```
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     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**OTSCVTTOL

```
  000000    TTTTTTTTTT    SSSSSSSS    CCCCCCCC  VV      VV  TTTTTTTTTT  TTTTTTTTTT    000000    LL
  000000    TTTTTTTTTT    SSSSSSSS    CCCCCCCC  VV      VV  TTTTTTTTTT  TTTTTTTTTT    000000    LL
00      00      TT      SS          CC          VV      VV      TT          TT      00      00  LL
00      00      TT      SS          CC          VV      VV      TT          TT      00      00  LL
00      00      TT      SS          CC          VV      VV      TT          TT      00      00  LL
00      00      TT      SS          CC          VV      VV      TT          TT      00      00  LL
00      00      TT        SSSSSS    CC          VV      VV      TT          TT      00      00  LL
00      00      TT        SSSSSS    CC          VV      VV      TT          TT      00      00  LL
00      00      TT              SS  CC          VV      VV      TT          TT      00      00  LL
00      00      TT              SS  CC          VV      VV      TT          TT      00      00  LL
00      00      TT              SS  CC            VV  VV        TT          TT      00      00  LL                ....
00      00      TT              SS  CC            VV  VV        TT          TT      00      00  LL                ....
  000000        TT      SSSSSSSS      CCCCCCCC      VV          TT          TT        000000    LLLLLLLLLL        ....
  000000        TT      SSSSSSSS      CCCCCCCC      VV          TT          TT        000000    LLLLLLLLLL        ....

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  OTS$CVTTOL      ; Convert text to integer
0000     2          .IDENT  /1-004/         ; File: OTSCVTTOL.MAR  Edit: SBL1004
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 ; FACILITY:  Language independent support library
0000    30 ;++
0000    31 ; ABSTRACT:
0000    32 ;
0000    33 ; This module contains routines to convert text representations of
0000    34 ; integers to longword (or other) values.  The text can be in
0000    35 ; hexadecimal, octal or binary bases.
0000    36 ;
0000    37 ;--
0000    38 ;
0000    39 ; VERSION 1
0000    40 ;
0000    41 ; HISTORY:
0000    42 ;
0000    43 ; AUTHOR:
0000    44 ;       Steven B. Lionel, 21-Feb-1979: Version 1
0000    45 ;
0000    46 ; MODIFIED BY:
0000    47 ;
```

```
                     0000     49          .SBTTL  HISTORY
                     0000     50
                     0000     51
                     0000     52 ; Edit History for OTS$CVT_TO_L
                     0000     53 ;
                     0000     54 ; 1-001 - Original.  Complete rewrite replacing FOR$CNV_IN_O and
                     0000     55 ;         FOR$CNV_IN_Z.  SBL 21-Feb-1979
                     0000     56 ; 1-002 - Add OTS$CVT_TB_L.  SBL 6-Nov-1980
                     0000     57 ; 1-003 - Fix undetected overflow bug.  SBL 4-Jan-1982
                     0000     58 ; 1-004 - Give error if character greater than 127.  SPR 11-52485
                     0000     59 ;         SBL 29-Dec-1982
```

```
                                        .SBTTL  DECLARATIONS

                                ;
                                ; INCLUDE FILES: None
                                ;

                                ;
                                ; EXTERNAL SYMBOLS:
                                ;
                                        .DSABL  GBL                     ; Prevent undefineds from being
                                                                        ; global.
                                        .EXTRN  OTS$_INPCONERR          ; Input conversion error

                                ;
                                ; MACROS: None
                                ;

                                ;
                                ; PSECT DECLARATIONS:
                                ;
                                        .PSECT  _OTS$CODE       PIC,SHR,LONG,EXE,NOWRT

                                ;
                                ; EQUATED SYMBOLS:
                                ;
                                        REGMASK = ^M<IV,R2,R3,R4,R5,R6> ; Register save mask

                                ;
                                ; OWN STORAGE:
                                ;

                                TABLE:                                  ; Converts characters to value
00                              .BYTE   0                               ; "0"
01                              .BYTE   1                               ; "1"
02                              .BYTE   2                               ; "2"
03                              .BYTE   3                               ; "3"
04                              .BYTE   4                               ; "4"
05                              .BYTE   5                               ; "5"
06                              .BYTE   6                               ; "6"
07                              .BYTE   7                               ; "7"
08                              .BYTE   8                               ; "8"
09                              .BYTE   9                               ; "9"
FF                              .BYTE   -1                              ; invalid character
FF                              .BYTE   -1                              ; invalid character
FF                              .BYTE   -1                              ; invalid character
FF                              .BYTE   -1                              ; invalid character
FF                              .BYTE   -1                              ; invalid character
FF                              .BYTE   -1                              ; invalid character
FF                              .BYTE   -1                              ; invalid character
0A                              .BYTE   10                              ; "A"
0B                              .BYTE   11                              ; "B"
0C                              .BYTE   12                              ; "C"
0D                              .BYTE   13                              ; "D"
0E                              .BYTE   14                              ; "E"
0F                              .BYTE   15                              ; "F"
```

```
                     0017   118          .SBTTL  OTS$CVT_TO_L - Convert text (octal) to long
                     0017   119
                     0017   120 ;++
                     0017   121 ; FUNCTIONAL DESCRIPTION:
                     0017   122 ;
                     0017   123 ;       Converts a text representation of a base 8 value to internal
                     0017   124 ;       form.  The usual result of this routine is one longword, but
                     0017   125 ;       values of any length (that is a multiple of 8 bits) may be
                     0017   126 ;       produced.
                     0017   127 ;
                     0017   128 ;       The valid characters for base 8 conversion are -space- and
                     0017   129 ;       "0" through "7".  No sign is permitted.
                     0017   130 ;
                     0017   131 ;       For compatibility with previous releases, the name
                     0017   132 ;       FOR$CNV_IN_O is equivalent to OTS$CVT_TO_L.
                     0017   133 ;
                     0017   134 ;
                     0017   135 ; CALLING SEQUENCE:
                     0017   136 ;
                     0017   137 ;       status.wlc.v = OTS$CVT_TO_L (in_string.rt.ds, value.wx.r
                     0017   138 ;                                   [, byte_count.rl.v
                     0017   139 ;                                   [, caller_flags.rl.v]]
                     0017   140 ;
                     0017   141 ; INPUT PARAMETERS:
                     0017   142 ;
00000004             0017   143          in_string       = 4             ; Input string by descriptor
0000000C             0017   144          byte_count      = 12            ; Optional count of bytes in
                     0017   145                                          ; value, defaults to 4 if
                     0017   146                                          ; omitted or zero.
00000010             0017   147          caller_flags    = 16            ; Optional.
00000000             0017   148              V_BN        = 0             ; If set, blanks are ignored;
                     0017   149                                          ; otherwise blanks are zeroes.
                     0017   150 ;
                     0017   151 ; IMPLICIT INPUTS: None
                     0017   152 ;
                     0017   153 ; OUTPUT PARAMETERS:
                     0017   154 ;
00000008             0017   155          value           = 8             ; Output value by reference.
                     0017   156                                          ; Length is dependent on
                     0017   157                                          ; byte_count.
                     0017   158 ;
                     0017   159 ; IMPLICIT OUTPUTS: None
                     0017   160 ;
                     0017   161 ; COMPLETION CODES:
                     0017   162 ;
                     0017   163 ;       SS$_NORMAL      - Successful completion
                     0017   164 ;       OTS$_INPCONERR  - Input conversion error.  Either an illegal
                     0017   165 ;                         character or overflow happened.
                     0017   166 ;
                     0017   167 ; SIDE EFFECTS: None
                     0017   168 ;
                     0017   169 ;--
                     0017   170
                     0017   171
                     0017   172 FOR$CNV_IN_O::                           ; For compatibility
     407C            0017   173          .ENTRY  OTS$CVT_TO_L, REGMASK
                     0019   174
```

```
                     009B   30  0019   175          BSBW    SETUP                   ; Get parameters and set up
                            001C   176                                              ; registers.
                     00D1   30  001C   177 10$:     BSBW    GET_CHAR                ; Get a character
                       38   19  001F   178          BLSS    EXIT_0                  ; Exit if done
                       2E   13  0021   179          BEQL    20$                     ; Skip if zero
                 08    50   91  0023   180          CMPB    R0, #8                  ; Invalid character?
                       34   18  0026   181          BGEQ    ERROR_0                 ; If so, error
                 03    56   D1  0028   182          CMPL    R6, #3                  ; Is there room for this digit?
                       1F   18  002B   183          BGEQ    15$                     ; Yes
                 01    56   D1  002D   184          CMPL    R6, #1                  ; 0, 1 or 2 bits left?
                       2A   19  0030   185          BLSS    ERROR_0                 ; 0 or less is error
                       0C   14  0032   186          BGTR    12$                     ; 2 bits
                 01    50   91  0034   187          CMPB    R0, #1                  ; 1 bit - digit 1?
                       23   14  0037   188          BGTR    ERROR_0                 ; No, error
     63    01    54    50   F0  0039   189          INSV    R0, R4, #1, (R3)        ; Insert bit
                       11   11  003E   190          BRB     20$                     ; Continue
                 03    50   91  0040   191 12$:     CMPB    R0, #3                  ; 2 bits -less than 4?
                       17   14  0043   192          BGTR    ERROR_0                 ; No, error
     63    02    54    50   F0  0045   193          INSV    R0, R4, #2, (R3)        ; Insert bits
                       05   11  004A   194          BRB     20$                     ; Continue
     63    03    54    50   F0  004C   195 15$:     INSV    R0, R4, #3, (R3)        ; Insert bits
                 54    03   C0  0051   196 20$:     ADDL    #3, R4                  ; Increment bit position
                 56    03   C2  0054   197          SUBL    #3, R6                  ; Decrement bit counter
                       C3   11  0057   198          BRB     10$                     ; Loop back for more digits
                            0059   199
                            0059   200 EXIT_0:
                     004C   31  0059   201          BRW     EXIT
                            005C   202
                            005C   203 ERROR_0:
                     004D   31  005C   204          BRW     ERROR
                            005F   205
```

```
                     005F   207          .SBTTL  OTS$CVT_TZ_L     - Convert text (hex) to long
                     005F   208
                     005F   209 ;++
                     005F   210 ; FUNCTIONAL DESCRIPTION:
                     005F   211 ;
                     005F   212 ;       Converts a text representation of a base 16 value to internal
                     005F   213 ;       form.  The usual result of this routine is one longword, but
                     005F   214 ;       values of any length (that is a multiple of 8 bits) may be
                     005F   215 ;       produced.
                     005F   216 ;
                     005F   217 ;       The valid characters for base 16 conversion are -space-,
                     005F   218 ;       "0" through "9" and "A" through "F".  No sign is permitted.
                     005F   219 ;       Lower case "a" through "f" are acceptible.
                     005F   220 ;
                     005F   221 ;       For compatibility with previous releases, the name
                     005F   222 ;       FOR$CNV_IN_Z is equivalent to OTS$CVT_TZ_L.
                     005F   223 ;
                     005F   224 ;
                     005F   225 ; CALLING SEQUENCE:
                     005F   226 ;
                     005F   227 ;       status.wlc.v = OTS$CVT_TZ_L (in_string.rt.ds, value.wx.r
                     005F   228 ;                                   [, byte_count.rl.v
                     005F   229 ;                                   [, caller_flags.rl.v]]
                     005F   230 ;
                     005F   231 ; INPUT PARAMETERS:
                     005F   232 ;
00000004             005F   233          in_string       = 4             ; Input string by descriptor
0000000C             005F   234          byte_count      = 12            ; Optional count of bytes in
                     005F   235                                          ; value, defaults to 4 if
                     005F   236                                          ; omitted or zero.
00000010             005F   237          caller_flags    = 16            ; Optional.
00000000             005F   238              V_BN        = 0             ; If set, blanks are ignored;
                     005F   239                                          ; otherwise blanks are zeroes.
                     005F   240 ;
                     005F   241 ; IMPLICIT INPUTS: None
                     005F   242 ;
                     005F   243 ; OUTPUT PARAMETERS:
                     005F   244 ;
00000008             005F   245          value           = 8             ; Output value by reference.
                     005F   246                                          ; Length is dependent on
                     005F   247                                          ; byte_count.
                     005F   248 ;
                     005F   249 ; IMPLICIT OUTPUTS: None
                     005F   250 ;
                     005F   251 ; COMPLETION CODES:
                     005F   252 ;
                     005F   253 ;       SS$_NORMAL      - Successful completion
                     005F   254 ;       OTS$_INPCONERR  - Input conversion error.  Either an illegal
                     005F   255 ;                         character or overflow happened.
                     005F   256 ;
                     005F   257 ; SIDE EFFECTS: None
                     005F   258 ;
                     005F   259 ;--
                     005F   260
                     005F   261
                     005F   262 FOR$CNV_IN_Z::                           ; For compatibility
           407C      005F   263          .ENTRY  OTS$CVT_TZ_L, REGMASK
```

```
                                0061   264
                         0053   0061   265          BSBW    SETUP                   ; Get parameters and set up
                                0064   266                                          ; registers.
                         0089   0064   267 10$:     BSBW    GET_CHAR                ; Get a character
                           13   0067   268          BLSS    EXIT_Z                  ; Exit if done
                           09   0069   269          BEQL    20$                     ; Skip if zero
                           56   006B   270          TSTL    R6                      ; Is there room for this digit?
                           10   006D   271          BLEQ    ERROR_Z                 ; No, must be multiple of 4!
                     63    50   006F   272          INSV    R0, R4, #4, (R3)        ; Insert bits
                     54    04   0074   273 20$:     ADDL    #4, R4                  ; Increment bit position
                     56    04   0077   274          SUBL    #4, R6                  ; Decrement bit counter
                           E8   007A   275          BRB     10$                     ; Loop back for more digits
                                007C   276
                                007C   277 EXIT_Z:
                         0029   007C   278          BRW     EXIT
                                007F   279
                                007F   280 ERROR_Z:
                         002A   007F   281          BRW     ERROR
                                0082   282
```

```
                     0082   284          .SBTTL  OTS$CVT_TB_L     - Convert text (binary) to long
                     0082   285
                     0082   286 ;++
                     0082   287 ; FUNCTIONAL DESCRIPTION:
                     0082   288 ;
                     0082   289 ;       Converts a text representation of a base 2 value to internal
                     0082   290 ;       form.  The usual result of this routine is one longword, but
                     0082   291 ;       values of any length (that is a multiple of 8 bits) may be
                     0082   292 ;       produced.
                     0082   293 ;
                     0082   294 ;       The valid characters for base 2 conversion are -space-,
                     0082   295 ;       "0" and "1".  No sign is permitted.
                     0082   296 ;
                     0082   297 ; CALLING SEQUENCE:
                     0082   298 ;
                     0082   299 ;       status.wlc.v = OTS$CVT_TB_L (in_string.rt.ds, value.wx.r
                     0082   300 ;                                   [, byte_count.rl.v
                     0082   301 ;                                   [, caller_flags.rl.v]]
                     0082   302 ;
                     0082   303 ; INPUT PARAMETERS:
                     0082   304 ;
00000004             0082   305          in_string       = 4             ; Input string by descriptor
0000000C             0082   306          byte_count      = 12            ; Optional count of bytes in
                     0082   307                                          ; value, defaults to 4 if
                     0082   308                                          ; omitted or zero.
00000010             0082   309          caller_flags    = 16            ; Optional.
00000000             0082   310              V_BN        = 0             ; If set, blanks are ignored;
                     0082   311                                          ; otherwise blanks are zeroes.
                     0082   312 ;
                     0082   313 ; IMPLICIT INPUTS: None
                     0082   314 ;
                     0082   315 ; OUTPUT PARAMETERS:
                     0082   316 ;
00000008             0082   317          value           = 8             ; Output value by reference.
                     0082   318                                          ; Length is dependent on
                     0082   319                                          ; byte_count.
                     0082   320 ;
                     0082   321 ; IMPLICIT OUTPUTS: None
                     0082   322 ;
                     0082   323 ; COMPLETION CODES:
                     0082   324 ;
                     0082   325 ;       SS$_NORMAL      - Successful completion
                     0082   326 ;       OTS$_INPCONERR  - Input conversion error.  Either an illegal
                     0082   327 ;                         character or overflow happened.
                     0082   328 ;
                     0082   329 ; SIDE EFFECTS: None
                     0082   330 ;
                     0082   331 ;--
                     0082   332
                     0082   333
     407C            0082   334          .ENTRY  OTS$CVT_TB_L, REGMASK
                     0084   335
0030   30            0084   336          BSBW    SETUP                   ; Get parameters and set up
                     0087   337                                          ; registers.
0066   30            0087   338 10$:     BSBW    GET_CHAR                ; Get a character
  16   19            008A   339          BLSS    EXIT_B                  ; Exit if done
  0E   13            008C   340          BEQL    20$                     ; Skip if zero
```

```
                    01    50   D1   008E   341          CMPL    R0, #1                  ; Illegal character?
                          12   14   0091   342          BGTR    ERROR_B                 ; Yes
                          56   D5   0093   343          TSTL    R6                      ; Is there room for this digit?
                          0E   15   0095   344          BLEQ    ERROR_B                 ; No.
                63  01    54   50   F0   0097   345          INSV    R0, R4, #1, (R3)        ; Insert bit
                          54   D6   009C   346 20$:      INCL    R4                      ; Increment bit position
                          56   D7   009E   347          DECL    R6                      ; Decrement bit counter
                          E5   11   00A0   348          BRB     10$                     ; Loop back for more digits
                                   00A2   349
                                   00A2   350 EXIT_B:
                     0003 31   00A2   351          BRW     EXIT
                                   00A5   352
                                   00A5   353 ERROR_B:
                     0004 31   00A5   354          BRW     ERROR
                                   00A8   355
```

```
                                00A8   357          .SBTTL  Local Subroutines
                                00A8   358
                                00A8   359 ;+
                                00A8   360 ; EXIT - Exit successfully
                                00A8   361 ;-
                                00A8   362
                                00A8   363 EXIT:
                     50   01    D0  00A8   364          MOVL    #1, R0                  ; Success status code
                                04  00AB   365          RET                             ; Return
                                    00AC   366
                                    00AC   367 ;+
                                    00AC   368 ; ERROR - Exit with error
                                    00AC   369 ;-
                                    00AC   370
                                    00AC   371 ERROR:
                           0008     30  00AC   372          BSBW    SETUP                   ; To re-zero the value
                50   00000000'8F    D0  00AF   373          MOVL    #OTS$_INPCONERR, R0     ; Error code
                                04  00B6   374          RET                             ; Return with failure
                                    00B7   375
                                    00B7   376
```

```
                        00B7   378 ;+
                        00B7   379 ; SETUP - Perform common initialization
                        00B7   380 ;
                        00B7   381 ;    1.   Determine if optional parameters are present.  Set defaults
                        00B7   382 ;         if any.
                        00B7   383 ;    2.   Zero the value.
                        00B7   384 ;    3.   Set up registers as follows:
                        00B7   385 ;                 R1 - Count of characters in string.
                        00B7   386 ;                 R2 - Points to one byte beyond string.
                        00B7   387 ;                 R3 - Address of value.
                        00B7   388 ;                 R4 - Set to zero, indicates current value bit offset.
                        00B7   389 ;                 R5 - Flag register, set to value of caller_flags.
                        00B7   390 ;                 R6 - Count of bits in value.
                        00B7   391 ;-
                        00B7   392
                        00B7   393 SETUP:
              56   04   D0  00B7   394          MOVL    #4, R6                          ; Initial size is 1 longword
              03   6C   91  00BA   395          CMPB    (AP), #<byte_count/4>           ; Is byte count present?
                   09   19  00BD   396          BLSS    10$                             ; No
                0C AC   D5  00BF   397          TSTL    byte_count(AP)                  ; Is it zero?
                   04   15  00C2   398          BLEQ    10$                             ; Yes (or neg), use default
           56   0C AC   D0  00C4   399          MOVL    byte_count(AP), R6              ; Get byte count
           50   08 AC   D0  00C8   400 10$:     MOVL    value(AP), R0                   ; Get value address
60 56 00   6E   00   2C  00CC   401          MOVC5   #0, (SP), #0, R6, (R0)          ; Fill with zero
                        00D2   402                                                  ; R5 gets zero
         56   56   03   78  00D2   403          ASHL    #3, R6, R6                      ; R6 gets bit count
           51   04 BC   7D  00D6   404          MOVQ    @in_string(AP), R1              ; R1 gets character count
                        00DA   405                                                  ; R2 gets pointer
              51   51   3C  00DA   406          MOVZWL  R1, R1                          ; Clear all but count
              52   51   C0  00DD   407          ADDL    R1, R2                          ; Position pointer to 1 beyond
                        00E0   408                                                  ; last character.
           53   08 AC   D0  00E0   409          MOVL    value(AP), R3                   ; Value address
                   54   D4  00E4   410          CLRL    R4                              ; Bit offset
              04   6C   91  00E6   411          CMPB    (AP), #<caller_flags/4>         ; Are caller_flags present?
                   04   19  00E9   412          BLSS    20$                             ; No
           55   10 AC   D0  00EB   413          MOVL    caller_flags(AP), R5            ; Set up caller flags
                   05   00  00EF   414 20$:     RSB                                     ; Return
```

```
                     00F0   416 ;++
                     00F0   417 ; GET_CHAR - Get a character
                     00F0   418 ;
                     00F0   419 ;     Looks at the next character in the input string, right to
                     00F0   420 ;     left.  If V_BN is set and character is a blank, it is ignored
                     00F0   421 ;     and another character is read.  If V_BN is not set and the
                     00F0   422 ;     character is a blank, it is converted to zero.
                     00F0   423 ;
                     00F0   424 ;     The character is translated to an actual value from 0 to 15.
                     00F0   425 ;     If the character is not 0-9 or A-F, it branches to ERROR.
                     00F0   426 ;
                     00F0   427 ;     The value obtained is moved to R0.  If no characters remain,
                     00F0   428 ;     R0 is set to -1.  Upon exit, the condition codes are set such
                     00F0   429 ;     that branches depending on the value in R0 can be made.
                     00F0   430 ;--
                     00F0   431
                     00F0   432 GET_CHAR:
          51    D5   00F0   433          TSTL    R1                      ; Any characters left?
          33    15   00F2   434          BLEQ    EOS                     ; No, exit with -1
     50   72    9A   00F4   435          MOVZBL  -(R2), R0               ; Put character in R0
          51    D7   00F7   436          DECL    R1                      ; Decrement count
     20   50    91   00F9   437          CMPB    R0, #^A/ /              ; Is it a blank?
          07    12   00FC   438          BNEQ    10$                     ; No if not equal
EE 55     00    E0   00FE   439          BBS     #V_BN, R5, GET_CHAR     ; Ignore if V_BN set
          50    D4   0102   440          CLRL    R0                      ; Blanks are zero
                05   0104   441          RSB                             ; Exit with zero
     50   30    C2   0105   442 10$:     SUBL    #^A/0/, R0              ; Subtract "0"
          21    19   0108   443          BLSS    ERROR_G                 ; Error if neg
     16   50    D1   010A   444          CMPL    R0, #<^A/F/-^A/0/>      ; Greater than "F"?
          0F    15   010D   445          BLEQ    20$                     ; Ok if not greater
     50   20    C2   010F   446          SUBL    #<^A/a/-^A/A/>, R0      ; Try lower case
          17    19   0112   447          BLSS    ERROR_G                 ; Error if between "F" and "P"
     11   50    D1   0114   448          CMPL    R0, #<^A/A/-^A/0/>      ; In between "0" and "a"?
          12    19   0117   449          BLSS    ERROR_G                 ; Yes, error
     16   50    D1   0119   450          CMPL    R0, #<^A/f/-^A/0/>      ; Greater than "f"?
          0D    14   011C   451          BGTR    ERROR_G                 ; Yes, error
50   FEDD CF40    90   011E   452 20$:     MOVB    TABLE[R0], R0           ; Get actual value
               05    19   0124   453          BLSS    ERROR_G                 ; Error if negative
                     05   0126   454          RSB                             ; Exit with condition codes set
                          0127   455
                          0127   456 EOS:
          50   01    CE   0127   457          MNEGL   #1, R0                  ; End of string
                     05   012A   458          RSB                             ; Exit
                          012B   459
                          012B   460 ERROR_G:
               FF7E  31   012B   461          BRW     ERROR                   ; Illegal character
                          012E   462
                          012E   463
                          012E   464          .END
```

```
BYTE_COUNT      = 0000000C
CALLER_FLAGS    = 00000010
EOS               00000127 R      01
ERROR             000000AC R      01
ERROR_B           000000A5 R      01
ERROR_G           0000012B R      01
ERROR_O           0000005C R      01
ERROR_Z           0000007F R      01
EXIT              000000A8 R      01
EXIT_B            000000A2 R      01
EXIT_O            00000059 R      01
EXIT_Z            0000007C R      01
FOR$CNV_IN_O      00000017 RG     01
FOR$CNV_IN_Z      0000005F RG     01
GET_CHAR          000000F0 R      01
IN_STRING       = 00000004
OTS$CVT_TB_L      00000082 RG     01
OTS$CVT_TO_L      00000017 RG     01
OTS$CVT_TZ_L      0000005F RG     01
OTS$_INPCONERR    ********    X   00
REGMASK         = 0000407C
SETUP             000000B7 R      01
TABLE             00000000 R      01
VALUE           = 00000008
V_BN            = 00000000
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
_OTS$CODE	0000012E (302.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.03	00:00:03.11
Command processing	117	00:00:00.32	00:00:03.94
Pass 1	83	00:00:00.63	00:00:03.88
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	88	00:00:00.53	00:00:02.23
Symbol table output	3	00:00:00.02	00:00:00.02
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	324	00:00:01.56	00:00:13.20

The working set limit was 1050 pages.
6504 bytes (13 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 25 non-local and 12 local symbols.
464 source lines were read in Pass 1, producing 17 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.

OTS$CVTTOL ; Convert text to integer 16-SEP-1984 00:30:37 VAX/VMS Macro V04-00 Page 14
VAX-11 Macro Run Statistics 6-SEP-1984 11:13:53 [LIBRTL.SRC]OTSCVTTOL.MAR;1 (9)

```
                                    +---------------------------+
                                    ! 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.

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

OTSCVTPD
LIS

OTSCVTTF
LIS

OTSCVTRFP
LIS

OTSCVTRT
LIS

OTSCVTTOL
LIS

OTSCVTPG
LIS

OTSCVTTR
LIS

OTSCVTRHP
LIS

OTSLINKAG
LIS

OTSCVTRDP
LIS

OTSCVTTIL
LIS

OTSCVTTLL
LIS

OTSCVTPF
LIS

OTSCVTRGP
LIS

OTSMOVE
LIS

OTSMSG
LIS

OTSCVTPH
LIS

OTSLUN
LIS