

[illegible]

3-

Sy

LI

LI
LILI
LI
LILI
LI

LI

LI
LILI
LI
LILI
LI

LI

LI
LI

LI

LI
LI

LI

LI
LI

LI

11

LI

LI
LI

LI

LI
LI

21

LI

LI
LI

22

11

LI
LI

LI

LI
LI

LI

[illegible]


```
1 0001 0 MODULE STR$POS_EXTR (      ! Extract a substring by position
2 0002 0
3 0003 0      IDENT = '1-012' ! File: STRPOSEXT.B32  Edit:  RKR1012
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 BASIC-PLUS-2
38 0038 1 syntax. It finds the substring of a main string starting at the
39 0039 1 character position specified by the second input parameter and
40 0040 1 continues for the number of characters specified by the third
41 0041 1 input parameter. This substring is copied to the destination
42 0042 1 string.
43 0043 1
44 0044 1 ENVIRONMENT: User mode, AST level or not or mixed
45 0045 1
46 0046 1 AUTHOR: R. Will, CREATION DATE: 21-Feb-79
47 0047 1
48 0048 1 MODIFIED BY:
49 0049 1
50 0050 1 R. Will, 28-Feb-79: VERSION 01
51 0051 1 1-001 - Original
52 0052 1 1-002 - Change linkage and call to COPY routine. 15-Mar-79 RW
53 0053 1 1-003 - Make the string linkages start with STR$. JBS 04-JUN-1979
54 0054 1 1-004 - Change call to STR$$COPY. JBS 16-JUL-1979
55 0055 1 1-005 - Change name to STR$, string cleanup. RW 1-Nov-79
56 0056 1 1-006 - Correct psect name. JBS 02-NOV-1979
57 0057 1 1-007 - Change to new interlock macros. JBS 06-NOV-1979
```


STR\$POS_EXTR
1-012

F 1
16-Sep-1984 01:44:44
14-Sep-1984 12:40:12

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]STRPOSEXT.B32;1

Page 2
(1)

```

: 58      0058 1 | 1-008 - Use string interlocks in CALL entry. RW 15-Nov-79
: 59      0059 1 | 1-009 - String speedup, remove edit 8. RW 8-Jan-1980
: 60      0060 1 | 1-010 - Enhance routine to recognize additional classes of descriptors
: 61      0061 1 | by using $STR$GET_LEN_ADDR to extract length and address
: 62      0062 1 | of 1st byte of source descriptor. Remove string interlocking
: 63      0063 1 | code. RKR 21-APR-81.
: 64      0064 1 | 1-011 - Speed up code. RKR 7-OCT-1981.
: 65      0065 1 | 1-012 - Use STR$COPY R R8 for copying operation. Use
: 66      0066 1 | $STR$SIGNAL_FATAL instead of $STR$CHECK_STATUS.
: 67      0067 1 | RKR 18-NOV-1981.
: 68      0068 1 | --
: 69      0069 1 |
: 70      0070 1 | <BLF/PAGE>
```



```
72 0071 1 |
73 0072 1 | SWITCHES:
74 0073 1 |
75 0074 1 |
76 0075 1 | SWITCHES ADDRESSING MODE
77 0076 1 | (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
78 0077 1 |
79 0078 1 |
80 0079 1 | LINKAGES:
81 0080 1 |
82 0081 1 |
83 0082 1 | REQUIRE 'RTLIN:STRLNK'; ! Use require file with string linkage
84 0267 1 |
85 0268 1 |
86 0269 1 | TABLE OF CONTENTS:
87 0270 1 |
88 0271 1 |
89 0272 1 | FORWARD ROUTINE
90 0273 1 | STR$POS_EXTR, ! Find a substr by position, CALL
91 0274 1 | STR$POS_EXTR_R8 : STR$JSB_POS_EXT; ! find a substr by position, JSB
92 0275 1 |
93 0276 1 |
94 0277 1 | INCLUDE FILES:
95 0278 1 |
96 0279 1 |
97 0280 1 | REQUIRE 'RTLIN:RTLPSECT'; ! Declare PSECTs code
98 0375 1 |
99 0376 1 | REQUIRE 'RTLIN:STRMACROS'; ! use string macros to code
100 1292 1 |
101 1293 1 | LIBRARY 'RTLSTARLE'; ! STARLET library for macros and symbol
102 1294 1 |
103 1295 1 |
104 1296 1 | MACROS:
105 1297 1 |
106 1298 1 | NONE
107 1299 1 |
108 1300 1 | EQUATED SYMBOLS:
109 1301 1 |
110 1302 1 | NONE
111 1303 1 |
112 1304 1 | PSECT DECLARATIONS
113 1305 1 |
114 1306 1 |
115 1307 1 | DECLARE_PSECTS (STR);
116 1308 1 |
117 1309 1 |
118 1310 1 | OWN STORAGE:
119 1311 1 |
120 1312 1 | NONE
121 1313 1 |
122 1314 1 | EXTERNAL REFERENCES:
123 1315 1 |
124 1316 1 |
125 1317 1 | EXTERNAL ROUTINE
126 1318 1 | STR$COPY_R_R8 : STR$JSB_COPY_R , ! to do copying
127 1319 1 | LIB$STOP; ! signal fatal errors
128 1320 1 |
```


STR\$POS_EXTR
1-012

H 1
16-Sep-1984 01:44:44
14-Sep-1984 12:40:12

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]STRPOSEXT.B32;1

Page 4
(2)

```
: 129      1321 1 EXTERNAL LITERAL
: 130      1322 1 STR$NORMAL,
: 131      1323 1 STR$_ILLSTR$PE,
: 132      1324 1 STR$_ILLSTRPOS;
```

```
: successful completion
: illegal string specification
: illegal string position
```



```
134 1325 1 GLOBAL ROUTINE STR$POS_EXTR (      ! substring by position
135 1326 1
136 1327 1     DEST_DESC,      ! Pointer to destination descriptor
137 1328 1     SRC_DESC,      ! Pointer to source descriptor
138 1329 1     START_POS,    ! First character to be included
139 1330 1     END_POS      ! Last character position to include
140 1331 1
141 1332 1     ) =
142 1333 1
143 1334 1 ++
144 1335 1 FUNCTIONAL DESCRIPTION:
145 1336 1
146 1337 1     This routine extracts the characters starting at the
147 1338 1     character position in the source string specified by the 3rd
148 1339 1     input parameter and continuing to the character position in the
149 1340 1     source string specified by the 4th input parameter, and copies
150 1341 1     that substring to the destination string (by JSB to
151 1342 1     STR$COPY_R_R8) according to the syntax of the class of the
152 1343 1     destination string.
153 1344 1     If the starting position is < 1, 1 is used. If the ending
154 1345 1     position is > length of the source string, the length of the
155 1346 1     source string is used. If the starting position > the ending
156 1347 1     position, a null string is returned. The call entry point is
157 1348 1     implemented by JSBing to the JSB entry point.
158 1349 1
159 1350 1 FORMAL PARAMETERS:
160 1351 1
161 1352 1     DEST_DESC.wt.dx    pointer to destination string descriptor
162 1353 1     SRC_DESC.rt.dx     pointer to source string descriptor
163 1354 1     START_POS.rl.r     character position in src to start
164 1355 1                     substring
165 1356 1     END_POS.rl.r       character position in src to end
166 1357 1                     substring
167 1358 1
168 1359 1 IMPLICIT INPUTS:
169 1360 1
170 1361 1     NONE
171 1362 1
172 1363 1 IMPLICIT OUTPUTS:
173 1364 1
174 1365 1     NONE
175 1366 1
176 1367 1 COMPLETION CODES:
177 1368 1
178 1369 1     $$$ NORMAL      Success
179 1370 1     STR$_ILLSTRPOS  Character position reference outside of string
180 1371 1     STR$_ILLSTRSPE  End pos < start pos, or length too long
181 1372 1     STR$_NEGSTRLEN  Negative length supplied, 0 used
182 1373 1     STR$_TRU        Truncation occurred in copying to destination
183 1374 1                     (Warning)
184 1375 1
185 1376 1 SIDE EFFECTS:
186 1377 1
187 1378 1     May signal:
188 1379 1     STR$_FATINTERR  Fatal internal error
189 1380 1     STR$_ILLSTRCLA  Illegal (or unsupported) string class
190 1381 1     STR$_INSVIRMEM  Insufficient virtual memory for
```


STR\$POS_EXTR
1-012

J 1
16-Sep-1984 01:44:44
14-Sep-1984 12:40:12

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]STRPOSEXT.B32;1

Page 6
(3)

reallocation of dynamic string

191 1382 1
192 1383 1
193 1384 1
194 1385 1
195 1386 2
196 1387 2
197 1388 2
198 1389 2
199 1390 2
200 1391 2
201 1392 2
202 1393 2
203 1394 2
204 1395 2
205 1396 1

BEGIN

MAP

SRC_DESC : REF \$STR\$DESCRIPTOR,
DEST_DESC : REF \$STR\$DESCRIPTOR;

RETURN STR\$POS_EXTR_R8 (DEST_DESC [0,0,0,0],
SRC_DESC [0,0,0,0],
..START_POS,
..END_POS);

END;

!End of STR\$POS_EXTR

.TITLE STR\$POS_EXTR
.IDENT \1-012\

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

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

.ENTRY STR\$POS_EXTR, Save R2,R3,R4,R5,R6,R7,R8
MOVL @END_POS, R3
MOVL @START_POS, R2
MOVQ DEST_DESC, R0
BSBW STR\$POS_EXTR_R8
RET

: 1325
: 1393
:
:
: 1396

53 10 BC D0 00002
52 0C BC D0 00006
50 04 AC 7D 0000A
0000V 30 0000E
04 00011

; Routine Size: 18 bytes, Routine Base: _STR\$CODE + 0000


```
207 1397 1 GLOBAL ROUTINE STR$POS_EXTR_R8 (      ! extract substring by position
208 1398 1
209 1399 1      DEST_DESC,      ! Pointer to destination descriptor
210 1400 1      SRC_DESC,      ! Pointer to source descriptor
211 1401 1      START_POS,      ! First character to be included
212 1402 1      END_POS        ! Last character position to include
213 1403 1
214 1404 1      ) : STR$JSB_POS_EXT =
215 1405 1
216 1406 1  !++
217 1407 1  FUNCTIONAL DESCRIPTION:
218 1408 1
219 1409 1      This routine extracts the characters starting at the
220 1410 1      character position in the source string specified by the 3rd
221 1411 1      input parameter and continuing to the character position in the
222 1412 1      source string specified by the 4th input parameter, and copies
223 1413 1      that substring to the destination string (by JSB to
224 1414 1      STR$COPY_R_R8) according to the syntax of the class of the
225 1415 1      destination string.
226 1416 1      If the starting position is < 1, 1 is used. If the ending
227 1417 1      position is > length of the source string, the length of the
228 1418 1      source string is used. If the starting position > the ending
229 1419 1      position, a null string is returned.
230 1420 1
231 1421 1  FORMAL PARAMETERS:
232 1422 1
233 1423 1      DEST_DESC.wt.dx      pointer to destination string descriptor
234 1424 1      SRC_DESC.rt.dx       pointer to source string descriptor
235 1425 1      START_POS.rl.v     character position in src to start
236 1426 1                      substring
237 1427 1      END_POS.rl.v        character position in src to end
238 1428 1                      substring
239 1429 1
240 1430 1  IMPLICIT INPUTS:
241 1431 1
242 1432 1      NONE
243 1433 1
244 1434 1  IMPLICIT OUTPUTS:
245 1435 1
246 1436 1      NONE
247 1437 1
248 1438 1  COMPLETION CODES:
249 1439 1
250 1440 1      SSS_NORMAL          Success
251 1441 1      STR$_ILLSTRPOS    Character position reference outside of string
252 1442 1      STR$_ILLSTRSPE    End pos < start pos, or length too long
253 1443 1      STR$_NEGSTRLEN    Negative length supplied, 0 used
254 1444 1      STR$_TRU          Truncation occurred in copying to destination
255 1445 1                      (Warning)
256 1446 1
257 1447 1  SIDE EFFECTS:
258 1448 1
259 1449 1      May signal:
260 1450 1      STR$_FATINTERR      Fatal internal error
261 1451 1      STR$_ILLSTRCLA     Illegal (or unsupported) string class
262 1452 1      STR$_INSVIRMEM     Insufficient virtual memory for
263 1453 1                      reallocation of dynamic string
```



```
264 1454 1 1 1--
265 1455 1 1 1--
266 1456 1 1 1--
267 1457 2 BEGIN
268 1458 2
269 1459 2 LOCAL
270 1460 2 IN_LEN, ! length of source string
271 1461 2 IN_ADDR, ! addr of 1st byte of source
272 1462 2 ACTUAL_START, ! MAX (1, .START_POS)
273 1463 2 LENGTH, ! length to copy
274 1464 2 COPY_STATUS, ! status returned by copy
275 1465 2 RETURN_STATUS; ! save status for this routine
276 1466 2
277 1467 2 MAP
278 1468 2 SRC_DESC : REF $STR$DESCRIPTOR,
279 1469 2 DEST_DESC : REF $STR$DESCRIPTOR;
280 1470 2
281 1471 2 RETURN_STATUS = 1 ; ! Assume success to follow
282 1472 2
283 1473 2 !+
284 1474 2 Extract length and address of 1st byte of source string.
285 1475 2 !-
286 1476 2 $STR$GET_LEN_ADDR ( SRC_DESC, IN_LEN, IN_ADDR ) ;
287 1477 2
288 1478 2 ACTUAL_START = .START_POS;
289 1479 2
290 1480 2 !+
291 1481 2 length is >=0 and the smaller of the
292 1482 2 (input end_pos and src len) - start pos (must be >0) + 1
293 1483 2 !-
294 1484 2 LENGTH = 1 + ! length = end - start + 1
295 1485 2 BEGIN
296 1486 2 IF .IN_LEN LSS .END_POS
297 1487 2 THEN
298 1488 2 BEGIN
299 1489 2 RETURN_STATUS = STR$_ILLSTRPOS; ! END_POS is too large
300 1490 2
301 1491 2 .IN_LEN ! use srclen, remember
302 1492 2 ! error
303 1493 2 END
304 1494 2 ELSE
305 1495 2 .END_POS ! use specified
306 1496 2 END
307 1497 2
308 1498 2 -
309 1499 2
310 1500 2 BEGIN
311 1501 2 IF .START_POS LSS 1
312 1502 2 THEN
313 1503 2 BEGIN
314 1504 2 RETURN_STATUS = STR$_ILLSTRPOS; ! START_POS is too small
315 1505 2 ACTUAL_START = 1; ! use 1, remember error
316 1506 2 1
317 1507 2 END
318 1508 2 ELSE
319 1509 2 .START_POS ! use specified
320 1510 2 END;
```



```

321 1511 2
322 1512 2
323 1513 2
324 1514 2
325 1515 2
326 1516 2
327 1517 2
328 1518 2
329 1519 2
330 1520 2
331 1521 2
332 1522 2
333 1523 2
334 1524 2
335 1525 2
336 1526 2
337 1527 2
338 1528 2
339 1529 2
340 1530 2
341 1531 2
342 1532 2
343 1533 2
344 1534 2
345 1535 2
346 1536 2
347 1537 1

IF .LENGTH LSS 0 THEN
    BEGIN
        LENGTH = 0;
        RETURN_STATUS = STR$_ILLSTRSPE;
    END;

! START_POS > END_POS
! null string, remember
! error

!+
Copy to destination descriptor for length as computed above.
Input pointer is the sum - 1 of the source pointer and input
start position ( which must be > 0).

COPY_STATUS =
    STR$COPY_R_R8 ( .DEST_DESC, .LENGTH,
        CH$PLOS (.IN_ADDR, .ACTUAL_START - 1) );

IF .COPY_STATUS NEQ SS$ NORMAL
THEN RETURN_STATUS = .COPY_STATUS;

! copy truncated,
! return truncate
! instead of previous
! status

$STR$SIGNAL_FATAL (RETURN_STATUS);
RETURN .RETURN_STATUS;
END;

! signal fatal errors
!End of STR$POS_EXTR_R8
```

```

.EXTRN STR$ANALYZE_SDESC_R1

56 50 D0 00000 STR$POS_EXTR_R8::
01 DD 00003
02 03 A1 91 00005
09 1A 00009
50 61 3C 0000B
55 04 A1 D0 0000E
0C 11 00012
50 51 D0 00014 1$:
00000000G 00 16 00017
55 51 D0 0001D
54 52 D0 00020 2$:
53 50 D1 00023
09 18 00026
6E 00000000G 8F D0 00028
03 11 0002F
50 53 D0 00031 3$:
52 D5 00034 4$:
0D 14 00036
6E 00000000G 8F D0 00038
54 01 D0 0003F
52 01 D0 00042
50 52 C2 00045 5$:
51 01 A0 9E 00048
09 18 0004C

MOVL R0, R6
PUSHL #1
CMPB 3(SRC_DESC), #2
BGTRU 1$
MOVZWL (SRC_DESC), IN_LEN
MOVL 4(SRC_DESC), IN_ADDR
BRB 2$
MOVL SRC_DESC, R0
JSB STR$ANALYZE_SDESC_R1
MOVL R1, R5
MOVL START_POS, ACTUAL_START
CMPL IN_LEN, END_POS
BGEQ 3$
MOVL #STR$_ILLSTRPOS, RETURN_STATUS
BRB 4$
MOVL END_POS, R0
TSTL START_POS
BGTR 5$
MOVL #STR$_ILLSTRPOS, RETURN_STATUS
MOVL #1, ACTUAL_START
MOVL #1, R2
SUBL2 R2, R0
MOVAB 1(R0), LENGTH
BGEQ 6$

: 1397
: 1471
: 1476
:
:
:
: 1478
: 1486
:
: 1489
: 1491
: 1495
: 1501
:
: 1504
: 1505
: 1503
: 1500
: 1498
: 1512
```


STR\$POS_EXTR
1-012

N 1
16-Sep-1984 01:44:44 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:40:12 [LIBRTL.SRC]STRPOSEXT.B32;1

Page 10
(4)

			51	D4	0004E	CLRL	LENGTH	: 1514		
	6E	00000000G	8F	D0	00050	MOVL	#STR\$_ILLSTRSPE, RETURN_STATUS	: 1515		
	52	FF A445	9E	00057	6\$:	MOVAB	-1(ACTUAL_START)[IN_ADDR], R2	: 1527		
	50		56	D0	0005C	MOVL	DEST_DESC, R0			
		00000000G	00	16	0005F	JSB	STR\$COPY_R R8			
	01		50	D1	00065	CMPL	COPY_STATUS, #1	: 1529		
			03	13	00068	BEQL	7\$			
	6E		50	D0	0006A	MOVL	COPY_STATUS, RETURN_STATUS	: 1530		
04			6E	E8	0006D	7\$:	BLBS	RETURN_STATUS, 8\$	: 1535	
	10		00	ED	00070	CMPZV	#0, #3, RETURN_STATUS, #4			
	03		09	12	00075	BNEQ	8\$			
			6E	DD	00077	PUSHL	RETURN_STATUS			
		00000000G	00	01	FB	00079	CALLS	#1, LIB\$STOP		
			50	8E	D0	00080	8\$:	MOVL	RETURN_STATUS, R0	: 1536
				05	00083	RSB		: 1537		

; Routine Size: 132 bytes, Routine Base: _STR\$CODE + 0012

:	348	1538	1		
:	349	1539	1	END	!End of module
:	350	1540	1		
:	351	1541	0	ELUDOM	

PSECT SUMMARY

Name	Bytes	Attributes
_STR\$CODE	150	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	7	0	581	00:00.7

COMMAND QUALIFIERS

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

; Size: 150 code + 0 data bytes

STR\$POS_EXTR
1-012

^{B 2}
16-Sep-1984 01:44:44

VAX-11 Bliss-32 V4.0-742

Page 11

; Run Time: 00:06.1
; Elapsed Time: 00:31.3
; Lines/CPU Min: 15257
; Lexemes/CPU-Min: 32683
; Memory Used: 87 pages
; Compilation Complete

0215 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

