D 8

```
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        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
```

```
**FILE**ID**STRDUPLCH                                  C  7
```

```
  SSSSSSSS  TTTTTTTTTT  RRRRRRRR    DDDDDDDD    UU      UU  PPPPPPPP    LL              CCCCCCCC  HH      HH
  SSSSSSSS  TTTTTTTTTT  RRRRRRRR    DDDDDDDD    UU      UU  PPPPPPPP    LL              CCCCCCCC  HH      HH
SS              TT      RR      RR  DD      DD  UU      UU  PP      PP  LL            CC          HH      HH
SS              TT      RR      RR  DD      DD  UU      UU  PP      PP  LL            CC          HH      HH
SS              TT      RR      RR  DD      DD  UU      UU  PP      PP  LL            CC          HH      HH
SS              TT      RR      RR  DD      DD  UU      UU  PP      PP  LL            CC          HH      HH
  SSSSSS        TT      RRRRRRRR    DD      DD  UU      UU  PPPPPPPP    LL            CC          HHHHHHHHHH
  SSSSSS        TT      RRRRRRRR    DD      DD  UU      UU  PPPPPPPP    LL            CC          HHHHHHHHHH
        SS      TT      RR  RR      DD      DD  UU      UU  PP          LL            CC          HH      HH
        SS      TT      RR  RR      DD      DD  UU      UU  PP          LL            CC          HH      HH
        SS      TT      RR    RR    DD      DD  UU      UU  PP          LL            CC          HH      HH       ....
        SS      TT      RR    RR    DD      DD  UU      UU  PP          LL            CC          HH      HH       ....
SSSSSSSS        TT      RR      RR  DDDDDDDD    UUUUUUUUUU  PP          LLLLLLLLLL      CCCCCCCC  HH      HH       ....
SSSSSSSS        TT      RR      RR  DDDDDDDD    UUUUUUUUUU  PP          LLLLLLLLLL      CCCCCCCC  HH      HH       ....


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
```

```
0 MODULE STR$DUPL_CHAR (          ! Duplicate a character in a string
0
0                  IDENT = '1-010' ! File: STRDUPLCH.B32   Edit: DG1010
0
0                        ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY: String support library
1 !
1 ! ABSTRACT:
1 !
1 !       This routine fills a string with an input number (defaults to
1 !       1) of an input character (defaults to space).
1 !
1 ! ENVIRONMENT: User mode, AST level or not or mixed
1 !
1 ! AUTHOR: R. Will,  CREATION DATE: 13-Mar-79
1 !
1 ! MODIFIED BY:
1 !
1 !       R. Will, 13-Mar-79: VERSION 01
1 ! 1-001 - Original
1 ! 1-002 - Use STR$K_FILL_CHAR.  JBS 15-APR-1979
1 ! 1-003 - String cleanup.  Change name to STR$.  RW  8-Nov-79
1 ! 1-004 - Don't use the string interlock macros from JSB entry
1 !         points.  JBS 15-NOV-1979
1 ! 1-005 - String speedup.  RW  7-Jan-1980
1 ! 1-006 - Enhance to accomodate additional classes of destination
1 !         descriptors by using $STR$GET_LEN_ADDR to extract length
1 !         and address of 1st data byte indicated by descriptor.
1 !         Remove string interlocking code.
1 !         RKR 20-APR-1981
```

```
;    58          0058  1 ! 1-007 - Speed up code.  RKR 7-OCT-1981.
;    59          0059  1 ! 1-008 - Use $STR$SIGNAL_FATAL rather that $STR$CHECK_STATUS.
;    60          0060  1 !         RKR 18-NOV-1981.
;    61          0061  1 ! 1-009 - Add support for class SO string descriptor.  DG 3-Oct-1983
;    62          0062  1 ! 1-010 - Change class SO string descriptor to SB.  DG 27-Feb-1984
;    63          0063  1 !--
;    64          0064  1
;    65          0065  1 !<BLF/PAGE>
```

ST
1-

```
;    67          0066  1 !
;    68          0067  1 ! SWITCHES:
;    69          0068  1 !
;    70          0069  1
;    71          0070  1 SWITCHES ADDRESSING_MODE
;    72          0071  1                 (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    73          0072  1
;    74          0073  1 !
;    75          0074  1 ! LINKAGES:
;    76          0075  1 !
;    77          0076  1
;    78          0077  1 REQUIRE 'RTLIN:STRLNK';          ! Use require file with string linkage
;    79          0262  1
;    80          0263  1 !
;    81          0264  1 ! TABLE OF CONTENTS:
;    82          0265  1 !
;    83          0266  1
;    84          0267  1 FORWARD ROUTINE
;    85          0268  1     STR$DUPL_CHAR,                      ! Fill a string with a character
;    86          0269  1     STR$DUPL_CHARR8 : STR$JSB_DUPL_CH;  ! JSB entry point
;    87          0270  1
;    88          0271  1 !
;    89          0272  1 ! INCLUDE FILES:
;    90          0273  1 !
;    91          0274  1
;    92          0275  1 REQUIRE 'RTLIN:RTLPSECT';               ! Use to declare PSECTs
;    93          0370  1
;    94          0371  1 REQUIRE 'RTLIN:STRMACROS';              ! Use string macros to code
;    95          1287  1
;    96          1288  1 LIBRARY 'RTLSTARLE';                    ! STARLET library for macros
;    97          1289  1                                         ! and symbols
;    98          1290  1
;    99          1291  1 !
;   100          1292  1 ! MACROS : NONE
;   101          1293  1 !
;   102          1294  1
;   103          1295  1 !
;   104          1296  1 ! EQUATED SYMBOLS:
;   105          1297  1 !
;   106          1298  1
;   107          1299  1 LITERAL
;   108          1300  1     DEFAULT_LENGTH = 1;         ! Default length of string produced
;   109          1301  1
;   110          1302  1 !
;   111          1303  1 ! PSECT DECLARATIONS
;   112          1304  1 !
;   113          1305  1 DECLARE_PSECTS (STR);
;   114          1306  1 !
;   115          1307  1 ! OWN STORAGE:
;   116          1308  1 !
;   117          1309  1 !       NONE
;   118          1310  1 !
;   119          1311  1 ! EXTERNAL REFERENCES:
;   120          1312  1 !
;   121          1313  1
;   122          1314  1 EXTERNAL ROUTINE
;   123          1315  1     LIB$STOP;                                       ! signal errors
;  124          1316  1
;  125          1317  1 EXTERNAL LITERAL
;  126          1318  1     STR$_ILLSTRCLA,                      ! illegal string class
;  127          1319  1     STR$_NEGSTRLEN,                      ! negative string length
;  128          1320  1     STR$_NORMAL,                         ! normal successful completion
;  129          1321  1     STR$_TRU,                            ! truncation occurred
;  130          1322  1     STR$_STRTOOLON;                      ! string too long, >65535
```

```
GLOBAL ROUTINE STR$DUPL_CHAR (                  ! Create a string of a char

        DEST_DESC,                              ! Pointer to dest str desc
        INPUT_LENGTH,                           ! Number of characters
        INPUT_CHAR                              ! Character to duplicate
                                ) : =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine writes LENGTH characters of CHAR into the string
!       pointer to by DEST_DESC.  If the destination is a fixed length
!       string, and LENGTH is greater than the length of the string,
!       only as many CHARs as will fit are copied.  If destination is
!       fixed length and LENGTH is less than the destination string
!       length then LENGTH CHARs are copied and the destination is
!       padded with blanks.  If the destination is a dynamic string,
!       after execution of this routine the destination will have a
!       length of LENGTH.
!       If the destination has varying string semantics and the LENGTH
!       exceeds MAXSTRLEN, STR$_TRU is returned.
!       The call entry point is implemented by
!       JSBing to the JSB entry point.
!
! FORMAL PARAMETERS:
!
!       DEST_DESC.wt.dx         pointer to destination string descrirtor
!       INPUT_LENGTH.rl.r       number of characters to duplicate
!       INPUT_CHAR.rbu.r        ASCII character to duplicate
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION CODES:
!
!       same as STR$DUPL_CHARR8
!
! SIDE EFFECTS:
!
!       same as STR$DUPL_CHARR8
!
!--

    BEGIN

    BUILTIN
        NULLPARAMETER;                          ! check for optional args

    LOCAL
        CHAR : BYTE,                                    ! character to use
        LENGTH;                                         ! length to use

```

```
;  189          1380  2      MAP
;  190          1381  2          DEST_DESC : REF $STR$DESCRIPTOR;
;  191          1382  2
;  192          1383  2      IF NULLPARAMETER (3)                 ! if character is not input
;  193          1384  2      THEN
;  194          1385  2          CHAR = STR$K_FILL_CHAR           ! use the default character
;  195          1386  2      ELSE
;  196          1387  2          CHAR = ..INPUT_CHAR;             ! else use the input character
;  197          1388  2
;  198          1389  2      IF NULLPARAMETER (2)                 ! if length is not input
;  199          1390  2      THEN
;  200          1391  2          LENGTH = DEFAULT_LENGTH          ! use the default
;  201          1392  2      ELSE
;  202          1393  2          LENGTH = ..INPUT_LENGTH;         ! else use the input value
;  203          1394  2
;  204          1395  2      RETURN STR$DUPL_CHARR8 (DEST_DESC [0,0,0,0], .LENGTH, .CHAR);
;  205          1396  2
;  206          1397  1      END;                                 !End of STR$DUPL_CHAR


                                                                   .TITLE  STR$DUPL_CHAR
                                                                   .IDENT  \1-010\

                                                                   .EXTRN  LIB$STOP, STR$_ILLSTRCLA
                                                                   .EXTRN  STR$_NEGSTRLEN, STR$_NORMAL
                                                                   .EXTRN  STR$_TRU, STR$_STRTOOLON

                                                                   .PSECT  _STR$CODE,NOWRT,  SHR,  PIC,2

                                      01FC 00000                   .ENTRY  STR$DUPL_CHAR, Save R2,R3,R4,R5,R6,R7,R8    ; 1323
                      03            6C  91 00002                   CMPB    (AP), #3                                    ; 1383
                                    05  1F 00005                   BLSSU   1$                                          ;
                              0C    AC  D5 00007                   TSTL    12(AP)                                      ;
                                    05  12 0000A                   BNEQ    2$                                          ;
                      53            20  90 0000C 1$:               MOVB    #32, CHAR                                   ; 1385
                                    04  11 0000F                   BRB     3$                                          ;
                      53      0C    BC  90 00011 2$:               MOVB    @INPUT_CHAR, CHAR                           ; 1387
                      02            6C  91 00015 3$:               CMPB    (AP), #2                                    ; 1389
                                    05  1F 00018                   BLSSU   4$                                          ;
                              08    AC  D5 0001A                   TSTL    8(AP)                                       ;
                                    05  12 0001D                   BNEQ    5$                                          ;
                      51            01  D0 0001F 4$:               MOVL    #1, LENGTH                                  ; 1391
                                    04  11 00022                   BRB     6$                                          ;
                      51      08    BC  D0 00024 5$:               MOVL    @INPUT_LENGTH, LENGTH                       ; 1393
                      52            53  9A 00028 6$:               MOVZBL  CHAR, R2                                    ; 1395
                      50      04    AC  D0 0002B                   MOVL    DEST_DESC, R0                               ;
                                 0000V  30 0002F                   BSBW    STR$DUPL_CHARR8                             ;
                                        04 00032                   RET                                                 ; 1397

; Routine Size:  51 bytes,    Routine Base:  _STR$CODE + 0000
```

```
1 GLOBAL ROUTINE STR$DUPL_CHARR8 (         ! Create a string of a char
1
1           DEST_DESC,                     ! Pointer to dest str desc
1           INPUT_LENGTH,                  ! Number of characters
1           INPUT_CHAR                     ! Character to duplicate
1
1                                   ) : STR$JSB_DUPL_CH =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !         This routine writes LENGTH characters of CHAR into the string
1 !         pointer to by DEST_DESC.  If the destination is a fixed length
1 !         string, and LENGTH is greater than the length of the string,
1 !         only as many CHARs as will fit are copied.  If destination is
1 !         fixed length and LENGTH is less than the destination string
1 !         length then LENGTH CHARs are copied and the destination is
1 !         padded with blanks.  If the destination is a dynamic string,
1 !         after execution of this routine the destination will have a
1 !         length of LENGTH.
1 !         If the destination has varying string semantics and the LENGTH
1 !         exceeds MAXSTRLEN, STR$_TRU is returned.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !         DEST_DESC.wt.dx         pointer to destination string descriptor
1 !         INPUT_LENGTH.rl.v       value of no. of characters to duplicate
1 !         INPUT_CHAR.rbu.v        value of ASCII character to duplicate
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         NONE
1 !
1 ! COMPLETION CODES:
1 !
1 !         STR$_NORMAL     if successful completion
1 !         STR$_NEGSTRLEN  if string length is negative
1 !         STR$_TRU        if input length is greater than fixed string
1 !                         length or length greater than MAXSTRLEN for
1 !                         varying string destination
1 !
1 ! SIDE EFFECTS:
1 !
1 !         may allocate or deallocate dynamic string space
1 !         may signal errors
1 !                 STR$_ILLSTRCLA  if not supported string class
1 !                 STR$_INSVIRMEM  if can't allocate more dynamic string
1 !                                 space
1 !                 STR$_STRTOOLON  if string length is > 65535 for Class_D
1 !                 STR$_FATINTERR  if debug set in STRMACROS and
1 !                                 consistency error
1 !
1 !--
```

```
;  265           1455  1
;  266           1456  2      BEGIN
;  267           1457  2
;  268           1458  2      LOCAL
;  269           1459  2          OUT_LEN,                          ! length of destination string
;  270           1460  2          OUT_ADDR,                         ! addr of 1st byte of
;  271           1461  2                                            ! destination string
;  272           1462  2          RETURN_STATUS;                    ! keep track of status
;  273           1463  2
;  274           1464  2      MAP
;  275           1465  2          DEST_DESC : REF $STR$DESCRIPTOR;
;  276           1466  2
;  277           1467  2  !+
;  278           1468  2  ! Check for fatal error.
;  279           1469  2  !-
;  280           1470  2      IF .INPUT_LENGTH GTR 65535
;  281           1471  2      THEN LIB$STOP (STR$_STRTOOLON);
;  282           1472  2
;  283           1473  2  !+
;  284           1474  2  ! Initialize return status.
;  285           1475  2  !-
;  286           1476  2
;  287           1477  2      RETURN_STATUS = STR$_NORMAL ;
;  288           1478  2
;  289           1479  2      IF .INPUT_LENGTH LSS 0
;  290           1480  2      THEN RETURN_STATUS = STR$_NEGSTRLEN;
;  291           1481  2  !+
;  292           1482  2  ! Determine length and address of 1st byte of destination string.
;  293           1483  2  !-
;  294           1484  2      $STR$GET_LEN_ADDR ( DEST_DESC, OUT_LEN, OUT_ADDR ) ;
```

```
2 !+
2 ! algorithm differs based on the class of the destination string
2 !-
2
2     CASE .DEST_DESC [DSC$B_CLASS] FROM DSC$K_CLASS_Z TO DSC$K_CLASS_SB OF
2         SET
2
2 !+
2 ! Classes using fixed-length semantics.
2 ! ************************************
2 !-
2
2         [DSC$K_CLASS_Z,
2          DSC$K_CLASS_A,
2          DSC$K_CLASS_NCA,
2          DSC$K_CLASS_SD,
2          DSC$K_CLASS_S,
2          DSC$K_CLASS_SB] :
2
2             IF .OUT_LEN LEQ .INPUT_LENGTH        ! if requested length
2             THEN                                 ! >= string length
3                 BEGIN
3                 CH$FILL (.INPUT_CHAR,    ! just fill the string
3                     .OUT_LEN,            !  for entire length
3                     .OUT_ADDR);          !  from beginning of string
3
3                 IF .OUT_LEN LSS .INPUT_LENGTH    ! if truncation
3                 THEN
3                     RETURN_STATUS = STR$_TRU;    ! return status
3
3                 END
3
2             ELSE                                 ! else
2
2                 !+
2                 ! Pad with fill character after filling with requested
2                 ! character for requested length.
2                 !-
2                 CH$FILL (STR$K_FILL_CHAR,
2                          .OUT_LEN - MAX (0, .INPUT_LENGTH),
2                          CH$FILL (.INPUT_CHAR,
2                                   MAX (0, .INPUT_LENGTH),
2                                   .OUT_ADDR));
2
```

```
!+
! dynamic destination string
! **************************
!-

        [DSC$K_CLASS_D] :
            BEGIN

            IF $STR$NEED_ALLOC (MAX (0, .INPUT_LENGTH), ! if allocation
                    ($STR$DYN_AL_LEN (DEST_DESC)))      ! needed
            THEN
                BEGIN                       ! cannot fill dest directly

                LOCAL
                    ALLOCATE_STATUS,    ! get status from allocate
                    TEMP_DESC : $STR$DESCRIPTOR;! create temp descrip

                !+
                ! If the allocate succeeds then create the string in
                ! the temp, switch the temp and the destination and
                ! deallocate the former destination.
                ! If the allocate fails, then return the fatal error
                ! status.
                !-
                IF (ALLOCATE_STATUS = $STR$ALLOCATE (
                    MAX (0, .INPUT_LENGTH),     ! alloc space to temp
                    TEMP_DESC))
                THEN
                    BEGIN
                    !+
                    ! Fill temp with request for requested length
                    ! from beginning of string.
                    CH$FILL (.INPUT_CHAR,
                             MAX (0, .INPUT_LENGTH),
                             .TEMP_DESC [DSC$A_POINTER]);

                    !+
                    ! Switch temp and destination descriptors.
                    !-
                    $STR$EXCH_DESCS (TEMP_DESC, DEST_DESC);

                    !+
                    ! If the deallocate fails, return the error status.
                    !-
                    IF (NOT (ALLOCATE_STATUS =
                        $STR$DEALLOCATE (TEMP_DESC)))   ! return former
                                                        ! string
                    THEN RETURN_STATUS = .ALLOCATE_STATUS;
                    END

                ELSE

                    RETURN_STATUS = .ALLOCATE_STATUS;   ! allocate
                                                        ! failed
                END

            ELSE                                    ! else directly fill
```

```
;  398          1586  3                                       ! destination string
;  399          1587  4             BEGIN
;  400          1588  4             CH$FILL (.INPUT_CHAR,                  ! with requested char.
;  401          1589  4                     MAX (0, .INPUT_LENGTH),        ! for requested length
;  402          1590  4                     .OUT_ADDR);                    ! from start of string
;  403          1591  4
;  404          1592  4             DEST_DESC [DSC$W_LENGTH] = MAX (0, .INPUT_LENGTH);
;  405          1593  3             END;
;  406          1594  3
;  407          1595  2         END;
```

```
2 !+
2 ! Class_VS Varying string destination
2 ! **********************************
2 !-
2
2          [DSC$K_CLASS_VS]:
3              BEGIN
3              IF .INPUT_LENGTH LEQU .DEST_DESC [DSC$W_MAXSTRLEN]
3              THEN
4                  BEGIN           ! fits within MAXSTRLEN
4                  !+
4                  ! Fill up to .INPUT_LENGTH chars into destination.
4                  !-
4                  CH$FILL (.INPUT_CHAR,
4                           MAX ( 0, .INPUT_LENGTH),
4                           .OUT_ADDR);
4
4                  !+
4                  ! Reset CURLEN field to the number of characters copied
4                  !-
4                  (.DEST_DESC [DSC$A_POINTER])<0,16> =
4                          MAX ( 0, .INPUT_LENGTH) ;
4
4                  RETURN_STATUS = STR$_NORMAL ;
4
4                  END             ! fits within MAXSTRLEN
4
3              ELSE
3
4                  BEGIN           ! doesn't fit within MAXSTRLEN
4                  !+
4                  ! Fill up to MAXSTRLEN chars into destination.
4                  !-
4                  CH$FILL (.INPUT_CHAR,
4                           MAX ( 0, .DEST_DESC [DSC$W_MAXSTRLEN]),
4                           .OUT_ADDR) ;
4
4                  !+
4                  ! Reset CURLEN field to the number of characters copied
4                  !-
4                  (.DEST_DESC [DSC$A_POINTER])<0,16> =
4                          MAX ( 0, .DEST_DESC [DSC$W_MAXSTRLEN]) ;
4
4                  RETURN_STATUS = STR$_TRU ;
4
3                  END;            ! doesn't fit within MAXSTRLEN
3
2              END;
2
2 !+
2 ! other classes of descriptors
2 ! ****************************
2 !-
2
2          [INRANGE, OUTRANGE] : RETURN_STATUS = STR$_ILLSTRCLA;
2          TES;
2
```

```
;  466           1653  2      $STR$SIGNAL_FATAL (RETURN_STATUS);  ! Signal the fatal errors
;  467           1654  2      RETURN .RETURN_STATUS;
;  468           1655  1      END;                                           !End of STR$DUPL_CHARR8


                                                               .EXTRN  STR$ANALYZE_SDESC_R1
                                                               .EXTRN  STR$$INIT, STR$$V_INIT
                                                               .EXTRN  STR$$ALOC_SHORT
                                                               .EXTRN  STR$$Q_SHORT_Q, LIB$GET_VM
                                                               .EXTRN  STR$_INSVIRMEM, STR$$MOVQ_R1
                                                               .EXTRN  LIB$FREE_VM, STR$_FATINTERR

                         5E             18  C2 00000 STR$DUPL_CHARR8::
                                                               SUBL2   #24, SP                                    ; 1398
                                        52  DD 00003           PUSHL   R2                                         ;
                         58             51  D0 00005           MOVL    R1, R8                                     ;
                         56             50  D0 00008           MOVL    R0, R6                                     ;
               0000FFFF  8F             58  D1 0000B           CMPL    INPUT_LENGTH, #65535                       ; 1470
                         0D             15  00012              BLEQ    1$                                         ;
                           00000000G 8F     DD  00014          PUSHL   #STR$_STRTOOLON                            ; 1471
               00000000G 00             01  FB 0001A           CALLS   #1, LIB$STOP                               ;
                    08   AE 00000000G 8F    D0 00021 1$:       MOVL    #STR$_NORMAL, RETURN_STATUS                ; 1477
                                        58  D5 00029           TSTL    INPUT_LENGTH                               ; 1479
                                        08  18 0002B           BGEQ    2$                                         ;
                    08   AE 00000000G 8F    D0 0002D           MOVL    #STR$_NEGSTRLEN, RETURN_STATUS             ; 1480
                         02         03  A6  91 00035 2$:       CMPB    3(DEST_DESC), #2                           ; 1484
                                        0A  1A 00039           BGTRU   3$                                         ;
                         57             66  3C 0003B           MOVZWL  (DEST_DESC), OUT_LEN                       ;
                    04   AE         04  A6  D0 0003E           MOVL    4(DEST_DESC), OUT_ADDR                     ;
                                        10  11 00043           BRB     4$                                         ;
                         50             56  D0 00045 3$:       MOVL    DEST_DESC, R0                              ;
                           00000000G 00     16 00048           JSB     STR$ANALYZE_SDESC_R1                       ;
                         57             50  D0 0004E           MOVL    R0, R7                                     ;
                    04   AE             51  D0 00051           MOVL    R1, 4(SP)                                  ;
              0F         00         03  A6  8F 00055 4$:       CASEB   3(DEST_DESC), #0, #15                      ; 1489
    0020      0058       002A         002A     0005A 5$:       .WORD   7$-5$,-                                    ;
    0020      0020       0020         002A     00062                   7$-5$,-                                    ;
    01F9      002A       002A         0020     0006A                   11$-5$,-                                   ;
    002A      0020       0020         0020     00072                   6$-5$,-                                    ;
                                                                       7$-5$,-                                    ;
                                                                       6$-5$,-                                    ;
                                                                       6$-5$,-                                    ;
                                                                       6$-5$,-                                    ;
                                                                       6$-5$,-                                    ;
                                                                       7$-5$,-                                    ;
                                                                       7$-5$,-                                    ;
                                                                       42$-5$,-                                   ;
                                                                       6$-5$,-                                    ;
                                                                       6$-5$,-                                    ;
                                                                       6$-5$,-                                    ;
                                                                       7$-5$                                      ;
                    08   AE 00000000G 8F    D0 0007A 6$:       MOVL    #STR$_ILLSTRCLA, RETURN_STATUS             ; 1650
                                        2B  11 00082           BRB     10$                                        ;
                         58             57  D1 00084 7$:       CMPL    OUT_LEN, INPUT_LENGTH                      ; 1504
                                        0F  14 00087           BGTR    8$                                         ;
     57       6E         6E             00  2C 00089           MOVC5   #0, (SP), INPUT_CHAR, OUT_LEN, @OUT_ADDR   ; 1509
```

```
STR$DUPL_CHAR                                          D  8
1-010                                        16-Sep-1984 01:36:47    VAX-11 Bliss-32 V4.0-742              Page  14
                                             14-Sep-1984 12:40:05    [LIBRTL.SRC]STRDUPLCH.B32;1                 (7)

                                   04   BE       0008E                                                     ;
                          58            57   D1  00090          CMPL    OUT_LEN, INPUT_LENGTH              ; 1511
                                        1A   18  00093          BGEQ    10$                                ;
                                      01EC   31  00095          BRW     45$                                ; 1513
                          51            58   D0  00098  8$:     MOVL    INPUT_LENGTH, R1                   ; 1524
                                        02   18  0009B          BGEQ    9$                                 ;
                                        51   D4  0009D          CLRL    R1                                 ;
                          57            51   C2  0009F  9$:     SUBL2   R1, R7                             ;
      51          6E      6E            00   2C  000A2          MOVC5   #0, (SP), INPUT_CHAR, R1, @OUT_ADDR ; 1527
                                   04   BE       000A7                                                     ;
      57          20      6E            00   2C  000A9          MOVC5   #0, (SP), #32, R7, (R3)            ; 1525
                                        63       000AE                                                     ;
                                      01DA   31  000AF  10$:    BRW     46$                                ; 1504
                          57            58   D0  000B2  11$:    MOVL    INPUT_LENGTH, R7                   ; 1538
                                        02   18  000B5          BGEQ    12$                                ;
                                        57   D4  000B7          CLRL    R7                                 ;
                          51       04   A6   D0  000B9  12$:    MOVL    4(DEST_DESC), R1                   ;
                                        52   D4  000BD          CLRL    R2                                 ;
                                        51   D5  000BF          TSTL    R1                                 ;
                                        06   12  000C1          BNEQ    13$                                ;
                                        52   D6  000C3          INCL    R2                                 ;
                                        50   D4  000C5          CLRL    R0                                 ;
                                        13   11  000C7          BRB     15$                                ;
                  00F0    8F            66   B1  000C9  13$:    CMPW    (DEST_DESC), #240                  ;
                                        05   1B  000CE          BLEQU   14$                                ;
                          50            66   3C  000D0          MOVZWL  (DEST_DESC), R0                    ;
                                        07   11  000D3          BRB     15$                                ;
                          50            51   D0  000D5  14$:    MOVL    R1, STRING_BLOCK                   ;
                          50       FE   A0   3C  000D8          MOVZWL  -2(STRING_BLOCK), R0               ;
              000000F0    8F            50   D1  000DC  15$:    CMPL    R0, #240                           ;
                                        21   1F  000E3          BLSSU   19$                                ;
                          04            52   E9  000E5          BLBC    R2, 16$                            ;
                                        50   D4  000E8          CLRL    R0                                 ;
                                        13   11  000EA          BRB     18$                                ;
                  00F0    8F            66   B1  000EC  16$:    CMPW    (DEST_DESC), #240                  ;
                                        05   1B  000F1          BLEQU   17$                                ;
                          50            66   3C  000F3          MOVZWL  (DEST_DESC), R0                    ;
                                        07   11  000F6          BRB     18$                                ;
                          50            51   D0  000F8  17$:    MOVL    R1, STRING_BLOCK                   ;
                          50       FE   A0   3C  000FB          MOVZWL  -2(STRING_BLOCK), R0               ;
                          50            57   D1  000FF  18$:    CMPL    R7, R0                             ;
                                        21   13  00102          BEQL    23$                                ;
                                        22   11  00104          BRB     24$                                ;
                          04            52   E9  00106  19$:    BLBC    R2, 20$                            ;
                                        50   D4  00109          CLRL    R0                                 ;
                                        13   11  0010B          BRB     22$                                ;
                  00F0    8F            66   B1  0010D  20$:    CMPW    (DEST_DESC), #240                  ;
                                        05   1B  00112          BLEQU   21$                                ;
                          50            66   3C  00114          MOVZWL  (DEST_DESC), R0                    ;
                                        07   11  00117          BRB     22$                                ;
                          50            51   D0  00119  21$:    MOVL    R1, STRING_BLOCK                   ;
                          50       FE   A0   3C  0011C          MOVZWL  -2(STRING_BLOCK), R0               ;
                          50            57   D1  00120  22$:    CMPL    R7, R0                             ;
                                        03   1A  00123          BGTRU   24$                                ;
                                      011F   31  00125  23$:    BRW     41$                                ;
                          07 00000000G  00   E8  00128  24$:    BLBS    STR$$V_INIT, 25$                   ; 1555
             00000000G    00            00   FB  0012F          CALLS   #0, STR$$INIT                      ;
```

```
                        50 00000000G  8F  D0 00136 25$:   MOVL    #STR$_NORMAL, RETURN_STATUS               :
            000000F0    8F            57  D1 0013D        CMPL    R7, #240                                  :
                                      4D  1A 00144        BGTRU   33$                                       :
                                      57  D5 00146        TSTL    R7                                        :
                                      04  12 00148        BNEQ    26$                                       :
                                      53  D4 0014A        CLRL    TEMP                                      :
                                      31  11 0014C        BRB     31$                                       :
                        51        FF  A7  9E 0014E 26$:   MOVAB   -1(R7), R1                                :
                        51            07  8A 00152        BICB2   #7, R1                                    :
                        54 00000000G0041  9E 00155        MOVAB   STR$$Q_SHORT_Q[R1], REMQUE_ADDR           :
                        53        00  B4  0F 0015D 27$:   REMQUE  @0(REMQUE_ADDR), TEMP                     :
                                      05  1D 00161        BVS     28$                                       :
                        52            01  D0 00163        MOVL    #1, ALLOC_DONE                            :
                                      0F  11 00166        BRB     30$                                       :
                                      52  D4 00168 28$:   CLRL    ALLOC_DONE                                :
                                      58  DD 0016A        PUSHL   INPUT_LENGTH                              :
                                      02  18 0016C        BGEQ    29$                                       :
                                      6E  D4 0016E        CLRL    (SP)                                      :
            00000000G   00            01  FB 00170 29$:   CALLS   #1, STR$$ALOC_SHORT                       :
                        05            52  E8 00177 30$:   BLBS    ALLOC_DONE, 31$                           :
                        37            50  E9 0017A        BLBC    RETURN_STATUS, 35$                        :
                                      DE  11 0017D        BRB     27$                                       :
                        32            50  E9 0017F 31$:   BLBC    RETURN_STATUS, 35$                        :
                  18    AE            53  D0 00182        MOVL    TEMP, TEMP_DESC+4                         :
                        51            58  D0 00186        MOVL    INPUT_LENGTH, R1                          :
                                      02  18 00189        BGEQ    32$                                       :
                                      51  D4 0018B        CLRL    R1                                        :
                  14    AE            51  B0 0018D 32$:   MOVW    R1, TEMP_DESC                             :
                                      21  11 00191        BRB     35$                                       :
                                  18  AE  9F 00193 33$:   PUSHAB  TEMP_DESC+4                               :
                  08    AE            57  D0 00196        MOVL    R7, 8(SP)                                 :
                                  08  AE  9F 0019A        PUSHAB  8(SP)                                     :
            00000000G   00            02  FB 0019D        CALLS   #2, LIB$GET_VM                            :
                        09            50  E8 001A4        BLBS    RETURN_STATUS, 34$                        :
                        50 00000000G  8F  D0 001A7        MOVL    #STR$_INSVIRMEM, RETURN_STATUS            :
                                      04  11 001AE        BRB     35$                                       :
                  14    AE            57  B0 001B0 34$:   MOVW    R7, TEMP_DESC                             :
                        57            50  D0 001B4 35$:   MOVL    RETURN_STATUS, ALLOCATE_STATUS            :
                        03            50  E8 001B7        BLBS    RETURN_STATUS, 36$                        :
                                    0084  31 001BA        BRW     40$                                       :
                        51            58  D0 001BD 36$:   MOVL    INPUT_LENGTH, R1                          : 1562
                                      02  18 001C0        BGEQ    37$                                       :
                                      51  D4 001C2        CLRL    R1                                        :
  51            6E      6E            00  2C 001C4 37$:   MOVC5   #0, (SP), INPUT_CHAR, R1, @TEMP_DESC+4    : 1563
                                  18  BE     001C9                                                      :
                  0C    AE            66  B0 001CB        MOVW    (DEST_DESC), $STR$TEMP_DESC               : 1568
                  10    AE        04  A6  D0 001CF        MOVL    4(DEST_DESC), $STR$TEMP_DESC+4            :
                  16    AE        02  A6  B0 001D4        MOVW    2(DEST_DESC), TEMP_DESC+2                 :
                        50        14  AE  9E 001D9        MOVAB   TEMP_DESC, R0                             :
                        51            56  D0 001DD        MOVL    DEST_DESC, R1                             :
                           00000000G  00  16 001E0        JSB     STR$$MOVQ_R1                              :
                  14    AE        0C  AE  B0 001E6        MOVW    $STR$TEMP_DESC, TEMP_DESC                 :
                  18    AE        10  AE  D0 001EB        MOVL    $STR$TEMP_DESC+4, TEMP_DESC+4             :
                        50 00000000G  8F  D0 001F0        MOVL    #STR$_NORMAL, RETURN_STATUS               : 1574
                        52        18  AE  D0 001F7        MOVL    TEMP_DESC+4, R2                           :
                                      3E  13 001FB        BEQL    39$                                       :
                00F0    8F        14  AE  B1 001FD        CMPW    TEMP_DESC, #240                           :
```

```
                                    1A    1A 00203         BGTRU   38$                                           ;
                      51            52    D0 00205         MOVL    R2, STRING_BLOCK                              ;
                      51      FE    A1    3C 00208         MOVZWL  -2(STRING_BLOCK), ALLOC_LENGTH                ;
                                    51    D7 0020C         DECL    R1                                            ;
                      51            07    8A 0020E         BICB2   #7, R1                                        ;
                      51 00000000G0041    9E 00211         MOVAB   STR$$Q_SHORT_Q[R1], INSQUE_ADDR               ;
                  00  B1            62    0E 00219         INSQUE  (R2), @0(INSQUE_ADDR)                         ;
                                    1C    11 0021D         BRB     39$                                           ;
                              18    AE    9F 0021F  38$:   PUSHAB  TEMP_DESC+4                                   ;
                  08  AE          18    AE    3C 00222         MOVZWL  TEMP_DESC, 8(SP)                      ;
                              08    AE    9F 00227         PUSHAB  8(SP)                                         ;
           00000000G  00            02    FB 0022A         CALLS   #2, LIB$FREE_VM                               ;
                      07            50    E8 00231         BLBS    RETURN_STATUS, 39$                            ;
                      50 00000000G  8F    D0 00234         MOVL    #STR$_FATINTERR, RETURN_STATUS                ;
                      57            50    D0 0023B  39$:   MOVL    RETURN_STATUS, ALLOCATE_STATUS                ;
                      4B            50    E8 0023E         BLBS    RETURN_STATUS, 46$                            ;
                  08  AE            57    D0 00241  40$:   MOVL    ALLOCATE_STATUS, RETURN_STATUS                ; 1581
                                    45    11 00245         BRB     46$                                           ; 1537
      57         6E       6E            00    2C 00247  41$:   MOVC5   #0, (SP), INPUT_CHAR, R7, @OUT_ADDR      ; 1590
                              04    BE       0024C                                                           ;
                      66            57    B0 0024E         MOVW    R7, (DEST_DESC)                               ; 1592
                                    39    11 00251         BRB     46$                                           ; 1489
      58         66       10            00    ED 00253  42$:   CMPZV   #0, #16, (DEST_DESC), INPUT_LENGTH       ; 1603
                                    1C    1F 00258         BLSSU   44$                                           ;
                      57            58    D0 0025A         MOVL    INPUT_LENGTH, R7                              ; 1610
                                    02    18 0025D         BGEQ    43$                                           ;
                                    57    D4 0025F         CLRL    R7                                            ;
      57         6E       6E            00    2C 00261  43$:   MOVC5   #0, (SP), INPUT_CHAR, R7, @OUT_ADDR      ; 1611
                              04    BE       00266                                                           ;
                  04  B6            57    B0 00268         MOVW    R7, @4(DEST_DESC)                             ; 1617
                  08  AE 00000000G  8F    D0 0026C         MOVL    #STR$_NORMAL, RETURN_STATUS                   ; 1619
                                    16    11 00274         BRB     46$                                           ; 1603
                      57            66    3C 00276  44$:   MOVZWL  (DEST_DESC), R7                               ; 1630
      57         6E       6E            00    2C 00279         MOVC5   #0, (SP), INPUT_CHAR, R7, @OUT_ADDR      ; 1631
                              04    BE       0027E                                                           ;
                  04  B6            57    B0 00280         MOVW    R7, @4(DEST_DESC)                             ; 1637
                  08  AE 00000000G  8F    D0 00284  45$:   MOVL    #STR$_TRU, RETURN_STATUS                      ; 1639
                      12      08    AE    E8 0028C  46$:   BLBS    RETURN_STATUS, 47$                            ; 1653
      04      08  AE          03            00    ED 00290         CMPZV   #0, #3, RETURN_STATUS, #4            ;
                                    0A    12 00296         BNEQ    47$                                           ;
                              08    AE    DD 00298         PUSHL   RETURN_STATUS                                 ;
           00000000G  00            01    FB 0029B         CALLS   #1, LIB$STOP                                  ;
                      50      08    AE    D0 002A2  47$:   MOVL    RETURN_STATUS, R0                             ; 1654
                      5E            1C    C0 002A6         ADDL2   #28, SP                                       ; 1655
                                          05 002A9         RSB                                                   ;

; Routine Size:  682 bytes,    Routine Base:  _STR$CODE + 0033


;   469          1656  1
;   470          1657  1 END                                          !End of module
;   471          1658  1
;   472          1659  0 ELUDOM
;                                     PSECT SUMMARY
;
;       Name                      Bytes                        Attributes
;
;  _STR$CODE                          733  NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)


;                          Library Statistics
;
;                                            -------- Symbols --------       Pages       Processing
;       File                                 Total    Loaded   Percent       Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1          9776        15         0        581          00:00.8




;                                     COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:STRDUPLCH/OBJ=OBJ$:STRDUPLCH MSRC$:STRDUPLCH/UPDATE=(ENH$:STRDUPLCH
;       )

; Size:          733 code + 0 data bytes
; Run Time:         00:11.7
; Elapsed Time:     00:54.1
; Lines/CPU Min:     8507
; Lexemes/CPU-Min: 34128
; Memory Used:  211 pages
; Compilation Complete
```

0214 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY