A	000D8	MOVZBL	(TRAN_POINT)+, R4	
			50		54	D1	000DB	CMPL	R4, QUOTE	
					04	13	000DE	BEQL	12\$	
					53	D6	000E0	INCL	P_LEN	0899
					EF	11	000E2	BRB	1T\$	
					0A	BB	000E4	PUSHR	#*M<R1,R3>	0906
					7E	D4	000E6	CLRL	-(SP)	
					6B	DD	000E8	PUSHL	EDT\$G_MESSAGE_LINE	
	00000000G	00			04	FB	000EA	CALLS	#4, EDT\$OUT_MSG	
					01	DD	000F1	PUSHL	#1	0907
	00000000G	00			01	FB	000F3	CALLS	#1, EDT\$TI_ENBLAUTREP	
	00000000G	00			53	C0	000FA	ADDL2	P_LEN, EDT\$G_TIN_ECHOPOS	0908
	00000000G	00			01	D0	00101	MOVL	#T, EDT\$G_TIN_ECHOFLG	0909
				04	AE	9F	0010B	PUSHAB	C	0914
	00000000G	00			01	FB	0010B	CALLS	#1, EDT\$TI_INPCH	

			50	D5	00112	TSTL	R0		
			14	13	00114	BEQL	13\$		
		08	AE	9F	00116	PUSHAB	TERM		0916
			56	DD	00119	PUSHL	ALT_TERM		
		0C	AE	DD	0011B	PUSHL	C		
00000000G	00		03	FB	0011E	CALLS	#3, EDT\$KPAD_INP		
	02		50	D1	00125	CMPL	STATUS, #2		0918
			04	12	00128	BNEQ	14\$		
	50		02	D0	0012A	MOVL	#2, R0		
				04	0012D	RET			
			50	D5	0012E	TSTL	STATUS		0920
			27	12	00130	BNEQ	16\$		
00000000G	00		00	FB	00132	CALLS	#0, EDT\$SSC_NONREVID		0926
	50	00000000G	00	D0	00139	MOVL	EDT\$G_CS_LNO, R0		0928
			0F	19	00140	BLSS	15\$		
		00000000G	00	DD	00142	PUSHL	EDT\$G_CUR_COL		
			50	DD	00148	PUSHL	R0		
00000000G	00		02	FB	0014A	CALLS	#2, EDT\$SSC_POSCSIF		
	67		68	9E	00151	MOVAB	EDT\$T_CMD_BUF, EDT\$A_CMD_BUF		0930
	6A		68	9E	00154	MOVAB	EDT\$T_CMD_BUF, EDT\$A_CMD_END		0931
			71	11	00157	BRB	23\$		0932
			7E	D4	00159	CLRL	-(SP)		0939
			6B	DD	0015B	PUSHL	EDT\$G_MESSAGE_LINE		
00000000G	00		02	FB	0015D	CALLS	#2, EDT\$SSC_POSCSIF		
00000000G	00		00	FB	00164	CALLS	#0, EDT\$OUT_FMTBUF		0940
			0D	11	0016B	BRB	18\$		0862
		0C	AC	DD	0016D	PUSHL	ECHO		0945
	7E		82	9A	00170	MOVZBL	(TRAN_POINT)+, -(SP)		
00000000G	00		02	FB	00173	CALLS	#2, EDT\$SPUT_CMDCH		
			FF18	31	0017A	BRW	4\$		0844
	50		67	D0	0017D	MOVL	EDT\$A_CMD_BUF, R0		0952
	2E		A0	91	00180	CMPL	-1(R0), #4\$		
		FF	48	12	00184	BNEQ	24\$		
			67	D7	00186	DECL	EDT\$A_CMD_BUF		0955
	0D	0C	AC	E9	00188	BLBC	ECHO, 20\$		0957
	50		67	D0	0018C	MOVL	EDT\$A_CMD_BUF, R0		
	7E		60	9A	0018F	MOVZBL	(R0), -(SP)		
00000000G	00		01	FB	00192	CALLS	#1, EDT\$TI_DELK		
			01	DD	00199	PUSHL	#1		0967
			03	DD	0019B	PUSHL	#3		
		FE57	CF	9F	0019D	PUSHAB	P.AAB		
		14	AE	DD	001A1	PUSHL	TERM		
	69		04	FB	001A4	CALLS	#4, EDT\$TST_KEYDEF		
	11		50	E8	001A7	BLBS	R0, 21\$		
			01	DD	001AA	PUSHL	#1		0968
			04	DD	001AC	PUSHL	#4		
		FE4A	CF	9F	001AE	PUSHAB	P.AAC		
		14	AE	DD	001B2	PUSHL	TERM		
	69		04	FB	001B5	CALLS	#4, EDT\$TST_KEYDEF		
	0C		50	E9	001B8	BLBC	R0, 22\$		
	7E	08	AC	7D	001BB	MOVQ	EXPAND, -(SP)		0970
		10	AE	DD	001BF	PUSHL	TERM		
FE39	CF		03	FB	001C2	CALLS	#3, EDT\$STRN_KSTR		
	6A		67	D0	001C7	MOVL	EDT\$A_CMD_BUF, EDT\$A_CMD_END		0972
	50		01	D0	001CA	MOVL	#1, R0		0973
				04	001CD	RET			
			50	D4	001CE	CLRL	R0		0976

EDT\$KEYFMTSTR - translate a key string
EDT\$STRN_KSTR - translate a key string

16^c-sep-1984 00:43:41
14⁹-sep-1984 12:23:22

```
VAX-11 BLISS-32 V4.0-742
[EDT.SRC]KEYFMTSTR.BLI;1
```

Page 12
(3)

04 001D0

RET

: 0977

```
; Routine Size: 465 bytes,    Routine Base: _EDT$CODE + 0010
```

```

: 408      0978 1
: 409      0979 1 !<BLF/PAGE>

```

EDT
V04

EDT\$KEYFMTSTR
V04-000

EDT\$KEYFMTSTR - translate a key string
EDT\$STRN_KSTR - translate a key string

D 9
16-Sep-1984 00:43:41 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:23:22 [EDT.SRC]KEYFMTSTR.BLI;1

: 411 0980 1 END
: 412 0981 1
: 413 0982 0 ELUDOM

! of module EDT\$KEYFMTSTR

PSECT SUMMARY

Name	Bytes	Attributes
_EDT\$CODE	481	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	6	1	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	5	14	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\$:KEYFMTSTR/OBJ=OBJ\$:KEYFMTSTR MSRC\$:KEYFMTSTR.BLI/UPDATE=(ENH\$:KEYFMTSTR)

: Size: 465 code + 16 data bytes
: Run Time: 00:23.6
: Elapsed Time: 00:28.0
: Lines/CPU Min: 2497
: Lexemes/CPU-Min: 7571
: Memory Used: 160 pages
: Compilation Complete

0135 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY