

```

EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD TTTTTTTTTTTTTTTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEEEEEEEEEEEEE DDD DDD TTT
EEEEEEEEEEEEEE DDD DDD TTT
EEEEEEEEEEEEEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEE DDD DDD TTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD TTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD TTT
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD TTT

```

EXE

Mod

EDT

ED

ED
EDED
EDED
EDED
ED

ED

ED

ED

ED

ED
EDSYN
LBA

110

```

KK      KK      EEEEEEEEEEE      YY      YY      CCCCCCCCC      HH      HH      RRRRRRRR
KK      KK      EEEEEEEEEEE      YY      YY      CCCCCCCCC      HH      HH      RRRRRRRR
KK      KK      EE      YY      YY      CC      HH      HH      RR      RR
KK      KK      EE      YY      YY      CC      HH      HH      RR      RR
KK      KK      EE      YY      YY      CC      HH      HH      RR      RR
KK      KK      EE      YY      YY      CC      HH      HH      RR      RR
KKKKKKK      EEEEEEEEE      YY      CC      HHHHHHHHHH      RRRRRRRR
KKKKKKK      EEEEEEEEE      YY      CC      HHHHHHHHHH      RRRRRRRR
KK      KK      EE      YY      CC      HH      HH      RR      RR
KK      KK      EE      YY      CC      HH      HH      RR      RR
KK      KK      EE      YY      CC      HH      HH      RR      RR
KK      KK      EE      YY      CC      HH      HH      RR      RR
KK      KK      EEEEEEEEEEE      YY      CCCCCCCCC      HH      HH      RR      RR
KK      KK      EEEEEEEEEEE      YY      CCCCCCCCC      HH      HH      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
LLLLLLLLLLLLLL  IIIIII               SSSSSSSS
LLLLLLLLLLLLLL  IIIIII               SSSSSSSS

```



```
0001 0 %TITLE 'EDT$KEYCHR - get next command character'
0002 0 MODULE EDT$KEYCHR ( ! Get next command character
0003 0 IDENT = 'V04-000' ! File: KEYCHR.BLI Edit: JBS1022
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 Get next command character.
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 EDT$$NXT CMDCH from module KEYTRAN.
0046 1 1-002 - Regularize headers. JBS 06-Mar-1981
0047 1 1-003 - Add a check for repeat counts allowed or not. STS 26-Aug-1981
0048 1 1-004 - Fixed problem with norepeat so it doesn't insert number.
0049 1 STS 27-Aug-1981
0050 1 1-005 - Add return value for end of journal file. JBS 02-Oct-1981
0051 1 1-006 - Don't pass values of keypad except arrow keys. STS 15-Apr-1982
0052 1 1-007 - Accept a flag indicating validity of repeat counts. STS 16-Jun-1982
0053 1 1-008 - Remove reference to TI STARTECHO. SMB 22-Jun-1982
0054 1 1-009 - Change numeric test. JBS 19-Jul-1982
0055 1 1-010 - Don't ring bell if quiet set. STS 09-Aug-1982
0056 1 1-011 - New implementation of defined keys. JBS 13-Aug-1982
0057 1 1-012 - Don't suppress all keys in NOKEYPAD mode. JBS 16-Aug-1982
```

EDTSKEYCHR
V04-000

EDTSKEYCHR - get next command character

6 5
16-Sep-1984 00:41:34
14-Sep-1984 12:23:20

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

```

: 58      0058 1 1-013 - Fix up norepeat case. JBS 16-Aug-1982
: 59      0059 1 1-014 - Allow for 8-bit keyboards. JBS 17-Aug-1982
: 60      0060 1 1-015 - Add SS3 for 8-bit keyboards. JBS 20-Aug-1982
: 61      0061 1 1-016 - Erase the message line after the first keystroke, if it has
: 62      0062 1 a message on it. JBS 06-Oct-1982
: 63      0063 1 1-017 - Output the format buffer just before waiting for input, in case
: 64      0064 1 there is anything in it. JBS 07-Oct-1982
: 65      0065 1 1-018 - Don't clear the message line until an entire key sequence has been read.
: 66      0066 1 JBS 09-Oct-1982
: 67      0067 1 1-019 - Output the format buffer in another case of waiting for input. JBS 09-Oct-1982
: 68      0068 1 1-020 - Change the call to EDT$STST_KEYDEF. JBS 14-Dec-1982
: 69      0069 1 1-021 - Complete the implementation of 8-bit keyboards. JBS 20-Jan-1983
: 70      0070 1 1-022 - Add a conditional for VT220 support. JBS 11-Feb-1983
: 71      0071 1 --
: 72      0072 1
```


EDTSKEYCHR
V04-000

EDTSKEYCHR - get next command character
Declarations

H 5
16-Sep-1984 00:41:34
14-Sep-1984 12:23:20

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

```
: 74      0073 1 %SBTTL 'Declarations'
: 75      0074 1
: 76      0075 1  TABLE OF CONTENTS:
: 77      0076 1
: 78      0077 1
: 79      0078 1  REQUIRE 'EDTSRC:TRAROUNAM';
: 80      0517 1
: 81      0518 1  FORWARD ROUTINE
: 82      0519 1      EDT$NXT_CMDCH;
: 83      0520 1
: 84      0521 1
: 85      0522 1  INCLUDE FILES:
: 86      0523 1
: 87      0524 1
: 88      0525 1  REQUIRE 'EDTSRC:EDTREQ';
: 89      0660 1
: 90      0661 1  LIBRARY 'EDTSRC:KEYPADDEF';
: 91      0662 1
: 92      0663 1  LIBRARY 'EDTSRC:SUPPORTS';
: 93      0664 1
: 94      0665 1
: 95      0666 1  MACROS:
: 96      0667 1
: 97      0668 1      NONE
: 98      0669 1
: 99      0670 1  EQUATED SYMBOLS:
: 100     0671 1
: 101     0672 1      NONE
: 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$$NXT_CMDCH - get next command character'
112 0682 1
113 0683 1 GLOBAL ROUTINE EDT$$NXT_CMDCH (
114 0684 1 C
115 0685 1 REPEAT
116 0686 1 ) =
117 0687 1
118 0688 1 !++
119 0689 1 FUNCTIONAL DESCRIPTION:
120 0690 1
121 0691 1 Get the next command character. Keypad keys are converted to their
122 0692 1 numeric equivalent and the functions of GOLD are handled here.
123 0693 1
124 0694 1 FORMAL PARAMETERS:
125 0695 1
126 0696 1 C The address of a fullword to receive the character.
127 0697 1
128 0698 1 REPEAT Flag indicating whether to accept repeat counts.
129 0699 1
130 0700 1 IMPLICIT INPUTS:
131 0701 1
132 0702 1 EDT$$T_CMD_BUF
133 0703 1 EDT$$G_KPAD
134 0704 1 EDT$$A_CMD_BUF
135 0705 1 EDT$$G_RPT
136 0706 1 EDT$$A_CMD_END
137 0707 1 EDT$$G_MSGFLG
138 0708 1
139 0709 1 IMPLICIT OUTPUTS:
140 0710 1
141 0711 1 EDT$$A_CMD_BUF
142 0712 1
143 0713 1 ROUTINE VALUE:
144 0714 1
145 0715 1 0 = control U typed, no command.
146 0716 1 1 = a command key was typed.
147 0717 1 2 = end of journal file.
148 0718 1
149 0719 1 SIDE EFFECTS:
150 0720 1
151 0721 1 NONE
152 0722 1
153 0723 1 --
154 0724 1
155 0725 2 BEGIN
156 0726 2
157 0727 2 EXTERNAL ROUTINE
158 0728 2 EDT$$SC_REVID,
159 0729 2 EDT$$PUT_CMDCH : NOVALUE,
160 0730 2 EDT$$TRN_KPADK,
161 0731 2 EDT$$TI_INPCH,
162 0732 2 EDT$$TI_DELK,
163 0733 2 EDT$$TI_ECHOCH,
164 0734 2 EDT$$ERA_MSGLN,
165 0735 2 EDT$$RING_BELL,
166 0736 2 EDT$$TST_KEYDEF,
167 0737 2 EDT$$OUT_FMTBUF;

! Turn on reverse video
! Put a character in the command buffer
! Read an escape sequence

! Erase the message lines
! Ring the bell on the terminal
! Test a key to see if it is defined as a particular string
! Output the format buffer, if it is non-empty
```



```
168 0738 2
169 0739 2
170 0740 2      EXTERNAL
171 0741 2          EDT$G_MSGFLG,
172 0742 2          EDT$G_QUIET,
173 0743 2          EDT$T_CMD_BUF,
174 0744 2          EDT$G_KPAD,
175 0745 2          EDT$G_RPT,
176 0746 2          EDT$A_CMD_END
177 0747 2
178 L 0748 2      %IF SUPPORT_VT220
179 0749 2      %THEN
180 0750 2
181 0751 2          EDT$B_CHAR_INFO : BLOCKVECTOR [256, 1, BYTE] ! Information about characters
182 0752 2      %FI
183 0753 2
184 0754 2      ;
185 0755 2
186 0756 2      LOCAL
187 0757 2          SAVE_POINT,
188 0758 2          MY_C;
189 0759 2
190 0760 2          SAVE_POINT = .EDT$A_CMD_BUF;
191 0761 2      !+
192 0762 2      ! Make sure the user sees anything which might be in the format buffer.
193 0763 2      !-
194 0764 2          EDT$OUT_FMTBUF ();
195 0765 2      !+
196 0766 2      ! Get a character.
197 0767 2      !-
198 0768 2
199 0769 2          IF (EDT$TI_INPCH (MY_C) EQL 0) THEN RETURN (2);
200 0770 2
201 0771 2      !+
202 0772 2      ! If the character is an escape, CSI or SS3, then look for a keypad sequence.
203 0773 2      !-
204 0774 2
205 L 0775 2      %IF SUPPORT_VT220
206 0776 2      %THEN
207 0777 2
208 0778 2          IF ((.MY_C EQL ASC_K_ESC) OR (.MY_C EQL ASC_K_CSI) OR (.MY_C EQL ASC_K_SS3))
209 0779 2          THEN
210 U 0780 2      %ELSE
211 U 0781 2
212 U 0782 2          IF (.MY_C EQL ASC_K_ESC)
213 U 0783 2          THEN
214 0784 2      %FI
215 0785 2
216 0786 2          BEGIN
217 0787 2      !+
218 0788 2      ! Translate keypad character.
219 0789 2      !-
220 0790 2
221 0791 2          IF (EDT$TRN_KPADK (MY_C) EQL 0) THEN RETURN (2);
222 0792 2
223 0793 2      !+
224 0794 2      ! If there is any text on the message line, erase it, since the user
```

```
225 0795 3 ! has now had an opportunity to read it.
226 0796 3 !-
227 0797 3
228 0798 3 IF (.EDT$$G_MSGFLG NEQ 0) THEN EDT$$ERA_MSGLN ();
229 0799 3
230 0800 3 IF ( NOT .EDT$$G_KPAD)
231 0801 3 THEN
232 0802 3 BEGIN
233 0803 3
234 0804 3 IF ((.MY_C EQL K_UP) OR (.MY_C EQL K_DOWN) OR (.MY_C EQL K_RIGHT) OR (.MY_C EQL K_LEFT))
235 0805 3 THEN
236 0806 3 .C = .MY_C
237 0807 3 ELSE
238 0808 3 .C = K_PF1;
239 0809 3
240 0810 3 RETURN (1);
241 0811 3 END;
242 0812 3
243 0813 3 END;
244 0814 3
245 0815 2 !+
246 0816 2 !- If the key is defined as GOLD, handle it here.
247 0817 2 !-
248 0818 2
249 0819 2 WHILE EDT$$TST_KEYDEF (.MY_C, UPLIT (BYTE ('GOLD')), 4, 0) DO
250 0820 2 BEGIN
251 0821 2 !+
252 0822 2 !- Look at the next character. It should be either a digit, a sign
253 0823 2 !- or a letter.
254 0824 2 !-
255 0825 2 EDT$$OUT_FMTBUF ();
256 0826 2
257 0827 2 IF (EDT$$TI_INPCH (MY_C) EQL 0) THEN RETURN (2);
258 0828 2
259 0829 2 EDT$$SC_REVID ();
260 0830 2
261 L 0831 2 %IF SUPPORT_VT220
262 0832 2 %THEN
263 0833 2
264 0834 2 IF (((.EDT$$B_CHAR_INFO [.MY_C, 0, 0, 8, 0] EQL %X'F0') OR (.MY_C EQL %C'-')) AND .REPEAT)
265 0835 2 THEN
266 U 0836 2 %ELSE
267 U 0837 2
268 U 0838 2 IF ((((((MY_C GEQ %C'0') AND (.MY_C LEQ %C'9')) OR (.MY_C EQL %C'-')) AND .REPEAT)
269 U 0839 2 THEN
270 0840 2 %FI
271 0841 2
272 0842 2 BEGIN
273 0843 2 !+
274 0844 2 !- Start of a repeat count. If this was not the first key pressed
275 0845 2 !- then re-start the count by clearing the buffer back to the
276 0846 2 !- point where we started.
277 0847 2 !-
278 0848 2
279 0849 2 IF (.EDT$$G_RPT EQL 0)
280 0850 2 THEN
281 0851 2 BEGIN
```



```
282 0852 5
283 0853 5 IF ( NOT .EDT$$G_QUIET) THEN EDT$$RING_BELL ();
284 0854 5
285 0855 5 EDT$$OUT_FMTBUF ();
286 0856 5
287 0857 5 IF (EDT$$TI_INPCH (MY_C) EQL 0) THEN RETURN (2);
288 0858 5
289 L 0859 5 %IF SUPPORT_VT220
290 0860 5 %THEN
291 0861 5
292 0862 6 IF ((.MY_C EQL ASC_K_ESC) OR (.MY_C EQL ASC_K_CSI) OR (.MY_C EQL ASC_K_SS3))
293 0863 5 THEN
294 U 0864 5 %ELSE
295 0865 5
296 0866 5 IF (.MY_C EQL ASC_K_ESC)
297 U 0867 5 THEN
298 0868 5 %FI
299 0869 5
300 0870 6 BEGIN
301 0871 6
302 0872 6 IF (EDT$$TRN_KPADK (MY_C) EQL 0) THEN RETURN (2);
303 0873 6
304 0874 5 END;
305 0875 5
306 0876 5 END
307 0877 4 ELSE
308 0878 5 BEGIN
309 0879 5
310 0880 6 IF CH$PTR_NEQ (.EDT$$A_CMD_BUF, .SAVE_POINT)
311 0881 5 THEN
312 0882 6 BEGIN
313 0883 6 EDT$$A_CMD_BUF = .SAVE_POINT;
314 0884 6 EDT$$ERA_MSGLN ();
315 0885 5 END;
316 0886 5
317 0887 5 !+
318 0888 5 !- Now continue reading and echoing characters until a non-digit is found.
319 0889 5 !-
320 0890 5
321 0891 5 DO
322 0892 6 BEGIN
323 0893 6 EDT$$TI_ECHOCH (.MY_C);
324 0894 6 EDT$$PUT_CMDCH (.MY_C, 0);
325 0895 6 EDT$$OUT_FMTBUF ();
326 0896 6
327 0897 6 IF (EDT$$TI_INPCH (MY_C) EQL 0) THEN RETURN (2);
328 0898 6
329 0899 6 !+
330 0900 6 !- Look for delete and CTRL/U
331 0901 6 !-
332 0902 6
333 0903 6 WHILE (.MY_C EQL ASC_K_DEL) DO
334 0904 7 BEGIN
335 0905 7
336 0906 8 IF CH$PTR_NEQ (.EDT$$A_CMD_BUF, .SAVE_POINT)
337 0907 7 THEN
338 0908 8 BEGIN
```

EDT\$KEYCHR
V04-000

EDT\$KEYCHR - get next command character
EDT\$\$NXT_CMDCH - get next command character

M 5
16-Sep-1984 00:41:34
14-Sep-1984 12:23:20

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]KEYCHR.BLI;1

Page 8
(3)

```

339      0909      8      EDT$$A_CMD_BUF = CH$PLUS (.EDT$$A_CMD_BUF, -1);
340      0910      8      EDT$$TI_DECK (CH$RCHAR (.EDT$$A_CMD_BUF));
341      0911      7      END;
342      0912      7
343      0913      7      EDT$$OUT_FMTBUF ();
344      0914      7
345      0915      7      IF (EDT$$TI_INPCH (MY_C) EQL 0) THEN RETURN (2);
346      0916      7
347      0917      6      END;
348      0918      6
349      0919      7      IF (.MY_C EQL ASC_K_CTRL_U)
350      0920      6      THEN
351      0921      7      BEGIN
352      0922      7      EDT$$ERA_MSGLN ();
353      0923      7      EDT$$A_CMD_END = EDT$$T_CMD_BUF;
354      0924      7      RETURN (0);
355      0925      6      END;
356      0926      6
357      0927      6      END
358      0928      5      UNTIL
359      0929      5
360      L 0930      5      %IF SUPPORT_VT220
361      0931      5      %THEN
362      0932      6      (.EDT$$B_CHAR_INFO [.MY_C, 0, 0, 8, 0] NEQ %X'FO')
363      0933      6      %ELSE
364      U 0934      6      ((.MY_C LSS %C'O') OR (.MY_C GTR %C'9'))
365      0935      6      %FI
366      0936      6
367      0937      5      ;
368      0938      5      !+
369      0939      5      ! If the repeat sequence was ended by an escape, CSI or SS3 then get the key.
370      0940      5      !-
371      0941      5
372      L 0942      5      %IF SUPPORT_VT220
373      0943      5      %THEN
374      0944      5
375      0945      6      IF ((.MY_C EQL ASC_K_ESC) OR (.MY_C EQL ASC_K_CSI) OR (.MY_C EQL ASC_K_SS3))
376      0946      5      THEN
377      U 0947      5      %ELSE
378      0948      5
379      U 0949      5      IF (.MY_C EQL ASC_K_ESC)
380      U 0950      5      THEN
381      0951      5      %FI
382      0952      5
383      0953      6      BEGIN
384      0954      6
385      0955      6      IF (EDT$$TRN_KPADK (MY_C) EQL 0) THEN RETURN (2);
386      0956      6
387      0957      5      END;
388      0958      5
389      0959      5      END
390      0960      5
391      0961      4      ELSE
392      0962      5      END
393      0963      5
394      L 0964      5      %IF SUPPORT_VT220
395      0965      5      %THEN
```


EDTSKEYCHR
V04-000

EDTSKEYCHR - get next command character
EDT\$\$NXT_CMDCH - get next command character

N 5
16-Sep-1984 00:41:34
14-Sep-1984 12:23:20

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]KEYCHR.BLI;1

Page 9
(3)

EDT
V04

```

396      0966      3
397      0967      4
398      0968      3      IF ((.MY_C EQL ASC_K_ESC) OR (.MY_C EQL ASC_K_CSI) OR (.MY_C EQL ASC_K_SS3))
399      0969      3      THEN
400      0970      3      %ELSE
401      0971      3
402      0972      3      IF (.MY_C EQL ASC_K_ESC)
403      0973      3      THEN
404      0974      3      %FI
405      0975      3      !+
406      0976      3      ! Here if we had gold followed by an escape, CSI or SS3.
407      0977      3      ! Translate the sequence and use the golded function of the key.
408      0978      3      !-
409      0979      4      BEGIN
410      0980      4
411      0981      4      IF (EDT$$TRN_KPADK (MY_C) EQL 0) THEN RETURN (2);
412      0982      4
413      0983      4      MY_C = .MY_C + K_GOLD_BASE;
414      0984      4      END
415      0985      3      ELSE
416      0986      4      BEGIN
417      0987      4      !+
418      0988      4      ! Here if we had gold followed by a character from the main keyboard.
419      0989      4      !-
420      0990      4
421      0991      4      %IF SUPPORT_VT220
422      0992      4      %THEN
423      0993      4
424      0994      4      IF .EDT$$B_CHAR_INFO [.MY_C, 0, 0, 1, 0]      ! Lower case
425      0995      4      THEN
426      0996      4      %ELSE
427      0997      4
428      0998      4      IF ((.MY_C GEQ %C'a') AND (.MY_C LEQ %C'z'))
429      0999      4      THEN
430      1000      4      %FI
431      1001      4
432      1002      4      MY_C = .MY_C - %C'a' + %C'A';      ! Force to upper
433      1003      4
434      1004      4      MY_C = .MY_C + K_GOLD_BASE;
435      1005      3      END;
436      1006      3
437      1007      2      END;
438      1008      2
439      1009      2      !+
440      1010      2      ! Return the coded character.
441      1011      2      !-
442      1012      2      .C = .MY_C;
443      1013      2      RETURN (T);
444      1014      1      END;

```

! of routine EDT\$\$NXT_CMDCH

.TITLE EDTSKEYCHR EDTSKEYCHR - get next command charac
ter
.IDENT \V04-000\
.PSECT _EDT\$CODE,NOWRT, SHR, PIC,2

44 4C 4F 47 00000 P.AAA: .ASCII \GOLD\

.EXTRN EDT\$\$SC REVID, EDT\$\$PUT_CMDCH
.EXTRN EDT\$\$STRN_KPADK, EDT\$\$TI_INPCH
.EXTRN EDT\$\$TI_DELK, EDT\$\$TI_ECHOCH
.EXTRN EDT\$\$ERA_MSGLN, EDT\$\$RING_BELL
.EXTRN EDT\$\$TST_KEYDEF
.EXTRN EDT\$\$OUT_FMTBUF
.EXTRN EDT\$\$G_MSGFLG, EDT\$\$G_QUIET
.EXTRN EDT\$\$T_CMD_BUF, EDT\$\$A_CMD_BUF
.EXTRN EDT\$\$G_KPAD, EDT\$\$G_RPT
.EXTRN EDT\$\$A_CMD_END, EDT\$\$B_CHAR_INFO

07FC 00000

.ENTRY EDT\$\$NXT_CMDCH, Save R2,R3,R4,R5,R6,R7,R8,- R9,R10 : 0683
MOVAB EDT\$\$B_CHAR_INFO, R10
MOVAB EDT\$\$ERA_MSGLN, R9
MOVAB EDT\$\$STRN_KPADK, R8
MOVAB EDT\$\$TI_INPCH, R7
MOVAB EDT\$\$OUT_FMTBUF, R6
MOVAB EDT\$\$A_CMD_BUF, R5
SUBL2 #4, SP
MOVL EDT\$\$A_CMD_BUF, SAVE_POINT : 0760
CALLS #0, EDT\$\$OUT_FMTBUF : 0764
PUSHL SP : 0769
CALLS #1, EDT\$\$TI_INPCH
TSTL R0
BEQL 2\$: 0778
CMPL MY_C, #27
BEQL 1\$
CMPL MY_C, #155
BEQL 1\$
CMPL MY_C, #143
BNEQ 7\$: 0791
PUSHL SP
CALLS #1, EDT\$\$STRN_KPADK
TSTL R0
BEQL 9\$: 0798
TSTL EDT\$\$G_MSGFLG
BEQL 3\$
CALLS #0, EDT\$\$ERA_MSGLN
BLBS EDT\$\$G_KPAD, 7\$: 0800
MOVL MY_C, R0 : 0804
CMPL R0, #312
BEQL 4\$
CMPL R0, #313
BEQL 4\$
CMPL R0, #314
BEQL 4\$
CMPL R0, #315
BNEQ 5\$
MOVL R0, ac : 0806
BRB 6\$
MOVZWL #320, ac : 0808
BRW 31\$: 0810
MOVQ #4, -(SP) : 0819
PUSHAB P.AAA

5A 00000000G 00 9E 00002
59 00000000G 00 9E 00009
58 00000000G 00 9E 00010
57 00000000G 00 9E 00017
56 00000000G 00 9E 0001E
55 00000000G 00 9E 00025
5E 04 C2 0002C
54 65 D0 0002F
66 00 FB 00032
5E DD 00035
67 01 FB 00037
50 D5 0003A
1E 13 0003C
1B 6E D1 0003E
12 13 00041
0000009B 8F 6E D1 00043
09 13 0004A
0000008F 8F 6E D1 0004C
51 12 00053
5E DD 00055 1\$:
68 01 FB 00057
50 D5 0005A
69 13 0005C 2\$:
00000000G 00 D5 0005E
03 13 00064
69 00 FB 00066
36 00000000G 00 E8 00069 3\$:
50 6E D0 00070
00000138 8F 50 D1 00073
1B 13 0007A
00000139 8F 50 D1 0007C
12 13 00083
0000013A 8F 50 D1 00085
09 13 0008C
0000013B 8F 50 D1 0008E
06 12 00095
04 BC 50 D0 00097 4\$:
06 11 0009B
04 BC 0140 8F 3C 0009D 5\$:
0146 31 000A3 6\$:
7E 04 7D 000A6 7\$:
FF4F CF 9F 000A9

00000000G	00	0C	AE	DD	000AD	PUSHL	MY_C	:	
	03		04	FB	000B0	CALLS	#4, EDT\$\$TST_KEYDEF	:	
			50	E8	000B7	BLBS	R0, 8\$	:	
	66		012B	31	000BA	BRW	30\$	:	
			00	FB	000BD	8\$:	CALLS #0, EDT\$\$OUT_FMTBUF	:	0825
	67		5E	DD	000C0	PUSHL	SP	:	0827
			01	FB	000C2	CALLS	#1, EDT\$\$TI_INPCH	:	
			50	D5	000C5	TSTL	R0	:	
			62	13	000C7	9\$:	BEQL 15\$	:	
00000000G	00		00	FB	000C9	CALLS	#0, EDT\$\$SC_REVID	:	0829
	52		6E	D0	000D0	MOVL	MY_C, R2	:	0834
53	52		5A	C1	000D3	ADDL3	R10, R2, R3	:	
	8F	F0	63	91	000D7	CMPB	(R3), #240	:	
			08	13	000DB	BEQL	11\$	:	
	2D		52	D1	000DD	CMPL	R2, #45	:	
			03	13	000E0	BEQL	11\$	:	
			00CE	31	000E2	10\$:	BRW 25\$	:	
	F9	08	AC	E9	000E5	11\$:	BLBC REPEAT, 10\$	:	
		00000000G	00	D5	000E9	TSTL	EDT\$\$G_RPT	:	0849
			3C	12	000EF	BNEQ	16\$	:	
00000000G	07	00000000G	00	E8	000F1	BLBS	EDT\$\$G_QUIET, 12\$	:	0853
	00		00	FB	000F8	CALLS	#0, EDT\$\$RING_BELL	:	
	66		00	FB	000FF	12\$:	CALLS #0, EDT\$\$OUT_FMTBUF	:	0855
			5E	DD	00102	PUSHL	SP	:	0857
	67		01	FB	00104	CALLS	#1, EDT\$\$TI_INPCH	:	
			50	D5	00107	TSTL	R0	:	
			6F	13	00109	BEQL	21\$	:	
	1B		6E	D1	0010B	CMPL	MY_C, #27	:	0862
			12	13	0010E	BEQL	14\$	:	
0000009B	8F		6E	D1	00110	CMPL	MY_C, #155	:	
			09	13	00117	BEQL	14\$	:	
0000008F	8F		6E	D1	00119	CMPL	MY_C, #143	:	
			84	12	00120	13\$:	BNEQ 7\$	:	
			5E	DD	00122	14\$:	PUSHL SP	:	0872
	66		01	FB	00124	CALLS	#1, EDT\$\$TRN_KPADK	:	
			50	D5	00127	TSTL	R0	:	
			F5	12	00129	BNEQ	13\$	:	
			4D	11	0012B	15\$:	BRB 21\$	:	
	54		65	D1	0012D	16\$:	CMPL EDT\$\$A_CMD_BUF, SAVE_POINT	:	0880
			06	13	00130	BEQL	17\$	:	
	65		54	D0	00132	MOVL	SAVE_POINT, EDT\$\$A_CMD_BUF	:	0883
	69		00	FB	00135	CALLS	#0, EDT\$\$ERA_MSGLN	:	0884
	53		6E	D0	00138	17\$:	MOVL MY_C, R3	:	0893
			53	DD	0013B	18\$:	PUSHL R3	:	
00000000G	00		01	FB	0013D	CALLS	#1, EDT\$\$TI_ECHOCH	:	
			7E	D4	00144	CLRL	-(SP)	:	0894
			53	DD	00146	PUSHL	R3	:	
00000000G	00		02	FB	00148	CALLS	#2, EDT\$\$PUT_CMDCH	:	
			1D	11	0014F	BRB	20\$	:	0895
0000007F	8F		6E	D1	00151	19\$:	CMPL MY_C, #127	:	0903
			22	12	00158	BNEQ	22\$	:	
	54		65	D1	0015A	CMPL	EDT\$\$A_CMD_BUF, SAVE_POINT	:	0906
			0F	13	0015D	BEQL	20\$	:	
			65	D7	0015F	DECL	EDT\$\$A_CMD_BUF	:	0909
	50		65	D0	00161	MOVL	EDT\$\$A_CMD_BUF, R0	:	0910
	7E		60	9A	00164	MOVZBL	(R0), =(SP)	:	
00000000G	00		01	FB	00167	CALLS	#1, EDT\$\$TI_DELK	:	

EDTSKEYCHR
V04-000

EDTSKEYCHR - get next command character
EDT\$\$NXT_CMDCH - get next command character

D 6
16-Sep-1984 00:41:34
14-Sep-1984 12:23:20

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]KEYCHR.BLI;1

Page 12
(3)

**F

66	00	FB	0016E	20\$:	CALLS	#0, EDT\$\$OUT_FMTBUF	: 0913
	5E	DD	00171		PUSHL	SP	: 0915
67	01	FB	00173		CALLS	#1, EDT\$\$TI_INPCH	
	50	D5	00176		TSTL	R0	
	D7	12	00178		BNEQ	19\$	
	57	11	0017A	21\$:	BRB	27\$	
53	6E	D0	0017C	22\$:	MOVL	MY_C, R3	: 0919
15	53	D1	0017F		CMPL	R3, #21	
	10	12	00182		BNEQ	23\$	
69	00	FB	00184		CALLS	#0, EDT\$\$ERA_MSGLN	: 0922
00000000G	00	9E	00187		MOVAB	EDT\$\$T_CMD_BUF, EDT\$\$A_CMD_END	: 0923
	5C	11	00192		BRB	32\$	: 0924
F0	8F	6A43	91 00194	23\$:	CMPB	EDT\$\$B_CHAR_INFO[R3], #240	: 0932
	A0	13	00199		BEQL	18\$	
1B	53	D1	0019B		CMPL	R3, #27	: 0945
	82	13	0019E	24\$:	BEQL	14\$	
0000009B	8F	53	D1 001A0		CMPL	R3, #155	
	F5	13	001A7		BEQL	24\$	
0000008F	8F	53	D1 001A9		CMPL	R3, #143	
	FF6D	31	001B0		BRW	13\$	
1B	52	D1	001B3	25\$:	CMPL	R2, #27	: 0967
	12	13	001B6		BEQL	26\$	
0000009B	8F	52	D1 001B8		CMPL	R2, #155	
	09	13	001BF		BEQL	26\$	
0000008F	8F	52	D1 001C1		CMPL	R2, #143	
	0D	12	001C8		BNEQ	28\$	
	5E	DD	001CA	26\$:	PUSHL	SP	: 0981
68	01	FB	001CC		CALLS	#1, EDT\$\$TRN_KPADK	
	50	D5	001CF		TSTL	R0	
	0B	12	001D1		BNEQ	29\$	
50	02	D0	001D3	27\$:	MOVL	#2, R0	
	04	001D6			RET		
04	63	E9	001D7	28\$:	BLBC	(R3), 29\$	: 0994
6E	A2	9E	001DA		MOVAB	-32(R2), MY_C	: 1002
6E	8F	C0	001DE	29\$:	ADDL2	#500, MY_C	: 0983
	FE	31	001E5		BRW	7\$	: 0819
04	BC	6E	D0 001E8	30\$:	MOVL	MY_C, @C	: 1012
	50	01	D0 001EC	31\$:	MOVL	#1, R0	: 1013
		04	001EF		RET		
	50	D4	001F0	32\$:	CLRL	R0	: 1014
		04	001F2		RET		

; Routine Size: 499 bytes, Routine Base: _EDT\$CODE + 0004

: 445 1015 1
: 446 1016 1 !<BLF/PAGE>

EDT\$KEYCHR
V04-000

EDT\$KEYCHR - get next command character
EDT\$\$NXT_CMDCH - get next command character

E 6
16-Sep-1984 00:41:34
14-Sep-1984 12:23:20

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]KEYCHR.BLI;1

Page 13
(4)

: 448
: 449
: 450
1017 1 END
1018 1
1019 0 ELUDOM

! of module EDT\$\$KEYCHR

PSECT SUMMARY

Name	Bytes	Attributes
_EDT\$CODE	503	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	6	17	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\$:KEYCHR/OBJ=OBJ\$:KEYCHR MSRC\$:KEYCHR.BLI/UPDATE=(ENHS\$:KEYCHR)

Size: 499 code + 4 data bytes
Run Time: 00:23.9
Elapsed Time: 00:28.3
Lines/CPU Min: 2558
Lexemes/CPU-Min: 7810
Memory Used: 158 pages
Compilation Complete

0135

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY