```
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBaB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**LIBTRAA2E

```
LL              IIIIII    BBBBBBBB    TTTTTTTTTT  RRRRRRRR      AAAAAA      AAAAAA      222222    EEEEEEEEEE
LL              IIIIII    BBBBBBBB    TTTTTTTTTT  RRRRRRRR      AAAAAA      AAAAAA      222222    EEEEEEEEEE
LL                II      BB      BB      TT      RR      RR  AA      AA  AA      AA  22      22  EE
LL                II      BB      BB      TT      RR      RR  AA      AA  AA      AA  22      22  EE
LL                II      BB      BB      TT      RR      RR  AA      AA  AA      AA          22  EE
LL                II      BB      BB      TT      RR      RR  AA      AA  AA      AA          22  EE
LL                II      BBBBBBBB        TT      RRRRRRRR    AA      AA  AA      AA        22    EEEEEEEE
LL                II      BBBBBBBB        TT      RRRRRRRR    AA      AA  AA      AA        22    EEEEEEEE
LL                II      BB      BB      TT      RR  RR      AAAAAAAAAA  AAAAAAAAAA      22      EE
LL                II      BB      BB      TT      RR  RR      AAAAAAAAAA  AAAAAAAAAA      22      EE
LL                II      BB      BB      TT      RR    RR    AA      AA  AA      AA    22        EE                ....
LL                II      BB      BB      TT      RR    RR    AA      AA  AA      AA    22        EE                ....
LLLLLLLLLL      IIIIII    BBBBBBBB        TT      RR      RR  AA      AA  AA      AA  2222222222  EEEEEEEEEE        ....
LLLLLLLLLL      IIIIII    BBBBBBBB        TT      RR      RR  AA      AA  AA      AA  2222222222  EEEEEEEEEE        ....

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
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
0000     1         .TITLE  LIB$TRA_ASC_EBC - Translate ASCII string to EBCDIC string
0000     2         .IDENT  /1-007/         ; File: LIBTRAA2E.MAR Edit: RKR1007
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 ;       Translates an ASCII string to an EBCDIC string
0000    35 ;
0000    36 ; ENVIRONMENT: User Mode, AST Reentrant
0000    37 ;
0000    38 ;--
0000    39 ; AUTHOR: R. Reichert,  CREATION DATE: 03-DEC-1979
0000    40 ;
0000    41 ; MODIFIED BY:
0000    42 ;
0000    43 ; REVISION HISTORY:
0000    44 ;
0000    45 ; 1-001 - Original. RKR 03-DEC-79
0000    46 ; 1-002 - Change EBCDIC string descriptor doucmentation to "bu.dx"
0000    47 ;         RKR 05-DEC-79
0000    48 ; 1-003 - Return LIB$ codes instead of STR$.  JBS 22-JAN-1980
0000    49 ; 1-004 - Refetch destination string length and address after calling
0000    50 ;         STR$GET1_DX.  SPR 11-38343  SBL 5-June-1981
0000    51 ; 1-005 - Enhance to recognize additional classes of string descriptors
0000    52 ;         by invoking LIB$ANALYZE_SDESC_R3 to extract length and address
0000    53 ;         of 1st data byte.
0000    54 ;         Redirect call to STR$GET1_DX to LIB$SGET1_DD and remove use of
0000    55 ;         handler.
0000    56 ;         RKR 26-MAY-1981
0000    57 ; 1-006 - Add special-case code to process string descriptors that
```

```
0000     58 ;                "read" like fixed string descriptors.  RKR 7-OCT-1981.
0000     59 ; 1-007 - Redirect jsb's from LIB$ANALYZE_SDESC_R3 to
0000     60 ;         LIB$ANALYZE_SDESC_R2.  RKR 18-NOV-1981.
```

```
                        62          .SBTTL  DECLARATIONS
                        63 ;
                        64 ; INCLUDE FILES: NONE
                        65 ;
                        66
                        67 ;
                        68 ; EXTERNAL DECLARATIONS:
                        69 ;
                        70          .DSABL  GBL                             ; Prevent undeclared
                        71                                                  ; symbols from being
                        72                                                  ; automatically global.
                        73
                        74          .EXTRN  LIB$SGET1_DD                    ; dynamic string allocator
                        75
                        76          .EXTRN  LIB$AB_ASC_EBC                  ; ASCII to EBCDIC translation
                        77                                                  ; table
                        78
                        79          .EXTRN  LIB$ANALYZE_SDESC_R2            ; Extract lenght and address of
                        80                                                  ; 1st data byte from descriptor
                        81
                        82          .EXTRN  LIB$_INVCHA                     ; Return code "invalid
                        83                                                  ; character"
                        84
                        85          .EXTRN  LIB$_INVARG                     ; Return code "invalid
                        86                                                  ; argument"
                        87
                        88 ;
                        89 ; MACROS:
                        90 ;
                        91
                        92          $SSDEF                                  ; Definition of SS$_ symbols
                        93          $DSCDEF                                 ; Descriptor components
                        94 ;
                        95 ; EQUATED SYMBOLS:
                        96 ;
0000003F  0000          97 ESC_CHAR         = ^X3F                  ; EBCDIC Character "SUB"
0000001A  0000          98 SOURCE_CODE      = ^X1A                  ; ASCII Character "SUB"
                        99
                       100
                       101 ;
                       102 ; PSECT DECLARATIONS:
                       103 ;
00000000               104          .PSECT _LIB$CODE PIC, USR, CON, REL, LCL, SHR, -
                       105                         EXE, RD, NOWRT, LONG
                       106
```

```
0000    108          .SBTTL  LIB$TRA_ASC_EBC - Translate ASCII to EBCDIC
0000    109 ;++
0000    110 ; FUNCTIONAL DESCRIPTION:
0000    111 ;
0000    112 ;       Translates an ASCII string to a EBCDIC string.
0000    113 ;       If destination string has fixed string semantics, its length
0000    114 ;       must match the length of the input string (no filling is done).
0000    115 ;       If destination string is varying, its length is forced (if
0000    116 ;       possible) to be the length of the source string.  If it can't
0000    117 ;       (MAXSTRLEN too small) LIB$_INVARG is returned.
0000    118 ;
0000    119 ; CALLING SEQUENCE:
0000    120 ;
0000    121 ;       status.wlc.v = LIB$TRA_ASC_EBC ( SRC_STR.rt.dx, DST_STR.wbu.dx)
0000    122 ;
0000    123 ; INPUT PARAMETERS:
0000    124 ;
0000    125 ;       SRC_STR.rt.dx   address of source string descriptor (ASCII)
0000    126 ;
0000    127 ;       DST_STR.wbu.dx  address of destination string descriptor (EBCDIC)
0000    128 ;
0000    129 ;
0000    130 ; IMPLICIT INPUTS:
0000    131 ;
0000    132 ;       The ASCII to EBCDIC translation table at LIB$AB_ASC_ESC
0000    133 ;
0000    134 ; OUTPUT PARAMETERS:
0000    135 ;
0000    136 ;       NONE
0000    137 ;
0000    138 ; IMPLICIT OUTPUTS:
0000    139 ;
0000    140 ;       NONE
0000    141 ;
0000    142 ; COMPLETION CODES:
0000    143 ;
0000    144 ;       SS$_NORMAL      - Routine successfully completed.
0000    145 ;
0000    146 ;       LIB$_INVCHA     - One or more occurences of an untranslatable
0000    147 ;                         character has been detected in the course
0000    148 ;                         of translation.
0000    149 ;
0000    150 ;       LIB$_INVARG     - If destination string is fixed string and
0000    151 ;                         its length is not the same as the source
0000    152 ;                         length.
0000    153 ;
0000    154 ; SIDE EFFECTS:
0000    155 ;
0000    156 ;       NONE
0000    157 ;
0000    158 ;--
0000    159
```

```
                                                H 14
                     0000   161 ; Displacements off AP of input arguments:
                     0000   162
           00000004  0000   163 SRC_STR =       4
           00000008  0000   164 DST_STR =       8
                     0000   165
                     0000   166
               40FC  0000   167         .ENTRY LIB$TRA_ASC_EBC, ^M<IV, R2, R3, R4, R5, R6, R7>
                     0002   168 ;+
                     0002   169 ; Extract the lengths and addresses we will need from the source and
                     0002   170 ; destination descriptors.
                     0002   171 ;-
 50   04 AC    D0   0002   172         MOVL    SRC_STR(AP), R0         ; Address of src descriptor
 02   03 A0    91   0006   173         CMPB    DSC$B_CLASS(R0), #DSC$K_CLASS_D ; read like fixed ?
        06    1A   000A   174         BGTRU   1$                      ; no
 56   04 BC    7D   000C   175         MOVQ    @SRC_STR(AP), R6        ; length->R6, address->R7
        09    11   0010   176         BRB     2$                      ; join common flow
                     0012   177
00000000'GF    16   0012   178 1$:     JSB     G^LIB$ANALYZE_SDESC_R2  ; Extract: length->R1, addr->R2
    56   51    7D   0018   179         MOVQ    R1, R6                  ; length->R6, address->R7
                     001B   180
 50   08 AC    D0   001B   181 2$:     MOVL    DST_STR(AP), R0 ; Address of dst descriptor
 02   03 A0    91   001F   182         CMPB    DSC$B_CLASS(R0), #DSC$K_CLASS_D ; read like fixed ?
        06    1A   0023   183         BGTRU   3$                      ; no
 54   08 BC    7D   0025   184         MOVQ    @DST_STR(AP), R4        ; length->R4, address->R5
        09    11   0029   185         BRB     4$                      ; join common flow
                     002B   186
00000000'GF    16   002B   187 3$:     JSB     G^LIB$ANALYZE_SDESC_R2  ; Extract: length->R1, addr->R2
    54   51    7D   0031   188         MOVQ    R1, R4                  ; length->R4, address->R5
                     0034   189
                     0034   190 ;+
                     0034   191 ; If destination is a dynamic string, reallocate a dynamic string which
                     0034   192 ; is the same length as the input string.
                     0034   193 ;-
                     0034   194
 52   08 AC    D0   0034   195 4$:     MOVL    DST_STR(AP), R2         ; address of dst descriptor
 02   03 A2    91   0038   196         CMPB    DSC$B_CLASS(R2)  #DSC$K_CLASS_D ; dynamic string dest ?
        1C    12   003C   197         BNEQ    5$                      ; not dynamic
    7E   56    3C   003E   198         MOVZWL  R6, -(SP)               ; length of source
      08 AC    DD   0041   199         PUSHL   DST_STR(AP)             ; caller's dest desc addr
      04 AE    DF   0044   200         PUSHAL  4(SP)                   ; address of length of source
00000000'GF    02   FB   0047   201         CALLS   #2, G^LIB$SGET1_DD      ; allocate a dynamic string
                     004E   202                                         ; equal in len to source
    5E   04    C0   004E   203         ADDL2   #4, SP                  ; restore stack
    54   56    3C   0051   204         MOVZWL  R6, R4                  ; dest length = source length
 55   04 A2    D0   0054   205         MOVL    DSC$A_POINTER(R2), R5   ; new destination address
        21    11   0058   206         BRB     10$                     ; join common flow
                     005A   207 ;+
                     005A   208 ; If destination is a varying string, force its CURLEN to the MIN (
                     005A   209 ; MAXSTRLEN, length of source).  If we can't, return LIB$_INVARG.
                     005A   210 ;-
                     005A   211 5$:
 0B   03 A2    91   005A   212         CMPB    DSC$B_CLASS(R2), #DSC$K_CLASS_VS ; varying string dest ?
        0E    12   005E   213         BNEQ    6$                      ; not varying
    56   62    B1   0060   214         CMPW    DSC$W_MAXSTRLEN(R2), R6 ; MAXSTRLEN ? length of source
        0E    1F   0063   215         BLSSU   9$                      ; won't fit, exit with error
    54   56    3C   0065   216         MOVZWL  R6, R4                  ; Set dest len to source length
 04 B2    54    B0   0068   217         MOVW    R4, @DSC$A_POINTER(R2)  ; Set CURLEN to new length
```

```
                      0D   11  006C    218          BRB     10$                     ; join common flow
                                  006E    219 ;+
                                  006E    220 ; Otherwise, we have a destination string with fixed-length semantics.
                                  006E    221 ; Its length must match the length of the input string,
                                  006E    222 ; else LIB$_INVARG is returned.
                                  006E    223 ;-
                                  006E    224 6$:
                    54    56    B1  006E    225          CMPW    R6, R4                  ; source len = dest len ?
                          08    13  0071    226          BEQL    10$                     ; ok if lengths match
                50  00000000'8F    D0  0073    227 9$:      MOVL    #LIB$_INVARG, R0        ; return "invalid argument'
                                  04  007A    228          RET
                                  007B    229
                                  007B    230 ;+
                                  007B    231 ; Set up registers for the MOVTUC and MOVTC to follow
                                  007B    232 ;-
                                  007B    233 10$:
                    50    56    B0  007B    234          MOVW    R6, R0                  ; get source length in R0
                    51    57    D0  007E    235          MOVL    R7, R1                  ; address of caller's source
                                  0081    236 ;+
                                  0081    237 ; registers at this point...
                                  0081    238 ;
                                  0081    239 ;       R0      length of source string
                                  0081    240 ;       R1      address of source string
                                  0081    241 ;       R2      address of destination descriptor
                                  0081    242 ;       R3      unknown
                                  0081    243 ;       R4      length of destination string (must be equal to R6)
                                  0081    244 ;       R5      address of destination string
                                  0081    245 ;       R6      length of source string (must be equal to R4)
                                  0081    246 ;       R7      address of source string (same as R1)
                                  0081    247 ;-
```

```
                          0081   249 ;+
                          0081   250 ; Actual translation loop.
                          0081   251 ; Repeated until we translate all error free, or fall through with
                          0081   252 ; an error and perform one final MOVTC to complete translation
                          0081   253 ;-
                          0081   254
                          0081   255 15$:
00000000'GF   3F   61  50  2F 0081   256         MOVTUC  R0, (R1), #ESC_CHAR, G^LIB$AB_ASC_EBC, R4, (R5)
                     65  54    008A
                          008C   257                                 ; State of regs after a MOVTUC instr.
                          008C   258                                 ; R0 = number of bytes remaining in
                          008C   259                                 ;      source string (including the
                          008C   260                                 ;      byte which caused the escape.
                          008C   261                                 ;      Is zero only if the entire source
                          008C   262                                 ;      string was translated and moved
                          008C   263                                 ;      without escape.
                          008C   264                                 ; R1 = address of the byte which
                          008C   265                                 ;      resulted in destination string
                          008C   266                                 ;      exhaustion or escape.  If no
                          008C   267                                 ;      exhaustion or escape, then
                          008C   268                                 ;      address of one byte beyond the
                          008C   269                                 ;      source string.
                          008C   270                                 ; R2 = 0
                          008C   271                                 ; R3 = address of the table
                          008C   272                                 ; R4 = number of bytes remaining in the
                          008C   273                                 ;      destinatin string.
                          008C   274                                 ; R5 = address of the byte in the
                          008C   275                                 ;      destination string which would
                          008C   276                                 ;      have received the translated byte
                          008C   277                                 ;      that caused the escape or would
                          008C   278                                 ;      have received a translated byte
                          008C   279                                 ;      if the source string were not
                          008C   280                                 ;      exhausted.  If not exhaustion
                          008C   281                                 ;      or escape, then address of one
                          008C   282                                 ;      byte beyond the destination
                          008C   283                                 ;      string.
                     1F   1C  008C   284         BVC     GOOD_COMPL              ; terminated by end of string
                          008E   285
                85   3F   90  008E   286         MOVB    #ESC_CHAR, (R5)+        ; store esc_char in output
                          0091   287                                 ; bumping R5 to next byte pos.
                     50   D7  0091   288         DECL    R0                      ; adjust input count for one
                          0093   289                                 ; done by hand
                     54   D7  0093   290         DECL    R4                      ; adjust output count for one
                          0095   291                                 ; done by hand
                1A   81   91  0095   292         CMPB    (R1)+, #SOURCE_CODE     ; was the input an escape char
                     E7   13  0098   293         BEQL    15$                     ; yes -- treat as success and
                          009A   294                                 ; and continue
                          009A   295
                          009A   296 ;+
                          009A   297 ; an untranslatable input char has been detected.  Translate rest of
                          009A   298 ; string and exit with an error status of LIB$_INVCHA
                          009A   299 ;-
                          009A   300
00000000'GF   6E   61  50  2E 009A   301         MOVTC   R0, (R1), 0(SP), G^LIB$AB_ASC_EBC, R4, (R5)
                     65  54    00A3
                          00A5   302                                 ; NOTE: lengths are guaranteed
                          00A5   303                                 ; to be equal, so fill_char is
```

```
                     00A5   304                                    ; not needed.
                     00A5   305                                ; State of regs after a MOVTC instr.
                     00A5   306                                ; R0 = number of translated bytes
                     00A5   307                                ;      remaining in source string.
                     00A5   308                                ;      Is non zero only if source string
                     00A5   309                                ;      is longer than destination
                     00A5   310                                ;      string.
                     00A5   311                                ; R1 = address of one byte beyond the
                     00A5   312                                ;      last byte in source string that
                     00A5   313                                ;      was translated.
                     00A5   314                                ; R2 = 0
                     00A5   315                                ; R3 = address of the translation table
                     00A5   316                                ; R4 = 0
                     00A5   317                                ; R5 = address of one byte beyond the
                     00A5   318                                ;      destination string.
                     00A5   319                                ;
     50  00000000'8F 00  00A5   320          MOVL    #LIB$_INVCHA, R0        ; return "invalid char" code
                 04  00AC   321          RET
                     00AD   322
                     00AD   323 GOOD_COMPL:
         50  01      D0  00AD   324          MOVL    #SS$_NORMAL, R0         ; return success
                 04  00B0   325          RET
                     00B1   326
                     00B1   327          .END
```

```
DSC$A_POINTER                   = 00000004
DSC$B_CLASS                     = 00000003
DSC$K_CLASS_D                   = 00000002
DSC$K_CLASS_VS                  = 0000000B
DSC$W_MAXSTRLEN                 = 00000000
DST_STR                         = 00000008
ESC_CHAR                        = 0000003F
GOOD_COMPL                        000000AD R      02
LIB$AB_ASC_EBC                    ********   X    00
LIB$ANALYZE_SDESC_R2              ********   X    00
LIB$SGET1_DD                      ********   X    00
LIB$TRA_ASC_EBC                   00000000 RG     02
LIB$_INVARG                       ********   X    00
LIB$_INVCHA                       ********   X    00
SOURCE_CODE                     = 0000001A
SRC_STR                         = 00000004
SS$_NORMAL                      = 00000001
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_LIB$CODE	0C00J0B1 (177.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.03	00:00:01.72
Command processing	105	00:00:00.30	00:00:01.72
Pass 1	218	00:00:03.45	00:00:14.65
Symbol table sort	0	00:00:00.51	00:00:03.59
Pass 2	70	00:00:00.82	00:00:03.02
Symbol table output	4	00:00:00.02	00:00:00.02
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	430	00:00:05.15	00:00:24.74

The working set limit was 1200 pages.
28450 bytes (56 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 549 non-local and 9 local symbols.
327 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.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:LIBTRAA2E/OBJ=OBJ$:LIBTRAA2E MSRC$:LIBTRAA2E/UPDATE=(ENH$:LIBTRAA2E)
```

LIBSPAWN
LIS

LIBSTATVM
LIS

LIBTRAA2E
LIS

LIBSPANC
LIS

LIBSYMBOL
LIS

LIBTRNLOG
LIS

LIBSKPC
LIS

LIBTIMER
LIS

LIBTPARSE
LIS

LIBTRIMFI
LIS

LIBSTRRET
LIS

LIBTRAE2A
LIS