

RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	

```

PPPPPPPP      EEEEEEEEEEE RRRRRRRR  MM      MM  UU      UU  TTTTTTTTTT  EEEEEEEEEEE
PPPPPPPP      EEEEEEEEEEE RRRRRRRR  MM      MM  UU      UU  TTTTTTTTTT  EEEEEEEEEEE
PP           PP  EE           RR      RR  MMMM  MMMM  UU      UU  TT      TT  EE
PP           PP  EE           RR      RR  MMMM  MMMM  UU      UU  TT      TT  EE
PP           PP  EE           RR      RR  MM  MM  MM  UU      UU  TT      TT  EE
PP           PP  EE           RR      RR  MM  MM  MM  UU      UU  TT      TT  EE
PPPPPPPP      EEEEEEEEEEE RRRRRRRR  MM      MM  UU      UU  TT      TT  EEEEEEEEEEE
PPPPPPPP      EEEEEEEEEEE RRRRRRRR  MM      MM  UU      UU  TT      TT  EEEEEEEEEEE
PP           EE           RR  RR      MM      MM  UU      UU  TT      TT  EE
PP           EE           RR  RR      MM      MM  UU      UU  TT      TT  EE
PP           EE           RR  RR      MM      MM  UU      UU  TT      TT  EE
PP           EE           RR  RR      MM      MM  UU      UU  TT      TT  EE
PP           EE           RR  RR      MM      MM  UU      UU  TT      TT  EE
PP           EEEEEEEEEEE RR      RR  MM      MM  UUUUUUUUUU  TT      TT  EEEEEEEEEEE
PP           EEEEEEEEEEE RR      RR  MM      MM  UUUUUUUUUU  TT      TT  EEEEEEEEEEE

```

```

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 'PERMUTE -- Permute index entries'
0002 0 MODULE PERMUTE (IDENT = 'V04-000'
0003 0 %BLISS32 [, ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE)]
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:
0033 1   DSR (Digital Standard RUNOFF) /DSRPLUS DSRINDEX/INDEX Utility
0034 1
0035 1 ABSTRACT:
0036 1   Routines to permute index entries
0037 1
0038 1 ENVIRONMENT:   Transportable
0039 1
0040 1 AUTHOR:        JPK
0041 1
0042 1 CREATION DATE: December 1981
0043 1
0044 1 MODIFIED BY:
0045 1
0046 1   004   JPK00024   23-May-1983
0047 1   Modified lowercasing algorithm in PERMUTE. Now lowercase only
0048 1   if word contains only 1 letter or if second letter in word is
0049 1   lowercase. Picked up modules PAGMRG and POOL from DSR/DSRPLUS
0050 1   since they are no longer used by DSR/DSRPLUS.
0051 1
0052 1   003   JPK00015   04-Feb-1983
0053 1   Cleaned up module names, modified revision history to
0054 1   conform with established standards. Updated copyright dates.
0055 1
0056 1   002   JPK00014   02-Feb-1983
0057 1   Modified PERMUTE to remove extra whitespace in index entries.
```

PERMUTE
V04-000

PERMUTE -- Permute index entries

D 6
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 2
(1)

```
: 58      0058 1 |      Changed module name.
: 59      0059 1 |
: 60      0060 1 | --
: 61      0061 1 |
: 62      0062 1 |
: 63      0063 1 | TABLE OF CONTENTS:
: 64      0064 1 |
: 65      0065 1 |
: 66      0066 1 | FORWARD ROUTINE
: 67      0067 1 |     PERMUT      : NOVALUE,
: 68      0068 1 |     MAKE_ENTRY  : NOVALUE,
: 69      0069 1 |     FIX_CASE    : NOVALUE,
: 70      0070 1 |     COPY_STR    : NOVALUE;
: 71      0071 1 |
: 72      0072 1 |
: 73      0073 1 |
: 74      0074 1 | INCLUDE FILES:
: 75      0075 1 |
: 76      0076 1 |
: 77      0077 1 | LIBRARY 'NXPORT:XPORT';
: 78      0078 1 |
: 79      0079 1 | SWITCHES LIST (REQUIRE);
: 80      0080 1 |
: 81      0081 1 | REQUIRE 'REQ:NDXXPL';
```

```
: ! Permute index entry
: ! Make a permuted index entry
: ! Fix up case of permuted entry
: ! Copy string from a to b and filter
: ! out the nopermute flag.
```

```
: ! Extended indexing attributes
```

PERMUTE
V04-000

PERMUTE -- Permute index entries

E 6
16-Sep-1984 01:27:55
15-Sep-1984 22:53:35

VAX-11 Bliss-32 V4.0-742
_S255SDUA28:[RUNOFF.SRC]NDXXPL.REQ;1 Page 3
(1)

Version: 'V04-000'

*
* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
* ALL RIGHTS RESERVED.
*

* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
* TRANSFERRED.
*

* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
* CORPORATION.
*

* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*

++
FACILITY: DSR (Digital Standard RUNOFF) / DSRPLUS

ABSTRACT:
This file contains definitions of data structures used to support
the extended indexing features of the DSRPLUS INDEX program.

ENVIRONMENT: Transportable BLISS

AUTHOR: J.P. Kellerman

CREATION DATE: January 1982

MODIFIED BY:

002 KAD00002 Keith Dawson 07-Mar-1983
Global edit of all modules. Updated module names, idents,
copyright dates. Changed require files to BLISS library.

--
! Extended INDEX attributes block.

\$FIELD XPL_FIELDS =
SET

XPL\$V_OPTIONS = [\$INTEGER], ! Attributes options

\$OVERLAY (XPL\$V_OPTIONS)

PERMUTE
V04-000

PERMUTE -- Permute index entries

F 6

16-Sep-1984 01:27:55

VAX-11 Bliss-32 V4.0-742

Page 4

15-Sep-1984 22:53:35

_S255\$DUA28:[RUNOFF.SRC]NDXXPL.REQ;1

(1)

```

: R0139 1      XPL$V_VALID      = [$BIT],      ! Attributes block contains valid information.
: R0140 1      XPL$V_BOLD       = [$BIT],      ! Bold page reference.
: R0141 1      XPL$V_UNDERLINE  = [$BIT],      ! Underlined page reference.
: R0142 1      XPL$V_BEGIN      = [$BIT],      ! Begin page range.
: R0143 1      XPL$V_END        = [$BIT],      ! End page range.
: R0144 1      XPL$V_MASTER     = [$BIT],      ! Master index entry.
: R0145 1      XPL$V_PERMUTE    = [$BIT],      ! Permute index entry.
: R0146 1      XPL$V_NOPERMUTE  = [$BIT],      ! Set if permute explicitly forbidden.
: R0147 1      XPL$V_SORT       = [$BIT],      ! Set if SORT string present.
: R0148 1      XPL$V_APPEND     = [$BIT],      ! Set if append string present.
: R0149 1
: R0150 1      $CONTINUE
: R0151 1
: R0152 1      XPL$T_SORT       = [$DESCRIPTOR(DYNAMIC)], ! SORT string.
: R0153 1      XPL$T_APPEND     = [$DESCRIPTOR(DYNAMIC)], ! APPEND string.
: R0154 1
: R0155 1      TES;
: R0156 1
: R0157 1      LITERAL
: R0158 1      XPL$K_LENGTH = $FIELD_SET_SIZE;
: R0159 1
: R0160 1      MACRO
: R0161 1      $XPL_BLOCK = BLOCK [XPL$K_LENGTH] FIELD (XPL_FIELDS) %;
: R0162 1
: R0163 1      !
: R0164 1      ! Macros for INDEX_ATTRIBUTES flags
: R0165 1      !
: R0166 1      MACRO
: R0167 1      XPLUS$V_VALID    = 0, 0, 1, 0 %, ! Set if attributes data is valid.
: R0168 1      XPLUS$V_BOLD     = 0, 1, 1, 0 %, ! Set if page reference is bolded.
: R0169 1      XPLUS$V_UNDERLINE = 0, 2, 1, 0 %, ! Set if page reference is underlined.
: R0170 1      XPLUS$V_BEGIN    = 0, 3, 1, 0 %, ! Set if entry begins a page range.
: R0171 1      XPLUS$V_END      = 0, 4, 1, 0 %, ! Set if entry ends a page range.
: R0172 1      XPLUS$V_MASTER   = 0, 5, 1, 0 %, ! Set if master index entry only.
: R0173 1      XPLUS$V_PERMUTE  = 0, 6, 1, 0 %, ! Set if entry is to be permuted.
: R0174 1      XPLUS$V_NOPERMUTE = 0, 7, 1, 0 %, ! Set if permute is explicitly forbidden.
: R0175 1      XPLUS$V_SORT     = 0, 8, 1, 0 %, ! Set if entry contains a SORT string.
: R0176 1      XPLUS$V_APPEND   = 0, 9, 1, 0 %, ! Set if entry contains an APPEND string.
: R0177 1
: R0178 1      !
:                               End of NDXXPL.REQ
```

PERMUTE
V04-000

PERMUTE -- Permute index entries

: 82
: 83

0179 1
0180 1 REQUIRE 'REQ:LETTER';

G 6
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 5
(1)

PE
VO

8.

PERMUTE
V04-000

PERMUTE -- Permute index entries

I 6
16-Sep-1984 01:27:55
15-Sep-1984 22:52:49

VAX-11 Bliss-32 V4.0-742
_S255\$DUA28:[RUNOFF.SRC]LETTER.REQ;1 Page 7
(1)

```
: MR0238 1      letter (khar) = ! See if any type of letter
: MR0239 1      (upper_letter (khar) OR lower_letter (khar))
: R0240 1      %;
: R0241 1
: R0242 1      MACRO
: MR0243 1      upper_case (khar) = ! Convert to upper case
: MR0244 1      (khar + %C'A' - %C'a')
: R0245 1      %,
: R0246 1
: MR0247 1      lower_case (khar) = ! Convert to lower case
: MR0248 1      (khar + %C'a' - %C'A')
: R0249 1      %;
: R0250 1
: R0251 1      !
                End of LETTER.REQ
```

PERMUTE
V04-000

PERMUTE -- Permute index entries

J 6
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 8
(1)

```

: 84      0252 1
: 85      0253 1 SWITCHES LIST (NOREQUIRE);
: 86      0254 1
: 87      0255 1
: 88      0256 1
: 89      0257 1
: 90      0258 1
: 91      0259 1
: 92      0260 1
: 93      0261 1
: 94      0262 1 LITERAL
: 95      0263 1     TRUE = 1
: 96      0264 1     FALSE = 0;
: 97      0265 1
: 98      0266 1
: 99      0267 1
: 100     0268 1
: 101     0269 1
: 102     0270 1
: 103     0271 1
: 104     0272 1
: 105     0273 1 EXTERNAL LITERAL
: 106     0274 1     TAB : UNSIGNED (8);
: 107     0275 1     RINTES : UNSIGNED (8);
: 108     0276 1
: 109     0277 1 EXTERNAL
: 110     0278 1     XPLBLK : $XPL_BLOCK;
: 111     0279 1
: 112     0280 1 EXTERNAL ROUTINE
: 113     0281 1     XOUT : NOVALUE;
: 114     0282 1
```

```
! TAB character
! RUNOFF escape sequence
```

```
! Extended indexing attributes block
```

```
! Insert index entry into internal pool
```

PERMUTE
V04-000

PERMUTE -- Permute index entries
PERMUT -- Permute Index Entry

K 6
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 9
(2)

```

: 116 0283 1 %SBTTL 'PERMUT -- Permute Index Entry'
: 117 0284 1 GLOBAL ROUTINE PERMUT (ENTRY_LEN, ENTRY_PTR, XTN, BAR_FLAG) : NOVALUE =
: 118 0285 1 ++
: 119 0286 1
: 120 0287 1 FUNCTIONAL DESCRIPTION:
: 121 0288 1
: 122 0289 1 This routine is called to permute and output index entries
: 123 0290 1
: 124 0291 1 FORMAL PARAMETERS:
: 125 0292 1
: 126 0293 1 ENTRY_LEN - Number of characters in entry
: 127 0294 1 ENTRY_PTR - A CH$PTR to index entry
: 128 0295 1 XTN - Transaction number
: 129 0296 1 BAR_FLAG - Change bar flag
: 130 0297 1
: 131 0298 1 IMPLICIT INPUTS:
: 132 0299 1
: 133 0300 1 XPLBLK - Extended indexing attributes block
: 134 0301 1
: 135 0302 1 IMPLICIT OUTPUTS:
: 136 0303 1
: 137 0304 1 None
: 138 0305 1
: 139 0306 1 ROUTINE VALUE:
: 140 0307 1 COMPLETION CODES:
: 141 0308 1
: 142 0309 1 None
: 143 0310 1
: 144 0311 1 SIDE EFFECTS:
: 145 0312 1
: 146 0313 1 None
: 147 0314 1 --
: 148 0315 1
: 149 0316 2 BEGIN
: 150 0317 2
: 151 0318 2 LOCAL
: 152 0319 2 WHITE_SPACE, ! Length of string up to white space
: 153 0320 2 SOURCE_PTR, ! Pointer to source string
: 154 0321 2 PERMUTE_PTR, ! Pointer to permuted text
: 155 0322 2 PERMUTE_LEN, ! Length of permuted text
: 156 0323 2 PERMUTE_STR : VECTOR [CH$ALLOCATION (1200)]; ! Put permuted text here
: 157 0324 2
: 158 0325 2 IF .ENTRY_LEN EQL 0 THEN RETURN; ! Nothing to do if no text
: 159 0326 2
: 160 0327 2 ! Initialization
: 161 0328 2
: 162 0329 2
: 163 0330 2 WHITE_SPACE = -1;
: 164 0331 2 SOURCE_PTR = .ENTRY_PTR; ! Copy pointer to string
: 165 0332 2 PERMUTE_PTR = CH$PTR (PERMUTE_STR);
: 166 0333 2 PERMUTE_LEN = 0;
: 167 0334 2
: 168 0335 2 IF .XPLBLK [XPL$V_VALID] AND .XPLBLK [XPL$V_PERMUTE]
: 169 0336 2 THEN
: 170 0337 2 BEGIN
: 171 0338 2 !
: 172 0339 2 ! Permute index entry
```

PERMUTE
V04-000

PERMUTE -- Permute index entries
PERMUT -- Permute Index Entry

L 6
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 10
(2)

```

173 0340 3      !
174 0341 3      ! LOCAL
175 0342 3      PERMUTE_FIRST,
176 0343 3      FIRST_CHAR_SEEN;
177 0344 3      ! TRUE if can permute on first word
178 0345 3      ! TRUE after 1st character found
179 0346 3      PERMUTE_FIRST = TRUE;
180 0347 3      FIRST_CHAR_SEEN = FALSE;
181 0348 3      INCR I FROM 1 TO .ENTRY_LEN DO
182 0349 4      BEGIN
183 0350 4      LOCAL
184 0351 4      CH;
185 0352 4      ! Current character
186 0353 4      CH = CH$RCHAR_A (SOURCE_PTR);
187 0354 4      SELECTONE .CH OF
188 0355 4      SET
189 0356 4      [INTES]:
190 0357 4      BEGIN
191 0358 4      ! Process RUNOFF escape sequence
192 0359 4      LOCAL
193 0360 5      FUNCTION_CODE,
194 0361 5      OPERAND;
195 0362 5      FUNCTION_CODE = CH$RCHAR_A (SOURCE_PTR);
196 0363 5      OPERAND = CH$RCHAR_A (SOURCE_PTR);
197 0364 5      I = .I + 2;
198 0365 5      SELECTONE .FUNCTION_CODE OF
199 0366 5      SET
200 0367 5      [%C'J'] :
201 0368 5      BEGIN
202 0369 5      ! A word mark. For index entries this starts a
203 0370 5      ! subindex entry. Since subindex entries are not
204 0371 5      ! permuted, copy subindex entries and exit.
205 0372 5      COPY_STR (3 + .ENTRY_LEN - .I,
206 0373 5      CH$PLUS (.SOURCE_PTR, -3),
207 0374 5      PERMUTE_LEN, PERMUTE_PTR);
208 0375 5      EXITLOOP;
209 0376 6      END;
210 0377 6      [%C'P']:
211 0378 6      ! Nopermute flag.
212 0379 6      ! If we haven't seen the first character yet, set
213 0380 6      ! flag saying that we can't permute on the first word.
214 0381 6      ! Remove escape sequence by doing nothing.
215 0382 6      COPY_STR (3 + .ENTRY_LEN - .I,
216 0383 6      CH$PLUS (.SOURCE_PTR, -3),
217 0384 6      PERMUTE_LEN, PERMUTE_PTR);
218 0385 6      EXITLOOP;
219 0386 6      END;
220 0387 5      [%C'P']:
221 0388 5      ! Nopermute flag.
222 0389 5      ! If we haven't seen the first character yet, set
223 0390 5      ! flag saying that we can't permute on the first word.
224 0391 5      ! Remove escape sequence by doing nothing.
225 0392 5      COPY_STR (3 + .ENTRY_LEN - .I,
226 0393 5      CH$PLUS (.SOURCE_PTR, -3),
227 0394 5      PERMUTE_LEN, PERMUTE_PTR);
228 0395 5      EXITLOOP;
229 0396 5      END;
```

PERMUTE
V04-000

PERMUTE -- Permute index entries
PERMUT -- Permute Index Entry

M 6
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 11
(2)

```

230 0397 5
231 0398 5
232 0399 5
233 0400 5
234 0401 6
235 0402 6
236 0403 6
237 0404 6
238 0405 6
239 0406 6
240 0407 6
241 0408 6
242 0409 6
243 0410 6
244 0411 6
245 0412 6
246 0413 6
247 0414 6
248 0415 6
249 0416 6
250 0417 6
251 0418 6
252 0419 6
253 0420 6
254 0421 5
255 0422 5
256 0423 5
257 0424 5
258 0425 5
259 0426 4
260 0427 4
261 0428 4
262 0429 5
263 0430 5
264 0431 5
265 0432 5
266 0433 5
267 0434 5
268 0435 5
269 0436 5
270 0437 6
271 0438 6
272 0439 6
273 0440 6
274 0441 6
275 0442 6
276 0443 6
277 0444 6
278 0445 5
279 0446 5
280 0447 4
281 0448 4
282 0449 4
283 0450 5
284 0451 5
285 0452 5
286 0453 5

      IF NOT .FIRST_CHAR_SEEN THEN PERMUTE_FIRST = FALSE;
[OTHERWISE] :
      BEGIN
      Other RUNOFF escape sequences signal a new word
      if white space was seen. In this case, It's time
      to make a permuted entry. Once the entry has been
      made (or if it's not necessary), the escape
      sequence is copied to the permuted text string.

      IF .WHITE_SPACE GEQ 0
      THEN
        MAKE_ENTRY (3 + .ENTRY_LEN - .I,
          CH$PLUS (.SOURCE_PTR, -3),
          .WHITE_SPACE, CH$PTR (PERMUTE_STR),
          .XTN, .BAR_FLAG);

        CH$WCHAR_A (.CH, PERMUTE_PTR);
        CH$WCHAR_A (.FUNCTION_CODE, PERMUTE_PTR);
        CH$WCHAR_A (.OPERAND, PERMUTE_PTR);
        PERMUTE_LEN = .PERMUTE_LEN + 3;
      END;

      TES;

      WHITE_SPACE = -1;          ! Blank or tab not seen
      END;

[XC' ', TAB]:
      BEGIN
      Blank or TAB.
      Remove extra whitespace by inserting a blank for
      only the first one seen.

      IF .WHITE_SPACE EQL -1
      THEN
        BEGIN
          No blanks or tabs seen yet. Save position and insert
          a blank in the output string.

          WHITE_SPACE = .PERMUTE_LEN;
          CH$WCHAR_A (XC' ', PERMUTE_PTR);
          PERMUTE_LEN = .PERMUTE_LEN + 1;
        END;

      END;

[OTHERWISE]:
      BEGIN
      Process normal character
```

PERMUTE
V04-000

PERMUTE -- Permute index entries
PERMUT -- Permute Index Entry

N 6
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 12
(2)

```

: 287      0454      5      FIRST CHAR_SEEN = TRUE;
: 288      0455      5      CH$WCHAR A-(.CH, PERMUTE_PTR);
: 289      0456      5      PERMUTE_LEN = .PERMUTE_LEN + 1;
: 290      0457      5
: 291      0458      5      IF .WHITE_SPACE GEQ 0
: 292      0459      5      THEN
: 293      0460      5      |
: 294      0461      5      |   White space was seen before this character. Hence we
: 295      0462      5      |   are starting a new word and it is time to permute.
: 296      0463      5      |
: 297      0464      5      |   MAKE ENTRY (.ENTRY_LEN - .I + 1,
: 298      0465      5      |   CH$PLUS (.SOURCE_PTR, -1), .WHITE_SPACE,
: 299      0466      5      |   CH$PTR (PERMUTE_STR), .XTN, .BAR_FLAG);
: 300      0467      5
: 301      0468      5      WHITE_SPACE = -1;      ! Blank or tab not seen
: 302      0469      4      END;
: 303      0470      4
: 304      0471      4      TES;
: 305      0472      4
: 306      0473      5      END;
: 307      0474      5
: 308      0475      5      IF .PERMUTE_FIRST
: 309      0476      5      THEN
: 310      0477      4      BEGIN
: 311      0478      4      FIX_CASE (.PERMUTE_LEN, CH$PTR (PERMUTE_STR), 0, 0);
: 312      0479      4      XOUT (.PERMUTE_LEN, CH$PTR (PERMUTE_STR), .XTN, .BAR_FLAG);
: 313      0480      5      END;
: 314      0481      5      END
: 315      0482      5      ELSE
: 316      0483      5      BEGIN
: 317      0484      5      |
: 318      0485      5      |   Do not permute index entry.
: 319      0486      5      |   Copy it filtering out nopermute flags, insert it in the internal
: 320      0487      5      |   pool and return.
: 321      0488      5      |
: 322      0489      5      |   COPY_STR (.ENTRY_LEN, .ENTRY_PTR, PERMUTE_LEN, PERMUTE_PTR);
: 323      0490      5      |   XOUT (.PERMUTE_LEN, CH$PTR (PERMUTE_STR), .XTN, .BAR_FLAG);
: 324      0491      5      |   END;
: 325      0492      2
: 326      0493      1      END;
```

.TITLE PERMUTE PERMUTE -- Permute index entries
.IDENT \V04-000\

.EXTRN TAB, RINTES, XPLBLK
.EXTRN XOUT

.PSECT \$CODE\$,NOWRT,2

.ENTRY PERMUT, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,- ; 0284
R11
MOVAB COPY_STR, R11
MOVAB -1208(SP), SP
MOVL ENTRY_LEN, R5
BNEQ 1\$; 0325
RET

OFFC 00000

```

5B 00000000V EF 9E 00002
5E FB48 CE 9E 00009
55 04 AC D0 0000E
01 12 00012
04 00014
```

PERMUTE
V04-000

PERMUTE -- Permute index entries
PERMUT -- Permute Index Entry

B 7
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 13
(2)

	54		01	CE	00015	1\$:	MNEGL	#1, WHITE_SPACE	0330	
	52	08	AC	D0	00018		MOVL	ENTRY_PTR, SOURCE_PTR	0331	
	6E	08	AE	9E	0001C		MOVAB	PERMUTE_STR, PERMUTE_PTR	0332	
		04	AE	D4	00020		CLRL	PERMUTE_LEN	0333	
	03	00000000G	EF	E8	00023		BLBS	XPLBLK, -3\$	0335	
			00F8	31	0002A	2\$:	BRW	15\$		
F5	00000000G		06	E1	0002D	3\$:	BBC	#6, XPLBLK, 2\$		
			59	01	D0	00035	MOVL	#1, PERMUTE_FIRST	0345	
				58	D4	00038	CLRL	FIRST_CHAR_SEEN	0346	
				56	D4	0003A	CLRL	I	0348	
			00CC	31	0003C		BRW	13\$		
			82	9A	0003F	4\$:	MOVZBL	(SOURCE_PTR)+, CH	0354	
	00000000G		53	D1	00042		CMPL	CH, #RINTES	0359	
			6D	12	00049		BNEQ	9\$		
			57	82	9A	0004B	MOVZBL	(SOURCE_PTR)+, FUNCTION_CODE	0368	
			5A	82	9A	0004E	MOVZBL	(SOURCE_PTR)+, OPERAND	0369	
			56	02	C0	00051	ADDL2	#2, I	0370	
0000004A			8F	57	D1	00054	CMPL	FUNCTION_CODE, #74	0375	
				15	12	0005B	BNEQ	5\$		
				5E	DD	0005D	PUSHL	SP	0383	
			08	AE	9F	0005F	PUSHAB	PERMUTE_LEN		
			FD	A2	9F	00062	PUSHAB	-3(SOURCE_PTR)		
50				56	C3	00065	SUBL3	I, R5, R0	0382	
			03	A0	9F	00069	PUSHAB	3(R0)		
				04	FB	0006C	CALLS	#4, COPY_STR	0383	
			009F	31	0006F		BRW	14\$	0376	
00000050				57	D1	00072	5\$:	CMPL	FUNCTION_CODE, #80	0389
				07	12	00079	BNEQ	6\$		
				58	E8	0007B	BLBS	FIRST_CHAR_SEEN, 8\$	0398	
				59	D4	0007E	CLRL	PERMUTE_FIRST		
				34	11	00080	BRB	8\$		
				54	D5	00082	6\$:	TSTL	WHITE_SPACE	0410
				1A	19	00084	BLSS	7\$		
			7E	0C	AC	7D	00086	MOVQ	XTN, -(SP)	0415
				10	AE	9F	0008A	PUSHAB	PERMUTE_STR	0414
					54	DD	0008D	PUSHL	WHITE_SPACE	
			FD	A2	9F	0008F	PUSHAB	-3(SOURCE_PTR)	0413	
50				56	C3	00092	SUBL3	I, R5, R0	0412	
			03	A0	9F	00096	PUSHAB	3(R0)		
				06	FB	00099	CALLS	#6, MAKE_ENTRY	0413	
00000000V			EF	53	90	000A0	7\$:	MOVB	CH, @PERMUTE_PTR	0417
00			BE	6E	D6	000A4	INCL	PERMUTE_PTR		
				57	90	000A6	MOVB	FUNCTION_CODE, @PERMUTE_PTR	0418	
			00	6E	D6	000AA	INCL	PERMUTE_PTR		
			00	5A	90	000AC	MOVB	OPERAND, @PERMUTE_PTR	0419	
				6E	D6	000B0	INCL	PERMUTE_PTR		
			04	03	C0	000B2	ADDL2	#3, PERMUTE_LEN	0420	
				50	11	000B6	8\$:	BRB	12\$	0425
				53	D1	000B8	9\$:	CMPL	CH, #32	0428
				09	13	000BB	BEQL	10\$		
00000000G			8F	53	D1	000BD	CMPL	CH, #TAB		
				18	12	000C4	BNEQ	11\$		
FFFFFFFF			8F	54	D1	000C6	10\$:	CMPL	WHITE_SPACE, #-1	0435
				3C	12	000CD	BNEQ	13\$		
				04	AE	D0	000CF	MOVL	PERMUTE_LEN, WHITE_SPACE	0442
			00	20	90	000D3	MOVB	#32, @PERMUTE_PTR	0443	
				6E	D6	000D7	INCL	PERMUTE_PTR		

PERMUTE
V04-000

PERMUTE -- Permute index entries
PERMUT -- Permute Index Entry

C 7
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 14
(2)

PER
V04

			04	AE	D6	000D9		INCL	PERMUTE_LEN	:	0444
				2D	11	000DC		BRB	13\$	:	0356
	00	58		01	D0	000DE	11\$:	MOVL	#1, FIRST_CHAR_SEEN	:	0454
		BE		53	90	000E1		MOVB	CH, @PERMUTE_PTR	:	0455
				6E	D6	000E5		INCL	PERMUTE_PTR	:	
			04	AE	D6	000E7		INCL	PERMUTE_LEN	:	0456
				54	D5	000EA		TSTL	WHITE_SPACE	:	0458
				1A	19	000EC		BLSS	12\$	:	
		7E	0C	AC	7D	000EE		MOVQ	XTN, -(SP)	:	0466
			10	AE	9F	000F2		PUSHAB	PERMUTE_STR	:	
				54	DD	000F5		PUSHL	WHITE_SPACE	:	0465
			FF	A2	9F	000F7		PUSHAB	-1(SOURCE_PTR)	:	
50		55		56	C3	000FA		SUBL3	I, R5, R0	:	0464
			01	A0	9F	000FE		PUSHAB	1(R0)	:	
	00000000V	EF		06	FB	00101		CALLS	#6, MAKE_ENTRY	:	0465
		54		01	CE	00108	12\$:	MNEGL	#1, WHITE_SPACE	:	0468
FF2E	56	01		55	F1	0010B	13\$:	ACBL	R5, #1, I-4\$	:	0348
		2F		59	E9	00111	14\$:	BLBC	PERMUTE_FIRST, 17\$	:	0475
				7E	7C	00114		CLRQ	-(SP)	:	0478
			10	AE	9F	00116		PUSHAB	PERMUTE_STR	:	
			10	AE	DD	00119		PUSHL	PERMUTE_LEN	:	
	00000000V	EF		04	FB	0011C		CALLS	#4, FIX_CASE	:	
				0D	11	00123		BRB	16\$	:	0479
				5E	DD	00125	15\$:	PUSHL	SP	:	0489
			08	AE	9F	00127		PUSHAB	PERMUTE_LEN	:	
			08	AC	DD	0012A		PUSHL	ENTRY_PTR	:	
				55	DD	0012D		PUSHL	R5	:	
		6B		04	FB	0012F		CALLS	#4, COPY_STR	:	
		7E	0C	AC	7D	00132	16\$:	MOVQ	XTN, -(SP)	:	0490
			10	AE	9F	00136		PUSHAB	PERMUTE_STR	:	
			10	AE	DD	00139		PUSHL	PERMUTE_LEN	:	
	00000000G	EF		04	FB	0013C		CALLS	#4, XOUT	:	
				04	00143	17\$:		RET		:	0493

; Routine Size: 324 bytes, Routine Base: \$CODE\$ + 0000

```

328 0494 1 %SBTTL 'MAKE_ENTRY -- Make a permuted entry'
329 0495 1 ROUTINE MAKE_ENTRY (SOURCE_LEN, SOURCE_PTR, PERMUTED_LEN, PERMUTED_PTR, XTN, BAR_FLAG) : NOVALUE =
330 0496 1 ++
331 0497 1
332 0498 1 FUNCTIONAL DESCRIPTION:
333 0499 1
334 0500 1 This routine makes a permuted index entry.
335 0501 1
336 0502 1 It copies from the string described by SOURCE_LEN and SOURCE_PTR
337 0503 1 until either a subindex entry or end of string is encountered.
338 0504 1 At this point, it appends a comma and a blank and then copies
339 0505 1 from the permuted string described by PERMUTED_LEN and PERMUTED_PTR.
340 0506 1 If the break was caused by a subindex entry, the subindex entry
341 0507 1 is then copied.
342 0508 1
343 0509 1 Once the permuted text has been built, it is written out.
344 0510 1
345 0511 1 FORMAL PARAMETERS:
346 0512 1
347 0513 1 SOURCE_LEN - Length of unpermuted portion of index entry
348 0514 1 SOURCE_PTR - CH$PTR to unpermuted portion of index entry
349 0515 1 PERMUTED_LEN - Length of permuted portion of index entry
350 0516 1 PERMUTED_PTR - CH$PTR to permuted portion of index entry
351 0517 1 XTN - Transaction number
352 0518 1 BAR_FLAG - Change bar flag
353 0519 1
354 0520 1 IMPLICIT INPUTS:
355 0521 1
356 0522 1 None
357 0523 1
358 0524 1 IMPLICIT OUTPUTS:
359 0525 1
360 0526 1 None
361 0527 1
362 0528 1 ROUTINE VALUE:
363 0529 1 COMPLETION CODES:
364 0530 1
365 0531 1 None
366 0532 1
367 0533 1 SIDE EFFECTS:
368 0534 1
369 0535 1 None
370 0536 1 --
371 0537 1
372 0538 2 BEGIN
373 0539 2
374 0540 2 LOCAL
375 0541 2 WHITE_SPACE, ! To delete trailing white space
376 0542 2 PTR, ! Copy of SOURCE_PTR
377 0543 2 LINE : VECTOR [CH$ALLOCATION (1200)], ! Build permuted string here
378 0544 2 L_PTR, ! CH$PTR into LINE
379 0545 2 L_LEN; ! Length of string in LINE
380 0546 2
381 0547 2 |
382 0548 2 | Initialization
383 0549 2 |
384 0550 2 WHITE_SPACE = -1;
```

```
385 0551 2 PTR = .SOURCE_PTR;  
386 0552 2 L_PTR = CH$PTR (LINE);  
387 0553 2 L_LEN = 0;  
388 0554 2  
389 0555 2 INCR I FROM 1 TO .SOURCE_LEN DO  
390 0556 2 BEGIN  
391 0557 2  
392 0558 2 LOCAL  
393 0559 2 CH; ! Current character  
394 0560 2  
395 0561 2 CH = CH$RCHAR_A (PTR);  
396 0562 2  
397 0563 2 SELECTONE .CH OF  
398 0564 2 SET  
399 0565 2  
400 0566 2 [RINTES]:  
401 0567 2 BEGIN  
402 0568 2  
403 0569 2 | RUNOFF escape sequence  
404 0570 2  
405 0571 2 LOCAL  
406 0572 2 FUNCTION_CODE,  
407 0573 2 OPERAND;  
408 0574 2  
409 0575 2 FUNCTION_CODE = CH$RCHAR_A (PTR);  
410 0576 2 OPERAND = CH$RCHAR_A (PTR);  
411 0577 2 I = .I + 2;  
412 0578 2  
413 0579 2 SELECTONE .FUNCTION_CODE OF  
414 0580 2 SET  
415 0581 2  
416 0582 2 [%C'J'] :  
417 0583 2 BEGIN  
418 0584 2  
419 0585 2 | Subindex flag. Delete trailing white space,  
420 0586 2 | and append comma, blank  
421 0587 2  
422 0588 2 IF .WHITE_SPACE NEQ -1  
423 0589 2 THEN  
424 0590 2 BEGIN  
425 0591 2 L_PTR = CH$PLUS (.L_PTR, (.WHITE_SPACE - .L_LEN));  
426 0592 2 L_LEN = .WHITE_SPACE;  
427 0593 2 END;  
428 0594 2  
429 0595 2 CH$WCHAR_A (%C',', L_PTR);  
430 0596 2 CH$WCHAR_A (%C',', L_PTR);  
431 0597 2 L_LEN = .L_LEN + 2;  
432 0598 2  
433 0599 2  
434 0600 2 | Raise case of first letter in permuted entry and  
435 0601 2 | drop case of original first letter if necessary.  
436 0602 2  
437 0603 2 FIX_CASE (.L_LEN, CH$PTR (LINE), .PERMUTED_LEN, .PERMUTED_PTR);  
438 0604 2  
439 0605 2  
440 0606 2 | Finally, append permuted text to end of line,  
441 0607 2 | append the subindex entry and output.
```

PERMUTE
V04-000

PERMUTE -- Permute index entries
MAKE_ENTRY -- Make a permuted entry

F 7
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 17
(3)

```

442 0608 5
443 0609 5      !
444 0610 5      COPY_STR (.PERMUTED_LEN, .PERMUTED_PTR, L_LEN, L_PTR);
445 0611 5      COPY_STR (3 + .SOURCE_LEN - .1, CH$PLUS (.PTR, -3), L_LEN, L_PTR);
446 0612 5      XOUT (.L_LEN, CH$PTR (LINE), .XTN, .BAR_FLAG);
447 0613 4      RETURN;
448 0614 4      END;
449 0615 4      [%C'P']:
450 0616 4      |
451 0617 4      |   Filter out nopermute flags by doing nothing
452 0618 4      |
453 0619 4      |
454 0620 4      |
455 0621 4      [OTHERWISE] :
456 0622 5      BEGIN
457 0623 5      |
458 0624 5      |   Other RUNOFF escape sequences are just copied
459 0625 5      |
460 0626 5      CH$WCHAR_A (.CH, L_PTR);
461 0627 5      CH$WCHAR_A (.FUNCTION_CODE, L_PTR);
462 0628 5      CH$WCHAR_A (.OPERAND, L_PTR);
463 0629 5      L_LEN = .L_LEN + 3;
464 0630 4      END;
465 0631 4
466 0632 4      TES;
467 0633 4
468 0634 4      WHITE_SPACE = -1;
469 0635 3      END;
470 0636 3
471 0637 3      [%C' ', TAB]:
472 0638 4      BEGIN
473 0639 4      |
474 0640 4      |   Blank or TAB.
475 0641 4      |   Remove extra whitespace by inserting a blank for
476 0642 4      |   only the first one seen.
477 0643 4      |
478 0644 4      IF .WHITE_SPACE EQL -1
479 0645 4      THEN
480 0646 5      BEGIN
481 0647 5      |
482 0648 5      |   No blanks or tabs seen yet. Save position and insert a
483 0649 5      |   blank in the output string.
484 0650 5      |
485 0651 5      WHITE_SPACE = .L_LEN;
486 0652 5      CH$WCHAR_A (%C' ', L_PTR);
487 0653 5      L_LEN = .L_LEN + 1;
488 0654 4      END;
489 0655 4
490 0656 3      END;
491 0657 3
492 0658 3      [OTHERWISE]:
493 0659 4      BEGIN
494 0660 4      |
495 0661 4      |   A normal character
496 0662 4      |
497 0663 4      WHITE_SPACE = -1;
498 0664 4      CH$WCHAR_A (.CH, L_PTR);
```

PERMUTE
V04-000

PERMUTE -- Permute index entries
MAKE_ENTRY -- Make a permuted entry

G 7
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 18
(3)

```

499 0665 4          L_LEN = .L_LEN + 1;
500 0666          END;
501 0667
502 0668          TES;
503 0669
504 0670          END;
505 0671
506 0672
507 0673          ! End of input string reached. Delete trailing white space,
508 0674          append comma, blank
509 0675
510 0676          IF .WHITE_SPACE NEQ -1
511 0677          THEN
512 0678              BEGIN
513 0679                  L_PTR = CH$PLUS (.L_PTR, (.WHITE_SPACE - .L_LEN));
514 0680                  L_LEN = .WHITE_SPACE;
515 0681                  END;
516 0682
517 0683          CH$WCHAR_A (%C',', L_PTR);
518 0684          CH$WCHAR_A (%C',', L_PTR);
519 0685          L_LEN = .L_LEN + 2;
520 0686
521 0687          !
522 0688          ! Raise case of first letter in permuted entry and
523 0689          ! drop case of original first letter if necessary.
524 0690
525 0691          FIX_CASE (.L_LEN, CH$PTR (LINE), .PERMUTED_LEN, .PERMUTED_PTR);
526 0692
527 0693          !
528 0694          ! Finally, append permuted text to end of line and output.
529 0695
530 0696          COPY_STR (.PERMUTED_LEN, .PERMUTED_PTR, L_LEN, L_PTR);
531 0697          XOUT (.L_LEN, CH$PTR (LINE), .XTN, .BAR_FLAG);
532 0698          END;

```

```

                                00FC 0000 MAKE_ENTRY:
57 00000000V EF 9E 00002          .WORD          Save R2,R3,R4,R5,R6,R7          : 0495
56 00000000V EF 9E 00009          MOVAB          COPY_STR, R7
5E FB4C CE 9E 00010          MOVAB          FIX_CASE, R6
52 01 CE 00015          MNEGL          #1, WHITE_SPACE
53 08 AC D0 00018          MOVL          SOURCE_PTR, PTR
04 AE 9F 0001C          PUSHAB          LINE
04 AE D4 0001F          CLRL          L_LEN
54 D4 00022          CLRL          I
00C3 31 00024          BRW          10$
50 83 9A 00027 1$:          MOVZBL          (PTR)+, CH
00000000G 8F 50 D1 0002A          CMPL          CH, #RINTES
03 13 00031          BEQL          2$
0087 31 00033          BRW          6$
51 83 9A 00036 2$:          MOVZBL          (PTR)+, FUNCTION_CODE
55 83 9A 00039          MOVZBL          (PTR)+, OPERAND
54 02 C0 0003C          ADDL2          #2, I          : 0577

```

PERMUTE
V04-000

PERMUTE -- Permute index entries
MAKE_ENTRY -- Make a permuted entry

H 7
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 19
(3)

	0000004A	8F		51	D1	0003F	CMPL	FUNCTION_CODE, #74	0582	
	FFFFFFF	8F		51	12	00046	BNEQ	4\$	0588	
				52	D1	00048	CMPL	WHITE_SPACE, #-1	0591	
50		52	04	0C	13	0004F	BEQL	3\$	0592	
		6E		AE	C3	00051	SUBL3	L_LEN, WHITE_SPACE, R0	0595	
	04	AE		50	C0	00056	ADDL2	R0, L_PTR	0596	
	00	BE		52	D0	00059	MOVL	WHITE_SPACE, L_LEN	0597	
				2C	90	0005D	3\$:	MOVB	#44, @L_PTR	0603
	00	BE		6E	D6	00061	INCL	L_PTR		
				20	90	00063	MOVB	#32, @L_PTR		
	04	AE		6E	D6	00067	INCL	L_PTR		
		7E		02	C0	00069	ADDL2	#2, L_LEN		
			0C	AC	7D	0006D	MOVQ	PERMUTED_LEN, -(SP)		
			10	AE	9F	00071	PUSHAB	LINE		
			10	AE	DD	00074	PUSHL	L_LEN		
		66		04	FB	00077	CALLS	#4, FIX_CASE	0609	
				5E	DD	0007A	PUSHL	SP		
		7E		AE	9F	0007C	PUSHAB	L_LEN		
		67		OC	AC	7D	MOVQ	PERMUTED_LEN, -(SP)		
				04	FB	00083	CALLS	#4, COPY_STR	0610	
				5E	DD	00086	PUSHL	SP		
			08	AE	9F	00088	PUSHAB	L_LEN		
			FD	A3	9F	0008B	PUSHAB	-3(PTR)		
50	04	AC		54	C3	0008E	SUBL3	I, SOURCE_LEN, R0		
			03	A0	9F	00093	PUSHAB	3(R0)		
				0093	31	00096	BRW	12\$		
	00000050	8F		51	D1	00099	4\$:	CMPL	FUNCTION_CODE, #80	0615
				16	13	000A0	BEQL	5\$	0626	
	00	BE		50	90	000A2	MOVB	CH, @L_PTR		
				6E	D6	000A6	INCL	L_PTR		
	00	BE		51	90	000A8	MOVB	FUNCTION_CODE, @L_PTR	0627	
				6E	D6	000AC	INCL	L_PTR		
	00	BE		55	90	000AE	MOVB	OPERAND, @L_PTR	0628	
				6E	D6	000B2	INCL	L_PTR		
	04	AE		03	C0	000B4	ADDL2	#3, L_LEN	0629	
		52		01	CE	000B8	5\$:	MNEGL	#1, WHITE_SPACE	0634
				2D	11	000BB	BRB	10\$	0563	
		20		50	D1	000BD	6\$:	CMPL	CH, #32	0637
				09	13	000C0	BEQL	7\$		
	00000000G	8F		50	D1	000C2	CMPL	CH, #TAB		
				13	12	000C9	BNEQ	8\$		
	FFFFFFF	8F		52	D1	000CB	7\$:	CMPL	WHITE_SPACE, #-1	0644
				16	12	000D2	BNEQ	10\$		
		52	04	AE	D0	000D4	MOVL	L_LEN, WHITE_SPACE	0651	
	00	BE		20	90	000D8	MOVB	#32, @L_PTR	0652	
				07	11	000DC	BRB	9\$		
		52		01	CE	000DE	8\$:	MNEGL	#1, WHITE_SPACE	0663
	00	BE		50	90	000E1	MOVB	CH, @L_PTR	0664	
				6E	D6	000E5	9\$:	INCL	L_PTR	
			04	AE	D6	000E7	INCL	L_LEN	0665	
FF36	54	01	04	AC	F1	000EA	10\$:	ACBL	SOURCE_LEN, #1, I, 1\$	0555
		8F		52	D1	000F1	CMPL	WHITE_SPACE, #-1	0676	
				0C	13	000F8	BEQL	11\$		
	50	52	04	AE	C3	000FA	SUBL3	L_LEN, WHITE_SPACE, R0	0679	
		6E		50	C0	000FF	ADDL2	R0, L_PTR		
	04	AE		52	D0	00102	MOVL	WHITE_SPACE, L_LEN	0680	
	00	BE		2C	90	00106	11\$:	MOVB	#44, @L_PTR	0683

PERMUTE
V04-000

PERMUTE -- Permute index entries
MAKE_ENTRY -- Make a permuted entry

1 7
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 20
(3)

00	BE		6E	D6	0010A	INCL	L_PTR		
			20	90	0010C	MOVB	#32, @L_PTR		0684
04	AE		6E	D6	00110	INCL	L_PTR		
	7E		02	C0	00112	ADDL2	#2, L_LEN		0685
		0C	AC	7D	00116	MOVQ	PERMUTED_LEN, -(SP)		0691
		10	AE	9F	0011A	PUSHAB	LINE		
		10	AE	DD	0011D	PUSHL	L_LEN		
	66		04	FB	00120	CALLS	#4, FIX_CASE		
			5E	DD	00123	PUSHL	SP		0696
		08	AE	9F	00125	PUSHAB	L_LEN		
	7E	0C	AC	7D	00128	MOVQ	PERMUTED_LEN, -(SP)		
	67		04	FB	0012C	CALLS	#4, COPY_STR		
	7E		AC	7D	0012F	MOVQ	XTN, -(SP)		0697
		14	AE	9F	00133	PUSHAB	LINE		
		10	AE	DD	00136	PUSHL	L_LEN		
00000000G	EF		04	FB	00139	CALLS	#4, XOUT		
			04	00140	RET				0698

; Routine Size: 321 bytes, Routine Base: \$CODE\$ + 0144

```
0699 1 %SBTTL 'FIX_CASE -- Fix up case of permuted entry'
0700 1 ROUTINE FIX_CASE (B_LEN, B_PTR, P_LEN, P_PTR) : NOVALUE =
0701 1 ++
0702 1
0703 1 FUNCTIONAL DESCRIPTION:
0704 1
0705 1     This routine is called to fix up the case of the permuted index
0706 1     entry. The case of the first character of the new entry is raised.
0707 1     The case of the first character of the permuted text is lowered
0708 1     if there are any lower case characters in the first word of the
0709 1     permuted text.
0710 1
0711 1 FORMAL PARAMETERS:
0712 1
0713 1     B_LEN - Length of string which begins new entry
0714 1     B_PTR - Pointer to string which begins new entry
0715 1     P_LEN - Length of permuted string
0716 1     P_PTR - Pointer to permuted string
0717 1
0718 1 IMPLICIT INPUTS:
0719 1
0720 1     None
0721 1
0722 1 IMPLICIT OUTPUTS:
0723 1
0724 1     None
0725 1
0726 1 ROUTINE VALUE:
0727 1 COMPLETION CODES:
0728 1
0729 1     None
0730 1
0731 1 SIDE EFFECTS:
0732 1
0733 1     None
0734 1 --
0735 2 BEGIN
0736 2 LOCAL
0737 2     PTR,
0738 2     LEN,
0739 2     FIRST_PTR,
0740 2     FIRST_CH,
0741 2     LOWER;
0742 2
0743 2     !
0744 2     ! Raise case of new first character if necessary
0745 2     !
0746 2     PTR = .B_PTR;
0747 2     LEN = .B_LEN;
0748 2
0749 2     INCR I FROM 1 TO .LEN DO
0750 2         BEGIN
0751 2             LOCAL
0752 2                 CH;
0753 2
0754 2             CH = CH$RCHAR (.PTR);
0755 2
```

```
591 0756 3      IF .CH EQL RINTES
592 0757 3      THEN
593 0758 4          BEGIN
594 0759 4              RUNOFF escape sequence - skip rest of sequence
595 0760 4
596 0761 4              CH$RCHAR_A (PTR);
597 0762 4              CH$RCHAR_A (PTR);
598 0763 4              CH$RCHAR_A (PTR);
599 0764 4              I = .I + 2;
600 0765 4          END
601 0766 4      ELSE
602 0767 3          BEGIN
603 0768 4              Have first character in string
604 0769 4
605 0770 4              IF LOWER_LETTER (.CH)
606 0771 4                  THEN
607 0772 5                      Character is a lowercase letter - convert to upper case
608 0773 4                      CH$WCHAR (UPPER_CASE (.CH), .PTR);
609 0774 4
610 0775 4              EXITLOOP;
611 0776 4              END;
612 0777 4          END;
613 0778 4
614 0779 4      END;
615 0780 3
616 0781 2
617 0782 2
618 0783 2
619 0784 2      : If there's no permuted text then just return
620 0785 2      IF .P_LEN EQL 0 THEN RETURN;
621 0786 2
622 0787 2
623 0788 2      : Find first character in permuted text
624 0789 2
625 0790 2      FIRST_PTR = .P_PTR;
626 0791 2
627 0792 2      DECR I FROM .P_LEN TO 1 DO
628 0793 2          BEGIN
629 0794 3              FIRST_CH = CH$RCHAR (.FIRST_PTR);
630 0795 3
631 0796 3              IF .FIRST_CH EQL RINTES
632 0797 3                  THEN
633 0798 3                      BEGIN
634 0799 3                          RUNOFF escape sequence - skip it
635 0800 4
636 0801 4                          CH$RCHAR_A (FIRST_PTR);
637 0802 4                          CH$RCHAR_A (FIRST_PTR);
638 0803 4                          CH$RCHAR_A (FIRST_PTR);
639 0804 4                          I = .I - 2;
640 0805 4                      END
641 0806 4                  ELSE
642 0807 4                      BEGIN
643 0808 4                          Have pointer to first character.
644 0809 3
645 0810 4
646 0811 4
647 0812 4
```

```

: 648      0813 4      ! Set length of remainder of string and exit.
: 649      0814 4
: 650      0815 4      LEN = .I - 1;
: 651      0816 4      EXITLOOP;
: 652      0817 3      END;
: 653      0818 2      END;
: 654      0819 2
: 655      0820 2
: 656      0821 3      IF UPPER_LETTER (.FIRST_CH)
: 657      0822 2      THEN
: 658      0823 3      BEGIN
: 659      0824 3      ! Lower case of first character in permuted text if the
: 660      0825 3      ! second letter in the word is lowercase.
: 661      0826 3
: 662      0827 3      PTR = CH$PLUS (.FIRST_PTR, 1);
: 663      0828 3
: 664      0829 3      LOWER = FALSE;
: 665      0830 3      ! No lower case characters seen
: 666      0831 3
: 667      0832 3      INCR I FROM 1 TO .LEN DO
: 668      0833 4      BEGIN
: 669      0834 4      LOCAL
: 670      0835 4      CH;
: 671      0836 4
: 672      0837 4      CH = CH$RCHAR_A (PTR);
: 673      0838 4
: 674      0839 4      SELECTONE TRUE OF
: 675      0840 4      SET
: 676      0841 4      [CH EQL RINTES]:
: 677      0842 4      BEGIN
: 678      0843 5      ! RUNOFF escape sequence - skip rest of sequence
: 679      0844 5
: 680      0845 5      CH$RCHAR_A (PTR);
: 681      0846 5      CH$RCHAR_A (PTR);
: 682      0847 5      I = .I + 2;
: 683      0848 5      END;
: 684      0849 5
: 685      0850 4
: 686      0851 4      [(CH EQL %C' ') OR (CH EQL TAB)]:
: 687      0852 4      BEGIN
: 688      0853 5      ! Whitespace - end of word
: 689      0854 5      ! Leave case alone.
: 690      0855 5
: 691      0856 5      EXITLOOP;
: 692      0857 5      END;
: 693      0858 5
: 694      0859 4
: 695      0860 4      [LOWER_LETTER (.CH)]:
: 696      0861 4      BEGIN
: 697      0862 5      ! Lower case letter - note it and exit loop
: 698      0863 5
: 699      0864 5      LOWER = TRUE;
: 700      0865 5      EXITLOOP;
: 701      0866 5      END;
: 702      0867 5
: 703      0868 4
: 704      0869 4
```

```

0870 4
0871 4
0872 4
0873 4
0874 4
0875 4
0876 4
0877 4
0878 4
0879 4
0880 4
0881 4
0882 4
0883 4
0884 4
0885 3
0886 3
0887 3
0888 2
0889 2
0890 1
UPPER_LETTER (.CH)]:
    Second letter is upper case. Terminate scan but don't
    lower case of first letter.
EXITLOOP;
[OTHERWISE]:
    Some other character - ignore it
;
TES;
END;
IF .LOWER THEN CH$WCHAR (LOWER_CASE (.FIRST_CH), .FIRST_PTR);
END;
END:

```

07FC 00000 FIX_CASE:				.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10	
	5A	00G	8F 9A 00002	MOVZBL	#RINTES, R10	: 0700
	51	08	AC D0 00006	MOVL	B_PTR, PTR	: 0746
	53	04	AC D0 0000A	MOVL	B_LEN, LEN	: 0747
			52 D4 0000E	CLRL	I	: 0749
			28 11 00010	BRB	3\$	: 0754
	50		61 9A 00012 1\$:	MOVZBL	(PTR), CH	: 0756
	5A		50 D1 00015	CMPL	CH, R10	: 0764
			08 12 00018	BNEQ	2\$	: 0765
	51		03 C0 0001A	ADDL2	#3, PTR	: 0756
	52		02 C0 0001D	ADDL2	#2, I	: 0772
			18 11 00020	BRB	3\$	: 0777
00000061	8F		50 D1 00022 2\$:	CMPL	CH, #97	: 0768
			13 19 00029	BLSS	4\$	: 0749
0000007A	8F		50 D1 0002B	CMPL	CH, #122	: 0786
			0A 14 00032	BGTR	4\$	: 0791
61	50		20 83 00034	SUBB3	#32, CH, (PTR)	: 0793
			04 11 00038	BRB	4\$	: 0796
D4	52		53 F3 0003A 3\$:	AOBLEQ	LEN, I, 1\$	: 0798
		0C	AC D5 0003E 4\$:	TSTL	P_LEN	: 0806
			01 12 00041	BNEQ	5\$	: 0807
			04 00043	RET		
	52	10	AC D0 00044 5\$:	MOVL	P_PTR, FIRST_PTR	: 0791
50	OC		01 C1 00048	ADDL3	#T, P_LEN, I	: 0793
			16 11 0004D	BRB	8\$	: 0796
	57		62 9A 0004F 6\$:	MOVZBL	(FIRST_PTR), FIRST_CH	: 0798
	5A		57 D1 00052	CMPL	FIRST_CH, R10	: 0806
			08 12 00055	BNEQ	7\$	: 0807
	52		03 C0 00057	ADDL2	#3, FIRST_PTR	: 0806
	50		02 C2 0005A	SUBL2	#2, I	: 0807

PERMUTE
V04-000

PERMUTE -- Permute index entries
FIX_CASE -- Fix up case of permuted entry

N 7
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 25
(4)

	53	FF	06	11	0005D		BRB	8\$		0798
			A0	9E	0005F	7\$:	MOVAB	-1(R0), LEN		0815
			03	11	00063		BRB	9\$		0810
00000041	E7		50	F5	00065	8\$:	SOBGR	I, 6\$		0793
	8F		57	D1	00068	9\$:	CMPL	FIRST_CH, #65		0821
			01	18	0006F		BGEQ	10\$		
				04	00071		RET			
0000005A	8F		57	D1	00072	10\$:	CMPL	FIRST_CH, #90		
			01	15	00079		BLEQ	11\$		
				04	0007B		RET			
	51	01	A2	9E	0007C	11\$:	MOVAB	1(R2), PTR		0828
			58	D4	00080		CLRL	LOWER		0830
			56	D4	00082		CLRL	I		0832
			7D	11	00084		BRB	21\$		
	54		81	9A	00086	12\$:	MOVZBL	(PTR)+, CH		0837
	5A		54	D1	00089		CMPL	CH, R10		0842
			08	12	0008C		BNEQ	13\$		
	51		02	C0	0008E		ADDL2	#2, PTR		0848
	56		02	C0	00091		ADDL2	#2, I		0849
			6D	11	00094		BRB	21\$		0839
			55	D4	00096	13\$:	CLRL	R5		0852
	20		54	D1	00098		CMPL	CH, #32		
			02	12	0009B		BNEQ	14\$		
			55	D6	0009D		INCL	R5		
			50	D4	0009F	14\$:	CLRL	R0		
00000000G	8F		54	D1	000A1		CMPL	CH, #TAB		
			02	12	000A8		BNEQ	15\$		
			50	D6	000AA		INCL	R0		
	50		55	C8	000AC	15\$:	BISL2	R5, R0		
	01		50	D1	000AF		CMPL	R0, #1		
			55	13	000B2		BEQL	22\$		
			55	D4	000B4		CLRL	R5		0861
00000061	8F		54	D1	000B6		CMPL	CH, #97		
			02	19	000BD		BLSS	16\$		
			55	D6	000BF		INCL	R5		
			50	D4	000C1	16\$:	CLRL	R0		
0000007A	8F		54	D1	000C3		CMPL	CH, #122		
			02	14	000CA		BGTR	17\$		
			50	D6	000CC		INCL	R0		
	59		55	D2	000CE	17\$:	MCOML	R5, R9		
	50		59	CA	000D1		BICL2	R9, R0		
	01		50	D1	000D4		CMPL	R0, #1		
			05	12	000D7		BNEQ	18\$		
	58		01	D0	000D9		MOVL	#1, LOWER		0866
			2B	11	000DC		BRB	22\$		0862
			55	D4	000DE	18\$:	CLRL	R5		0870
00000041	8F		54	D1	000E0		CMPL	CH, #65		
			02	19	000E7		BLSS	19\$		
			55	D6	000E9		INCL	R5		
			50	D4	000EB	19\$:	CLRL	R0		
0000005A	8F		54	D1	000ED		CMPL	CH, #90		
			02	14	000F4		BGTR	20\$		
			50	D6	000F6		INCL	R0		
	59		55	D2	000F8	20\$:	MCOML	R5, R9		
	50		59	CA	000FB		BICL2	R9, R0		
	01		50	D1	000FE		CMPL	R0, #1		
			06	13	00101		BEQL	22\$		

PERMUTE
V04-000

PERMUTE -- Permute index entries
FIX_CASE -- Fix up case of permuted entry

B 8
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 26
(4)

FF7D

56

01

53

F1 00103 21\$:

ACBL

LEN, #1, I, 12\$

: 0832

62

04

58

E9 00109 22\$:

BLBC

LOWER, 23\$

: 0887

20

81 0010C

ADDB3

#32, FIRST_CH, (FIRST_PTR)

: 0890

04

00110 23\$:

RET

; Routine Size: 273 bytes, Routine Base: \$CODE\$ + 0285

PERMUTE
V04-000

PERMUTE -- Permute index entries
COPY_STR -- Copy string source to destination

C 8
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 27
(5)

```
: 727 0891 1 %SBTTL 'COPY_STR -- Copy string source to destination'
: 728 0892 1 ROUTINE COPY_STR (SOURCE_LEN, SOURCE_PTR, DEST_LEN, DEST_PTR) : NOVALUE =
: 729 0893 1 ++
: 730 0894 1
: 731 0895 1 FUNCTIONAL DESCRIPTION:
: 732 0896 1
: 733 0897 1 This routine copies the string described by SOURCE_LEN and
: 734 0898 1 SOURCE_PTR to the area pointed to by DEST_PTR.
: 735 0899 1
: 736 0900 1 Upon completion, DEST_PTR points to the next character position
: 737 0901 1 in the output string and DEST_LEN has been updated to reflect
: 738 0902 1 the number of characters copied.
: 739 0903 1
: 740 0904 1 Trailing white space is deleted from the source string.
: 741 0905 1
: 742 0906 1 FORMAL PARAMETERS:
: 743 0907 1
: 744 0908 1 SOURCE_LEN - Length of source string
: 745 0909 1 SOURCE_PTR - CH$PTR to source string
: 746 0910 1 DEST_LEN - Address of destination string length variable
: 747 0911 1 DEST_PTR - Address of a CH$PTR to destination string
: 748 0912 1
: 749 0913 1 IMPLICIT INPUTS:
: 750 0914 1
: 751 0915 1 None
: 752 0916 1
: 753 0917 1 IMPLICIT OUTPUTS:
: 754 0918 1
: 755 0919 1 None
: 756 0920 1
: 757 0921 1 ROUTINE VALUE:
: 758 0922 1 COMPLETION CODES:
: 759 0923 1
: 760 0924 1 None
: 761 0925 1
: 762 0926 1 SIDE EFFECTS:
: 763 0927 1
: 764 0928 1 None
: 765 0929 1 --
: 766 0930 1
: 767 0931 2 BEGIN
: 768 0932 2
: 769 0933 2 BIND
: 770 0934 2 TO_LEN = .DEST_LEN,
: 771 0935 2 TO_PTR = .DEST_PTR;
: 772 0936 2
: 773 0937 2 LOCAL
: 774 0938 2 WHITE_SPACE,
: 775 0939 2 FROM_PTR;
: 776 0940 2
: 777 0941 2 FROM_PTR = .SOURCE_PTR;
: 778 0942 2 WHITE_SPACE = -1;
: 779 0943 2
: 780 0944 2 INCR I FROM 1 TO .SOURCE_LEN DO
: 781 0945 3 BEGIN
: 782 0946 3
: 783 0947 3 LOCAL
```

```

: 784      0948      3      CH;                                ! Current character
: 785      0949      3
: 786      0950      3      CH = CH$RCHAR_A (FROM_PTR);
: 787      0951      3
: 788      0952      3      SELECTONE .CH OF
: 789      0953      3      SET
: 790      0954      3
: 791      0955      3      [RINTES]:
: 792      0956      4      BEGIN
: 793      0957      4      |
: 794      0958      4      | RUNOFF escape sequence
: 795      0959      4      |
: 796      0960      4
: 797      0961      4      LOCAL
: 798      0962      4      FUNCTION_CODE,
: 799      0963      4      OPERAND;
: 800      0964      4
: 801      0965      4      FUNCTION_CODE = CH$RCHAR_A (FROM_PTR);
: 802      0966      4      OPERAND = CH$RCHAR_A (FROM_PTR);
: 803      0967      4      I = .I + 2;
: 804      0968      4
: 805      0969      4      IF .FUNCTION_CODE NEQ %C'P'
: 806      0970      4      THEN
: 807      0971      5      BEGIN
: 808      0972      5      |
: 809      0973      5      | Copy any runoff sequence except the nopermute sequence
: 810      0974      5      |
: 811      0975      5      | CH$WCHAR_A (.CH, TO_PTR);
: 812      0976      5      | CH$WCHAR_A (.FUNCTION_CODE, TO_PTR);
: 813      0977      5      | CH$WCHAR_A (.OPERAND, TO_PTR);
: 814      0978      5      | TO_LEN = .TO_LEN + 3;
: 815      0979      4      | END;
: 816      0980      4
: 817      0981      4      WHITE_SPACE = -1;                    ! No trailing white space
: 818      0982      3      END;
: 819      0983      3
: 820      0984      3      [%C' ', TAB]:
: 821      0985      4      BEGIN
: 822      0986      4      |
: 823      0987      4      | Blank or TAB.
: 824      0988      4      | Remove extra whitespace by inserting a blank for
: 825      0989      4      | only the first one seen.
: 826      0990      4      |
: 827      0991      4      | IF .WHITE_SPACE EQL -1
: 828      0992      4      | THEN
: 829      0993      5      | BEGIN
: 830      0994      5      | |
: 831      0995      5      | | No blanks or tabs seen yet. Save position and insert a
: 832      0996      5      | | blank in the output string.
: 833      0997      5      | |
: 834      0998      5      | | WHITE_SPACE = .TO_LEN;
: 835      0999      5      | | CH$WCHAR_A (%C' ', TO_PTR);
: 836      1000      5      | | TO_LEN = .TO_LEN + 1;
: 837      1001      4      | | END;
: 838      1002      4
: 839      1003      3      END;
: 840      1004      3
```

```

: 841      1005 3      [OTHERWISE]:
: 842      1006 4      BEGIN
: 843      1007 4      |
: 844      1008 4      |   Copy a normal character
: 845      1009 4      |
: 846      1010 4      |   WHITE_SPACE = -1;
: 847      1011 4      |   CH$WCHAR_A (.CH, TO_PTR);
: 848      1012 4      |   TO_LEN = .TO_LEN + 1;
: 849      1013 3      |   END;
: 850      1014 3
: 851      1015 3      TES;
: 852      1016 3
: 853      1017 2      END;
: 854      1018 2
: 855      1019 2      IF .WHITE_SPACE NEQ -1
: 856      1020 2      THEN
: 857      1021 3      BEGIN
: 858      1022 3      |
: 859      1023 3      |   Delete trailing whitespace
: 860      1024 3      |
: 861      1025 3      |   TO_PTR = CH$PLUS (.TO_PTR, (.WHITE_SPACE - .TO_LEN));
: 862      1026 3      |   TO_LEN = .WHITE_SPACE;
: 863      1027 2      |   END;
: 864      1028 2
: 865      1029 1      END;
```

```

                                00FC 00000 COPY_STR:
                                .WORD      Save R2,R3,R4,R5,R6,R7
                                53          0C AC D0 00002      MOVL      DEST_LEN, R3
                                51          10 AC D0 00006      MOVL      DEST_PTR, R1
                                55          08 AC D0 0000A      MOVL      SOURCE_PTR, FROM_PTR
                                50          01 CE 0000E      MNEGL     #1, WHITE_SPACE
                                54          D4 00011      CLRL      I
                                63          11 00013      BRB       7$
                                52          85 9A 00015      MOVZBL   (FROM_PTR)+, CH
                                8F          52 D1 00018      CMPL     CH, #RINTES
                                00000000G 2C 12 0001F      BNEQ      3$
                                56          85 9A 00021      MOVZBL   (FROM_PTR)+, FUNCTION_CODE
                                57          85 9A 00024      MOVZBL   (FROM_PTR)+, OPERAND
                                54          02 C0 00027      ADDL2     #2, I
                                00000050 8F 56 D1 0002A      CMPL     FUNCTION_CODE, #80
                                00          15 13 00031      BEQL     2$
                                00 B1          52 90 00033      MOVVB    CH, @0(R1)
                                00 B1          61 D6 00037      INCL     (R1)
                                00 B1          56 90 00039      MOVVB    FUNCTION_CODE, @0(R1)
                                00 B1          61 D6 0003D      INCL     (R1)
                                63          03 C0 00045      ADDL2     #3, (R3)
                                50          01 CE 00048      MNEGL     #1, WHITE_SPACE
                                20          2B 11 0004B      BRB       7$
                                09          52 D1 0004D      CMPL     CH, #32
                                13 00050      BEQL     4$
                                0892
                                0934
                                0935
                                0941
                                0942
                                0944
                                0950
                                0955
                                0965
                                0966
                                0967
                                0969
                                0975
                                0976
                                0977
                                0978
                                0981
                                0952
                                0984
```

PERMUTE
V04-000

PERMUTE -- Permute index entries
COPY_STR -- Copy string source to destination

F 8
16-Sep-1984 01:27:55
14-Sep-1984 13:07:46

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]PERMUTE.BLI;1

Page 30
(5)

00000000G	8F	52	D1	00052	CMPL	CH, #TAB	:	
FFFFFFFF	8F	12	12	00059	BNEQ	5\$	:	
		50	D1	0005B	4\$:	CMPL	WHITE_SPACE, #-1	0991
		14	12	00062	BNEQ	7\$	:	
	50	63	D0	00064	MOVL	(R3), WHITE_SPACE	0998	
00	B1	20	90	00067	MOVB	#32, @0(R1)	0999	
		07	11	0006B	BRB	6\$	:	
	50	01	CE	0006D	5\$:	MNEGL	#1, WHITE_SPACE	1010
00	B1	52	90	00070	MOVB	CH, @0(R1)	1011	
		61	D6	00074	6\$:	INCL	(R1)	:
		63	D6	00076	INCL	(R3)	1012	
98	54	AC	F3	00078	7\$:	AOBLEQ	SOURCE_LEN, 1, 1\$	0944
FFFFFFFF	8F	50	D1	0007D	CMPL	WHITE_SPACE, #-1	1019	
		0A	13	00084	BEQL	8\$	:	
52	50	63	C3	00086	SUBL3	(R3), WHITE_SPACE, R2	1025	
	61	52	C0	0008A	ADDL2	R2, (R1)	:	
	63	50	D0	0008D	MOVL	WHITE_SPACE, (R3)	1026	
		04	00090	8\$:	RET		1029	

; Routine Size: 145 bytes, Routine Base: \$CODE\$ + 0396

: 866	1030	1		
: 867	1031	1	END	! End of module
: 868	1032	0	ELUDOM	

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	1063	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]XPORT.L32;1	590	28	4	252	00:00.1

COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:PERMUTE/OBJ=OBJ\$:PERMUTE MSRC\$:PERMUTE/UPDATE=(ENH\$:PERMUTE)

PERMUTE
V04-000

PERMUTE -- Permute index entries
COPY_STR -- Copy string source to destination

G 8
16-Sep-1984 01:27:55

VAX-11 Bliss-32 V4.0-742

Page 31

: Size: 1063 code + 0 data bytes
: Run Time: 00:21.9
: Elapsed Time: 00:43.7
: Lines/CPU Min: 2824
: Lexemes/CPU-Min: 24725
: Memory Used: 133 pages
: Compilation Complete

0347 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY