I 3

```
   CCCCCCCCCCCC  LLL                 IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCCC  LLL                 IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCCC  LLL                 IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
```

```
                                                  E  4
**FILE**ID**CHRSUB




  CCCCCCCC  HH      HH  RRRRRRRR      SSSSSSSS  UU      UU  BBBBBBBB
  CCCCCCCC  HH      HH  RRRRRRRR      SSSSSSSS  UU      UU  BBBBBBBB
CC          HH      HH  RR      RR  SS          UU      UU  BB      BB
CC          HH      HH  RR      RR  SS          UU      UU  BB      BB
CC          HH      HH  RR      RR  SS          UU      UU  BB      BB
CC          HH      HH  RR      RR  SS          UU      UU  BB      BB
CC          HHHHHHHHHH  RRRRRRRR      SSSSSS    UU      UU  BBBBBBBB
CC          HHHHHHHHHH  RRRRRRRR      SSSSSS    UU      UU  BBBBBBBB
CC          HH      HH  RR  RR              SS  UU      UU  BB      BB
CC          HH      HH  RR  RR              SS  UU      UU  BB      BB
CC          HH      HH  RR    RR            SS  UU      UU  BB      BB      ....
CC          HH      HH  RR    RR            SS  UU      UU  BB      BB      ....
  CCCCCCCC  HH      HH  RR      RR  SSSSSSSS    UUUUUUUUUU  BBBBBBBB        ....
  CCCCCCCC  HH      HH  RR      RR  SSSSSSSS    UUUUUUUUUU  BBBBBBBB        ....


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
```

F 4

Table of contents

```
0000            .TITLE  CHRSUB - CHARACTER MANIPULATION SUBROUTINES
0000            .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ; FACILITY:     UTILITY SUBROUTINES
0000    ;
0000    ; ABSTRACT:     CHARACTER MANIPUATION SUBROUTINES
0000    ;
0000    ; ENVIRONMENT:  NATIVE/USER MODE CODE
0000    ;
0000    ; AUTHOR:       W.H.BROWN, CREATION DATE:       19-MAY-1977
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       . : VERSION
0000    ; 01    -
0000    ;--
```

```
                    0000     44             .SBTTL  DECLARATIONS
                    0000     45     ;
                    0000     46     ; INCLUDE FILES:
                    0000     47     ;
                    0000     48
                    0000     49     ;
                    0000     50     ; MACROS:
                    0000     51     ;
                    0000     52     ; MACRO TO GENERATE AN ENTRY IN THE CHARACTER CLASSIFICATION TABLE
                    0000     53     ;
                    0000     54     ; CALL:
                    0000     55     ;       CHAR    NAME,CHR
                    0000     56     ; WHERE:
                    0000     57     ;       NAME IS THE SYMBOLIC NAME SUFFIX TO "CHR$K_" FOR THE CHAR
                    0000     58     ;       CHR IS THE ASCII CHAR.
                    0000     59     ;
                    0000     60             .MACRO  CHAR    NAME,CHR,N
                    0000     61             CHR$K_'NAME == N
                    0000     62             .BYTE   ^A\CHR\
                    0000     63             .ENDM
                    0000     64
                    0000     65     ;
                    0000     66     ; EQUATED SYMBOLS:
                    0000     67     ;
                    0000     68     ; DEFINE SPECIAL SYMBOLS FOR ALPHA/NUMERIC SETS
                    0000     69     ;
          00000001  0000     70             CHR$K_ALPHA == 1
          00000002  0000     71             CHR$K_NUMERIC == 2
                    0000     72
                    0000     73     ;
                    0000     74     ; OWN STORAGE:
                    0000     75     ;
                00000000     76             .PSECT  _PURE   RD,NOWRT,BYTE
                    0000     77
                    0000     78 CHRTBL:
                    0000     79             CHAR    SLASH   </>     12
                    0001     80             CHAR    SEMI    <;>     11
                    0002     81             CHAR    LBRAKT  <[>     10
                    0003     82             CHAR    RBRAKT  <]>     9
                    0004     83             CHAR    COMMA   <,>     8
                    0005     84             CHAR    DOT     <.>     7
                    0006     85             CHAR    COLON   <:>     6
                    0007     86             CHAR    BLANK   < >     5
                    0008     87             CHAR    DOLLAR  <$>     4
                    0009     88             CHAR    UNDRSCR <_>     3
             00 00  000A     89             .BYTE   0,0                     ; EOL AND FILLER FOR REMAINING COUNT
                    000C     90
          0000000C  000C     91 CHRTBLSIZ = . - CHRTBL
                    000C     92
          2B 25 2D  000C     93 SPCNUM: .ASCII  \-%+\                       ; SPECIAL CHARACTERS TREATED AS NUMERIC
          00000003  000F     94 SPCNUMSIZ = . - SPCNUM
                    000F     95
```

```
                     000F         .SBTTL  TEST A CHARACTER FOR CLASS
                     000F   ;++
                     000F   ; FUNCTIONAL DESCRIPTION:
                     000F   ;
                     000F   ;         THIS ROUTINE IS CALLED TO CLASSIFY AN ASCII CHARATER INTO
                     000F   ;         ONE OF SEVERAL CLASSES. AN ALTERNATE ENTRY PROVIDES LOWER
                     000F   ;         TO UPPER CASE CONVERSION AS WELL.
                     000F   ;
                     000F   ; CALLING SEQUENCE:
                     000F   ;
                     000F   ;         BSB/JSB CHR$TSTCHR                  ; TEST THE CHARACTER
                     000F   ;         BSB/JSB CHR$CVT                     ; CONVERT AND TEST
                     000F   ;
                     000F   ; INPUT PARAMETERS:
                     000F   ;
                     000F   ;         R6 CONTAINS ADDRESS OF BYTE TO TEST
                     000F   ;
                     000F   ; IMPLICIT INPUTS:
                     000F   ;
                     000F   ;         STRING IS TERMINATED BY A ZERO BYTE
                     000F   ;
                     000F   ; OUTPUT PARAMETERS:
                     000F   ;
                     000F   ;         RO SET TO "CHR$K_<CLASS_NAME>" IF ONE OF RECOGNIZED CHARACTERS
                     000F   ;            ELSE SET TO MINUS 1
                     000F   ;
                     000F   ; IMPLICIT OUTPUTS:
                     000F   ;
                     000F   ;         NONE
                     000F   ;
                     000F   ; COMPLETION CODES:
                     000F   ;
                     000F   ;         NONE
                     000F   ;
                     000F   ; SIDE EFFECTS:
                     000F   ;
                     000F   ;         NONE
                     000F   ;
                     000F   ;--
                     000F
                     000F   CHR$CVT::                                 ; CONVERT TO UPPER CASE
61 8F   66    91     000F           CMPB    (R6),#<^A/A/+^X20>        ; LOWER CASE A?
        0D    19     0013           BLSS    CHR$TSTCHR                ; BR IF NOT LOWER
7A 8F   66    91     0015           CMPB    (R6),#<^A/Z/+^X20>        ; LOWER CASE Z?
        07    14     0019           BGTR    CHR$TSTCHR                ; BR IF NOT LOWER
   66   20    82     001B           SUBB    #^X20,(R6)                ; CONVERT TO UPPER
        02    11     001E           BRB     CHR$TSTCHR                ;
                     0020
                     0020   CHR$TSTNXT::                              ; TEST NEXT CHAR
        56    D6     0020           INCL    R6                        ; ADD ONE TI ADDRESS
                     0022
                     0022   CHR$TSTCHR::                              ; TEST A CHARACTER FOR CLASS
        50    D4     0022           CLRL    R0                        ; ASSUME END-OF-LINE
        66    95     0024           TSTB    (R6)                      ; END-OF-LINE?
        50    13     0026           BEQL    90$                       ; BR IF YES
        50    D6     0028           INCL    R0                        ; SET TYPE TO ALPHA
41 8F   66    91     002A           CMPB    (R6),#^A/A/               ; CHECK AGAINST LOW LIMIT
```

```
                 12    1F  002E   154          BLSSU   20$                         ; BR IF BELOW ALPHA
         5A 8F   66    91  0030   155          CMPB    (R6),#^A/Z/                 ; NOW CHECK HI END
                 42    15  0034   156          BLEQ    90$                         ; BR IF ALPHA
         61 8F   66    91  0036   157          CMPB    (R6),#<^A/A/+^X20>          ; CHECK FOR LOWER CASE ALPHA
                 06    19  003A   158          BLSS    20$                         ; BR IF NO
         7A 8F   66    91  003C   159          CMPB    (R6),#<^A/Z/+^X20>          ; OTHER LIMIT
                 36    15  0040   160          BLEQ    90$                         ; FOUND THE CLASS
                 02    DD  0042   161 20$:     PUSHL   S^#CHR$K_NUMERIC            ; SET VALUE FOR NUMERIC CHARATERS
      C3 AF   03 66    3A  0044   162          LOCC    (R6),#SPCNUMSIZ,SPCNUM      ; CHECK FOR SPECIAL NUMERIC CHARACTERS
                 01    BA  0049   163          POPR    #^M<R0>                     ; GET VALUE FOR NUMERIC CHARACTER
                 2B    12  004B   164          BNEQ    90$                         ; BR IF CHARACTER IS SPECIAL NUMERIC
              30 66    91  004D   165          CMPB    (R6),#^A/0/                 ; CHECK LOW LIMIT
                 05    19  0050   166          BLSS    30$                         ; BR IF NOT NUMERIC
              39 66    91  0052   167          CMPB    (R6),#^A/9/                 ; WHAT ABOUT THE HI LIMIT
                 21    15  0055   168          BLEQ    90$                         ; BR IF NUMERIC
      A4 AF   0C 66    3A  0057   169 30$:     LOCC    (R6),#CHRTBLSIZ,CHRTBL      ; CHECK IF ONE OF SPECIALS
                 1A    12  005C   170          BNEQ    90$                         ; BR IF YES
              50 05    D0  005E   171          MOVL    #CHR$K_BLANK,R0             ; ASSUME TAB
              09 66    91  0061   172          CMPB    (R6),#^A/       /           ; IS IT A TAB?
                 12    13  0064   173          BEQL    90$                         ; BR IF YES
              50 0A    D0  0066   174          MOVL    #CHR$K_LBRAKT,R0            ; ASSUME LEFT BRACKET
              3C 66    91  0069   175          CMPB    (R6),#^A/</                 ; IS IT THE FUNNY BRAKET?
                 0A    13  006C   176          BEQL    90$                         ; BR IF YES
                 50    D6  006E   177          INCL    R0                          ; CHANGE CODE TO RIGHT BRACKET
              3E 66    91  0070   178          CMPB    (R6),#^A/>/                 ; CHECK CLOSE BRAKET
                 03    13  0073   179          BEQL    90$                         ; BR IF YES
              50 01    CE  0075   180          MNEGL   #1,R0                       ; SET AS GENERAL SPECIAL
                 50    D5  0078   181 90$:     TSTL    R0                          ; SET STATUS BASED ON VALUE
                       05  007A   182          RSB                                 ;
```

```
                  007B          .SBTTL  GET TOKEN
                  007B   ;++
                  007B   ; FUNCTIONAL DESCRIPTION:
                  007B   ;
                  007B   ;      THIS ROUTINE IS CALLED TO PARSE THE NEXT TOKEN FROM THE
                  007B   ;      COMMAND LINE.
                  007B   ;
                  007B   ; CALLING SEQUENCE:
                  007B   ;
                  007B   ;      BSB/JSB CHR$GETOKEN                 ; GET TOKEN FROM LINE
                  007B   ;      BSB/JSB CHR$NXTOKEN                 ; TOKEN AFTER NEXT CHARACTER
                  007B   ;
                  007B   ; INPUT PARAMETERS:
                  007B   ;
                  007B   ;      R6 CONTAINS ADDRESS OF NEXT BYTE ON THE LINE
                  007B   ;
                  007B   ; IMPLICIT INPUTS:
                  007B   ;
                  007B   ;      STRING IS TERMINATED BY ZERO BYTE
                  007B   ;
                  007B   ; OUTPUT PARAMETERS:
                  007B   ;
                  007B   ;      R6 IS ADVANCED TO THE FIRST NONE BLANK CHARACTER AFTER THE TOKEN.
                  007B   ;      R3,R4 ARE A DESCRIPTOR TO THE TOKEN
                  007B   ;
                  007B   ; IMPLICIT OUTPUTS:
                  007B   ;
                  007B   ;      "Z" BIT IS SET IF ZERO LENGTH TOKEN IS PARSED.
                  007B   ;
                  007B   ; COMPLETION CODES:
                  007B   ;
                  007B   ;      R0 IS SET TO THE TYPE OF THE CHARACTER
                  007B   ;
                  007B   ; SIDE EFFECTS:
                  007B   ;
                  007B   ;      NONE
                  007B   ;
                  0076   ;--
                  007B          .ENABL  LSB
                  007B
                  007B   CHR$GETOKEN::                              ; GET TOKEN
        56    D7  007B          DECL    R6                          ; BACK UP ONE FOR SKIP
                  007D   CHR$NXTOKEN::                              ; TOKEN FOLLOWING CURRENT CHAR
        1C    10  007D          BSBB    CHR$NXTNBLK                 ; FIND NON-BLANK
     54 66    9E  007F          MOVAB   (R6),R4                     ; SET START ADDRESS OF TOKEN
        56    D7  0082          DECL    R6                          ; BACK UP SO SKIP WILL START HERE
53   01 A6    9E  0084   10$:   MOVAB   1(R6),R3                    ; SET ADDRESS OF NEXT BYTE
        96    10  0088          BSBB    CHR$TSTNXT                  ; LOOK AT NEXT CHAR
        09    13  008A          BEQL    40$                         ; BR ON END OF LINE
     05 50    91  008C          CMPB    R0,#CHR$K_BLANK             ; VALID CHARACTER FOR TOKEN?
        F3    1F  008F          BLSSU   10$                         ; IF LSSU YES-KEEP LOOKING FOR TERMIATOR
        02    12  0091          BNEQ    40$                         ; BR IF NOT A SPACE
        06    10  0093          BSBB    CHR$NXTNBLK                 ; SKIP TO NON-BLANK
     53 54    C2  0095   40$:   SUBL    R4,R3                       ; FIND LENGTH OF TOKEN
              05  0098   50$:   RSB                                 ; GET OUT
```

```
                0099          .DSABL  LSB
                0099          .SBTTL  SET NONE BLANK
                0099  ;++
                0099  ; FUNCTIONAL DESCRIPTION:
                0099  ;
                0099  ;       THIS ROUTINE IS CALLED TO ADVANCE THE CHARACTER POINTER
                0099  ;       TO  THE FIRST NONE BLANK CHARATER ON THE LINE.
                0099  ;
                0099  ; CALLING SEQUENCE:
                0099  ;
                0099  ;       BSB/JSB CHR$SETNB                  ; SET NONE BLANK
                0099  ;
                0099  ; INPUT PARAMETERS:
                0099  ;
                0099  ;       R6 CONTAINS ADDRESS OF NEXT BYTE ON THE LINE
                0099  ;
                0099  ; IMPLICIT INPUTS:
                0099  ;
                0099  ;       NONE
                0099  ;
                0099  ; OUTPUT PARAMETERS:
                0099  ;
                0099  ;       R6 IS ADVANCED TO THE FIRST NONE BLANK CHARACTER
                0099  ;
                0099  ; IMPLICIT OUTPUTS:
                0099  ;
                0099  ;       NONE
                0099  ;
                0099  ; COMPLETION CODES:
                0099  ;
                0099  ;       R0 = 1 IF MORE DATA ON LINE, 0 IS NO NONE BLANK CHARACTERS
                0099  ;
                0099  ; SIDE EFFECTS:
                0099  ;
                0099  ;       NONE
                0099  ;
                0099  ;--
                0099          .ENABL  LSB
                0099
                0099  CHR$SETNBLK::                      ; SET NONE BLANK
        56  D7  0099          DECL    R6                 ; BACK UP SO SKIP ONE WILL BE NOP
                009B  CHR$NXTNBLK::                      ; SKIP THEN-THEN NEXT NONE BLANK
      FF82  30  009B  20$:    BSBW    CHR$TSTNXT         ;
        08  13  009E          BEQL    40$                ; BR IF END-OF-LINE
    05  50  91  00A0          CMPB    R0,#CHR$K_BLANK    ; NEXT CHAR BLANK
        F6  13  00A3          BEQL    20$                ; IF YES-KEEP LOOKING
    50  01  D0  00A5          MOVL    #1,R0              ; SUCESS
            05  00A8  40$:    RSB                        ; ALL DONE
                00A9
                00A9          .DSABL  LSB
                00A9          .END
```

```
CHR$CVT              0000000F RG       01
CHR$GETOKEN          0000007B RG       01
CHR$K_ALPHA        = 00000001  G
CHR$K_BLANK        = 00000005  G
CHR$K_COLON        = 00000006  G
CHR$K_COMMA        = 00000008  G
CHR$K_DOLLAR       = 00000004  G
CHR$K_DOT          = 00000007  G
CHR$K_LBRAKT       = 0000000A  G
CHR$K_NUMERIC      = 00000002  G
CHR$K_RBRAKT       = 00000009  G
CHR$K_SEMI         = 0000000B  G
CHR$K_SLASH        = 0000000C  G
CHR$K_UNDRSCR      = 00000003  G
CHR$NXTNBLK          0000009B RG       01
CHR$NXTOKEN          0000007D RG       01
CHR$SETNBLK          00000099 RG       01
CHR$TSTCHR           00000022 RG       01
CHR$TSTNXT           00000020 RG       01
CHRTBL               00000000 R        01
CHRTBLSIZ          = 0000000C
SPCNUM               0000000C R        01
SPCNUMSIZ          = 00000003


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

PSECT name                         Allocation          PSECT No.  Attributes
----------                         ----------          ---------  ----------
.  ABS  .                          00000000  (     0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
_PURE                              000000A9  (   169.)  01 (  1.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

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

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        12    00:00:00.12    00:00:01.49
Command processing                   105    00:00:00.94    00:00:03.15
Pass 1                                94    00:00:00.76    00:00:03.16
Symbol table sort                      0    00:00:00.01    00:00:00.01
Pass 2                                62    00:00:00.53    00:00:01.85
Symbol iable output                    4    00:00:00.04    00:00:00.04
Psect synopsis output                  1    00:00:00.02    00:00:00.03
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                 280    00:00:02.43    00:00:09.74

The working set limit was 750 pages.
4246 bytes (9 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 23 non-local and 8 local symbols.
291 source lines were read in Pass 1, producing 11 object records in Pass 2.
1 page of virtual memory was used to define 1 macro.
```

```
                                        +-----------------------------+
                                        ! Macro library statistics !
                                        +-----------------------------+

Macro library name                      Macros defined
------------------                      --------------
_$255$DUA28:[CLIUTL.OBJ]CLIUTL.MLB;1             0
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                   0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                0
TOTALS (all libraries)                           0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:CHRSUB/OBJ=OBJ$:CHRSUB MSRC$:CHRSUB/UPDATE=(ENH$:CHRSUB)+EXECML$/LIB+LIB$:CLIUTL/LIB
```

0049 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY