

[illegible]

Sy

LI

LI  
LILI  
LILI  
LILI  
LILI  
LI

LI

LI

LI

LI  
LILI  
LILI  
LI

11

LI  
LI

LI

LI

21

100

LI  
LI

11

100

10

111

LI

LI

LI  
LI

11

10

1000

10

10

1

1

10

```
LL      IIIIII  BBBB8888  MM      MM      000000  VV      VV      TTTTTTTTTT  CCCCCCCC
LL      IIIIII  BBBB8888  MM      MM      000000  VV      VV      TTTTTTTTTT  CCCCCCCC
LL      II      BB      BB  MMMM  MMMM  00      00  VV      VV      TT      CC
LL      II      BB      BB  MMMM  MMMM  00      00  VV      VV      TT      CC
LL      II      BB      BB  MM      MM      00      00  VV      VV      TT      CC
LL      II      BBBB8888  MM      MM      00      00  VV      VV      TT      CC
LL      II      BBBB8888  MM      MM      00      00  VV      VV      TT      CC
LL      II      BB      BB  MM      MM      00      00  VV      VV      TT      CC
LL      II      BB      BB  MM      MM      00      00  VV      VV      TT      CC
LL      II      BB      BB  MM      MM      00      00  VV      VV      TT      CC
LL      II      BB      BB  MM      MM      00      00  VV      VV      TT      CC
LLLLLLLLLLLL IIIIII  BBBB8888  MM      MM      000000  VV      VV      TT      CC
LLLLLLLLLLLL IIIIII  BBBB8888  MM      MM      000000  VV      VV      TT      CC
```

```
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
```



(2) 80  
(3) 110

DECLARATIONS  
LIB\$MOVTC - Translate and Move Characters

```
0000 1      .TITLE  LIB$MOVTC - Move Translated Characters
0000 2      .IDENT  /1-012/      ; File: LIBMOVTC.MAR      Edit: RKR1012
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: General Utility Library
0000 31
0000 32 ABSTRACT:
0000 33
0000 34      Move translated characters from the source to the destination.
0000 35      The fill character is employed if the destination is longer
0000 36      than the source and the destination semantics are fixed-length.
0000 37
0000 38 ENVIRONMENT: User Mode, AST Reentrant
0000 39
0000 40
0000 41 AUTHOR: Donald G. Petersen, CREATION DATE: 30-Dec-77
0000 42
0000 43 MODIFIED BY:
0000 44
0000 45      DGP, 30-Dec-77 : Version 00
0000 46 01 - Original
0000 47 02 - bug fix. Add @ to fill character pointer. DGP 9-May-78
0000 48 1-001 - Update version number and copyright notice. JBS 16-NOV-78
0000 49 1-002 - Add "" to PSECT directive. JBS 21-DEC-78
0000 50 1-003 - If the destination string is dynamic, allocate enough space
0000 51          to hold the whole translated source string. JBS 20-MAR-1979
0000 52 1-004 - Change LIB$$ and OT$$ to STR$. JBS 21-MAY-1979
0000 53 1-005 - Put in .EXTRN assembler directives. JBS 19-JUN-1979
0000 54 1-006 - Use handler to change STR errors to LIB. RW 22-Jan-1980
0000 55 1-007 - Enhance to recognize additional classes of string descriptors
0000 56          by invoking LIB$ANALYZE_SDESC_R3 to extract length and address
0000 57          of 1st data byte from descriptor.
```



```
0000 58 : Use LIB$SGET1_DD instead of STR$GET1_DX to allocate space.
0000 59 : This should be faster and eliminate the need for establishing
0000 60 : LIB$STRTO_RET as a handler.
0000 61 : RKR 22-MAY-1981
0000 62 : 1-008 - Add checks for errors returned by LIB$ANALYZE_SDEC, so that
0000 63 : we returned error status to caller. RKR 19-AUG-1981.
0000 64 : 1-009 - Analyze source before allocating dest string. SBL 23-Sept-1981
0000 65 : 1-010 - Add special-case code to process string descriptors that
0000 66 : "read" like fixed string descriptors.
0000 67 : Fix error in calling sequence to LIB$SGET1_DD. Needs length
0000 68 : by reference.
0000 69 : RKR 8-OCT-1981.
0000 70 : 1-011 - Redirect jsb's from LIB$ANALYZE_SDESC_R3 to
0000 71 : LIB$ANALYZE_SDESC_R2. Reorganize register usage to free
0000 72 : R7 and drop R7 from entry mask.
0000 73 : RKR 18-NOV-1981.
0000 74 : 1-012 - Check for truncation and return LIB$STRTRU if it occurs.
0000 75 : This feature has been in the documentation, but not in the
0000 76 : code.
0000 77 : RKR 17-AUG-1982
0000 78 :--
```

```
0000 80      .SBTTL  DECLARATIONS
0000 81      :
0000 82      : INCLUDE FILES:
0000 83      :
0000 84      $DSCDEF      ; Descriptor symbols
0000 85      $SSDEF      ; SS$_NORMAL
0000 86
0000 87      : EXTERNAL SYMBOLS
0000 88      .DSABL  GBL      ; Explicit externals only
0000 89      .EXTRN  LIB$ANALYZE_SDESC_R2 ; Extract length and address of
0000 90
0000 91      .EXTRN  LIB$SGET1 DD      ; 1st data byte
0000 92      .EXTRN  LIB$_STRTRU      ; Allocate a string
0000 93
0000 94      : MACROS:
0000 95
0000 96      NONE
0000 97
0000 98      : EQUATED SYMBOLS:
0000 99
0000 100     NONE
0000 101
0000 102     : OWN STORAGE:
0000 103
0000 104     NONE
0000 105
0000 106     : PSECT DECLARATIONS:
0000 107
00000000 108     .PSECT _LIB$CODE PIC, SHR, LONG, EXE, NOWRT
```



```
0000 110 .SBTTL LIB$MOVTC - Translate and Move Characters
0000 111 :++
0000 112 : FUNCTIONAL DESCRIPTION:
0000 113 :
0000 114 : Each character in the source is used as an index into the table.
0000 115 : The byte entry found is then placed into the destination as a
0000 116 : translation. The fill character is used if the destination
0000 117 : string is longer than the source string and the destination
0000 118 : semantics are fixed-length. If the source is
0000 119 : longer than the destination, then truncation results.
0000 120 : Overlap of the source and destination strings does not affect
0000 121 : execution.
0000 122 : Overlap of the destination string and translation table yields
0000 123 : unpredictable results.
0000 124 : If the destination string is dynamic, enough space is allocated
0000 125 : for it to hold the entire translated source string.
0000 126 : Destination strings which have varying-length string semantics
0000 127 : will end up with a CURLEN field that is the MIN( MAXSTRLEN,
0000 128 : SOURCE string length).
0000 129 :
0000 130 : CALLING SEQUENCE:
0000 131 :
0000 132 : CALL LIB$MOVTC (src.rt.dx, fill.rt.dx, table.rt.dx, dst.wt.dx)
0000 133 :
0000 134 :
0000 135 : INPUT PARAMETERS:
0000 136 :
00000004 0000 137 : SOURCE = 4 ; Adr. of desc. of source string
00000008 0000 138 : FILL = 8 ; Adr. of desc. of fill character
0000000C 0000 139 : TABLE = 12 ; Adr. of desc. of translation table
0000 140 :
0000 141 : IMPLICIT INPUTS:
0000 142 :
0000 143 : NONE
0000 144 :
0000 145 : OUTPUT PARAMETERS:
0000 146 :
00000010 0000 147 : DEST = 16 ; Adr. of desc. of destination string
0000 148 :
0000 149 : IMPLICIT OUTPUTS:
0000 150 :
0000 151 : NONE
0000 152 :
0000 153 : COMPLETION CODES:
0000 154 :
0000 155 : SSS NORMAL The MOVTC went OK
0000 156 : LIB$_FATERRLIB Fatal error in library
0000 157 : LIB$_INSVIRMEM Insufficient virtual memory to allocate
0000 158 : the destination string.
0000 159 : LIB$_INVSTRDES Invalid string descriptor
0000 160 : LIB$_STRTRU The destination string could not contain
0000 161 : all of the characters.
0000 162 :
0000 163 : SIDE EFFECTS:
0000 164 :
0000 165 : NONE
0000 166 :
```

```
0000 167 ;--
0000 168
007C 0000 169 .ENTRY LIB$MOVTC , ^M<R2, R3, R4, R5, R6> ; Entry point
0002 170
0002 171 ;+
0002 172 ; Get the length and address of the source string.
0002 173 ;--
50 04 AC D0 0002 174 MOVL SOURCE(AP), R0 ; Address of SOURCE descriptor
02 03 A0 91 0006 175 CMPB DSC$B_CLASS(R0), #DSC$K_CLASS_D ; read like fixed ?
06 1A 000A 176 BGTRU 5$ ; no
55 04 BC 7D 000C 177 MOVQ @SOURCE(AP), R5 ; length->R5, address->R6
0D 11 0010 178 BRB 15$ ; join common flow
0012 179
00000000'GF 16 0012 180 5$: JSB G^LIB$ANALYZE_SDESC_R2 ; Extract: length->R1, addr->R2
01 50 E8 0018 181 BLBS R0, 10$ ; If ok, continue
04 001B 182 RET ; else, error -- return
001C 183
55 51 7D 001C 184 10$: MOVQ R1, R5 ; save length and addr of SOURCE
001F 185 ; length -> R5, addr-> R6
001F 186
001F 187 ;+
001F 188 ; If the destination string is dynamic, allocate enough space for it
001F 189 ; that it will hold the whole translated source string.
001F 190 ;--
50 10 AC D0 001F 191 15$: MOVL DEST(AP), R0 ; Point to dest descr.
03 A0 02 91 0023 192 CMPB #DSC$K_CLASS_D, DSC$B_CLASS(R0) ; Dynamic?
15 12 0027 193 BNEQ 20$ ; No, use it as is.
55 DD 0029 194 PUSHL R5 ; length
50 DD 002B 195 PUSHL R0 ; address
04 AE 3F 002D 196 PUSHL 4(SP) ; address of length
00000000'GF 02 FB 0030 197 CALLS #2, G^LIB$SGET1_DD
5E 04 C0 0037 198 ADDL2 #4, SP ; realign stack
01 50 E8 003A 199 BLBS R0, 20$
04 003D 200 RET ; If alloc. fails
003E 201 ;+
003E 202 ; Extract the various lengths and addresses we will need and leave
003E 203 ; in registers for the actual MOVTC instruction to follow.
003E 204 ;--
003E 205 20$:
50 0C AC D0 003E 206 MOVL TABLE(AP), R0 ; Address of TABLE descriptor
02 03 A0 91 0042 207 CMPB DSC$B_CLASS(R0), #DSC$K_CLASS_D ; read like fixed ?
06 1A 0046 208 BGTRU 25$ ; no
53 04 A0 D0 0048 209 MOVL DSC$A_POINTER(R0), R3 ; address of TABLE
0C 11 004C 210 BRB 30$ ; join common flow
004E 211
00000000'GF 16 004E 212 25$: JSB G^LIB$ANALYZE_SDESC_R2 ; Extract: length->R1, addr->R2
69 50 E9 0054 213 BLBC R0, 70$ ; If error, return error
53 52 D0 0057 214 MOVL R2, R3 ; save addr of TABLE
005A 215
50 08 AC D0 005A 216 30$: MOVL FILL(AP), R0 ; Address of FILL descriptor
02 03 A0 91 005E 217 CMPB DSC$B_CLASS(R0), #DSC$K_CLASS_D ; read like fixed ?
06 1A 0062 218 BGTRU 35$ ; no
54 04 B0 9A 0064 219 MOVZBL @DSC$A_POINTER(R0), R4 ; value of fill character
0C 11 0068 220 BRB 40$ ; join common flow
006A 221
00000000'GF 16 006A 222 35$: JSB G^LIB$ANALYZE_SDESC_R2 ; Extract: length->R1, addr->R2
4D 50 E9 0070 223 BLBC R0, 70$ ; If error, return error
```



```
54 62 9A 0073 224 MOVZBL (R2), R4 ; value of fill character
0076 225
50 10 AC D0 0076 226 40$: MOVL DEST(AP), R0 ; Address of DEST descriptor
02 03 A0 91 007A 227 CMPB DSC$B_CLASS(R0), #DSC$K_CLASS_D ; read like fixed ?
06 1A 007E 228 BGTRU 45$ ; no
51 10 BC 7D 0080 229 MOVQ @DEST(AP), R1 ; length->R1, address->R2
09 11 0084 230 BRB 50$ ; join common flow
0086 231
00000000'GF 16 0086 232 45$: JSB G^LIB$ANALYZE_SDESC_R2 ; Extract: length->R1, addr->R2
31 50 E9 008C 233 BLBC R0, 70$ ; If error, return error
008F 234
008F 235 ;+
008F 236 ; Class_VS destination is handled as a special case. We must try to
008F 237 ; force its current length (CURLen) field to be the same size as the
008F 238 ; source length. However, this new length must be the MIN( source_len,
008F 239 ; and MAXSTRLEN). If source len is greater than MAXSTRLEN, then output
008F 240 ; will be truncated to MAXSTRLEN chars.
008F 241 ;+
50 10 AC D0 008F 242 50$: MOVL DEST(AP), R0 ; Address of DEST descriptor
0B 03 A0 91 0093 243 CMPB DSC$B_CLASS(R0), #DSC$K_CLASS_VS ; Class_VS ?
11 12 0097 244 BNEQ 65$ ; no, no special action needed
60 55 B1 0099 245 CMPW R5, DSC$W_MAXSTRLEN(R0) ; SOURCE len : MAXSTRLEN
05 15 009C 246 BLEQ 55$ ; if SOURCE len leq
51 60 3C 009E 247 MOVZWL DSC$W_MAXSTRLEN(R0), R1 ; use MAXSTRLEN for CURLen
03 11 00A1 248 BRB 60$
51 55 3C 00A3 249 55$: MOVZWL R5, R1 ; use SOURCE len for CURLen
04 B0 51 B0 00A6 250 60$: MOVW R1, @DSC$A_POINTER(R0) ; rewrite CURLen field
00AA 251 ;+
00AA 252 ; prototype movtc
00AA 253 ; movtc srclen, srcaddr, fillchar, tableaddr, dstlen, dstaddr
00AA 254 ;+
62 51 63 54 66 55 2E 00AA 255 65$: MOVTC R5, (R6), R4, (R3), R1, (R2) ; move translated chars
00B1 256 ; State of regs after a MOVTC instr.
00B1 257 ; R0 = number of translated bytes
00B1 258 ; remaining in source string.
00B1 259 ; Is non zero only if source string
00B1 260 ; is longer than destination
00B1 261 ; string.
00B1 262 ; R1 = address of one byte beyond the
00B1 263 ; last byte in source string that
00B1 264 ; was translated.
00B1 265 ; R2 = 0
00B1 266 ; R3 = address of the translation table
00B1 267 ; R4 = 0
00B1 268 ; R5 = address of one byte beyond the
00B1 269 ; destination string.
00B1 270
00B1 271
50 50 D5 00B1 272 TSTL R0 ; Were all moved ?
08 13 00B3 273 BEQL 68$ ; yes
50 00000000'8F D0 00B5 274 MOVL #LIB$_STRTRU, R0 ; Set up truncation status
04 00BC 275 RET
00BD 276
50 01 D0 00BD 277 68$: MOVL #SS$_NORMAL, R0 ; Indicate normal completion
04 00C0 278 70$: RET ; Return to caller
00C1 279 .END
```

LIB\$MOVTC  
Symbol table

- Move Translated Characters

L 11

16-SEP-1984 00:14:05  
6-SEP-1984 11:09:18

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

Page 7  
(3)

|                       |          |          |    |    |
|-----------------------|----------|----------|----|----|
| DEST                  | =        | 00000010 |    |    |
| DSC\$A_POINTER        | =        | 00000004 |    |    |
| DSC\$B_CLASS          | =        | 00000003 |    |    |
| DSC\$K_CLASS_D        | =        | 00000002 |    |    |
| DSC\$K_CLASS_VS       | =        | 0000000B |    |    |
| DSC\$W_MAXSTRLEN      | =        | 00000000 |    |    |
| FILL                  | =        | 00000008 |    |    |
| LIB\$ANALYZE_SDESC_R2 | *****    |          | X  | 00 |
| LIB\$MOVTC            | 00000000 |          | RG | 02 |
| LIB\$SGET1_DD         | *****    |          | X  | 00 |
| LIB\$STRTRU           | *****    |          | X  | 00 |
| SOURCE                | =        | 00000004 |    |    |
| SS\$ NORMAL           | =        | 00000001 |    |    |
| TABCE                 | =        | 0000000C |    |    |

-----  
! Psect synopsis !  
-----

| PSECT name | Allocation       | PSECT No. | Attributes |     |     |     |     |       |       |      |       |       |      |  |  |  |  |  |
|------------|------------------|-----------|------------|-----|-----|-----|-----|-------|-------|------|-------|-------|------|--|--|--|--|--|
| . ABS .    | 00000000 ( 0.)   | 00 ( 0.)  | NOPIC      | USR | CON | ABS | LCL | NOSHR | NOEXE | NORD | NOWRT | NOVEC | BYTE |  |  |  |  |  |
| \$ABSS     | 00000000 ( 0.)   | 01 ( 1.)  | NOPIC      | USR | CON | ABS | LCL | NOSHR | EXE   | RD   | WRT   | NOVEC | BYTE |  |  |  |  |  |
| _LIB\$CODE | 000000C1 ( 193.) | 02 ( 2.)  | PIC        | USR | CON | REL | LCL | SHR   | EXE   | RD   | NOWRT | NOVEC | LONG |  |  |  |  |  |

-----  
! Performance indicators !  
-----

| Phase                  | Page faults | CPU Time    | Elapsed Time |
|------------------------|-------------|-------------|--------------|
| Initialization         | 29          | 00:00:00.04 | 00:00:00.62  |
| Command processing     | 102         | 00:00:00.33 | 00:00:02.43  |
| Pass 1                 | 214         | 00:00:03.41 | 00:00:11.80  |
| Symbol table sort      | 0           | 00:00:00.58 | 00:00:01.75  |
| Pass 2                 | 64          | 00:00:00.76 | 00:00:04.21  |
| Symbol table output    | 3           | 00:00:00.03 | 00:00:00.03  |
| Psect synopsis output  | 2           | 00:00:00.01 | 00:00:00.01  |
| Cross-reference output | 0           | 00:00:00.00 | 00:00:00.00  |
| Assembler run totals   | 416         | 00:00:05.17 | 00:00:20.85  |

The working set limit was 1200 pages.  
28230 bytes (56 pages) of virtual memory were used to buffer the intermediate code.  
There were 30 pages of symbol table space allocated to hold 546 non-local and 15 local symbols.  
279 source lines were read in Pass 1, producing 13 object records in Pass 2.  
9 pages of virtual memory were used to define 8 macros.

-----  
! Macro library statistics !  
-----

| Macro library name                  | Macros defined |
|-------------------------------------|----------------|
| _\$255\$DUA28:[SYSLIB]STARLET.MLB;2 | 5              |

604 GETS were required to define 5 macros.



LIB\$MOVTC  
VAX-11 Macro Run Statistics

- Move Translated Characters

M 11

16-SEP-1984 00:14:05 VAX/VMS Macro V04-00  
6-SEP-1984 11:09:18 [LIBRTL.SRC]LIBMOVTC.MAR;1

Page 8  
(3)

There were no errors, warnings or information messages.

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



0208 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

