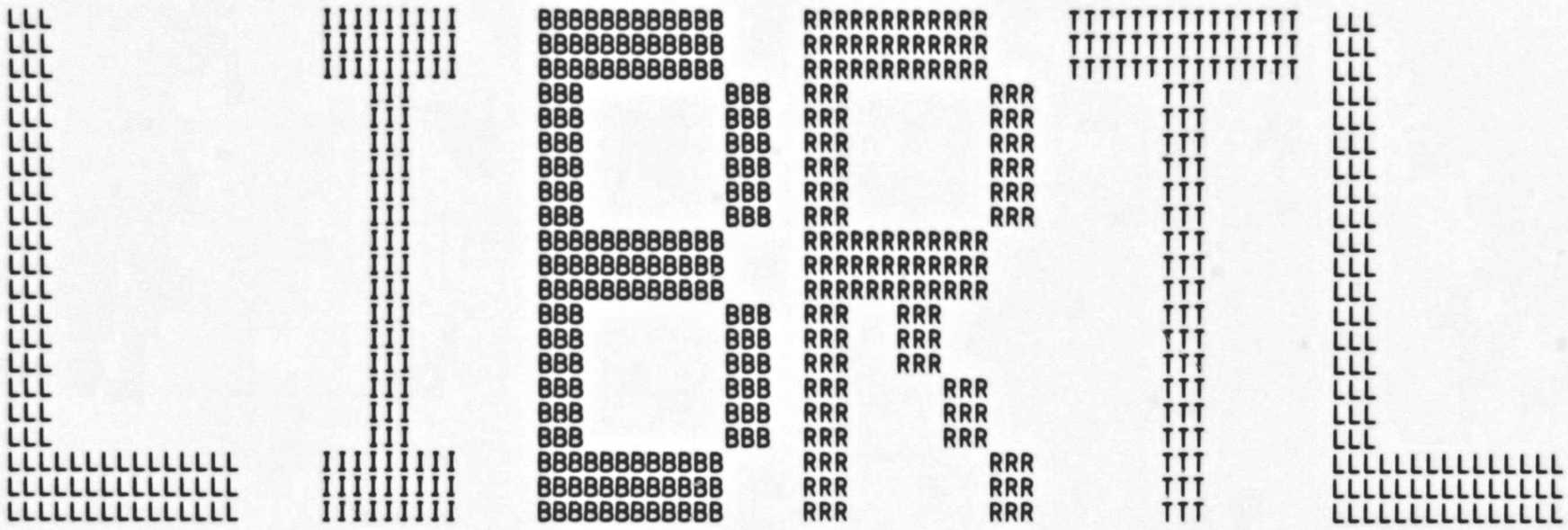
LLL IIIIIIIII BBBBBBBBBBB RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
LLL IIIIIIIII BBBBBBBBBBBB RRRRRRRRRRRR TTTTTTTTTTTTTTT LLL
LLL IIIIIIIII BBBBBBBBBBBB RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBB BBB RRR RRR TTT LLL
LLL III BBB BBB RRR RRR TTT 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**STRPREFIX

```
  SSSSSSSS  TTTTTTTTTT  RRRRRRRR    PPPPPPPP    RRRRRRRR    EEEEEEEEEE  FFFFFFFFFF    IIIIII    XX      XX
  SSSSSSSS  TTTTTTTTTT  RRRRRRRR    PPPPPPPP    RRRRRRRR    EEEEEEEEEE  FFFFFFFFFF    IIIIII    XX      XX
SS              TT      RR      RR  PP      PP  RR      RR  EE          FF             II      XX      XX
SS              TT      RR      RR  PP      PP  RR      RR  EE          FF             II      XX      XX
SS              TT      RR      RR  PP      PP  RR      RR  EE          FF             II        XX  XX
SS              TT      RR      RR  PP      PP  RR      RR  EE          FF             II        XX  XX
  SSSSSS        TT      RRRRRRRR    PPPPPPPP    RRRRRRRR    EEEEEEEE    FFFFFFFF       II          XX
  SSSSSS        TT      RRRRRRRR    PPPPPPPP    RRRRRRRR    EEEEEEEE    FFFFFFFF       II          XX
        SS      TT      RR  RR      PP          RR  RR      EE          FF             II        XX  XX
        SS      TT      RR  RR      PP          RR  RR      EE          FF             II        XX  XX
        SS      TT      RR    RR    PP          RR    RR    EE          FF             II      XX      XX      ....
        SS      TT      RR    RR    PP          RR    RR    EE          FF             II      XX      XX      ....
SSSSSSSS        TT      RR      RR  PP          RR      RR  EEEEEEEEEE  FF         IIIIII    XX      XX      ....
SSSSSSSS        TT      RR      RR  PP          RR      RR  EEEEEEEEEE  FF         IIIIII    XX      XX      ....

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$PREFIX (! Prefix a string to the beginning of the destination
;     2          0002  0
;     3          0003  0                 IDENT = '1-007' ! File: STRPREFIX.B32   Edit: DG1007
;     4          0004  0
;     5          0005  0                   ) =
;     6          0006  1 BEGIN
;     7          0007  1
;     8          0008  1 !
;     9          0009  1 !*****************************************************************************
;    10          0010  1 !*                                                                           *
;    11          0011  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    12          0012  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    13          0013  1 !*  ALL RIGHTS RESERVED.                                                      *
;    14          0014  1 !*                                                                           *
;    15          0015  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    16          0016  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    17          0017  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    18          0018  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    19          0019  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    20          0020  1 !*  TRANSFERRED.                                                              *
;    21          0021  1 !*                                                                           *
;    22          0022  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    23          0023  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    24          0024  1 !*  CORPORATION.                                                              *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    27          0027  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*                                                                           *
;    30          0030  1 !*****************************************************************************
;    31          0031  1 !
;    32          0032  1
;    33          0033  1 !++
;    34          0034  1 ! FACILITY: String support library
;    35          0035  1 !
;    36          0036  1 ! ABSTRACT:
;    37          0037  1 !       This routine prefixes the input string onto the beginning of the
;    38          0038  1 !       the destination string.  It will handle strings of any supported
;    39          0039  1 !       dtype or class.
;    40          0040  1 !
;    41          0041  1 ! ENVIRONMENT: User mode, AST level or not or mixed
;    42          0042  1 !
;    43          0043  1 ! AUTHOR: R. Will, CREATION DATE: 1-Dec-79
;    44          0044  1 !
;    45          0045  1 ! MODIFIED BY:
;    46          0046  1 !
;    47          0047  1 ! R. Will, 1-Dec-79 : VERSION 01
;    48          0048  1 ! 1-001 - Original
;    49          0049  1 ! 1-002 - String speedup, status from macros.  RW  11-Jan-1980
;    50          0050  1 ! 1-003 - Enhance to recognize additional classes of descriptors by
;    51          0051  1 !         using $STR$GET_LEN_ADDR to extract length and address
;    52          0052  1 !         of 1st byte of data of source string.  Remove string
;    53          0053  1 !         interlocking code.    RKR 22-APR-81.
;    54          0054  1 ! 1-004 - Fix bug in code where class_vs destination must be truncated.
;    55          0055  1 !         (non-overlap case).
;    56          0056  1 !         RKR 25-AUG-1981
;    57          0057  1 ! 1-005 - Speed up code.  RKR 7-OCT-1981.
```

```
;   58          0058  1 ! 1-006 - Add support for class SO string descriptors.  DG 3-Oct-1983.
;   59          0059  1 ! 1-007 - Change class SO string descriptors to SB.  DG 27-Feb-1984.
;   60          0060  1 !--
;   61          0061  1 !<BLF/PAGE>
```

```
;    63          0062  1 !
;    64          0063  1 ! SWITCHES:
;    65          0064  1 !
;    66          0065  1
;    67          0066  1 SWITCHES ADDRESSING_MODE
;    68          0067  1                 (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    69          0068  1
;    70          0069  1 !
;    71          0070  1 ! LINKAGES:
;    72          0071  1 !
;    73          0072  1
;    74          0073  1 REQUIRE 'RTLIN:STRLNK';         ! Use require file with string linkages
;    75          0258  1
;    76          0259  1 !
;    77          0260  1 ! TABLE OF CONTENTS:
;    78          0261  1 !
;    79          0262  1
;    80          0263  1 FORWARD ROUTINE
;    81          0264  1     STR$PREFIX;                 ! prefix the input string to the
;    82          0265  1                                 ! beginning of the destination string
;    83          0266  1
;    84          0267  1 !
;    85          0268  1 ! INCLUDE FILES:
;    86          0269  1 !
;    87          0270  1
;    88          0271  1 REQUIRE 'RTLIN:RTLPSECT';       ! Declare PSECTS code
;    89          0366  1 REQUIRE 'RTLIN:STRMACROS';      ! use string macros to write code
;    90          1282  1 LIBRARY 'RTLSTARLE';            ! STARLET library for macros and symbols
;    91          1283  1
;    92          1284  1 !
;    93          1285  1 ! MACROS : NONE
;    94          1286  1 !
;    95          1287  1
;    96          1288  1 !
;    97          1289  1 ! EQUATED SYMBOLS: NONE
;    98          1290  1 !
;    99          1291  1
;   100          1292  1 !
;   101          1293  1 ! PSECT DECLARATIONS
;   102          1294  1 !
;   103          1295  1
;   104          1296  1 DECLARE_PSECTS (STR);
;   105          1297  1
;   106          1298  1 !
;   107          1299  1 ! OWN STORAGE: NONE
;   108          1300  1 !
;   109          1301  1
;   110          1302  1 !
;   111          1303  1 ! EXTERNAL REFERENCES:
;   112          1304  1 !
;   113          1305  1 EXTERNAL LITERAL
;   114          1306  1     STR$_ILLSTRCLA,                ! signal illegal class error
;   115          1307  1     STR$_TRU,                      ! warning, truncation
;   116          1308  1     STR$_NORMAL;                   ! successful append
;   117          1309  1
;   118          1310  1 EXTERNAL ROUTINE
;   119          1311  1     LIB$STOP;                      ! signal errors
```

```
;  121          1312  1 GLOBAL ROUTINE STR$PREFIX ( ! Prefix a string to the start of another
;  122          1313  1
;  123          1314  1                     DEST_DESC,   ! pointer to destination descriptor
;  124          1315  1                     SRC_DESC     ! pointer to source descriptor
;  125          1316  1
;  126          1317  1                           ) =
;  127          1318  1
;  128          1319  1 !++
;  129          1320  1 ! FUNCTIONAL DESCRIPTION:
;  130          1321  1 !
;  131          1322  1 !       This routine takes a source string of any supported dtype and
;  132          1323  1 !       class, and prefixes that string to the beginning of the
;  133          1324  1 !       destination string, which may be of any supported class or
;  134          1325  1 !       dtype , except that it is impossible to add something to the
;  135          1326  1 !       beginning of a string having fixed length semantics so an error
;  136          1327  1 !       will always be signalled in that case
;  137          1328  1 !
;  138          1329  1 ! FORMAL PARAMETERS:
;  139          1330  1 !
;  140          1331  1 !       DEST_DESC.wt.dx         pointer to destination descriptor
;  141          1332  1 !       SRC_DESC.rt.dx          pointer to source descriptor
;  142          1333  1 !
;  143          1334  1 ! IMPLICIT INPUTS:
;  144          1335  1 !
;  145          1336  1 !       NONE
;  146          1337  1 !
;  147          1338  1 ! IMPLICIT OUTPUTS:
;  148          1339  1 !
;  149          1340  1 !       NONE
;  150          1341  1 !
;  151          1342  1 ! COMPLETION CODES:
;  152          1343  1 !
;  153          1344  1 !       STR$_NORMAL             Success
;  154          1345  1 !       STR$_TRU                Truncation occurred.   Warning.
;  155          1346  1 !
;  156          1347  1 ! SIDE EFFECTS:
;  157          1348  1 !
;  158          1349  1 !       STR$_ILLSTRCLA may be signalled if the destination string has
;  159          1350  1 !       fixed length semantics or undefined class.
;  160          1351  1 !       Dynamic string space may be allocated or deallocated
;  161          1352  1 !
;  162          1353  1 !--
;  163          1354  1
;  164          1355  2     BEGIN
;  165          1356  2
;  166          1357  2     LOCAL
;  167          1358  2         IN_LEN,                 ! length of source string
;  168          1359  2         IN_ADDR,                ! address of 1st byte of source string
;  169          1360  2         RETURN_STATUS;          ! statuses from macros
;  170          1361  2
;  171          1362  2     MAP
;  172          1363  2         SRC_DESC  : REF $STR$DESCRIPTOR,
;  173          1364  2         DEST_DESC : REF $STR$DESCRIPTOR;
;  174          1365  2
;  175          1366  2     RETURN_STATUS = 1 ;         ! Assume success to follow
;  176          1367  2
;  177          1368  2 !+
```

```
;  178          1369  2 ! Extract length and address of 1st data byte of source string.
;  179          1370  2 ! Signal if a fatal error results.
;  180          1371  2 !-
;  181          1372  2     $STR$GET_LEN_ADDR ( SRC_DESC, IN_LEN, IN_ADDR) ;
;  182          1373  2
;  183          1374  2 !+
;  184          1375  2 ! algorithm differs based on the class of the destination descriptor
;  185          1376  2 !-
;  186          1377  2
;  187          1378  2     CASE .DEST_DESC [DSC$B_CLASS]
;  188          1379  2     FROM DSC$K_CLASS_Z TO DSC$K_CLASS_SB OF
;  189          1380  2         SET
;  190          1381  2
;  191          1382  2 !+
;  192          1383  2 ! dynamic destination strings
;  193          1384  2 ! ***************************
;  194          1385  2 !-
;  195          1386  2
;  196          1387  2         [DSC$K_CLASS_D]:
;  197          1388  3             BEGIN
;  198          1389  3             IF
;  199        L 1390  3             %IF %BLISS (BLISS16) OR %BLISS (BLISS36)
;  200        U 1391  3             %THEN                                   ! except on VAX
;  201        U 1392  3             $STR$OVERLAP (                          ! If dest overlaps
;  202        U 1393  3                 .DEST_DESC [DSC$A_POINTER],         ! with where it will be
;  203        U 1394  3                 .DEST_DESC [DSC$W_LENGTH],          ! written
;  204        U 1395  3                 CH$PLUS (.DEST_DESC [DSC$A_POINTER],
;  205        U 1396  3                          .DEST_DESC [DSC$W_LENGTH]),
;  206        U 1397  3                 .DEST_DESC [DSC$W_LENGTH])
;  207        U 1398  3             OR
;  208          1399  3             %FI
;  209        P 1400  3             $STR$OVERLAP (                          ! or if dest will be
;  210        P 1401  3                                                     ! written on top of
;  211        P 1402  3                 .IN_ADDR,                           ! source when moved
;  212        P 1403  3                 .IN_LEN,
;  213        P 1404  3                 CH$PLUS (.DEST_DESC [DSC$A_POINTER], .IN_ADDR),
;  214          1405  4                 .DEST_DESC [DSC$W_LENGTH])
;  215          1406  3             OR                                      ! or if destination not
;  216          1407  3                                                     ! large enough for
;  217        P 1408  4             ($STR$NEED_ALLOC (                      ! prefix
;  218        P 1409  4                 .IN_ADDR + .DEST_DESC [DSC$W_LENGTH],
;  219          1410  4                 ($STR$DYN_AL_LEN (DEST_DESC) ) )
;  220          1411  3             THEN                                    ! then allocate a temp
;  221          1412  3                                                     ! and use it for
;  222          1413  4                 BEGIN                               ! building output string
;  223          1414  4                 LOCAL TEMP_DESC : $STR$DESCRIPTOR;
;  224          1415  4                 !+
;  225          1416  4                 ! If allocate is successful, continue the operation,
;  226          1417  4                 ! otherwise remember a fatal error
;  227          1418  4                 !-
;  228        P 1419  5                 IF (RETURN_STATUS = $STR$ALLOCATE (
;  229        P 1420  5                     .IN_LEN + .DEST_DESC [DSC$W_LENGTH],
;  230          1421  5                     TEMP_DESC))
;  231          1422  4                 THEN
;  232          1423  5                     BEGIN
;  233          1424  5                     !+
;  234          1425  5                     ! move source to temp
```

```
;  235          1426  5                !-
;  236          1427  5                CH$MOVE (.IN_LEN, .IN_ADDR,
;  237          1428  5                        .TEMP_DESC [DSC$A_POINTER]);
;  238          1429  5
;  239          1430  5                !+
;  240          1431  5                ! move destination to end of temp
;  241          1432  5                !-
;  242          1433  5                CH$MOVE (.DEST_DESC [DSC$W_LENGTH],
;  243          1434  5                        .DEST_DESC [DSC$A_POINTER],
;  244          1435  5                        CH$PLUS (.TEMP_DESC [DSC$A_POINTER],
;  245          1436  5                                .IN_LEN));
;  246          1437  5
;  247          1438  5                !+
;  248          1439  5                ! switch temp and destination descriptors
;  249          1440  5                !-
;  250          1441  5                $STR$EXCH_DESCS (TEMP_DESC, DEST_DESC);
;  251          1442  5
;  252          1443  5                !+
;  253          1444  5                ! deallocate temp
;  254          1445  5                !-
;  255          1446  5                RETURN_STATUS = $STR$DEALLOCATE (TEMP_DESC);
;  256          1447  4                END;
;  257          1448  4            END
;  258          1449  4
;  259          1450  3        ELSE
;  260          1451  3
;  261          1452  4            BEGIN
;  262          1453  4            !+
;  263          1454  4            ! move destination down
;  264          1455  4            !-
;  265          1456  4            CH$MOVE (.DEST_DESC [DSC$W_LENGTH],
;  266          1457  4                     .DEST_DESC [DSC$A_POINTER],
;  267          1458  4                     CH$PLUS (.DEST_DESC [DSC$A_POINTER],
;  268          1459  4                              .IN_LEN));
;  269          1460  4
;  270          1461  4            !+
;  271          1462  4            ! move source in front of it
;  272          1463  4            !-
;  273          1464  4            CH$MOVE (.IN_LEN, .IN_ADDR, .DEST_DESC [DSC$A_POINTER]);
;  274          1465  4
;  275          1466  4            !+
;  276          1467  4            ! readjust length of output
;  277          1468  4            !-
;  278          1469  4            DEST_DESC [DSC$W_LENGTH] = .DEST_DESC [DSC$W_LENGTH] +
;  279          1470  4                                       .IN_ADDR;
;  280          1471  3            END;
;  281          1472  2        END;
;  282          1473  2
```

```
;  284          1474  2 !+
;  285          1475  2 ! Varying string destination
;  286          1476  2 ! ****************************
;  287          1477  2 !-
;  288          1478  2
;  289          1479  2         [DSC$K_CLASS_VS]:
;  290          1480  3             BEGIN
;  291          1481  3             LOCAL
;  292          1482  3                 OUT_LEN,            ! current destination length
;  293          1483  3                 OUT_ADDR,           ! current pointer to destination
;  294          1484  3                 TOT_LEN;            ! MIN of sum of IN_LEN + OUT_LEN
;  295          1485  3                                     ! and MAXSTRLEN
;  296          1486  3
;  297          1487  3                 !+
;  298          1488  3                 ! set up current length and address of 1st byte of data for
;  299          1489  3                 ! a varying string destination.
;  300          1490  3                 !-
;  301          1491  3                 OUT_LEN = .(.DEST_DESC [DSC$A_POINTER])<0,16> ;
;  302          1492  3                 OUT_ADDR = .DEST_DESC [DSC$A_POINTER] + 2 ;
;  303          1493  3                 TOT_LEN = MIN ( .IN_LEN + .OUT_LEN,
;  304          1494  3                                 .DEST_DESC [DSC$W_MAXSTRLEN]) ;
;  305          1495  3
;  306          1496  3                 IF
;  307        L 1497  3                 %IF %BLISS (BLISS16) OR %BLISS (BLISS36)
;  308        U 1498  3                 %THEN                                   ! except on VAX
;  309        U 1499  3                 $STR$OVERLAP (                          ! If dest overlaps
;  310        U 1500  3                     .OUT_ADDR,                          ! with where it will be
;  311        U 1501  3                     .OUT_LEN,                           ! written
;  312        U 1502  3                     CH$PLUS (.OUT_ADDR,
;  313        U 1503  3                              .OUT_LEN),
;  314        U 1504  3                     .OUT_LEN)
;  315        U 1505  3                 OR
;  316          1506  3                 %FI
;  317        P 1507  3                 $STR$OVERLAP (                          ! or if dest will be
;  318        P 1508  3                                                         ! written on top of
;  319        P 1509  3                     .IN_ADDR,                           ! source when moved
;  320        P 1510  3                     .IN_LEN,
;  321        P 1511  3                     CH$PLUS (.OUT_ADDR, .IN_ADDR),
;  322          1512  4                     .OUT_LEN)
;  323          1513  3                 THEN                                    ! then allocate a temp
;  324          1514  3                                                         ! and use it for
;  325          1515  3                                                         ! building output string
;  326          1516  4                     BEGIN           ! Overlap case
;  327          1517  4                     LOCAL TEMP_DESC : $STR$DESCRIPTOR;
;  328          1518  4                     !+
;  329          1519  4                     ! If allocate is successful, continue the operation,
;  330          1520  4                     ! otherwise remember a fatal error
;  331          1521  4                     !-
;  332        P 1522  5                     IF (RETURN_STATUS = $STR$ALLOCATE (
;  333        P 1523  5                         .IN_LEN + .OUT_LEN,
;  334          1524  5                         TEMP_DESC))
;  335          1525  4                     THEN
;  336          1526  5                         BEGIN       ! copy via temp descr after succ alloc
;  337          1527  5                         !+
;  338          1528  5                         ! move source to temp
;  339          1529  5                         !-
;  340          1530  5                         CH$MOVE (.IN_LEN, .IN_ADDR,
```

```
;   341          1531  5                      .TEMP_DESC [DSC$A_POINTER]);
;   342          1532  5
;   343          1533  5                  !+
;   344          1534  5                  ! move destination to end of temp
;   345          1535  5                  !-
;   346          1536  5                  CH$MOVE (.OUT_LEN,
;   347          1537  5                           .OUT_ADDR,
;   348          1538  5                           CH$PLUS (.TEMP_DESC [DSC$A_POINTER],
;   349          1539  5                                    .IN_LEN));
;   350          1540  5
;   351          1541  5                  !+
;   352          1542  5                  ! copy temp to varying string destination
;   353          1543  5                  !-
;   354          1544  5                  CH$MOVE ( .TOT_LEN,
;   355          1545  5                           .TEMP_DESC [DSC$A_POINTER],
;   356          1546  5                           .OUT_ADDR) ;
;   357          1547  5
;   358          1548  5                  !+
;   359          1549  5                  ! deallocate temp
;   360          1550  5                  !-
;   361          1551  5                  RETURN_STATUS = $STR$DEALLOCATE (TEMP_DESC);
;   362          1552  4                  END;            ! copy via temp descr after succ alloc
;   363          1553  4              END                 ! Overlap case
;   364          1554  4
;   365          1555  3          ELSE
;   366          1556  3
;   367          1557  4              BEGIN               ! non-overlap case
;   368          1558  4              !+
;   369          1559  4              ! move destination down within destination string
;   370          1560  4              !-
;   371          1561  4              CH$MOVE (MIN ( .OUT_LEN,
;   372          1562  4                             MAX (
;   373          1563  4                             .DEST_DESC [DSC$W_MAXSTRLEN] - .IN_LEN,
;   374          1564  4                              0)),
;   375          1565  4                       .OUT_ADDR,
;   376          1566  4                       CH$PLUS (.OUT_ADDR,
;   377          1567  4                                .IN_LEN));
;   378          1568  4
;   379          1569  4              !+
;   380          1570  4              ! move source in front of it
;   381          1571  4              !-
;   382          1572  4              CH$MOVE (MIN (.IN_LEN,
;   383          1573  4                            .DEST_DESC [DSC$W_MAXSTRLEN]),
;   384          1574  4                       .IN_ADDR,
;   385          1575  4                       .OUT_ADDR);
;   386          1576  4
;   387          1577  3              END;                ! non-overlap case
;   388          1578  3
;   389          1579  3          !+
;   390          1580  3          ! readjust length of output -- the CURLEN field
;   391          1581  3          !-
;   392          1582  3          (.DEST_DESC [DSC$A_POINTER])<0,16> = .TOT_LEN ;
;   393          1583  3
;   394          1584  3          !+
;   395          1585  3          ! if truncation occurred in copying, make a note of it
;   396          1586  3          !-
;   397          1587  3          IF .IN_LEN + .OUT_LEN GTRU .DEST_DESC [DSC$W_MAXSTRLEN]
```

```
;  398          1588  3          THEN RETURN_STATUS = STR$_TRU ;
;  399          1589  3
;  400          1590  2          END;                    ! of DSC$K_CLASS_VS
```

```
;   2 !+
;   2 ! all other classes of descriptors describe strings that can't be
;   2 ! prefixed or are unsupported classes, or are unknown classes.
;   2 !-
;   2         [INRANGE, OUTRANGE]:
;   2             RETURN_STATUS = STR$_ILLSTRCLA;
;   2         TES;
;   2
;   2     $STR$SIGNAL_FATAL (RETURN_STATUS);              ! signal severe errors
;   2     RETURN .RETURN_STATUS;                          !
;   1     END;                                            ! End of STR$PREFIX

                                              .TITLE  STR$PREFIX
                                              .IDENT  \1-007\

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

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

                            0FFC 00000        .ENTRY  STR$PREFIX, Save R2,R3,R4,R5,R6,R7,R8,R9,-  ; 1312
                                                      R10,R11                                     ;
                   5E      18  C2 00002       SUBL2   #24, SP                                     ;
                   5B      01  D0 00005       MOVL    #1, RETURN_STATUS                           ; 1366
                   50  08  AC  D0 00008       MOVL    SRC_DESC, R0                                ; 1372
                   02  03  A0  91 0000C       CMPB    3(R0), #2                                   ;
                       09  1A 00010           BGTRU   1$                                          ;
                   59      60  3C 00012       MOVZWL  (R0), IN_LEN                                ;
                   5A  04  A0  D0 00015       MOVL    4(R0), IN_ADDR                              ;
                       09  11 00019           BRB     2$                                          ;
          00000000G        00  16 0001B 1$:   JSB     STR$ANALYZE_SDESC_R1                        ;
                   59      50  7D 00021       MOVQ    R0, R9                                      ;
                   58  04  AC  D0 00024 2$:   MOVL    DEST_DESC, R8                               ; 1378
            0F     00  03  A8  8F 00028       CASEB   3(R8), #0, #15                              ;
0020    002A    0020        0020    0002D 3$: .WORD   4$-3$,-                                     ;
0020    0020    0020        0020    00035             4$-3$,-                                     ;
01F5    0020    0020        0020    0003D             5$-3$,-                                     ;
0020    0020    0020        0020    00045             4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      35$-3$,-                                    ;
                                                      4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      4$-3$,-                                     ;
                                                      4$-3$                                       ;
```

```
                5B 00000000G   8F   D0 0004D 4$:     MOVL    #STR$_ILLSTRCLA, RETURN_STATUS          ; 1596
                          033B 31   00054           BRW     57$
                56        04   A8   D0 00057 5$:     MOVL    4(R8), R6                               ; 1405
          50    56             5A   C1 0005B         ADDL3   IN_ADDR, R6, R0
                50             5A   D1 0005F         CMPL    IN_ADDR, R0
                               0B   1E 00062         BGEQU   6$
          51    5A             59   C1 00064         ADDL3   IN_LEN, IN_ADDR, R1
                51             50   D1 00068         CMPL    R0, R1
                               0D   18 0006B         BGEQ    7$
                               58   11 0006D         BRB     14$
                51             68   3C 0006F 6$:     MOVZWL  (R8), R1
                50             51   C0 00072         ADDL2   R1, R0
                50             5A   D1 00075         CMPL    IN_ADDR, R0
                               77   19 00078         BLSS    20$
                52             D4   0007A 7$:     CLRL    R2                                      ; 1410
                56             D5   0007C         TSTL    R6
                               06   12 0007E         BNEQ    8$
                52             D6   00080         INCL    R2
                50             D4   00082         CLRL    R0
                               13   11 00084         BRB     10$
          00F0  8F             68   B1 00086 8$:     CMPW    (R8), #240
                               05   1B 0008B         BLEQU   9$
                50             68   3C 0008D         MOVZWL  (R8), R0
                               07   11 00090         BRB     10$
                50             56   D0 00092 9$:     MOVL    R6, STRING_BLOCK
                50        FE   A0   3C 00095         MOVZWL  -2(STRING_BLOCK), R0
    000000F0    8F             50   D1 00099 10$:    CMPL    R0, #240
                               27   1F 000A0         BLSSU   15$
                51             68   3C 000A2         MOVZWL  (R8), R1
                51             5A   C0 000A5         ADDL2   IN_ADDR, R1
                04             52   E9 000A8         BLBC    R2, 11$
                               50   D4 000AB         CLRL    R0
                               13   11 000AD         BRB     13$
          00F0  8F             68   B1 000AF 11$:     CMPW    (R8), #240
                               05   1B 000B4         BLEQU   12$
                50             68   3C 000B6         MOVZWL  (R8), R0
                               07   11 000B9         BRB     13$
                50             56   D0 000BB 12$:     MOVL    R6, STRING_BLOCK
                50        FE   A0   3C 000BE         MOVZWL  -2(STRING_BLOCK), R0
                50             51   D1 000C2 13$:     CMPL    R1, R0
                               27   13 000C5         BEQL    19$
                               28   11 000C7 14$:     BRB     20$
                51             68   3C 000C9 15$:     MOVZWL  (R8), R1
                51             5A   C0 000CC         ADDL2   IN_ADDR, R1
                04             52   E9 000CF         BLBC    R2, 16$
                               50   D4 000D2         CLRL    R0
                               13   11 000D4         BRB     18$
          00F0  8F             68   B1 000D6 16$:     CMPW    (R8), #240
                               05   1B 000DB         BLEQU   17$
                50             68   3C 000DD         MOVZWL  (R8), R0
                               07   11 000E0         BRB     18$
                50             56   D0 000E2 17$:     MOVL    R6, STRING_BLOCK
                50        FE   A0   3C 000E5         MOVZWL  -2(STRING_BLOCK), R0
                50             51   D1 000E9 18$:     CMPL    R1, R0
                               03   1A 000EC         BGTRU   20$
                          0122 31   000EE 19$:     BRW     33$
                07 00000000G   00   E8 000F1 20$:     BLBS    STR$$V_INIT, 21$                        ; 1421
```

```
                00000000G  00              00  FB  000F8              CALLS   #0, STR$$INIT
                           51  00000000G    8F  D0  000FF  21$:       MOVL    #STR$_NORMAL, RETURN_STATUS
                           52               68  3C  00106            MOVZWL  (R8), R2
                           52               59  C0  00109            ADDL2   IN_LEN, R2
                000000F0   8F               52  D1  0010C            CMPL    R2, #240
                                            4C  1A  00113            BGTRU   27$
                                            52  D5  00115            TSTL    R2
                                            04  12  00117            BNEQ    22$
                                            53  D4  00119            CLRL    TEMP
                                            35  11  0011B            BRB     26$
                           50         FF    A2  9E  0011D  22$:       MOVAB   -1(R2), R0
                           50               07  8A  00121            BICB2   #7, R0
                           54  00000000G00040  9E  00124            MOVAB   STR$$Q_SHORT_Q[R0], REMQUE_ADDR
                           53         00    B4  0F  0012C  23$:       REMQUE  @0(REMQUE_ADDR), TEMP
                                            05  1D  00130            BVS     24$
                           52               01  D0  00132            MOVL    #1, ALLOC_DONE
                                            13  11  00135            BRB     25$
                                            52  D4  00137  24$:       CLRL    ALLOC_DONE
                           50         04    BC  3C  00139            MOVZWL  @DEST_DESC, R0
                                    6049    9F  00013D               PUSHAB  (R0)[IN_LEN]
                00000000G  00               01  FB  00140            CALLS   #1, STR$$ALOC_SHORT
                           51               50  D0  00147            MOVL    R0, RETURN_STATUS
                           05               52  E8  0014A  25$:       BLBS    ALLOC_DONE, 26$
                           35               51  E9  0014D            BLBC    RETURN_STATUS, 29$
                                            DA  11  00150            BRB     23$
                           30               51  E9  00152  26$:       BLBC    RETURN_STATUS, 29$
                      14   AE               53  D0  00155            MOVL    TEMP, TEMP_DESC+4
                10    AE                    59  04  BC  A1  00159    ADDW3   @DEST_DESC, IN_LEN, TEMP_DESC
                                            24  11  0015F            BRB     29$
                                14   AE      9F  00161  27$:       PUSHAB  TEMP_DESC+4
                           08   AE           52  D0  00164            MOVL    R2, 8(SP)
                                08   AE      9F  00168               PUSHAB  8(SP)
                00000000G  00               02  FB  0016B            CALLS   #2, LIB$GET_VM
                           51               50  D0  00172            MOVL    R0, RETURN_STATUS
                           09               51  E8  00175            BLBS    RETURN_STATUS, 28$
                           51  00000000G    8F  D0  00178            MOVL    #STR$_INSVIRMEM, RETURN_STATUS
                                            04  11  0017F            BRB     29$
                           10   AE          52  B0  00181  28$:       MOVW    R2, TEMP_DESC
                           5B               51  D0  00185  29$:       MOVL    RETURN_STATUS, RETURN_STATUS
                           03               51  E8  00188            BLBS    RETURN_STATUS, 30$
                                    0204    31  00018B               BRW     57$
14    BE                    6A               59  28  0018E  30$:       MOVC3   IN_LEN, (IN_ADDR), @TEMP_DESC+4                 1428
                           56         04    AC  D0  00193            MOVL    DEST_DESC, R6                                   1433
14 BE49          04   B6                    66  28  00197            MOVC3   (R6), @4(R6), @TEMP_DESC+4[IN_LEN]              1436
                 08   AE                    66  B0  0019E            MOVW    (R6), $STR$TEMP_DESC                            1441
                 0C   AE         04    A6    D0  001A2                MOVL    4(R6), $STR$TEMP_DESC+4
                 12   AE         02    A6    B0  001A7                MOVW    2(R6), TEMP_DESC+2
                           50         10    AE  9E  001AC            MOVAB   TEMP_DESC, R0
                           51               56  D0  001B0            MOVL    R6, R1
                                00000000G   00  16  001B3            JSB     STR$$MOVQ_R1
                      10   AE          08   AE  B0  001B9            MOVW    $STR$TEMP_DESC, TEMP_DESC
                      14   AE          0C   AE  D0  001BE            MOVL    $STR$TEMP_DESC+4, TEMP_DESC+4
                           50  00000000G    8F  D0  001C3            MOVL    #STR$_NORMAL, RETURN_STATUS                     1446
                           52               14  AE  D0  001CA        MOVL    TEMP_DESC+4, R2
                                            3E  13  001CE            BEQL    32$
                      00F0   8F         10    AE  B1  001D0        CMPW    TEMP_DESC, #240
                                            1A  1A  001D6            BGTRU   31$
```

```
                    51            52  D0 001D8         MOVL     R2, STRING_BLOCK
                    51       FE   A1  3C 001DB         MOVZWL   -2(STRING_BLOCK), ALLOC_LENGTH
                                  51  D7 001DF         DECL     R1
                    51            07  8A 001E1         BICB2    #7, R1
                    51 00000000G0041  9E 001E4         MOVAB    STR$$Q_SHORT_Q[R1], INSQUE_ADDR
           00       B1            62  0E 001EC         INSQUE   (R2), @0(INSQUE_ADDR)
                                  1C  11 001F0         BRB      32$
                          14      AE  9F 001F2 31$:    PUSHAB   TEMP_DESC+4
           08       AE    14      AE  3C 001F5         MOVZWL   TEMP_DESC, 8(SP)
                          08      AE  9F 001FA         PUSHAB   8(SP)
    00000000G       00            02  FB 001FD         CALLS    #2, LIB$FREE_VM
                    07            50  E8 00204         BLBS     RETURN_STATUS, 32$
                    50 00000000G  8F  D0 00207         MOVL     #STR$_FATINTERR, RETURN_STATUS
                    5B            50  D0 0020E 32$:    MOVL     RETURN_STATUS, RETURN_STATUS
                                  0C  11 00211         BRB      34$                                 1389
       6946         66            68  28 00213 33$:    MOVC3    (R8), (R6), (IN_LEN)[R6]            1459
         66         6A            59  28 00218         MOVC3    IN_LEN, (IN_ADDR), (R6)             1464
                    68            5A  A0 0021C         ADDW2    IN_ADDR, (R8)                       1470
                               0170   31 0021F 34$:    BRW      57$                                 1378
                    57       04   B8  3C 00222 35$:    MOVZWL   @4(R8), OUT_LEN                     1491
         56         04   A8       02  C1 00226         ADDL3    #2, 4(R8), OUT_ADDR                 1492
         52              59       57  C1 0022B         ADDL3    OUT_LEN, IN_LEN, R2                 1493
                         50       52  D0 0022F         MOVL     R2, R0                              1494
 50      68              10       00  ED 00232         CMPZV    #0, #16, (R8), R0
                                  03  18 00237         BGEQ     36$
                    50            68  3C 00239         MOVZWL   (R8), R0
           04       AE            50  D0 0023C 36$:    MOVL     R0, TOT_LEN                         1493
         50         56            5A  C1 00240         ADDL3    IN_ADDR, OUT_ADDR, R0               1512
                    50            5A  D1 00244         CMPL     IN_ADDR, R0
                                  09  1E 00247         BGEQU    37$
         51         5A            59  C1 00249         ADDL3    IN_LEN, IN_ADDR, R1
                    51            50  D1 0024D         CMPL     R0, R1
                                  06  11 00250         BRB      38$
                    50            57  C0 00252 37$:    ADDL2    OUT_LEN, R0
                    50            5A  D1 00255         CMPL     IN_ADDR, R0
                                  03  19 00258 38$:    BLSS     39$
                               00EF   31 0025A         BRW      52$
                    07 00000000G  00  E8 0025D 39$:    BLBS     STR$$V_INIT, 40$                    1524
    00000000G       00            00  FB 00264         CALLS    #0, STR$$INIT
                    51 00000000G  8F  D0 0026B 40$:    MOVL     #STR$_NORMAL, RETURN_STATUS
    000000F0        8F            52  D1 00272         CMPL     R2, #240
                                  47  1A 00279         BGTRU    46$
                                  52  D5 0027B         TSTL     R2
                                  04  12 0027D         BNEQ     41$
                                  53  D4 0027F         CLRL     TEMP
                                  31  11 00281         BRB      45$
                    50       FF   A2  9E 00283 41$:    MOVAB    -1(R2), R0
                    50            07  8A 00287         BICB2    #7, R0
                    54 00000000G0040  9E 0028A         MOVAB    STR$$Q_SHORT_Q[R0], REMQUE_ADDR
                    53       00   B4  0F 00292 42$:    REMQUE   @0(REMQUE_ADDR), TEMP
                                  05  1D 00296         BVS      43$
                    52            01  D0 00298         MOVL     #1, ALLOC_DONE
                                  0F  11 0029B         BRB      44$
                                  52  D4 0029D 43$:    CLRL     ALLOC_DONE
                               6749   9F 0029F         PUSHAB   (OUT_LEN)[IN_LEN]
    00000000G       00            01  FB 002A2         CALLS    #1, STR$$ALOC_SHORT
                    51            50  D0 002A9         MOVL     R0, RETURN_STATUS
```

```
                        05              52  E8  002AC 44$:     BLBS    ALLOC_DONE, 45$                                 ;
                        34              51  E9  002AF           BLBC    RETURN_STATUS, 48$                              ;
                                        DE  11  002B2           BRB     42$                                             ;
                        2F              51  E9  002B4 45$:     BLBC    RETURN_STATUS, 48$                              ;
                    14  AE              53  D0  002B7           MOVL    TEMP, TEMP_DESC+4                               ;
              10    AE                  57  A1  002BB           ADDW3   OUT_LEN, IN_LEN, TEMP_DESC                      ;
                                        24  11  002C0           BRB     48$                                             ;
                            14      AE  9F  002C2 46$:     PUSHAB  TEMP_DESC+4                                     ;
                        04  AE          52  D0  002C5           MOVL    R2, 4(SP)                                       ;
                            04      AE  9F  002C9           PUSHAB  4(SP)                                           ;
                00000000G   00          02  FB  002CC           CALLS   #2, LIB$GET_VM                                  ;
                            51          50  D0  002D3           MOVL    R0, RETURN_STATUS                               ;
                            09          51  E8  002D6           BLBS    RETURN_STATUS, 47$                              ;
                            51  00000000G  8F  D0  002D9           MOVL    #STR$_INSVIRMEM, RETURN_STATUS                  ;
                                        04  11  002E0           BRB     48$                                             ;
                        10  AE          52  B0  002E2 47$:     MOVW    R2, TEMP_DESC                                   ;
                            5B          51  D0  002E6 48$:     MOVL    RETURN_STATUS, RETURN_STATUS                    ;
                            5E          51  E9  002E9           BLBC    RETURN_STATUS, 51$                              ;
                            58      14  AE  D0  002EC           MOVL    TEMP_DESC+4, R8                                 ; 1531
                        68  6A          59  28  002F0           MOVC3   IN_LEN, (IN_ADDR), (R8)                         ;
                      6948  66          57  28  002F4           MOVC3   OUT_LEN, (OUT_ADDR), (IN_LEN)[R8]               ; 1539
                        66  68      04  AE  28  002F9           MOVC3   TOT_LEN, (R8), (OUT_ADDR)                       ; 1546
                            50  00000000G  8F  D0  002FE           MOVL    #STR$_NORMAL, RETURN_STATUS                     ; 1551
                                        58  D5  00305           TSTL    R8                                              ;
                                        3E  13  00307           BEQL    50$                                             ;
                    00F0    8F      10  AE  B1  00309           CMPW    TEMP_DESC, #240                                 ;
                                        1A  1A  0030F           BGTRU   49$                                             ;
                            51          58  D0  00311           MOVL    R8, STRING_BLOCK                                ;
                            51      FE  A1  3C  00314           MOVZWL  -2(STRING_BLOCK), ALLOC_LENGTH                  ;
                                        51  D7  00318           DECL    R1                                              ;
                            51          07  8A  0031A           BICB2   #7, R1                                          ;
                            51  00000000G0041  9E  0031D           MOVAB   STR$$Q_SHORT_Q[R1], INSQUE_ADDR                 ;
                        00  B1              68  0E  00325           INSQUE  (R8), @0(INSQUE_ADDR)                           ;
                                        1C  11  00329           BRB     50$                                             ;
                                14      AE  9F  0032B 49$:     PUSHAB  TEMP_DESC+4                                     ;
                        04  AE          14  AE  3C  0032E           MOVZWL  TEMP_DESC, 4(SP)                                ;
                                04      AE  9F  00333           PUSHAB  4(SP)                                           ;
                00000000G   00          02  FB  00336           CALLS   #2, LIB$FREE_VM                                 ;
                            07          50  E8  0033D           BLBS    RETURN_STATUS, 50$                              ;
                            50  00000000G  8F  D0  00340           MOVL    #STR$_FATINTERR, RETURN_STATUS                  ;
                            5B          50  D0  00347 50$:     MOVL    RETURN_STATUS, RETURN_STATUS                    ;
                                        2B  11  0034A 51$:     BRB     56$                                             ; 1496
                            50          68  3C  0034C 52$:     MOVZWL  (R8), R0                                        ; 1563
                            50          59  C2  0034F           SUBL2   IN_LEN, R0                                      ;
                                        02  18  00352           BGEQ    53$                                             ; 1562
                                        50  D4  00354           CLRL    R0                                              ;
                            51          57  D0  00356 53$:     MOVL    OUT_LEN, R1                                     ;
                            50          51  D1  00359           CMPL    R1, R0                                          ;
                                        03  15  0035C           BLEQ    54$                                             ;
                            51          50  D0  0035E           MOVL    R0, R1                                          ;
                      6946  66          51  28  00361 54$:     MOVC3   R1, (OUT_ADDR), (IN_LEN)[OUT_ADDR]              ; 1567
                            50          59  D0  00366           MOVL    IN_LEN, R0                                      ; 1573
                    50      68      10  00  ED  00369           CMPZV   #0, #16, (R8), R0                               ;
                                        03  18  0036E           BGEQ    55$                                             ;
                            50          68  3C  00370           MOVZWL  (R8), R0                                        ;
                        66  6A          50  28  00373 55$:     MOVC3   R0, (IN_ADDR), (OUT_ADDR)                       ; 1575
                            51      04  AC  D0  00377 56$:     MOVL    DEST_DESC, R1                                   ; 1582
```

```
                        04   B1      04   AE  B0 0037B          MOVW    TOT_LEN, @4(R1)                                  ;
                  50         59           57  C1 00380          ADDL3   OUT_LEN, IN_LEN, R0                              ; 1587
            50    61         10           00  ED 00384          CMPZV   #0, #16, (R1), R0                                ;
                               10           07  1E 00389          BGEQU   57$                                              ;
                               5B 00000000G 8F  D0 0038B          MOVL    #STR$_TRU, RETURN_STATUS                         ; 1588
                               10           5B  E8 00392 57$:     BLBS    RETURN_STATUS, 58$                               ; 1599
            04    5B         03           00  ED 00395          CMPZV   #0, #3, RETURN_STATUS, #4                        ;
                               09           12  0039A          BNEQ    58$                                              ;
                               5B           DD  0039C          PUSHL   RETURN_STATUS                                    ;
                  00000000G    00           01  FB  0039E          CALLS   #1, LIB$STOP                                     ;
                               50           5B  D0  003A5 58$:    MOVL    RETURN_STATUS, R0                                ; 1600
                                            04  003A8          RET                                                      ; 1601

; Routine Size:  937 bytes,    Routine Base:  _STR$CODE + 0000

;  413          1602  1 END                                 !End of module
;  414          1603  0 ELUDOM



;                                  PSECT SUMMARY
;
;      Name                      Bytes                          Attributes
;
;  _STR$CODE                      937  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         11          0         581       00:00.8



;                                   COMMAND QUALIFIERS

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

; Size:         937 code + 0 data bytes
; Run Time:       00:14.2
; Elapsed Time:   00:54.9
; Lines/CPU Min:   6782
; Lexemes/CPU-Min: 37117
; Memory Used:  294 pages
```

```
; Compilation Complete
```

LINKER
LIS

STRPOSIT
LIS

STRUNWDEQ
LIS

STRPOSEXT
LIS

STRREPLAC
LIS

STRSRCHN
LIS

STRRIGHT
LIS

LINKER

STRTRIM
LIS

LINK
MAP

PREFIX
REQ

ISDSORT
LIS

STRUPCASE
LIS

STRTRANSL
LIS

DATBAS
MDL

TIRAUX
REQ

ISGENC
REQ

STRPREFIX
LIS

DATBAS
LIS