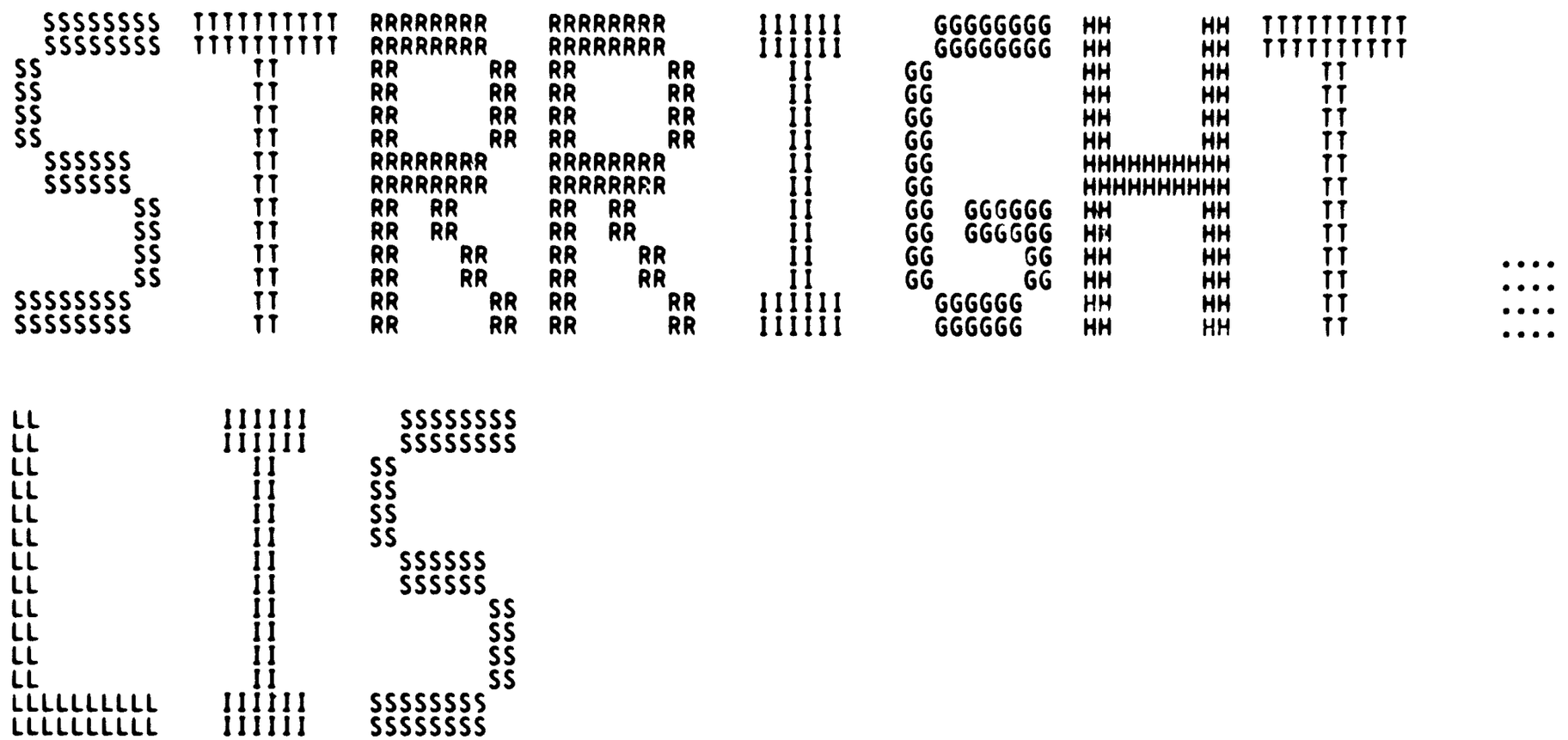
```
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```


SSSSSSSS TTTTTTTTTT RRRRRRRR RRRRRRRR IIIIII GGGGGGGG HH HH TTTTTTTTTT
SSSSSSSS TTTTTTTTTT RRRRRRRR RRRRRRRR IIIIII GGGGGGGG HH HH TTTTTTTTTT
SS TT RR RR RR RR II GG HH HH TT
SS TT RR RR RR RR II GG HH HH TT
SS TT RR RR RR RR II GG HH HH TT
SS TT RR RR RR RR II GG HH HH TT
SSSSSS TT RRRRRRRR RRRRRRRR II GG HHHHHHHHHH TT
SSSSSS TT RRRRRRRR RRRRRRRR II GG HHHHHHHHHH TT
SS TT RR RR RR RR II GG GGGGGG HH HH TT
SS TT RR RR RR RR II GG GGGGGG HH HH TT
SS TT RR RR RR RR II GG GG HH HH TT
SS TT RR RR RR RR II GG GG HH HH TT
SSSSSSSS TT RR RR RR RR IIIIII GGGGGG HH HH TT
SSSSSSSS TT RR RR RR RR IIIIII GGGGGG HH HH TT
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
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS

```
;     1          0001  0 MODULE STR$RIGHT (                    ! Extract a substr on the right
;     2          0002  0
;     3          0003  0                     IDENT = '1-013' ! File: STRRIGHT.B32    Edit: RKR1013
;     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 module extracts a substring according to the
;    38          0038  1 !       BASIC-PLUS-2 syntax.  It finds the substring of a main string
;    39          0039  1 !       starting at the character position specified by the input
;    40          0040  1 !       parameter and continues through the last character of the
;    41          0041  1 !       string.  This substring is copied to the destination string.
;    42          0042  1 !
;    43          0043  1 ! ENVIRONMENT: User mode, AST level or not or mixed
;    44          0044  1 !
;    45          0045  1 ! AUTHOR: R. Will, CREATION DATE: 19-Feb-79
;    46          0046  1 !
;    47          0047  1 ! MODIFIED BY:
;    48          0048  1 !
;    49          0049  1 !  R. Will, 19-Feb-79: VERSION 01
;    50          0050  1 ! 1-001 - Original
;    51          0051  1 ! 1-002 - Change linkage and call to COPY routine. 15-mar-79
;    52          0052  1 ! 1-003 - Change string linkages to start with STR$.  JBS 04-JUN-1979
;    53          0053  1 ! 1-004 - Change call to STR$$COPY.  JBS 16-JUL-1979
;    54          0054  1 ! 1-006 - Define an aux variable to improve the code generated.
;    55          0055  1 !         JBS 11-OCT-1979
;    56          0056  1 ! 1-007 - Change name to STR$, string cleanup.  RW  31-Oct-79
;    57          0057  1 ! 1-008 - Change to new interlock macros.  JBS 06-NOV-1979
```

```
;    58          0058  1 ! 1-009 - Use string interlocks from CALL entry.  RW  15-Nov-79
;    59          0059  1 ! 1-010 - String speedup, remove edit 9.  RW  8-Jan-1980
;    60          0060  1 ! 1-011 - Enhance to recognize additonal classes of descriptors by
;    61          0061  1 !         using $STR$GET_LEN_ADDR to extract length and address of
;    62          0062  1 !         1st data byte from descriptor.  Remove all string interlocking
;    63          0063  1 !         code. RKR 27-APR-81
;    64          0064  1 ! 1-012 - Speed up code.  RKR 7-OCT-1981.
;    65          0065  1 ! 1-013 - Use STR$COPY_R_R8 for copy operation.  Use $STR$SIGNAL_FATAL
;    66          0066  1 !         instead of $STR$CHECK_STATUS.  RKR 18-NOV-1981.
;    67          0067  1 !--
;    68          0068  1
;    69          0069  1 !<BLF/PAGE>
```

```
;    71          0070  1 !
;    72          0071  1 ! SWITCHES:
;    73          0072  1 !
;    74          0073  1
;    75          0074  1 SWITCHES ADDRESSING_MODE
;    76          0075  1                 (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    77          0076  1
;    78          0077  1 !
;    79          0078  1 ! LINKAGES:
;    80          0079  1 !
;    81          0080  1
;    82          0081  1 REQUIRE 'RTLIN:STRLNK';         ! Use require file with string linkage
;    83          0266  1
;    84          0267  1 !
;    85          0268  1 ! TABLE OF CONTENTS:
;    86          0269  1 !
;    87          0270  1
;    88          0271  1 FORWARD ROUTINE
;    89          0272  1     STR$RIGHT,                  ! Find the RIGHT of a str, CALL
;    90          0273  1     STR$RIGHT_R8 : STR$JSB_RIGHT; ! Find the RIGHT of a str, JSB
;    91          0274  1
;    92          0275  1 !
;    93          0276  1 ! INCLUDE FILES:
;    94          0277  1 !
;    95          0278  1
;    96          0279  1 REQUIRE 'RTLIN:RTLPSECT';       ! Declare PSECTs code
;    97          0374  1
;    98          0375  1 REQUIRE 'RTLIN:STRMACROS';      ! use string macros to code
;    99          1291  1
;   100          1292  1 LIBRARY 'RTLSTARLE';            ! STARLET library for macros and symbol
;   101          1293  1
;   102          1294  1 !
;   103          1295  1 ! MACROS: NONE
;   104          1296  1 !
;   105          1297  1
;   106          1298  1 !
;   107          1299  1 ! EQUATED SYMBOLS: NONE
;   108          1300  1 !
;   109          1301  1
;   110          1302  1 !
;   111          1303  1 ! PSECT DECLARATIONS
;   112          1304  1 !
;   113          1305  1
;   114          1306  1 DECLARE_PSECTS (STR);
;   115          1307  1
;   116          1308  1 !
;   117          1309  1 ! OWN STORAGE: NONE
;   118          1310  1 !
;   119          1311  1
;   120          1312  1 !
;   121          1313  1 ! EXTERNAL REFERENCES:
;   122          1314  1 !
;   123          1315  1
;   124          1316  1 EXTERNAL ROUTINE
;   125          1317  1     LIB$STOP,                                  ! signal fatal errors
;   126          1318  1     STR$COPY_R_R8 : STR$JSB_COPY_R ;           ! Routine to do the copy
;   127          1319  1
```

```
;  128          1320  1 EXTERNAL LITERAL
;  129          1321  1     STR$_NORMAL,                ! Successful completion
;  130          1322  1     STR$_ILLSTRPOS;             . pos was - or > str length
```

```
;  132     1323  1 GLOBAL ROUTINE STR$RIGHT (                    ! Extract a substr on the right
;  133     1324  1
;  134     1325  1         DEST_DESC,                  ! Pointer to destination descriptor
;  135     1326  1         SRC_DESC,                   ! Pointer to source descriptor
;  136     1327  1         CHAR_POS                    ! First character to be included
;  137     1328  1
;  138     1329  1                                 ) =
;  139     1330  1
;  140     1331  1 !++
;  141     1332  1 ! FUNCTIONAL DESCRIPTION:
;  142     1333  1 !
;  143     1334  1 !       This routine extracts the characters starting at the
;  144     1335  1 !       character position in the source string specified by the input
;  145     1336  1 !       parameter and continuing through the last character of the
;  146     1337  1 !       source string and copies that substring
;  147     1338  1 !       to the destination string (by JSB to STR$COPY_R_R8) according
;  148     1339  1 !       to the syntax of the class of the destination string.
;  149     1340  1 !       If the input character position is < 2 then the entire source
;  150     1341  1 !       string is copied.  If the input character position is > the
;  151     1342  1 !       length of the source string, then the destination string becomes
;  152     1343  1 !       a null string.  It computes the RIGHT by JSBing to the JSB entry
;  153     1344  1 !       point.
;  154     1345  1 !
;  155     1346  1 ! FORMAL PARAMETERS:
;  156     1347  1 !
;  157     1348  1 !       DEST_DESC.wt.dx         pointer to destination string descriptor
;  158     1349  1 !       SRC_DESC.rt.dx          pointer to source string descriptor
;  159     1350  1 !       CHAR_POS.rl.r           character position in source to start
;  160     1351  1 !                               substring
;  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$RIGHT_R8
;  173     1364  1 !
;  174     1365  1 ! SIDE EFFECTS:
;  175     1366  1 !
;  176     1367  1 !       same as STR$RIGHT_R8 since it JSBs to that routine.
;  177     1368  1 !
;  178     1369  1 !--
;  179     1370  1
;  180     1371  2     BEGIN
;  181     1372  2
;  182     1373  2     MAP
;  183     1374  2         SRC_DESC  : REF $STR$DESCRIPTOR,
;  184     1375  2         DEST_DESC : REF $STR$DESCRIPTOR;
;  185     1376  2
;  186     1377  2     RETURN STR$RIGHT_R8 (DEST_DESC [0,0,0,0],
;  187     1378  2                          SRC_DESC [0,0,0,0],
;  188     1379  2                          ..CHAR_POS);
```

```
;  189          1380  1     END;                                    !End of STR$RIGHT


                                                                     .TITLE  STR$RIGHT
                                                                     .IDENT  \1-013\

                                                                     .EXTRN  LIB$STOP, STR$COPY_R_R8
                                                                     .EXTRN  STR$_NORMAL, STR$_ILLSTRPOS

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

                                                01FC 00000           .ENTRY  STR$RIGHT, Save R2,R3,R4,R5,R6,R7,R8        ; 1323
                                      52        0C   BC   D0 00002  MOVL    @CHAR_POS, R2                              ; 1378
                                      50        04   AC   7D 00006  MOVQ    DEST_DESC, R0                              ;
                                                     0000V 30 0000A  BSBW    STR$RIGHT_R8                               ;
                                                           04 0000D  RET                                                ; 1380

; Routine Size:  14 bytes,    Routine Base:  _STR$CODE + 0000
```

```
;  191     1381  1 GLOBAL ROUTINE STR$RIGHT_R8 (          ! Extract a substr on the right
;  192     1382  1
;  193     1383  1         DEST_DESC,              ! Pointer to destination descriptor
;  194     1384  1         SRC_DESC,               ! Pointer to source descriptor
;  195     1385  1         CHAR_POS                ! First character to be included
;  196     1386  1
;  197     1387  1                               ) : STR$JSB_RIGHT =
;  198     1388  1
;  199     1389  1 !++
;  200     1390  1 ! FUNCTIONAL DESCRIPTION:
;  201     1391  1 !
;  202     1392  1 !       This routine extracts the characters starting at the
;  203     1393  1 !       character position in the source string specified by the input
;  204     1394  1 !       parameter (where the first character in the string is at
;  205     1395  1 !       position 1) and continuing through the last character of the
;  206     1396  1 !       source string and copies that substring to the destination
;  207     1397  1 !       string (by JSB to STR$$COPY_R_R8) according
;  208     1398  1 !       to the syntax of the class of the destination string.
;  209     1399  1 !       If the input character position is < 2 then the entire source
;  210     1400  1 !       string is copied.  If the input character position is > the
;  211     1401  1 !       length of the source string, then the destination string
;  212     1402  1 !       becomes a null string.
;  213     1403  1 !
;  214     1404  1 ! FORMAL PARAMETERS:
;  215     1405  1 !
;  216     1406  1 !       DEST_DESC.wt.dx         pointer to destination string descriptor
;  217     1407  1 !       SRC_DESC.rt.dx          pointer to source string descriptor
;  218     1408  1 !       CHAR_POS.rl.v           character position in src to start
;  219     1409  1 !                               substring
;  220     1410  1 !
;  221     1411  1 ! IMPLICIT INPUTS:
;  222     1412  1 !
;  223     1413  1 !       NONE
;  224     1414  1 !
;  225     1415  1 ! IMPLICIT OUTPUTS:
;  226     1416  1 !
;  227     1417  1 !       NONE
;  228     1418  1 !
;  229     1419  1 ! COMPLETION CODES:
;  230     1420  1 !
;  231     1421  1 !       SS$_NORMAL      Success
;  232     1422  1 !       STR$_ILLSTRPOS  Character position reference outside of string
;  233     1423  1 !       STR$_ILLSTRSPE  End pos < start pos, or length too long
;  234     1424  1 !       STR$_NEGSTRLEN  Negative length supplied, 0 used
;  235     1425  1 !       STR$_TRU        Truncation occured in copying to destination
;  236     1426  1 !                       (Warning)
;  237     1427  1 !
;  238     1428  1 ! SIDE EFFECTS:
;  239     1429  1 !
;  240     1430  1 !       May signal:
;  241     1431  1 !           STR$_FATINTERR      Fatal internal error
;  242     1432  1 !           STR$_ILLSTRCLA      Illegal (or unsupported) string class
;  243     1433  1 !           STR$_INSVIRMEM      Insufficient virtual memory for
;  244     1434  1 !                               reallocation of dynamic string
;  245     1435  1 !
;  246     1436  1 !--
;  247     1437  1
```

```
;  248     1438  2     BEGIN
;  249     1439  2
;  250     1440  2     LOCAL
;  251     1441  2         IN_LEN,                 ! length of input string
;  252     1442  2         IN_ADDR,                ! address of 1st data byte of input str
;  253     1443  2         RETURN_STATUS,          ! keep the status of this routine
;  254     1444  2         COPY_STATUS,            ! status returned from STR$COPY
;  255     1445  2         LENGTH;                 ! length to be copied
;  256     1446  2
;  257     1447  2     MAP
;  258     1448  2         SRC_DESC  : REF $STR$DESCRIPTOR,
;  259     1449  2         DEST_DESC : REF $STR$DESCRIPTOR;
;  260     1450  2
;  261     1451  2     RETURN_STATUS = 1 ;         ! Assume success to follow
;  262     1452  2
;  263     1453  2 !+
;  264     1454  2 ! compute length and address of 1st data byte of input string
;  265     1455  2 !-
;  266     1456  2     $STR$GET_LEN_ADDR ( SRC_DESC, IN_LEN, IN_ADDR) ;
;  267     1457  2
;  268     1458  2 !+
;  269     1459  2 ! Compute the correct substring and call the COPY routine using
;  270     1460  2 ! the substring length and pointer.  LENGTH is length to be skipped is
;  271     1461  2 ! the actual starting position (after testing for errors) - 1
;  272     1462  2 ! (which must be >= 0 and <= src len)
;  273     1463  2 !-
;  274     1464  2
;  275     1465  2     LENGTH =
;  276     1466  3         BEGIN
;  277     1467  3         IF .IN_LEN LSS .CHAR_POS - 1
;  278     1468  3         THEN
;  279     1469  4             BEGIN
;  280     1470  4             RETURN_STATUS = STR$_ILLSTRPOS;     ! input start to big
;  281     1471  4             .IN_LEN                             ! use srclen and
;  282     1472  4                                                 ! remember error
;  283     1473  4             END
;  284     1474  3         ELSE .CHAR_POS - 1
;  285     1475  2         END;
;  286     1476  2
;  287     1477  2     IF .LENGTH LSS 0 THEN
;  288     1478  3         BEGIN
;  289     1479  3         LENGTH = 0;                             ! input length is
;  290     1480  3                                                 ! negative
;  291     1481  3         RETURN_STATUS = STR$_ILLSTRPOS;         ! use 0 and remember
;  292     1482  3                                                 ! error
;  293     1483  2         END;
;  294     1484  2
;  295     1485  2 !+
;  296     1486  2 ! copy to destination descriptor for srclen - (startpos - 1) starting
;  297     1487  2 ! srcptr + startpos - 1
;  298     1488  2 !-
;  299     1489  2     COPY_STATUS = STR$COPY_R_R8 ( .DEST_DESC,
;  300     1490  2                                   .IN_LEN - .LENGTH,
;  301     1491  2                                   CH$PLUS (.IN_ADDR, .LENGTH) );
;  302     1492  2
;  303     1493  2     IF .COPY_STATUS NEQ SS$_NORMAL
;  304     1494  2     THEN RETURN_STATUS = .COPY_STATUS;  ! copy truncated, return
```

```
;  305          1495  2                                   ! truncate
;  306          1496  2                                   ! instead of previous status
;  307          1497  2
;  308          1498  2     $STR$SIGNAL_FATAL (RETURN_STATUS);          ! signal fatal errors
;  309          1499  2     RETURN .RETURN_STATUS;
;  310          1500  1     END;                                        !End of STR$RIGHT


                                                .EXTRN  STR$ANALYZE_SDESC_R1

                                STR$RIGHT_R8::
                    50  D0 00000              MOVL    R0, R4                          ; 1381
                    01  DD 00003              PUSHL   #1                              ; 1451
                02  03  A1 91 00005          CMPB    3(SRC_DESC), #2                 ; 1456
                    09  1A 00009              BGTRU   1$
                50      61 3C 0000B          MOVZWL  (SRC_DESC), IN_LEN
                53  04  A1 D0 0000E          MOVL    4(SRC_DESC), IN_ADDR
                    0C  11 00012              BRB     2$
                50      51 D0 00014 1$:      MOVL    SRC_DESC, R0
                00000000G   00 16 00017      JSB     STR$ANALYZE_SDESC_R1
                53      51 D0 0001D          MOVL    R1, R3
                    52  D7 00020 2$:         DECL    R2                              ; 1467
                52      50 D1 00022          CMPL    IN_LEN, R2
                    0C  18 00025              BGEQ    3$
                6E 00000000G   8F D0 00027   MOVL    #STR$_ILLSTRPOS, RETURN_STATUS  ; 1470
                51      50 D0 0002E          MOVL    IN_LEN, LENGTH                  ; 1471
                    03  11 00031              BRB     4$
                51      52 D0 00033 3$:      MOVL    R2, LENGTH                      ; 1474
                    09  18 00036 4$:         BGEQ    5$                              ; 1477
                    51  D4 00038             CLRL    LENGTH                          ; 1479
                    6E 00000000G   8F D0 0003A  MOVL  #STR$_ILLSTRPOS, RETURN_STATUS ; 1481
                52      53  51 C1 00041 5$:  ADDL3   LENGTH, IN_ADDR, R2            ; 1491
                51      50  51 C3 00045      SUBL3   LENGTH, IN_LEN, R1              ; 1490
                        50  54 D0 00049      MOVL    DEST_DESC, R0                   ; 1491
                    00000000G   00 16 0004C  JSB     STR$COPY_R_R8
                        01  50 D1 00052      CMPL    COPY_STATUS, #1                 ; 1493
                            03 13 00055      BEQL    6$
                        6E  50 D0 00057      MOVL    COPY_STATUS, RETURN_STATUS      ; 1494
                        10  6E F8 0005A 6$:  BLBS    RETURN_STATUS, 7$               ; 1498
            04      6E      03  00 ED 0005D  CMPZV   #0, #3, RETURN_STATUS, #4
                            09  12 00062      BNEQ    7$
                            6E  DD 00064     PUSHL   RETURN_STATUS
                00000000G   00  01 FB 00066  CALLS   #1, LIB$STOP
                            50  8E D0 0006D 7$: MOVL  RETURN_STATUS, R0              ; 1499
                                05 00070     RSB                                     ; 1500

; Routine Size:  113 bytes,    Routine Base:  _STR$CODE + 000E

;  311          1501  1
;  312          1502  1 END                                     !End of module
;  313          1503  1
;  314          1504  0 ELUDOM
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _STR$CODE                        127  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         7         0       581         00:00.8
;
;
;
;                                   COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:STRRIGHT/OBJ=OBJ$:STRRIGHT MSRC$:STRRIGHT/UPDATE=(ENH$:STRRIGHT)
;
; Size:         127 code + 0 data bytes
; Run Time:       00:05.8
; Elapsed Time:   00:18.6
; Lines/CPU Min:  15585
; Lexemes/CPU-Min: 33730
; Memory Used:  82 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