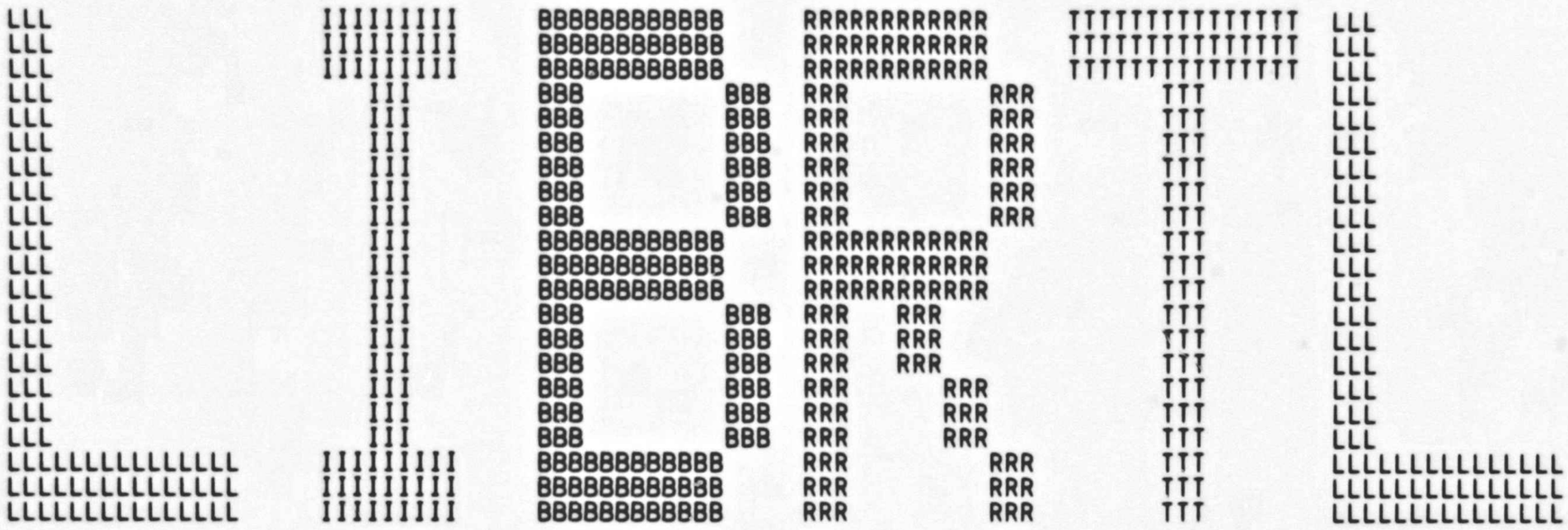
LLL IIIIIIIII BBBBBBBBBBB RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
LLL IIIIIIIII BBBBBBBBBBBB RRRRRRRRRRRR TTTTTTTTTTTTTTT LLL
LLL IIIIIIIII BBBBBBBBBBBB RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBBBBBBBBBB RRRRRRRRRRR 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
LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBBB RRR RRR TTT LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBBB RRR RRR TTT LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBBB RRR RRR TTT LLLLLLLLLLLLLLL

FILEID**STRDUPLCH

```
  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
```

```
;     1          0001  0 MODULE STR$DUPL_CHAR (          ! Duplicate a character in a string
;     2          0002  0
;     3          0003  0                 IDENT = '1-010' ! File: STRDUPLCH.B32   Edit: DG1010
;     4          0004  0
;     5          0005  0                      ) =
;     6          0006  1 BEGIN
;     7          0007  1 !
;     8          0008  1 !**************************************************************************
;     9          0009  1 !*                                                                        *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                   *
;    13          0013  1 !*                                                                        *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    19          0019  1 !*  TRANSFERRED.                                                           *
;    20          0020  1 !*                                                                        *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    23          0023  1 !*  CORPORATION.                                                           *
;    24          0024  1 !*                                                                        *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
;    27          0027  1 !*                                                                        *
;    28          0028  1 !*                                                                        *
;    29          0029  1 !**************************************************************************
;    30          0030  1 !
;    31          0031  1
;    32          0032  1 !++
;    33          0033  1 ! FACILITY: String support library
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       This routine fills a string with an input number (defaults to
;    38          0038  1 !       1) of an input character (defaults to space).
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT: User mode, AST level or not or mixed
;    41          0041  1 !
;    42          0042  1 ! AUTHOR: R. Will,  CREATION DATE: 13-Mar-79
;    43          0043  1 !
;    44          0044  1 ! MODIFIED BY:
;    45          0045  1 !
;    46          0046  1 !       R. Will, 13-Mar-79: VERSION 01
;    47          0047  1 ! 1-001 - Original
;    48          0048  1 ! 1-002 - Use STR$K_FILL_CHAR.  JBS 15-APR-1979
;    49          0049  1 ! 1-003 - String cleanup.  Change name to STR$.  RW  8-Nov-79
;    50          0050  1 ! 1-004 - Don't use the string interlock macros from JSB entry
;    51          0051  1 !         points.  JBS 15-NOV-1979
;    52          0052  1 ! 1-005 - String speedup.  RW  7-Jan-1980
;    53          0053  1 ! 1-006 - Enhance to accomodate additional classes of destination
;    54          0054  1 !         descriptors by using $STR$GET_LEN_ADDR to extract length
;    55          0055  1 !         and address of 1st data byte indicated by descriptor.
;    56          0056  1 !         Remove string interlocking code.
;    57          0057  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>
```

```
;    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
```

```
;  132          1323  1 GLOBAL ROUTINE STR$DUPL_CHAR (          ! Create a string of a char
;  133          1324  1
;  134          1325  1         DEST_DESC,                      ! Pointer to dest str desc
;  135          1326  1         INPUT_LENGTH,                   ! Number of characters
;  136          1327  1         INPUT_CHAR                      ! Character to duplicate
;  137          1328  1                               ) : =
;  138          1329  1
;  139          1330  1 !++
;  140          1331  1 ! FUNCTIONAL DESCRIPTION:
;  141          1332  1 !
;  142          1333  1 !       This routine writes LENGTH characters of CHAR into the string
;  143          1334  1 !       pointer to by DEST_DESC.  If the destination is a fixed length
;  144          1335  1 !       string, and LENGTH is greater than the length of the string,
;  145          1336  1 !       only as many CHARs as will fit are copied.  If destination is
;  146          1337  1 !       fixed length and LENGTH is less than the destination string
;  147          1338  1 !       length then LENGTH CHARs are copied and the destination is
;  148          1339  1 !       padded with blanks.  If the destination is a dynamic string,
;  149          1340  1 !       after execution of this routine the destination will have a
;  150          1341  1 !       length of LENGTH.
;  151          1342  1 !       If the destination has varying string semantics and the LENGTH
;  152          1343  1 !       exceeds MAXSTRLEN, STR$_TRU is returned.
;  153          1344  1 !       The call entry point is implemented by
;  154          1345  1 !       JSBing to the JSB entry point.
;  155          1346  1 !
;  156          1347  1 ! FORMAL PARAMETERS:
;  157          1348  1 !
;  158          1349  1 !       DEST_DESC.wt.dx         pointer to destination string descriptor
;  159          1350  1 !       INPUT_LENGTH.rl.r       number of characters to duplicate
;  160          1351  1 !       INPUT_CHAR.rbu.r        ASCII character to duplicate
;  161          1352  1 !
;  162          1353  1 ! IMPLICIT INPUTS:
;  163          1354  1 !
;  164          1355  1 !       NONE
;  165          1356  1 !
;  166          1357  1 ! IMPLICIT OUTPUTS:
;  167          1358  1 !
;  168          1359  1 !       NONE
;  169          1360  1 !
;  170          1361  1 ! COMPLETION CODES:
;  171          1362  1 !
;  172          1363  1 !       same as STR$DUPL_CHARR8
;  173          1364  1 !
;  174          1365  1 ! SIDE EFFECTS:
;  175          1366  1 !
;  176          1367  1 !       same as STR$DUPL_CHARR8
;  177          1368  1 !
;  178          1369  1 !--
;  179          1370  1
;  180          1371  2     BEGIN
;  181          1372  2
;  182          1373  2     BUILTIN
;  183          1374  2         NULLPARAMETER;                  ! check for optional args
;  184          1375  2
;  185          1376  2     LOCAL
;  186          1377  2         CHAR : BYTE,                        ! character to use
;  187          1378  2         LENGTH;                             ! length to use
;  188          1379  2
```

```
;  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
```

```
;  208     1398  1 GLOBAL ROUTINE STR$DUPL_CHARR8 (          ! Create a string of a char
;  209     1399  1
;  210     1400  1         DEST_DESC,                        ! Pointer to dest str desc
;  211     1401  1         INPUT_LENGTH,                     ! Number of characters
;  212     1402  1         INPUT_CHAR                        ! Character to duplicate
;  213     1403  1
;  214     1404  1                                 ) : STR$JSB_DUPL_CH =
;  215     1405  1
;  216     1406  1 !++
;  217     1407  1 ! FUNCTIONAL DESCRIPTION:
;  218     1408  1 !
;  219     1409  1 !       This routine writes LENGTH characters of CHAR into the string
;  220     1410  1 !       pointer to by DEST_DESC.  If the destination is a fixed length
;  221     1411  1 !       string, and LENGTH is greater than the length of the string,
;  222     1412  1 !       only as many CHARs as will fit are copied.  If destination is
;  223     1413  1 !       fixed length and LENGTH is less than the destination string
;  224     1414  1 !       length then LENGTH CHARs are copied and the destination is
;  225     1415  1 !       padded with blanks.  If the destination is a dynamic string,
;  226     1416  1 !       after execution of this routine the destination will have a
;  227     1417  1 !       length of LENGTH.
;  228     1418  1 !       If the destination has varying string semantics and the LENGTH
;  229     1419  1 !       exceeds MAXSTRLEN, STR$_TRU is returned.
;  230     1420  1 !
;  231     1421  1 ! FORMAL PARAMETERS:
;  232     1422  1 !
;  233     1423  1 !       DEST_DESC.wt.dx         pointer to destination string descriptor
;  234     1424  1 !       INPUT_LENGTH.rl.v       value of no. of characters to duplicate
;  235     1425  1 !       INPUT_CHAR.rbu.v        value of ASCII character to duplicate
;  236     1426  1 !
;  237     1427  1 ! IMPLICIT INPUTS:
;  238     1428  1 !
;  239     1429  1 !       NONE
;  240     1430  1 !
;  241     1431  1 ! IMPLICIT OUTPUTS:
;  242     1432  1 !
;  243     1433  1 !       NONE
;  244     1434  1 !
;  245     1435  1 ! COMPLETION CODES:
;  246     1436  1 !
;  247     1437  1 !       STR$_NORMAL     if successful completion
;  248     1438  1 !       STR$_NEGSTRLEN  if string length is negative
;  249     1439  1 !       STR$_TRU        if input length is greater than fixed string
;  250     1440  1 !                       length or length greater than MAXSTRLEN for
;  251     1441  1 !                       varying string destination
;  252     1442  1 !
;  253     1443  1 ! SIDE EFFECTS:
;  254     1444  1 !
;  255     1445  1 !       may allocate or deallocate dynamic string space
;  256     1446  1 !       may signal errors
;  257     1447  1 !               STR$_ILLSTRCLA  if not supported string class
;  258     1448  1 !               STR$_INSVIRMEM  if can't allocate more dynamic string
;  259     1449  1 !                               space
;  260     1450  1 !               STR$_STRTOOLON  if string length is > 65535 for Class_D
;  261     1451  1 !               STR$_FATINTERR  if debug set in STRMACROS and
;  262     1452  1 !                               consistency error
;  263     1453  1 !
;  264     1454  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 ) ;
```

```
;  296     1485  2 !+
;  297     1486  2 ! algorithm differs based on the class of the destination string
;  298     1487  2 !-
;  299     1488  2
;  300     1489  2     CASE .DEST_DESC [DSC$B_CLASS] FROM DSC$K_CLASS_Z TO DSC$K_CLASS_SB OF
;  301     1490  2         SET
;  302     1491  2
;  303     1492  2 !+
;  304     1493  2 ! Classes using fixed-length semantics.
;  305     1494  2 ! ************************************
;  306     1495  2 !-
;  307     1496  2
;  308     1497  2         [DSC$K_CLASS_Z,
;  309     1498  2          DSC$K_CLASS_A,
;  310     1499  2          DSC$K_CLASS_NCA,
;  311     1500  2          DSC$K_CLASS_SD,
;  312     1501  2          DSC$K_CLASS_S,
;  313     1502  2          DSC$K_CLASS_SB] :
;  314     1503  2
;  315     1504  2             IF .OUT_LEN LEQ .INPUT_LENGTH       ! if requested length
;  316     1505  2             THEN                                ! >= string length
;  317     1506  2                 BEGIN
;  318     1507  2                 CH$FILL (.INPUT_CHAR,   ! just fill the string
;  319     1508  2                     .OUT_LEN,           !  for entire length
;  320     1509  2                     .OUT_ADDR);         !  from beginning of string
;  321     1510  2
;  322     1511  2                 IF .OUT_LEN LSS .INPUT_LENGTH   ! if truncation
;  323     1512  2                 THEN
;  324     1513  2                     RETURN_STATUS = STR$_TRU;   ! return status
;  325     1514  2
;  326     1515  2                 END
;  327     1516  3
;  328     1517  2             ELSE                                ! else
;  329     1518  2
;  330     1519  2                 !+
;  331     1520  2                 ! Pad with fill character after filling with requested
;  332     1521  2                 ! character for requested length.
;  333     1522  2                 !-
;  334     1523  2                 CH$FILL (STR$K_FILL_CHAR,
;  335     1524  2                          .OUT_LEN - MAX (0, .INPUT_LENGTH),
;  336     1525  2                          CH$FILL (.INPUT_CHAR,
;  337     1526  2                                   MAX (0, .INPUT_LENGTH),
;  338     1527  2                                   .OUT_ADDR));
;  339     1528  2
```

```
;  341          1529  2 !+
;  342          1530  2 ! dynamic destination string
;  343          1531  2 ! **************************
;  344          1532  2 !-
;  345          1533  2
;  346          1534  2         [DSC$K_CLASS_D] :
;  347          1535  3             BEGIN
;  348          1536  3
;  349        P 1537  3             IF $STR$NEED_ALLOC (MAX (0, .INPUT_LENGTH), ! if allocation
;  350          1538  4                 ($STR$DYN_AL_LEN (DEST_DESC)))      ! needed
;  351          1539  3             THEN
;  352          1540  4                 BEGIN                       ! cannot fill dest directly
;  353          1541  4
;  354          1542  4                 LOCAL
;  355          1543  4                     ALLOCATE_STATUS,    ! get status from allocate
;  356          1544  4                     TEMP_DESC : $STR$DESCRIPTOR;! create temp descrip
;  357          1545  4
;  358          1546  4                 !+
;  359          1547  4                 ! If the allocate succeeds then create the string in
;  360          1548  4                 ! the temp, switch the temp and the destination and
;  361          1549  4                 ! deallocate the former destination.
;  362          1550  4                 ! If the allocate fails, then return the fatal error
;  363          1551  4                 ! status.
;  364          1552  4                 !-
;  365        P 1553  5                 IF (ALLOCATE_STATUS = $STR$ALLOCATE (
;  366        P 1554  5                     MAX (0, .INPUT_LENGTH),     ! alloc space to temp
;  367          1555  5                     TEMP_DESC))
;  368          1556  4                 THEN
;  369          1557  5                     BEGIN
;  370          1558  5                     !+
;  371          1559  5                     ! Fill temp with request for requested length
;  372          1560  5                     ! from beginning of string.
;  373          1561  5                     CH$FILL (.INPUT_CHAR,
;  374          1562  5                              MAX (0, .INPUT_LENGTH),
;  375          1563  5                              .TEMP_DESC [DSC$A_POINTER]);
;  376          1564  5
;  377          1565  5                     !+
;  378          1566  5                     ! Switch temp and destination descriptors.
;  379          1567  5                     !-
;  380          1568  5                     $STR$EXCH_DESCS (TEMP_DESC, DEST_DESC);
;  381          1569  5
;  382          1570  5                     !+
;  383          1571  5                     ! If the deallocate fails, return the error status.
;  384          1572  5                     !-
;  385          1573  7                     IF (NOT (ALLOCATE_STATUS =
;  386          1574  6                         $STR$DEALLOCATE (TEMP_DESC)))   ! return former
;  387          1575  6                                                         ! string
;  388          1576  5                     THEN RETURN_STATUS = .ALLOCATE_STATUS;
;  389          1577  5                     END
;  390          1578  5
;  391          1579  4                 ELSE
;  392          1580  4
;  393          1581  4                     RETURN_STATUS = .ALLOCATE_STATUS;   ! allocate
;  394          1582  4                                                         ! failed
;  395          1583  4                 END
;  396          1584  4
;  397          1585  3             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;
```

```
;  409     1596  2 !+
;  410     1597  2 ! Class_VS Varying string destination
;  411     1598  2 ! ************************************
;  412     1599  2 !-
;  413     1600  2
;  414     1601  2         [DSC$K_CLASS_VS]:
;  415     1602  3             BEGIN
;  416     1603  3             IF .INPUT_LENGTH LEQU .DEST_DESC [DSC$W_MAXSTRLEN]
;  417     1604  3             THEN
;  418     1605  4                 BEGIN           ! fits within MAXSTRLEN
;  419     1606  4                 !+
;  420     1607  4                 ! Fill up to .INPUT_LENGTH chars into destination.
;  421     1608  4                 !-
;  422     1609  4                 CH$FILL (.INPUT_CHAR,
;  423     1610  4                          MAX ( 0, .INPUT_LENGTH),
;  424     1611  4                          .OUT_ADDR);
;  425     1612  4
;  426     1613  4                 !+
;  427     1614  4                 ! Reset CURLEN field to the number of characters copied
;  428     1615  4                 !-
;  429     1616  4                 (.DEST_DESC [DSC$A_POINTER])<0,16> =
;  430     1617  4                         MAX ( 0, .INPUT_LENGTH) ;
;  431     1618  4
;  432     1619  4                 RETURN_STATUS = STR$_NORMAL ;
;  433     1620  4
;  434     1621  4                 END             ! fits within MAXSTRLEN
;  435     1622  4
;  436     1623  3             ELSE
;  437     1624  3
;  438     1625  4                 BEGIN           ! doesn't fit within MAXSTRLEN
;  439     1626  4                 !+
;  440     1627  4                 ! Fill up to MAXSTRLEN chars into destination.
;  441     1628  4                 !-
;  442     1629  4                 CH$FILL (.INPUT_CHAR,
;  443     1630  4                          MAX ( 0, .DEST_DESC [DSC$W_MAXSTRLEN]),
;  444     1631  4                          .OUT_ADDR) ;
;  445     1632  4
;  446     1633  4                 !+
;  447     1634  4                 ! Reset CURLEN field to the number of characters copied
;  448     1635  4                 !-
;  449     1636  4                 (.DEST_DESC [DSC$A_POINTER])<0,16> =
;  450     1637  4                         MAX ( 0, .DEST_DESC [DSC$W_MAXSTRLEN]) ;
;  451     1638  4
;  452     1639  4                 RETURN_STATUS = STR$_TRU ;
;  453     1640  4
;  454     1641  3                 END;            ! doesn't fit within MAXSTRLEN
;  455     1642  3
;  456     1643  2             END;
;  457     1644  2
;  458     1645  2 !+
;  459     1646  2 ! other classes of descriptors
;  460     1647  2 ! ****************************
;  461     1648  2 !-
;  462     1649  2
;  463     1650  2         [INRANGE, OUTRANGE] : RETURN_STATUS = STR$_ILLSTRCLA;
;  464     1651  2         TES;
;  465     1652  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$-5$
                04  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
```

```
                                  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 00000000G00041   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   BA 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
```

STRDUPLCH
LIS

STRCOMPAR
LIS

STRFINDSB
LIS

STRMATCH
LIS

STRMSG
LIS

STRLENEXT
LIS

STRCOPY
LIS

STRFINDFI
LIS

STRLEFT
LIS

STRMULTI
LIS

STRCOMEQL
LIS

STRCONCAT
LIS

STRMOVQ
LIS

STRGETFRE
LIS