

[illegible]

```

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 ZTITLE 'EDT$RANRPOS - position to the first line of a range'
0002 0 MODULE EDT$RANRPOS ( ! Position to the first line of a range
0003 0 IDENT = 'V04-000' ! File: RANRPOS.BLI Edit: SMB1017
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 Position to the first line of a range.
0037 1
0038 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0039 1
0040 1 AUTHOR: Bob Kushlis, CREATION DATE: February 3, 1978
0041 1
0042 1 MODIFIED BY:
0043 1
0044 1 1-001 - Original. DJS 19-FEB-1981. This module was created by
0045 1 extracting routine RPOS from module RANGE.
0046 1 1-002 - Regularize headers. JBS 12-Mar-1981
0047 1 1-003 - Change to use the ASSERT macro. JBS 01-Jun-1981
0048 1 1-004 - Use the new message codes. JBS 04-Aug-1981
0049 1 1-005 - Use the new PREV_RANGE field for ALL. JBS 02-Nov-1981
0050 1 1-006 - In THRU, if part of the range is not found the whole
0051 1 range fails. JBS 19-Nov-1981
0052 1 1-007 - Fix any size problems in arithmetic and compares. SMB 25-Jan-1982
0053 1 1-008 - Remove original line numbers. SMB 28-Jan-1982
0054 1 1-009 - Add error message and change display for original line numbers. SMB 6-Feb-1982
0055 1 1-010 - Change the way buffer pos. is saved for AND ranges. SMB 15-Feb-1982
0056 1 1-011 - Don't change buffer pos if we have /STAY. STS 21-Apr-1982
0057 1 1-012 - Worry about string truncation. JBS 05-May-1982

```

EDT\$RANRPOS
V04-000

EDT\$RANRPOS - position to the first line of a r ^{L 8}
16-Sep-1984 01:26:05 VAX-11 Bliss-32 V4.0-742 Page 2
14-Sep-1984 12:24:19 DISK\$VMSMASTER:[EDT.SRC]RANRPOS.BLI;1 (1)

:	58	0058	1	:	1-013	- Add error checks for incomplete select ranges. SMB 01-Jul-1982
:	59	0059	1	:	1-014	- Make sure range ok on "No select range". SMB 02-Jul-1982
:	60	0060	1	:	1-015	- Put edt\$srng_retfirst in line. STS 11-Oct-1982
:	61	0061	1	:	1-016	- Modify to use new compare macro. STS 20-Oct-1982
:	62	0062	1	:	1-017	- Remove setting of G_TXT_ONSCR with select range. SMB 13-Dec-1982
:	63	0063	1	:	--	
:	64	0064	1	:		

EDTSRANRPOS
V04-000

EDTSRANRPOS - position to the first line of a r
Declarations

M 8
16-Sep-1984 01:26:05
14-Sep-1984 12:24:19

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

```
: 66      0065 1 %SBTTL 'Declarations'
: 67      0066 1
: 68      0067 1 | TABLE OF CONTENTS:
: 69      0068 1 |
: 70      0069 1
: 71      0070 1 REQUIRE 'EDTSRC:TRAROUNAM';
: 72      0509 1
: 73      0510 1 FORWARD ROUTINE
: 74      0511 1     EDT$SRNG_REPOS;
: 75      0512 1
: 76      0513 1 |
: 77      0514 1 | INCLUDE FILES:
: 78      0515 1 |
: 79      0516 1
: 80      0517 1 REQUIRE 'EDTSRC:EDTREQ';
: 81      0652 1
: 82      0653 1 |
: 83      0654 1 | MACROS:
: 84      0655 1 |
: 85      0656 1 |     NONE
: 86      0657 1 |
: 87      0658 1 | EQUATED SYMBOLS:
: 88      0659 1 |
: 89      0660 1 |     NONE
: 90      0661 1 |
: 91      0662 1 | OWN STORAGE:
: 92      0663 1 |
: 93      0664 1 |     NONE
: 94      0665 1 |
: 95      0666 1 | EXTERNAL REFERENCES:
: 96      0667 1 |
: 97      0668 1 |     In the routine
```

EDT\$RANRPOS
V04-000

N 8
EDT\$RANRPOS - position to the first line of a r 16-Sep-1984 01:26:05
EDT\$\$RNG_REPOS - position to the first line of 14-Sep-1984 12:24:19

VAX-11 Bliss-32 V4.0-742 Page 4
DISK\$VMMASTER:[EDT.SRC]RANRPOS.BLI;1 (3)

```

: 99      0669 1 %SBTTL 'EDT$$RNG_REPOS - position to the first line of a range'
: 100     0670 1
: 101     0671 1 GLOBAL ROUTINE EDT$$RNG_REPOS (           ! Position to the first line of a range
: 102     0672 1     RANGE                               ! Range to position to
: 103     0673 1     ) =
: 104     0674 1
: 105     0675 1 ++
: 106     0676 1 FUNCTIONAL DESCRIPTION:
: 107     0677 1
: 108     0678 1     This routine positions to the first line of a range.
: 109     0679 1
: 110     0680 1 FORMAL PARAMETERS:
: 111     0681 1
: 112     0682 1     RANGE                               The range node.
: 113     0683 1
: 114     0684 1 IMPLICIT INPUTS:
: 115     0685 1
: 116     0686 1     EDT$$A_SEL_BUF
: 117     0687 1     EDT$$L_SEL_LN
: 118     0688 1     EDT$$A_CUR_BUF
: 119     0689 1     EDT$$A_PRV_BUF
: 120     0690 1     EDT$$T_LN_BUF
: 121     0691 1     EDT$$Z_RNG_ORIGPOS
: 122     0692 1     EDT$$A_SEL_POS
: 123     0693 1     EDT$$A_WK_LN
: 124     0694 1
: 125     0695 1 IMPLICIT OUTPUTS:
: 126     0696 1
: 127     0697 1     EDT$$G_RNG_MORELN
: 128     0698 1     EDT$$G_RNG_NOOFLN
: 129     0699 1     EDT$$L_RNG_FOL
: 130     0700 1     EDT$$A_PRV_BUF
: 131     0701 1     EDT$$A_CUR_BUF
: 132     0702 1     EDT$$A_SEL_BUF
: 133     0703 1     EDT$$Z_RNG_SAVPOS
: 134     0704 1     EDT$$Z_RNG_CURRNG
: 135     0705 1     EDT$$Z_RNG_ORIGPOS
: 136     0706 1
: 137     0707 1 ROUTINE VALUE:
: 138     0708 1
: 139     0709 1     0 = no such line
: 140     0710 1     1 = positioned successfully
: 141     0711 1
: 142     0712 1 SIDE EFFECTS:
: 143     0713 1
: 144     0714 1     Current text buffer is re-positioned
: 145     0715 1
: 146     0716 1 --
: 147     0717 1
: 148     0718 2 BEGIN
: 149     0719 2
: 150     0720 2 EXTERNAL ROUTINE
: 151     0721 2     EDT$$FMT_MSG,
: 152     0722 2     EDT$$RNG_REPOS,
: 153     0723 2     EDT$$G_EXE_SBITS,
: 154     0724 2     EDT$$FND_STR,
: 155     0725 2     EDT$$FND_BUF,
```



```
156 0726 2 EDT$$SET SEASTR,  
157 0727 2 EDT$$WF_BOT,  
158 0728 2 EDT$$RD_PRVLN,  
159 0729 2 EDT$$RD_CURLN,  
160 0730 2 EDT$$RD_NXTLN,  
161 0731 2 EDT$$TOP_BUF,  
162 0732 2 EDT$$LOC_LN;  
163 0733  
164 0734 2 EXTERNAL  
165 0735 2 EDT$$G_CUR_COL,      ! Current cursor column  
166 0736 2 EDT$$L_LNOS : LN_BLOCK, 10**5  
167 0737 2 EDT$$A_SEL_BUF,      ! Select buffer  
168 0738 2 EDT$$L_SEL_LN,      ! Select line  
169 0739 2 EDT$$A_PRV_BUF,      ! The previous TBCB for LAST range.  
170 0740 2 EDT$$A_CUR_BUF : REF TBCB_BLOCK, ! The current text buffer  
171 0741 2 EDT$$T_LN_BUF,      ! Line buffer  
172 0742 2 EDT$$G_RNG_MORELN,    ! Used by EDT$$NXT_LNRNG to indicate more lines.  
173 0743 2 EDT$$G_RNG_FRSTLN,    ! Indicates first line in a range.  
174 0744 2 EDT$$G_RNG_NOOFLN,    ! Count of number of lines in a range.  
175 0745 2 EDT$$Z_RNG_ORIGPOS : POS_BLOCK, ! To save the position at start of command.  
176 0746 2 EDT$$L_RNG_EOL : LN_BLOCK, ! The line number at which this range ends  
177 0747 2 EDT$$Z_RNG_CURRNG : REF NODE_BLOCK, ! The current range node  
178 0748 2 EDT$$Z_RNG_SAVPOS : POS_BLOCK, ! To save the beginning of range  
179 0749 2 EDT$$A_SEL_POS,      ! Select position  
180 0750 2 EDT$$A_WK_LN : REF LIN_BLOCK; ! The current line pointer.  
181 0751  
182 0752 2 MESSAGES ((NOSELN, NOSUCHLIN, STRNOTFND, INSMEM, NOORIGNUM, INVSTR, I^ $RAN));  
183 0753  
184 0754 2 MAP  
185 0755 2 RANGE : REF NODE_BLOCK;  
186 0756  
187 0757 2 !+  
188 0758 2 ! Make sure the parameter is a range node.  
189 0759 2 !-  
190 0760 2 ASSERT (.RANGE [NODE_TYPE] EQL RANGE_NODE);  
191 0761 2 !+  
192 0762 2 ! Initialize for first line of range.  
193 0763 2 !-  
194 0764 2 EDT$$G_RNG_MORELN = 1;  
195 0765 2 EDT$$G_RNG_NOOFLN = 0;  
196 0766 2 !+  
197 0767 2 ! Case on range type.  
198 0768 2 !-  
199 0769  
200 0770 2 CASE .RANGE [RAN_TYPE] FROM 0 TO NUM_RAN OF  
201 0771 2 SET  
202 0772 2 !+  
203 0773 2 ! If range is '.' or REST, the original position is the first line.  
204 0774 2 !-  
205 0775  
206 0776 2 [RAN_DOT, RAN_NULL, RAN_REST] : ! Use the current position  
207 0777 2 BEGIN  
208 0778 2 EDT$$CPY_MEM (POS_SIZE, EDT$$Z_RNG_ORIGPOS, .EDT$$A_CUR_BUF);  
209 0779 2 EDT$$RD_CURLN ();  
210 0780 2 END;  
211 0781 2 !+  
212 0782 2 ! Line number range. Find the line.
```



```
213 0783 2 :-  
214 0784 2  
215 0785 2 [RAN_NUMBER] :  
216 0786 2 BEGIN  
217 0787 2 EDT$$G_RNG_MORELN = EDT$$LOC_LN (RANGE [RAN_VAL]);  
218 0788 2 END;  
219 0789 2  
220 0790 2 + Range is BEFORE. Try going back a line to find out if there are any,  
221 0791 2 then save that number as the last line number in the range.  
222 0792 2 -  
223 0793 2  
224 0794 2 [RAN_BEFORE] :  
225 0795 2 BEGIN  
226 0796 2 EDT$$G_RNG_MORELN = EDT$$RD_PRVLN ();  
227 0797 2 MOVE LINE (EDT$$A_WK_LN [LIN_NUM], EDT$$L_RNG_EOL);  
228 0798 2 EDT$$TOP_BUF ();  
229 0799 2 END;  
230 0800 2  
231 0801 2 + Range is WHOLE or BEGIN. Position to the first line in the buffer.  
232 0802 2 -  
233 0803 2  
234 0804 2 [RAN_WHOLE, RAN_BEGIN] :  
235 0805 2 EDT$$TOP_BUF ();  
236 0806 2  
237 0807 2 + Range is SELECT. Check to see if there is a select range in effect.  
238 0808 2 If it not in the current buffer, then switch to the buffer which has  
239 0809 2 the select. Then position to the first line of the select range.  
240 0810 2 -  
241 0811 2  
242 0812 2 [RAN_SELECT] :  
243 0813 2 BEGIN  
244 0814 2  
245 0815 2 LOCAL  
246 0816 2 TMPRAN : LN_BLOCK;  
247 0817 2  
248 0818 2 IF (.EDT$$A_SEL_BUF EQLA 0)  
249 0819 2 THEN  
250 0820 2 BEGIN  
251 0821 2 EDT$$FMT MSG (EDT$ NOSELRAN);  
252 0822 2 EDT$$G_RNG_MORELN = 0;  
253 0823 2 EDT$$Z_RNG_CURRNG = .RANGE;  
254 0824 2  
255 0825 2 IF (.EDT$$G_RNG_FRSTLN) THEN EDT$$CPY_MEM (POS_SIZE, .EDT$$A_CUR_BUF, EDT$$Z_RNG_SAVPOS);  
256 0826 2  
257 0827 2 RETURN (0);  
258 0828 2 END  
259 0829 2 ELSE  
260 0830 2 BEGIN  
261 0831 2  
262 0832 2 IF (.EDT$$A_SEL_BUF NEQA .EDT$$A_CUR_BUF)  
263 0833 2 THEN  
264 0834 2 BEGIN  
265 0835 2 EDT$$A_PRV_BUF = .EDT$$A_CUR_BUF;  
266 0836 2 EDT$$A_CUR_BUF = .EDT$$A_SEL_BUF;  
267 0837 2 EDT$$RD_CURLN ();  
268 0838 2 END;  
269 0839 2
```



```
270 0840 4          SUBLINE (EDT$$A_CUR_BUF [TBCB_CUR_LIN], EDT$$L_SEL_LN, TMPRAN);
271 0841 4          RANGE [RAN_VAL] = .TMPRAN;          !Only want to move a word
272 0842 4  !+
273 0843 4  !- For line mode commands the select range must be in whole lines
274 0844 4  !-
275 0845 4
276 0846 5          IF (.EDT$$G_CUR_COL NEQ 0) OR (.TMPRAN EQL 0)
277 0847 4          THEN
278 0848 5              BEGIN
279 0849 5                  EDT$$FMT_MSG (EDT$ INVS_RAN);
280 0850 5                  EDT$$G_RNG_MORELN = 0;
281 0851 5                  EDT$$Z_RNG_CURRNG = .RANGE;
282 0852 5
283 0853 5                  IF (.EDT$$G_RNG_FRSTLN) THEN EDT$$CPY_MEM (POS_SIZE, .EDT$$A_CUR_BUF, EDT$$Z_RNG_SAVPOS)
284 0854 5
285 0855 5                  RETURN (0);
286 0856 5              END
287 0857 4          ELSE
288 0858 4
289 0859 5              IF (.RANGE [RAN_VAL] LSS 0)
290 0860 4              THEN
291 0861 5                  BEGIN
292 0862 5                      RANGE [RAN_VAL] = -.RANGE [RAN_VAL];
293 0863 5
294 0864 5                      DECR I FROM .RANGE [RAN_VAL] - 1 TO 0 DO
295 0865 5                          EDT$$RD_PRVLN ();
296 0866 5
297 0867 4                      END;
298 0868 4
299 0869 5              IF CH$PTR_NEQ (.EDT$$A_SEL_POS, CH$PTR (EDT$$T_LN_BUF))
300 0870 4              THEN
301 0871 4                  RANGE [RAN_VAL] = .RANGE [RAN_VAL] + 1;
302 0872 4
303 0873 4                  EDT$$A_SEL_BUF = 0;
304 0874 4              END;
305 0875 4
306 0876 4          END;
307 0877 4  !+
308 0878 4  !- Range is END. Position to the end of the buffer.
309 0879 4  !-
310 0880 4
311 0881 4          [RAN_END] :
312 0882 4              BEGIN
313 0883 4                  EDT$$WF_BOT ();
314 0884 4                  EDT$$G_RNG_MORELN = 0;
315 0885 4              END;
316 0886 4  !+
317 0887 4  !- Range is ORIGINAL. This is no longer a feature of EDT, so print a warning
318 0888 4  !- and type the real line number corresponding to the line input
319 0889 4  !-
320 0890 4
321 0891 4          [RAN_ORIG] :
322 0892 4              BEGIN
323 0893 4
324 0894 4                  LOCAL
325 0895 4                      LINNO : LN_BLOCK;
326 0896 4
```



```
327 0897 3 EDT$$CPY_MEM (POS_SIZE, .EDT$$A_CUR_BUF, EDT$$Z_RNG_SAVPOS);
328 0898 EDT$$FMT_MSG (EDT$ NOORIGNUM);
329 0899 MULTLINE (EDT$$L_LN05, RANGE [RAN_VAL], LINNO);
330 0900
331 0901 WHILE NOT (LINNOEQL (EDT$$A_WK_LN [LIN_NUM], LINNO)) DO
332 0902
333 0903 IF ( NOT EDT$$RD_NXTLN ())
334 0904 THEN
335 0905 BEGIN
336 0906 EDT$$G_RNG_MORELN = 0;
337 0907 EDT$$CPY_MEM (POS_SIZE, EDT$$Z_RNG_SAVPOS, .EDT$$A_CUR_BUF);
338 0908 EDT$$RD_CURLN ();
339 0909 EDT$$FMT_MSG (EDT$ NOSUCHLIN);
340 0910 RETURN (0);
341 0911 END;
342 0912
343 0913 END;
344 0914 + Range is MINUS. Make a recursive call to position to the range then
345 0915 move back the specified number of lines.
346 0916
347 0917
348 0918
349 0919 [RAN_MINUS] :
350 0920 BEGIN
351 0921
352 0922 IF EDT$$RNG_REPOS (.RANGE [SUB_RANGE])
353 0923 THEN
354 0924 BEGIN
355 0925 EDT$$G_RNG_MORELN = 1;
356 0926
357 0927 INCR I FROM 1 TO .RANGE [RAN_VAL] DO
358 0928
359 0929 IF ( NOT EDT$$RD_PRVLN ())
360 0930 THEN
361 0931 BEGIN
362 0932 EDT$$G_RNG_MORELN = 0;
363 0933 EXITLOOP;
364 0934 END
365 0935
366 0936 END
367 0937 ELSE
368 0938 EDT$$G_RNG_MORELN = 0;
369 0939
370 0940 END;
371 0941 + Range is Plus. Make a recursive call to position to the range then
372 0942 move forward the specified number of lines.
373 0943
374 0944
375 0945
376 0946 [RAN_PLUS] :
377 0947 BEGIN
378 0948
379 0949 IF EDT$$RNG_REPOS (.RANGE [SUB_RANGE])
380 0950 THEN
381 0951 BEGIN
382 0952 EDT$$G_RNG_MORELN = 1;
383 0953
```



```
384      0954      4      INCR I FROM 1 TO .RANGE [RAN_VAL] DO
385      0955      4
386      0956      5      IF ( NOT EDT$RD_NXTLN ( ) )
387      0957      4      THEN
388      0958      3      BEGIN
389      0959      3      EDT$$G_RNG_MORELN = 0;
390      0960      3      EXITLOOP;
391      0961      3      END
392      0962      3
393      0963      4      END
394      0964      3      ELSE
395      0965      3      EDT$$G_RNG_MORELN = 0;
396      0966      3
397      0967      3      END;
398      0968      3
399      0969      3      + Range is FOR or #. Just position to the original range.
400      0970      3      -
401      0971      3
402      0972      3      [RAN_FOR] :
403      0973      3      EDT$$G_RNG_MORELN = EDT$$RNG_REPOS (.RANGE [SUB_RANGE]);
404      0974      3
405      0975      3      + Range is a search string. Save the current position, search for the
406      0976      3      string in the specified direction. If the string is not found, then
407      0977      3      reposition and return failure.
408      0978      3      -
409      0979      3
410      0980      3      [RAN_STR, RAN_MINSTR] :
411      0981      3      BEGIN
412      0982      3
413      0983      3      LOCAL
414      0984      3      FIND_STATUS;
415      0985      3
416      0986      3      EDT$$CPY_MEM (POS_SIZE, .EDT$$A_CUR_BUF, EDT$$Z_RNG_SAVPOS);
417      0987      3      FIND_STATUS = EDT$$FND_STR (.RANGE [STR_PNT], .RANGE [RAN_VAL], (.RANGE [RAN_TYPE] EQL RAN_STR))
418      0988      3
419      0989      3      CASE .FIND_STATUS FROM 0 TO 2 OF
420      0990      3      SET
421      0991      3
422      0992      3      [0] :                               ! String not found
423      0993      4      BEGIN
424      0994      4      EDT$$FMT_MSG (EDT$ _STRNOTFND);
425      0995      4      EDT$$G_RNG_MORELN = 0;
426      0996      4      EDT$$CPY_MEM (POS_SIZE, EDT$$Z_RNG_SAVPOS, .EDT$$A_CUR_BUF);
427      0997      4      EDT$RD_CURLN ( );
428      0998      4      RETURN (0);
429      0999      4      END;
430      1000      3
431      1001      3      [1] :                               ! String found
432      1002      4      BEGIN
433      1003      4      0
434      1004      4      END;
435      1005      3
436      1006      3      [2] :                               ! String invalid
437      1007      4      BEGIN
438      1008      4      EDT$$FMT_MSG (EDT$ _INVSTR);
439      1009      4      EDT$$G_RNG_MORELN = 0;
440      1010      4      EDT$$CPY_MEM (POS_SIZE, EDT$$Z_RNG_SAVPOS, .EDT$$A_CUR_BUF);
```



```

441      1011      4      EDT$$RD_CURLN ();
442      1012      4      RETURN (0);
443      1013      4      END;
444      1014      4      TES;
445      1015      4
446      1016      4      END;
447      1017      4
448      1018      4      Range is THRU or : . Position to the first range, then find the line
449      1019      4      number of the end of the range. Special case when the end range is a
450      1020      4      line number or END. Otherwise, save the current position and position to
451      1021      4      the end to get the end range. If either the first or last cannot be
452      1022      4      found the whole range fails.
453      1023      4
454      1024      4
455      1025      4      [RAN_THRU] :
456      1026      4      BEGIN
457      1027      4
458      1028      4      LOCAL
459      1029      4      EDT$$SAV_BUFPOS : POS_BLOCK;
460      1030      4
461      1031      4      BIND
462      1032      4      END_RANGE = .RANGE [RANGE2] : NODE_BLOCK;
463      1033      4
464      1034      4      BIND
465      1035      4      START_RANGE = .RANGE [RANGE1] : NODE_BLOCK;
466      1036      4
467      1037      4      IF ( NOT EDT$$RNG_REPOS (START_RANGE)) THEN RETURN (0);
468      1038      4
469      1039      4      IF (.END_RANGE [RAN_TYPE] EQL RAN_NUMBER) !
470      1040      4      THEN
471      1041      4      EDT$$CPY_MEM (LN_SIZE, END_RANGE [RAN_VAL], EDT$$L_RNG_EOL)
472      1042      4      ELSE
473      1043      4
474      1044      4      IF (.END_RANGE [RAN_TYPE] EQL RAN_END)
475      1045      4      THEN
476      1046      4      RANGE [RAN_TYPE] = RAN_REST
477      1047      4      ELSE
478      1048      4      BEGIN
479      1049      4
480      1050      4      LOCAL
481      1051      4      END_FOUND;
482      1052      4
483      1053      4      EDT$$CPY_MEM (POS_SIZE, .EDT$$A_CUR_BUF, EDT$$SAV_BUFPOS);
484      1054      4      END_FOUND = EDT$$RNG_REPOS (END_RANGE);
485      1055      4
486      1056      4      IF .END_FOUND THEN MOVELINE (EDT$$A_WK_LN [LIN_NUM], EDT$$L_RNG_EOL);
487      1057      4
488      1058      4      EDT$$CPY_MEM (POS_SIZE, EDT$$SAV_BUFPOS, .EDT$$A_CUR_BUF);
489      1059      4      EDT$$RD_CURLN ();
490      1060      4
491      1061      4      IF ( NOT .END_FOUND) THEN RETURN (0);
492      1062      4
493      1063      4      END;
494      1064      4
495      1065      4      EDT$$G_RNG_MORELN = 1;
496      1066      4      END;
497      1067      4      !+
```



```

498 1068 2 | Range is ALL range. Look at the range to which ALL applies. If it is
499 1069 2 | null, assume WHOLE. Position to that range, then set up the ALL string
500 1070 2 | as the current search string. Note that EDT$$Z_RNG_CURRNG will be not
501 1071 2 | that ALL range but its subordinate. PREV_RANGE will point back to the
502 1072 2 | ALL range.
503 1073 2 |
504 1074 2 |
505 1075 2 | [RAN_ALL] :
506 1076 2 | BEGIN
507 1077 2 |
508 1078 2 | LOCAL
509 1079 2 | SUB_RAN : REF NODE_BLOCK;
510 1080 2 |
511 1081 2 | SUB_RAN = .RANGE [NEXT_RANGE];
512 1082 2 | ASSERT (.SUB_RAN [PREV_RANGE] EQLA .RANGE);
513 1083 2 |
514 1084 2 | IF (.SUB_RAN [RAN_TYPE] EQL RAN_NULL) THEN SUB_RAN [RAN_TYPE] = RAN_WHOLE;
515 1085 2 |
516 1086 2 | EDT$$G_RNG_FRSTLN = 1;
517 1087 2 | EDT$$CPY_MEM (POS_SIZE, .EDT$$A_CUR_BUF, EDT$$Z_RNG_ORIGPOS);
518 1088 2 |
519 1089 2 | IF ( NOT EDT$$RNG_REPOS (.RANGE [NEXT_RANGE])) THEN RETURN (0);
520 1090 2 |
521 1091 2 | IF ( NOT EDT$$SET_SEASTR (.RANGE [STR_PNT], .RANGE [RAN_VAL]))
522 1092 2 | THEN
523 1093 2 | BEGIN
524 1094 2 | EDT$$FMT_MSG (EDT$_INVSTR);
525 1095 2 | RETURN (0);
526 1096 2 | END;
527 1097 2 |
528 1098 2 | RETURN (1);
529 1099 2 | END;
530 1100 2 |
531 1101 2 | + The range contains a buffer specification. Switch to the new buffer, then
532 1102 2 | position to the range within the buffer. If the range within the buffer
533 1103 2 | was null, assume WHOLE.
534 1104 2 |
535 1105 2 |
536 1106 2 | [RAN_BUFFER] :
537 1107 2 | BEGIN
538 1108 2 |
539 1109 2 | LOCAL
540 1110 2 | SUB_RANGE : REF NODE_BLOCK;
541 1111 2 |
542 1112 2 | IF EDT$$FND_BUF (.RANGE [BUF_NAME], .RANGE [BUF_LEN])
543 1113 2 | THEN
544 1114 2 | BEGIN
545 1115 2 | SUB_RANGE = .RANGE [RANGE1];
546 1116 2 |
547 1117 2 | IF ( NOT .EDT$$G_EXE_SBITS<OPB_STAY>) !
548 1118 2 | THEN
549 1119 2 | EDT$$CPY_MEM (POS_SIZE, .EDT$$A_CUR_BUF, EDT$$Z_RNG_ORIGPOS);
550 1120 2 |
551 1121 2 | EDT$$TOP_BUF ();
552 1122 2 |
553 1123 2 | IF (.SUB_RANGE [RAN_TYPE] EQL RAN_NULL) THEN SUB_RANGE [RAN_TYPE] = RAN_WHOLE;
554 1124 2 |
```

```
555      RETURN (EDT$$RNG_REPOS (.SUB_RANGE));
556      END
557      ELSE
558      BEGIN
559      EDT$$FMT MSG (EDT$_INSMEM);
560      RETURN (0);
561      END
562      END;
563
564      + Range is LAST. Switch back to the buffer pointed to by EDT$$A_PRV_BUF
565      + at it's current position.
566      -
567
568      [RAN_LAST] :
569      BEGIN
570
571      LOCAL
572      TEMP;
573
574      TEMP = .EDT$$A_CUR_BUF;
575      EDT$$A_CUR_BUF = .EDT$$A_PRV_BUF;
576      EDT$$A_PRV_BUF = .TEMP;
577      EDT$$RD_CURLN ();
578      END;
579
580      [INRANGE, OUTRANGE] :
581      ASSERT (0);
582      TES;
583
584      + Save the range node and the current position.
585      -
586      EDT$$Z_RNG_CURRNG = .RANGE;
587
588      IF (.EDT$$G_RNG_FRSTLN) THEN EDT$$CPY_MEM (POS_SIZE, .EDT$$A_CUR_BUF, EDT$$Z_RNG_SAVPOS);
589
590      RETURN (1);
591      END;
```

! of routine EDT\$\$RNG_REPOS

```
.TITLE EDT$RANRPOS EDT$RANRPOS - position to the first
line of a r
```

```
.IDENT \V04-000\
```

```
.EXTRN EDT$$FMT MSG, EDT$$RNG_REPOS
.EXTRN EDT$$G_EXE_SBITS
.EXTRN EDT$$FND_STR, EDT$$FND_BUF
.EXTRN EDT$$SET_SEASTR
.EXTRN EDT$$WF_BOT, EDT$$RD_PRVLN
.EXTRN EDT$$RD_CURLN, EDT$$RD_NXTLN
.EXTRN EDT$$STOP_BUF, EDT$$LOC_LN
.EXTRN EDT$$G_COR_COL, EDT$$L_LNOS
.EXTRN EDT$$A_SEL_BUF, EDT$$L_SEL_LN
.EXTRN EDT$$A_PRV_BUF, EDT$$A_CUR_BUF
.EXTRN EDT$$T_LN_BUF, EDT$$G_RNG_MORELN
.EXTRN EDT$$G_RNG_FRSTLN
```


[illegible]

.EXTRN	EDT\$\$G_RNG_NOOFLN	
.EXTRN	EDT\$\$Z_RNG_ORIGPOS	
.EXTRN	EDT\$\$L_RNG_EOL, EDT\$\$Z_RNG_CURRNG	
.EXTRN	EDT\$\$Z_RNG_SAVPOS	
.EXTRN	EDT\$\$A_SEL_POS, EDT\$\$A_WK_LN	
.EXTRN	EDT\$_NOSEL_RAN, EDT\$_NOSUCH_LIN	
.EXTRN	EDT\$_STRNOTFND, EDT\$_INSMEM	
.EXTRN	EDT\$_NOORIGNUM, EDT\$_INVSTR	
.EXTRN	EDT\$_INVS_RAN, EDT\$\$INTER_ERR	
.PSECT	_EDT\$CODE, NOWRT, SHR, PIC, 2	
.ENTRY	EDT\$\$RNG_REPOS, Save R2,R3,R4,R5,R6,R7,R8,-	0671
MOVAB	R9,R10,RT1	
MOVAB	EDT\$\$Z_RNG_SAVPOS, R11	
MOVAB	EDT\$\$RNG_REPOS, R10	
MOVAB	EDT\$\$G_RNG_MORELN, R9	
MOVAB	EDT\$\$A_CUR_BUF, R8	
SUBL2	#24, SP	
MOVL	RANGE, R6	0760
CMPB	(R6), #2	
BEQL	1\$	
CALLS	#0, EDT\$\$INTER_ERR	
MOVL	#1, EDT\$\$G_RNG_MORELN	0764
CLRL	EDT\$\$G_RNG_NOOFLN	0765
CASEB	1(R6), -#0, -#20	0770
.WORD	4\$-2\$,-	
	5\$-2\$,-	
	4\$-2\$,-	
	37\$-2\$,-	
	7\$-2\$,-	
	21\$-2\$,-	
	22\$-2\$,-	
	3\$-2\$,-	
	58\$-2\$,-	
	6\$-2\$,-	
	4\$-2\$,-	
	7\$-2\$,-	
	9\$-2\$,-	
	52\$-2\$,-	
	30\$-2\$,-	
	27\$-2\$,-	
	34\$-2\$,-	
	44\$-2\$,-	
	37\$-2\$,-	
	49\$-2\$,-	
	3\$-2\$	
CALLS	#0, EDT\$\$INTER_ERR	1152
BRB	8\$	0770
MOVL	EDT\$\$A_CUR_BUF, R0	0778
MOVC3	#14, EDT\$\$Z_RNG_ORIGPOS, (R0)	
BRW	59\$	0779
PUSHAB	4(R6)	0787
CALLS	#1, EDT\$\$LOC_LN	
BRW	35\$	
CALLS	#0, EDT\$\$RD_PrvLN	0796
MOVL	R0, EDT\$\$G_RNG_MORELN	

00000000G	00	01	50	00000000G	00	D0	00097	MOVL	EDT\$\$A_WK_LN, R0	:	0797
			A0		06	28	0009E	MOV3	#6, 1(R0), EDT\$\$L_RNG_EOL	:	
			00		00	FB	000A7	CALLS	#0, EDT\$\$TOP_BUF	:	0805
					02F5	31	000AE	BRW	60\$	:	
			52	00000000G	00	D0	000B1	MOVL	EDT\$\$A_SEL_BUF, R2	:	0818
					08	12	000B8	BNEQ	10\$	:	
				00000000G	8F	DD	000BA	PUSHL	#EDT\$_NOSEL_RAN	:	0821
					56	11	000C0	BRB	13\$	:	
			50		68	D0	000C2	MOVL	EDT\$\$A_CUR_BUF, R0	:	0832
			50		52	D1	000C5	CMPL	R2, R0	:	
					11	13	000C8	BEQL	11\$	:	
				00000000G	00	D0	000CA	MOVL	R0, EDT\$\$A_PRV_BUF	:	0835
			68		52	D0	000D1	MOVL	R2, EDT\$\$A_CUR_BUF	:	0836
				00000000G	00	FB	000D4	CALLS	#0, EDT\$\$RD_CURLN	:	0837
			51	16	AE	B0	000DB	MOVW	UPPER_WORD, -SAVE	:	0840
			50		68	D0	000DF	MOVL	EDT\$\$A_CUR_BUF, R0	:	
10	AE	00000000G	00	06	A0	C3	000E2	SUBL3	6(R0), -EDT\$\$L_SEL_LN, TMPRAN	:	
		14	AE	00000000G	00	D0	000EC	MOVL	EDT\$\$L_SEL_LN+4, TMPRAN	:	
		14	AE	0A	A0	D9	000F4	SBWC	10(R0), TMPRAN	:	
		16	AE		51	B0	000F9	MOVW	SAVE, UPPER_WORD	:	
			57	04	A6	9E	000FD	MOVAB	4(R6), R7	:	0841
			67	10	AE	D0	00101	MOVL	TMPRAN, (R7)	:	
				00000000G	00	D5	00105	TSTL	EDT\$\$G_CUR_COL	:	0846
					05	12	0010B	BNEQ	12\$	:	
				10	AE	D5	0010D	TSTL	TMPRAN	:	
					27	12	00110	BNEQ	15\$	:	
				00000000G	8F	DD	00112	PUSHL	#EDT\$_INVS_RAN	:	0849
			00		01	FB	00118	CALLS	#1, EDT\$\$FMT_MSG	:	
					69	D4	0011F	CLRL	EDT\$\$G_RNG_MORELN	:	0850
				00000000G	00	D0	00121	MOVL	R6, EDT\$\$Z_RNG_CURRNG	:	0851
			07		00	E9	00128	BLBC	EDT\$\$G_RNG_FRSTLN, 14\$	:	0853
			50		68	D0	0012F	MOVL	EDT\$\$A_CUR_BUF, R0	:	
6B			60		0E	28	00132	MOV3	#14, (R0), EDT\$\$Z_RNG_SAVPOS	:	
					0286	31	00136	BRW	62\$	:	0855
					67	D5	00139	TSTL	(R7)	:	0859
					12	18	0013B	BGEQ	18\$	:	
			67		67	CE	0013D	MNEGL	(R7), (R7)	:	0862
			52		67	D0	00140	MOVL	(R7), I	:	0864
					07	11	00143	BRB	17\$	:	
				00000000G	00	FB	00145	CALLS	#0, EDT\$\$RD_PRVLN	:	0865
			F6		52	F4	0014C	SOBGEQ	I, 16\$	:	
			50	00000000G	00	9E	0014F	MOVAB	EDT\$\$T_LN_BUF, R0	:	0869
			50	00000000G	00	D1	00156	CMPL	EDT\$\$A_SEC_POS, R0	:	
					02	13	0015D	BEQL	19\$	:	
					67	D6	0015F	INCL	(R7)	:	0871
				00000000G	00	D4	00161	CLRL	EDT\$\$A_SEL_BUF	:	0873
					023C	31	00167	BRW	60\$	:	0770
			00		00	FB	0016A	CALLS	#0, EDT\$\$WF_BOT	:	0883
					00BE	31	00171	BRW	33\$	:	0884
			50		68	D0	00174	MOVL	EDT\$\$A_CUR_BUF, R0	:	0897
6B			60		0E	28	00177	MOV3	#14, (R0), EDT\$\$Z_RNG_SAVPOS	:	
				00000000G	8F	DD	0017B	PUSHL	#EDT\$_NOORIGNUM	:	0898
					01	FB	00181	CALLS	#1, EDT\$\$FMT_MSG	:	
			08	04	A6	D0	00188	MOVL	4(R6), M2	:	0899
			OC	08	A6	3C	0018D	MOVZWL	8(R6), M2+4	:	
					6E	7C	00192	CLRQ	P	:	
			50		10	D0	00194	MOVL	#16, I	:	

6E	09	00000000G	00	08	01	79	00197	23\$:	ASHQ	#1, P, P	:
			00		50	E1	00198		BBC	I, M1, 24\$	:
	04		6E	0C	AE	C0	001A3		ADDL2	M2, P	:
			AE		AE	D8	001A7		ADWC	M2, P	:
	10		E8		50	F4	001AC	24\$:	SOBGEQ	I, 23\$	:
	14		AE	04	6E	D0	001AF		MOVL	P, LINNO	:
			AE		AE	B0	001B3		MOVW	P+4, LINNO+4	:
50	00000000G	00	00		01	C1	001B8	25\$:	ADDL3	#1, EDTSSA_WK_LN, R0	0901
	10	AE			60	D1	001C0		CMPL	(R0), LOW_2	:
					07	12	001C4		BNEQ	26\$	:
	14	AE	04		A0	B1	001C6		CMPW	4(R0), HIGH_2	:
					9A	13	001CB		BEQL	20\$	:
	00000000G	00			00	FB	001CD	26\$:	CALLS	#0, EDTSSRD_NXTLN	0903
		E1			50	E8	001D4		BLBS	R0, 25\$	:
					69	D4	001D7		CLRL	EDTSSG_RNG_MORELN	0906
					68	D0	001D9		MOVL	EDTSSA_CUR_BUF, R0	0907
60			50		0E	28	001DC		MOVC3	#14, EDTSSZ_RNG_SAVPOS, (R0)	:
	00000000G	00	6B		00	FB	001E0		CALLS	#0, EDTSSRD_CURLN	0908
			00	00000000G	8F	DD	001E7		PUSHL	#EDTSS_NOSUCRLIN	0909
					0195	31	001ED		BRW	56\$	:
				08	A6	DD	001F0	27\$:	PUSHL	8(R6)	0922
		6A			01	FB	001F3		CALLS	#1, EDTSSRNG_REPOS	:
		39			50	E9	001F6		BLBC	R0, 33\$	:
		69			01	D0	001F9		MOVL	#1, EDTSSG_RNG_MORELN	0925
					52	D4	001FC		CLRL	I	0927
					0A	11	001FE		BRB	29\$	:
	00000000G	00			00	FB	00200	28\$:	CALLS	#0, EDTSSRD_PRVLN	0929
		28			50	E9	00207		BLBC	R0, 33\$	:
F1		52	04		A6	F3	0020A	29\$:	AOBLEQ	4(R6), I, 28\$	:
					2E	11	0020F		BRB	36\$	0927
			08		A6	DD	00211	30\$:	PUSHL	8(R6)	0949
		6A			01	FB	00214		CALLS	#1, EDTSSRNG_REPOS	:
		18			50	E9	00217		BLBC	R0, 33\$	:
		69			01	D0	0021A		MOVL	#1, EDTSSG_RNG_MORELN	0952
					52	D4	0021D		CLRL	I	0954
					0A	11	0021F		BRB	32\$	:
	00000000G	00			00	FB	00221	31\$:	CALLS	#0, EDTSSRD_NXTLN	0956
		07			50	E9	00228		BLBC	R0, 33\$	:
F1		52	04		A6	F3	0022B	32\$:	AOBLEQ	4(R6), I, 31\$	:
					0D	11	00230		BRB	36\$	0954
					69	D4	00232	33\$:	CLRL	EDTSSG_RNG_MORELN	0965
					09	11	00234		BRB	36\$	0770
			08		A6	DD	00236	34\$:	PUSHL	8(R6)	0973
		6A			01	FB	00239		CALLS	#1, EDTSSRNG_REPOS	:
		69			50	D0	0023C	35\$:	MOVL	R0, EDTSSG_RNG_MORELN	:
					0164	31	0023F	36\$:	BRW	60\$	:
		50			68	D0	00242	37\$:	MOVL	EDTSSA_CUR_BUF, R0	0986
6B		60			0E	28	00245		MOVC3	#14, (R0), EDTSSZ_RNG_SAVPOS	:
					7E	D4	00249		CLRL	-(SP)	0987
		03	01		A6	91	0024B		CMPB	1(R6), #3	:
					02	12	0024F		BNEQ	38\$	:
					6E	D6	00251		INCL	(SP)	:
			04		A6	DD	00253	38\$:	PUSHL	4(R6)	:
			08		A6	DD	00256		PUSHL	8(R6)	:
					03	FB	00259		CALLS	#3, EDTSSFND_STR	:
02	00000000G	00			50	CF	00260		CASEL	FIND_STATUS, #0, #2	0989
000E		0142			0006		00264	39\$:	.WORD	40\$-39\$,-	:

Address	Hex	Label	Instruction	Comment	Page
00000000G	8F DD 0026A	40\$:	PUSHL	60\$-39\$,-	0994
	06 11 00270		BRB	41\$-39\$	
00000000G	8F DD 00272	41\$:	PUSHL	#EDT\$ _STRNOTFND	1008
00000000G	01 FB 00278	42\$:	CALLS	42\$	
	69 D4 0027F		CLRL	#1, EDT\$\$FMT MSG	1009
	68 D0 00281		MOVL	EDT\$\$G_RNG_MORELN	1010
	0E 28 00284		MOVC3	EDT\$\$A_CUR_BUF, R0	
	00 FB 00288		CALLS	#14, EDT\$\$Z_RNG_SAVPOS, (R0)	1011
	012D 31 0028F	43\$:	BRW	#0, EDT\$\$RD_CURLN	1012
	08 A6 D0 00292	44\$:	MOVL	62\$	1012
	04 A6 DD 00296		PUSHL	8(R6), R7	1032
	01 01 FB 00299		CALLS	4(R6)	1037
	50 E9 0029C		BLBC	#1, EDT\$\$RNG_REPOS	
	01 A7 91 0029F		CMPB	R0, 43\$	1039
	0B 12 002A3		BNEQ	1(R7), #1	
	06 28 002A5		MOVC3	45\$	1041
	41 11 002AE		BRB	#6, 4(R7), EDT\$\$L_RNG_EOL	
	01 A7 91 002B0	45\$:	CMPB	48\$	1044
	06 12 002B4		BNEQ	1(R7), #5	
	0A 90 002B6		MOVB	46\$	1046
	35 11 002BA		BRB	#10, 1(R6)	
	68 D0 002BC	46\$:	MOVL	48\$	1053
	0E 28 002BF		MOVC3	EDT\$\$A_CUR_BUF, R0	
	57 DD 002C4		PUSHL	#14, (R0), EDT\$\$SAV_BUFPOS	1054
	01 FB 002C6		CALLS	R7	1056
	50 D0 002C9		MOVL	#1, EDT\$\$RNG_REPOS	
	57 E9 002CC		BLBC	R0, END FOUND	1056
	00 D0 002CF		MOVL	END FOUND, 47\$	
	06 28 002D6		MOVC3	EDT\$\$A_WK_LN, R0	1058
	68 D0 002DF	47\$:	MOVL	#6, 1(R0), EDT\$\$L_RNG_EOL	
	0E 28 002E2		MOVC3	EDT\$\$A_CUR_BUF, R0	1059
	00 FB 002E7		CALLS	#14, EDT\$\$SAV_BUFPOS, (R0)	1061
	57 E9 002EE		BLBC	#0, EDT\$\$RD_CURLN	1065
	01 D0 002F1	48\$:	MOVL	END FOUND, 43\$	0770
	00AF 31 002F4		BRW	#1, EDT\$\$G_RNG_MORELN	
	10 A6 D0 002F7	49\$:	MOVL	60\$	1081
	14 A2 D1 002FB		CPL	16(R6), SUB_RAN	1082
	07 13 002FF		BEQL	20(SUB_RAN), R6	
	00 FB 00301		CALLS	50\$	
	01 A2 95 00308	50\$:	TSTB	#0, EDT\$\$INTER_ERR	1084
	04 12 0030B		BNEQ	1(SUB_RAN)	
	0B 90 0030D		MOVB	51\$	
	01 D0 00311	51\$:	MOVL	#11, 1(SUB_RAN)	1086
	68 D0 00318		MOVL	#1, EDT\$\$G_RNG_FRSTLN	1087
	0E 28 0031B		MOVC3	EDT\$\$A_CUR_BUF, R0	
	A6 DD 00323		PUSHL	#14, (R0), EDT\$\$Z_RNG_ORIGPOS	1089
	01 FB 00326		CALLS	16(R6)	
	50 E9 00329		BLBC	#1, EDT\$\$RNG_REPOS	1091
	04 A6 DD 0032C		PUSHL	R0, 57\$	
	08 A6 DD 0032F		PUSHL	4(R6)	
	02 FB 00332		CALLS	8(R6)	
	50 E8 00339		BLBS	#2, EDT\$\$SET_SEASTR	1094
	8F DD 0033C		PUSHL	R0, 61\$	
	41 11 00342		BRB	#EDT\$ _INVSTR	1112
	7E 08 A6 7D 00344	52\$:	MOVQ	56\$	
				8(R6), -(SP)	

EDT\$RANRPOS
V04-000

EDT\$RANRPOS - position to the first line of a r 16-Sep-1984 01:26:05
EDT\$\$RNG_REPOS - position to the first line of 14-Sep-1984 12:24:19

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

Page 17
(3)

00000000G	00	02	FB	00348	CALLS	#2, EDT\$\$FND_BUF	:	
	2D	50	E9	0034F	BLBC	R0, 55\$	:	
	57	04	A6	D0	00352	MOVL	4(R6), SUB_RANGE	1115
		00000000G	00	95	00356	TSTB	EDT\$\$G_EXE_SBITS	1117
			0B	19	0035C	BLSS	53\$	
	50		68	D0	0035E	MOVL	EDT\$\$A_CUR_BUF, R0	1119
00000000G	00		0E	28	00361	MOVC3	#14, (R0), -EDT\$\$Z_RNG_ORIGPOS	
			00	FB	00369	CALLS	#0, EDT\$\$TOP_BUF	1121
		01	A7	95	00370	TSTB	1(SUB_RANGE)	1123
			04	12	00373	BNEQ	54\$	
	01	A7	0B	90	00375	MOVB	#11, 1(SUB_RANGE)	
			57	DD	00379	PUSHL	SUB_RANGE	1125
			6A	01	FB	0037B	CALLS	#1, -EDT\$\$RNG_REPOS
				04	0037E	RET		1128
		00000000G	8F	DD	0037F	PUSHL	#EDT\$ INSMEM	1129
			01	FB	00385	CALLS	#1, EDT\$\$FMT_MSG	
			31	11	0038C	BRB	62\$	1130
	50		68	D0	0038E	MOVL	EDT\$\$A_CUR_BUF, TEMP	1145
		00000000G	00	D0	00391	MOVL	EDT\$\$A_PRV_BUF, EDT\$\$A_CUR_BUF	1146
			50	D0	00398	MOVL	TEMP, EDT\$\$A_PRV_BUF	1147
00000000G	00		00	FB	0039F	CALLS	#0, EDT\$\$RD_CURLN	1148
00000000G	00		56	D0	003A6	MOVL	R6, EDT\$\$Z_RNG_CURRNG	1158
00000000G	00		00	E9	003AD	BLBC	EDT\$\$G_RNG_FRSTLN, 61\$	1160
	07	00000000G	00	D0	003B4	MOVL	EDT\$\$A_CUR_BUF, R0	
	50		68	D0	003B7	MOVC3	#14, (R0), -EDT\$\$Z_RNG_SAVPOS	
6B	60		0E	28	003B7	MOVL	#1, R0	1162
	50		01	D0	003BB	RET		
				04	003BE	CLRL	R0	1163
			50	D4	003BF	RET		
			04	003C1				

; Routine Size: 962 bytes, Routine Base: _EDT\$CODE + 0000

; 594 1164 1
; 595 1165 1 !<BLF/PAGE>

EDT\$RANRPOS
V04-000

EDT\$RANRPOS - position to the first line of a r 16-Sep-1984 01:26:05
EDT\$SRNG_REPOS - position to the first line of 14-Sep-1984 12:24:19

B 10
VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[EDT.SRC]RANRPOS.BLI;1
Page 18
(4)

: 597 1166 1 END
: 598 1167 1
: 599 1168 0 ELUDOM

! of module EDT\$RANRPOS

PSECT SUMMARY

: Name Bytes Attributes
: _EDT\$CODE 962 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	125	33	40	00:00.2
_\$255\$DUA28:[EDT.SRC]PSECTS.L32;1	2	1	50	7	00:00.1

COMMAND QUALIFIERS

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

: Size: 962 code + 0 data bytes
: Run Time: 00:40.2
: Elapsed Time: 00:50.3
: Lines/CPU Min: 1744
: Lexemes/CPU-Min: 9933
: Memory Used: 295 pages
: Compilation Complete

0138

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY