

[illegible]

3-

Sy

LI

LI
LILI
LI
LILI
LI

LI

LI
LILI
LI
LILI
LI

LI

LI
LI

LI

LI
LI

LI

LI
LI

LI

11

LI

LI
LI

LI

LI
LI

21

LI

LI
LI

22

11

LI
LI

LI

LI
LI

LI

```
000000  TTTTTTTTTT  SSSSSSSSS  MM  MM  000000  VV  VV  EEEEEEEEEEE
000000  TTTTTTTTTT  SSSSSSSSS  MM  MM  000000  VV  VV  EEEEEEEEEEE
00  00  TT  SS  MMMM  MMMM  00  00  VV  VV  EE
00  00  TT  SS  MMMM  MMMM  00  00  VV  VV  EE
00  00  TT  SS  MM  MM  00  00  VV  VV  EE
00  00  TT  SSSSSS  MM  MM  00  00  VV  VV  EEEEEEEE
00  00  TT  SSSSSS  MM  MM  00  00  VV  VV  EEEEEEEEEEE
00  00  TT  SS  MM  MM  00  00  VV  VV  EE
00  00  TT  SS  MM  MM  00  00  VV  VV  EE
00  00  TT  SS  MM  MM  00  00  VV  VV  EE
00  00  TT  SS  MM  MM  00  00  VV  VV  EE
000000  TT  SSSSSSSS  MM  MM  000000  VV  VV  EEEEEEEEEEE
000000  TT  SSSSSSSS  MM  MM  000000  VV  VV  EEEEEEEEEEE
```

```
LL  IIIIIII  SSSSSSSSS
LL  IIIIIII  SSSSSSSSS
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
LLLLLLLLLLL  IIIIIII  SSSSSSSS
LLLLLLLLLLL  IIIIIII  SSSSSSSS
```


(2) 50
(3) 78
(4) 128
(5) 226
(6) 286

DECLARATIONS
OTSSMOVE3 - Move characters without fill
OTSSMOVE3_R5 - Move characters without fill
OTSSMOVE5 - Move characters with fill
OTSSMOVE5_R5 - Move characters with fill

OTSSMOVE3 - Move characters without fill
OTSSMOVE3_R5 - Move characters without fill
OTSSMOVE5 - Move characters with fill
OTSSMOVE5_R5 - Move characters with fill

```

0000 1 .TITLE OTSS$MOVE - Move characters
0000 2 .IDENT /1-005/ ; File: OTSS$MOVE.MAR Edit: SBL1005
0000 3
0000 4
0000 5 *****
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 *
0000 25 *
0000 26 *****
0000 27
0000 28
0000 29 :++
0000 30 : FACILITY: Language-independent Compiled Code Support
0000 31
0000 32 : ABSTRACT:
0000 33
0000 34 : This module contains a procedure which moves up to 2**31-1
0000 35 : characters.
0000 36
0000 37 : ENVIRONMENT: Runs at any access mode, AST Reentrant
0000 38
0000 39 : AUTHOR: Steven B. Lionel, CREATION DATE: 14-SEP-1981
0000 40
0000 41 : MODIFIED BY:
0000 42
0000 43 : 1-001 - Original. SBL 14-SEP-1981
0000 44 : 1-002 - Code improvement. SBL 8-Dec-1981
0000 45 : 1-003 - Add OTSS$MOVE5 and OTSS$MOVE5_R5 entry points
0000 46 : 1-004 - Fix problem with just filling of >65K bytes. SBL 27-July-1982
0000 47 : 1-005 - Fix register-saving bug with backwards copy. SBL 17-May-1983
0000 48 :--

```



```

0000 50      .SBTTL  DECLARATIONS
0000 51      :
0000 52      : LIBRARY MACRO CALLS:
0000 53      :
0000 54      :     NONE
0000 55      :
0000 56      : EXTERNAL DECLARATIONS:
0000 57      :
0000 58      :     NONE
0000 59      :
0000 60      : MACROS:
0000 61      :
0000 62      :     NONE
0000 63      :
0000 64      : EQUATED SYMBOLS:
0000 65      :
0000 66      :     NONE
0000 67      :
0000 68      : OWN STORAGE:
0000 69      :
0000 70      :     NONE
0000 71      :
0000 72      : PSECT DECLARATIONS:
0000 73      :
0000 74      : .PSECT _OTSS$CODE PIC, USR, CON, REL, LCL, SHR, -
0000 75      :     EXE, RD, NOWRT, LONG
0000 76

```

OTSS\$MOVE3 - Move characters without fill

```

0000 78 .SBTTL OTSS$MOVE3 - Move characters without fill
0000 79 :++
0000 80 : FUNCTIONAL DESCRIPTION:
0000 81 :
0000 82 : This procedure moves up to 2**31-1 characters from a specified
0000 83 : source address to a specified destination address. Overlapping
0000 84 : fields are handled correctly.
0000 85 :
0000 86 : CALLING SEQUENCE:
0000 87 :
0000 88 : CALL OTSS$MOVE3 (length.rl.v, source.rbu.ra, dest.wbu.ra)
0000 89 :
0000 90 : FORMAL PARAMETERS:
0000 91 :
0000 92 :
00000004 0000 93 : length = 4 ; Number of bytes to move, passed by
0000 94 : ; immediate value. Value may range from
0000 95 : ; 0 through 2147483647.
0000 96 :
00000008 0000 97 : source = 8 ; Characters to move, passed by reference.
0000 98 :
0000000C 0000 99 : dest = 12 ; Area to receive moved characters, passed
0000 100 : ; by reference.
0000 101 :
0000 102 :
0000 103 : IMPLICIT INPUTS:
0000 104 :
0000 105 : NONE
0000 106 :
0000 107 : IMPLICIT OUTPUTS:
0000 108 :
0000 109 : NONE
0000 110 :
0000 111 : FUNCTION VALUE:
0000 112 :
0000 113 : NONE
0000 114 :
0000 115 : SIDE EFFECTS:
0000 116 :
0000 117 : NONE
0000 118 :
0000 119 : --
0000 120 :
003C 0000 121 : .ENTRY OTSS$MOVE3, ^M<R2,R3,R4,R5>
0002 122 :
50 04 AC 7D 0002 123 : MOVQ length(AP), R0 ; Get length and source address
52 0C AC D0 0006 124 : MOVL dest(AP), R2 ; Get destination address
01 10 000A 125 : BSBB OTSS$MOVE3_R5 ; Do the move
04 000C 126 : RET ; Return to caller

```



```

000D 128 .SBTTL OTSS$MOVE3_R5 - Move characters without fill
000D 129 :++
000D 130 : FUNCTIONAL DESCRIPTION:
000D 131 :
000D 132 : This procedure moves up to 2**31-1 characters from a specified
000D 133 : source address to a specified destination address. Overlapping
000D 134 : fields are handled correctly.
000D 135 :
000D 136 : CALLING SEQUENCE:
000D 137 :
000D 138 : JSB OTSS$MOVE3_R5 (length.rl.v, source.rbu.ra, dest.wbu.ra)
000D 139 :
000D 140 : FORMAL PARAMETERS:
000D 141 :
000D 142 :
000D 143 : length = R0 ; Number of bytes to move. Value may range
000D 144 : ; from 0 through 2147483647.
000D 145 :
000D 146 : source = R1 ; Address of characters to move.
000D 147 :
000D 148 : dest = R2 ; Address of area to receive moved characters.
000D 149 :
000D 150 :
000D 151 : IMPLICIT INPUTS:
000D 152 :
000D 153 : NONE
000D 154 :
000D 155 : IMPLICIT OUTPUTS:
000D 156 :
000D 157 : R0 = 0
000D 158 : R1 = Address of one byte beyond the source string.
000D 159 : R2 = 0
000D 160 : R3 = Address of one byte beyond the destination string.
000D 161 : R4 = 0
000D 162 : R5 = 0
000D 163 :
000D 164 : FUNCTION VALUE:
000D 165 :
000D 166 : NONE
000D 167 :
000D 168 : SIDE EFFECTS:
000D 169 :
000D 170 : NONE
000D 171 :
000D 172 : --
000D 173 :
000D 174 : OTSS$MOVE3_R5::
000D 175 :
000D 176 : CMPL R0, #65535 ; Is length greater than 65535?
000D 177 : BGTR BIGSTRING ; If so, can't do simple move.
000D 178 : MOVC3 R0, (R1), (R2) ; Simple case; do the move
000D 179 : RSB
000D 180 :
000D 181 : BIGSTRING:
000D 182 : CMPL R1, R2 ; Check for overlap that would prevent
000D 183 : BGEQU FORWARDS ; a forward copy.
000D 184 : SUBL3 R1, R2, R3 ; Get distance between start points

```

OTSSMOVE
1-005

- Move characters

F 16

16-SEP-1984 00:32:51 VAX/VMS Macro V04-00
6-SEP-1984 11:15:07 [LIBRTL.SRC]OTSMOVE.MAR;1

Page 5
(4)

OTSSMOVE3_R5 - Move characters without f

```

50 53 D1 0024 185      CMPL   R3, R0      ; and compare it to string size.
42 1E 0027 186      BGEQU  FORWARDS      ; If distance larger than string size,
      0029 187      ; ok.
      0029 188
      0029 189      ;+
      0029 190      ; Come here if we have to do the copy from right to left because of
      0029 191      ; overlap.
      0029 192      ; -
      0029 193
      0029 194 BACKWARDS:
53 52 50 C1 0029 195      ADDL3  R0, R2, R3      ; R3 points past end of dest
51 50 C0 002D 196      ADDL2  R0, R1      ; R1 points past end of source
0B BB 0030 197      PUSHHR #^M<R0,R1,R3>      ; Save length remaining, drc and dest end
      0032 198 10$:      MOVCL3 #65535, -65535(R1), -      ; Move a segment
      003B 200      -65535(R3)      ; R1 and R3 get set by the MOVCL3
      0040 201      ; to point past the source and dest.
51 0000FFFF 8F C2 0040 202      SUBL2  #65535, R1      ; Get new source address
53 0000FFFF 8F C2 0047 203      SUBL2  #65535, R3      ; Get new dest address
6E FFFF0001 8F 00010000 8F F1 004E 204      ACBL  #65536, #-65535, (SP), 10$ ; Loop until <65536 bytes left
      FFD6 005A
      50 8E D0 005C 205      MOVL   (SP)+, R0      ; Get length remaining in R0
      52 50 CE 005F 206      MNEGL  R0, R2      ; Get -length in R2
6342 6142 50 28 0062 207      MOVCL3 R0, (R1)[R2], (R3)[R2] ; Do the final move
      0A BA 0068 208      POPR   #^M<R1,R3>      ; Set R1 and R3 to proper end values
      05 006A 209      RSB
      006B 210
      006B 211      ;+
      006B 212      ; Come here if it's ok to do the copy from left to right.
      006B 213      ; -
      006B 214
      006B 215 FORWARDS:
      50 DD 006B 216      PUSHL  R0      ; Save length remaining
63 61 53 52 D0 006D 217      MOVL  R2, R3      ; Move dest address to R3
      FFFF 8F 28 0070 218 10$:      MOVCL3 #65535, (R1), (R3) ; Move a segment. R1 and R3 get
      0076 219      ; updated with new source and dest
      0076 220      ; addresses.
6E FFFF0001 8F 00010000 8F F1 0076 221      ACBL  #65536, #-65535, (SP), 10$ ; Repeat until <65536 bytes left
      FFEC 0082
      50 D0 0084 222      MOVL   (SP)+, R0      ; Get final length
63 61 50 28 0087 223      MOVCL3 R0, (R1), (R3) ; Do the final move
      05 008B 224      RSB      ; Return
```



```
008C 226 .SBTTL OTSS$MOVE5 - Move characters with fill
008C 227 :++
008C 228 : FUNCTIONAL DESCRIPTION:
008C 229 :
008C 230 : This procedure moves up to 2**31-1 characters from a specified
008C 231 : source address to a specified destination address, with separate
008C 232 : source and destination lengths, and with fill. Overlapping
008C 233 : fields are handled correctly.
008C 234 :
008C 235 : CALLING SEQUENCE:
008C 236 :
008C 237 : CALL OTSS$MOVE5 (srclen.rl.v, source.rbu.ra, fill.rbu.v,
008C 238 : dstlen.rl.v, dest.wbu.ra)
008C 239 :
008C 240 : FORMAL PARAMETERS:
008C 241 :
008C 242 :
00000004 008C 243 srclen = 4 ; Number of bytes in the source, passed by
008C 244 ; immediate value. Value may range from
008C 245 ; 0 through 2147483647.
008C 246 :
00000008 008C 247 source = 8 ; Characters to move, passed by reference.
008C 248 :
0000000C 008C 249 fill = 12 ; Fill byte to use when the srclen is less than
008C 250 ; dstlen. Passed by immediate value.
008C 251 :
00000010 008C 252 dstlen = 16 ; Number of bytes in the destination, passed by
008C 253 ; immediate value. Value may range from
008C 254 ; 0 through 2147483647.
008C 255 :
00000014 008C 256 dest = 20 ; Area to receive moved characters, passed
008C 257 ; by reference.
008C 258 :
008C 259 :
008C 260 : IMPLICIT INPUTS:
008C 261 :
008C 262 : NONE
008C 263 :
008C 264 : IMPLICIT OUTPUTS:
008C 265 :
008C 266 : NONE
008C 267 :
008C 268 : FUNCTION VALUE:
008C 269 :
008C 270 : NONE
008C 271 :
008C 272 : SIDE EFFECTS:
008C 273 :
008C 274 : NONE
008C 275 :
008C 276 :--
008C 277 :
003C 008C 278 .ENTRY OTSS$MOVE5, ^M<R2,R3,R4,R5>
008E 279 :
50 04 AC 7D 008E 280 MOVQ srclen(AP), R0 ; Get source length and address
52 0C AC 7D 0092 281 MOVQ fill(AP), R2 ; Get fill and destination length
54 14 AC D0 0096 282 MOVL dest(AP), R4 ; Get destination address
```

OTSS\$MOVE
1-005

- Move characters
OTSS\$MOVE5 - Move characters with fill

H 16

16-SEP-1984 00:32:51 VAX/VMS Macro V04-00
6-SEP-1984 11:15:07 [LIBRTL.SRC]OTSS\$MOVE.MAR;1

Page 7
(5)

01	10	009A	283	BSBB	OTSS\$MOVE5_R5	; Do the move
	04	009C	284	RET		; Return to caller


```

009D 286 .SBTTL OTSS$MOVE5_R5 - Move characters with fill
009D 287 :++
009D 288 : FUNCTIONAL DESCRIPTION:
009D 289 :
009D 290 : This procedure moves up to 2**31-1 characters from a specified
009D 291 : source address to a specified destination address, with separate
009D 292 : source and destination lengths, and with fill. Overlapping
009D 293 : fields are handled correctly.
009D 294 :
009D 295 : CALLING SEQUENCE:
009D 296 :
009D 297 : JSB OTSS$MOVE5_R5 (srcLen.rl.v, source.rbu.ra, fill.rbu.v,
009D 298 : dstLen.rl.v, dest.wbu.ra)
009D 299 :
009D 300 : FORMAL PARAMETERS:
009D 301 :
009D 302 :
009D 303 : srcLen = R0 ; Number of bytes in the source
009D 304 : ; Value may range from
009D 305 : ; 0 through 2147483647.
009D 306 :
009D 307 : source = R1 ; Address of characters to move
009D 308 :
009D 309 : fill = R2 ; Fill byte to use when the srcLen is less than
009D 310 : ; dstLen.
009D 311 :
009D 312 : dstLen = R3 ; Number of bytes in the destination.
009D 313 : ; Value may range from
009D 314 : ; 0 through 2147483647.
009D 315 :
009D 316 : dest = R4 ; Address of area to receive moved characters.
009D 317 :
009D 318 :
009D 319 : IMPLICIT INPUTS:
009D 320 :
009D 321 : NONE
009D 322 :
009D 323 : IMPLICIT OUTPUTS:
009D 324 :
009D 325 : R0 = Number of unmoved bytes remaining in source string.
009D 326 : R1 = Address of one byte beyond the source string.
009D 327 : R2 = 0
009D 328 : R3 = Address of one byte beyond the destination string.
009D 329 : R4 = 0
009D 330 : R5 = 0
009D 331 :
009D 332 : FUNCTION VALUE:
009D 333 :
009D 334 : NONE
009D 335 :
009D 336 : SIDE EFFECTS:
009D 337 :
009D 338 : NONE
009D 339 :
009D 340 : --
009D 341 :
009D 342 OTSS$MOVE5_R5::

```

```

0000FFFF 8F 50 D1 009D 343
10 14 00A4 344
0000FFFF 8F 53 D1 00A6 345
07 14 00AD 346
64 53 52 61 50 2C 00AF 347
05 00B5 348
00B6 349
00B6 350
00B6 351
7E 53 50 C3 00B6 352
0F 14 00BA 353
03 13 00BC 354
50 53 D0 00BE 355
52 54 D0 00C1 356
FF46 30 00C4 357
50 8E D0 00C7 358
05 00CA 359
00CB 360
00CB 361
00CB 362
00CB 363
00CB 364
00CB 365
00CB 366
7E 7E 52 D0 00CB 367
53 50 C3 00CE 368
52 54 D0 00D2 369
53 54 D0 00D5 370
50 54 D5 00D8 371
03 13 00DA 372
FF2E 30 00DC 373
08 AE 53 D0 00DF 374
0000FFFF 8F 6E D1 00E3 375
17 15 00EA 376
00EC 377
00EC 378
63 FFFF 8F 04 AE 6E 00 2C 00EC 379
6E FFFF0001 8F 00010000 8F F1 00F5
FFE9 0101
63 50 51 50 8E 7D 0103 380
6E 00 2C 0106 381
51 8E D0 010C 382
05 010F 383
0110 384
0110 385
.END

CMPL R0, #65535 ; Is srclen greater than 65535?
BGTR BIG5 ; If so, can't do simple move.
CMPL R3, #65535 ; Is dstlen greater than 65535?
BGTR BIG5 ; If so, can't do simple move.
MOVC5 R0, (R1), R2, R3, (R4) ; Simple case; do the move
RSB

BIG5:
SUBL3 R0, R3, -(SP) ; Fill, exact or truncate?
BGTR NEED_FILL ; Move with fill
BEQL EXACT ; Same length source and dest?
MOVL R3, R0 ; Truncate - use dstlen
EXACT:
MOVL R4, R2 ; If exact, use source length
BSBW OTSS$MOVE3_R5 ; Get dest address
MOVL (SP)+, R0 ; Just move the characters
RSB ; Restore unmoved source bytes
; Return

;+
; We get here if we have to do some filling. Save the information we need
; to do the fill, use OTSS$MOVE3_R5 to do the data copy, then do the fill.
;-
NEED_FILL:
MOVL R2, -(SP) ; Save fill byte
SUBL3 R0, R3, -(SP) ; Length of fill
MOVL R4, R2 ; Get destination address
MOVL R4, R3 ; In case we're just filling
TSTL R0 ; Just fill?
BEQL 10$ ; If so, skip the move
BSBW OTSS$MOVE3_R5 ; Move <srclen> characters - R3 has next byt
MOVL R3, 8(SP) ; Save end-of-source pointer
CMPL (SP), #65535 ; More than 65535 characters of fill?
BLEQ 30$ ; If not, just fill the rest

20$:
MOVC5 #0, (SP), 4(SP), #65535, (R3) ; Fill 65535 bytes
ACBL #65536, #-65535, (SP), 20$ ; Loop until <65536 bytes left

30$:
MOVQ (SP)+, R0 ; Pop length and fill byte
MOVC5 #0, (SP), R1, R0, (R3) ; Fill the rest
MOVL (SP)+, R1 ; Restore end-of-source pointer
RSB ; Return

; End of module OTSS$MOVE
```


OTSSMOVE
Symbol table

- Move characters

K 16

16-SEP-1984 00:32:51
6-SEP-1984 11:15:07

VAX/VMS Macro V04-00
[LIBRTL.SRC]OTSMOVE.MAR;1

Page 10
(6)

BACKWARDS	00000029	R	01
BIG5	000000B6	R	01
BIGSTRING	0000001B	R	01
DEST	= 00000014		
EXACT	000000C1	R	01
FILL	= 0000000C		
FORWARDS	0000006B	R	01
LENGTH	= 00000004		
NEED FILL	000000CB	R	01
OTSSMOVE3	00000000	RG	01
OTSSMOVE3_R5	0000000D	RG	01
OTSSMOVE5	0000008C	RG	01
OTSSMOVE5_R5	0000009D	RG	01
SRCLN	= 00000004		

! Psect synopsis !

PSECT name

Allocation

PSECT No.

Attributes

ABS	00000000 (0.)	00 (0.)	NOPI	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE
_OTSSCODE	00000110 (272.)	01 (1.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	LONG

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.05	00:00:00.89
Command processing	114	00:00:00.33	00:00:02.22
Pass 1	76	00:00:00.51	00:00:03.07
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	76	00:00:00.47	00:00:01.34
Symbol table output	2	00:00:00.01	00:00:00.01
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	302	00:00:01.40	00:00:07.56

The working set limit was 750 pages.

5152 bytes (11 pages) of virtual memory were used to buffer the intermediate code.

There were 10 pages of symbol table space allocated to hold 16 non-local and 5 local symbols.

385 source lines were read in Pass 1, producing 14 object records in Pass 2.

0 pages of virtual memory were used to define 0 macros.

! Macro library statistics !

Macro library name

Macros defined

_\$255\$DUA28:[SYSLIB]STARLET.MLB;2

0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

OTSSMOVE
VAX-11 Macro Run Statistics

- Move characters

L 16

16-SEP-1984 00:32:51 VAX/VMS Macro V04-00
6-SEP-1984 11:15:07 [LIBRTL.SRC]OTSMOVE.MAR;1

Page 11
(6)

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS\$:OTSMOVE/OBJ=OBJ\$:OTSMOVE MSRC\$:OTSMOVE/UPDATE=(ENH\$:OTSMOVE)

0212 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

