

[illegible]

[illegible]


```
1 0001 0 MODULE STR$LEFT ( ! Extract a substring from the left
2 0002 0
3 0003 0 IDENT = '1-011' ! File: STRLEFT.B32 Edit: RKR1011
4 0004 0
5 0005 0 ) =
6 0006 1 BEGIN
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 * ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 * TRANSFERRED.
20 0020 1 *
21 0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 * CORPORATION.
24 0024 1 *
25 0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *
29 0029 1 *****
30 0030 1
31 0031 1
32 0032 1 ++
33 0033 1 FACILITY: String support library
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1 This module extracts a substring according to the
38 0038 1 BASIC-PLUS-2 syntax. It finds the substring of a main string
39 0039 1 starting at the left end of the string (character position 1)
40 0040 1 and continues through the nth character of the string. This
41 0041 1 substring is copied to the destination string.
42 0042 1
43 0043 1 ENVIRONMENT: User mode, AST level or not or mixed
44 0044 1
45 0045 1 AUTHOR: R. Will, CREATION DATE: 19-Feb-79
46 0046 1
47 0047 1 MODIFIED BY:
48 0048 1
49 0049 1 R. Will, 19-Feb-79: VERSION 01
50 0050 1 1-001 - Original
51 0051 1 1-002 - Change linkage and call to COPY routine. 13-Mar-79 RW
52 0052 1 1-003 - Change string linkages to start with STR$. JBS 04-JUN-1979
53 0053 1 1-004 - Change call to STR$$COPY. JBS 16-JUL-1979
54 0054 1 1-005 - String cleanup. Change name to STR$. RW 31-Oct-79
55 0055 1 1-006 - Change to new interlock macros. JBS 06-NOV-1979
56 0056 1 1-007 - The new interlock macros cannot be used in routines called
57 0057 1 with a JSB instruction. JBS 15-NOV-1979
```

STR\$LEFT
1-011

K 11
16-Sep-1984 01:40:48
14-Sep-1984 12:40:07

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

Page 2
(1)

```

: 58      0058 1 | 1-008 - String speedup, undo edit 7. RW 7-Jan-1980
: 59      0059 1 | 1-009 - Enhance to accomodate additional classes of descriptors by
: 60      0060 1 | using $STR$GET_LEN_ADDR to extract length and 1st data
: 61      0061 1 | byte from descriptor. Remove string interlocking code.
: 62      0062 1 | RKR 21-APR-81
: 63      0063 1 | 1-010 - Speed up code. RKR 7-OCT-1981.
: 64      0064 1 | 1-011 - Use STR$COPY R R8 to do the copy. Use $STR$SIGNAL_FATAL
: 65      0065 1 | instead of $STR$CHECKSTATUS. RKR 18-NOV-1981.
: 66      0066 1 | --
: 67      0067 1 |
: 68      0068 1 | <BLF/PAGE>
```


STR\$LEFT
1-011

L 11
16-Sep-1984 01:40:48
14-Sep-1984 12:40:07

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

Page 3
(2)

```

70 0069 1 |
71 0070 1 | SWITCHES:
72 0071 1 |
73 0072 1 |
74 0073 1 | SWITCHES ADDRESSING MODE
75 0074 1 | (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
76 0075 1 |
77 0076 1 |
78 0077 1 | LINKAGES:
79 0078 1 |
80 0079 1 |
81 0080 1 | REQUIRE 'RTLIN:STRLNK'; ! Use require file with string linkage
82 0265 1 |
83 0266 1 |
84 0267 1 | TABLE OF CONTENTS:
85 0268 1 |
86 0269 1 |
87 0270 1 | FORWARD ROUTINE
88 0271 1 | STR$LEFT, ! Find the LEFT of a string, CALL
89 0272 1 | STR$LEFT_R8 : STR$JSB_LEFT; ! Find the LEFT of a string, JSB
90 0273 1 |
91 0274 1 |
92 0275 1 | INCLUDE FILES:
93 0276 1 |
94 0277 1 |
95 0278 1 | REQUIRE 'RTLIN:RTLPSECT'; ! Declare PSECTS code
96 0373 1 |
97 0374 1 | REQUIRE 'RTLIN:STRMACROS'; ! use string macros to code
98 1290 1 |
99 1291 1 | LIBRARY 'RTLSTARLE'; ! STARLET library for macros and symbol
100 1292 1 |
101 1293 1 |
102 1294 1 | MACROS:
103 1295 1 |
104 1296 1 | NONE
105 1297 1 |
106 1298 1 | EQUATED SYMBOLS:
107 1299 1 |
108 1300 1 | NONE
109 1301 1 |
110 1302 1 | PSECT DECLARATIONS:
111 1303 1 |
112 1304 1 | DECLARE_PSECTS (STR);
113 1305 1 |
114 1306 1 | OWN STORAGE:
115 1307 1 |
116 1308 1 | NONE
117 1309 1 |
118 1310 1 | EXTERNAL REFERENCES:
119 1311 1 |
120 1312 1 |
121 1313 1 | EXTERNAL ROUTINE
122 1314 1 | STR$COPY_R_R8 : STR$JSB_COPY_R ; ! Routine to do the copying
123 1315 1 |
124 1316 1 | EXTERNAL LITERAL
125 1317 1 | STR$_NORMAL, ! successful completion
126 1318 1 | STR$_ILLSTRPOS; ! error status
```

STR\$LEFT
1-011

: 127

1319 1

M 11
16-Sep-1984 01:40:48
14-Sep-1984 12:40:07

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

Page 4
(2)

ST
1-


```
129 1320 1 GLOBAL ROUTINE STR$LEFT (          ! extract the left substring
130 1321 1
131 1322 1     DEST_DESC,          ! Pointer to destination descriptor
132 1323 1     SRC_DESC,        ! Pointer to source descriptor
133 1324 1     END_POS         ! Last character to be included
134 1325 1
135 1326 1          ) =
136 1327 1
137 1328 1 ++
138 1329 1 FUNCTIONAL DESCRIPTION:
139 1330 1
140 1331 1     This routine extracts the characters starting at the leftmost
141 1332 1     character (character position 1) and continuing through the
142 1333 1     character position specified by the input and copies that
143 1334 1     substring to the destination string (by JSB to STR$COPY_R_R8)
144 1335 1     according to the syntax of the class of the destination string.
145 1336 1     If the input character position is > the length of the input
146 1337 1     string, then the length of the input string is used. If the
147 1338 1     input character position is < 1, the destination becomes a null
148 1339 1     string.
149 1340 1     The call entry point executes a JSB to the JSB entry point.
150 1341 1
151 1342 1 FORMAL PARAMETERS:
152 1343 1
153 1344 1     DEST_DESC.wt.dx      pointer to destination string descriptor
154 1345 1     SRC_DESC.rt.dx       pointer to source string descriptor
155 1346 1     END_POS.rl.r        last character position to include
156 1347 1
157 1348 1 IMPLICIT INPUTS:
158 1349 1
159 1350 1     NONE
160 1351 1
161 1352 1 IMPLICIT OUTPUTS:
162 1353 1
163 1354 1     NONE
164 1355 1
165 1356 1 COMPLETION CODES:
166 1357 1
167 1358 1     any of the codes returned by the JSB entry point
168 1359 1
169 1360 1 SIDE EFFECTS:
170 1361 1
171 1362 1     JSBs to the JSB entry point so may signal any of its errors or
172 1363 1     have any of its side effects.
173 1364 1
174 1365 1 --
175 1366 1
176 1367 2 BEGIN
177 1368 2
178 1369 2 MAP
179 1370 2     SRC_DESC : REF $STR$DESCRIPTOR,
180 1371 2     DEST_DESC : REF $STR$DESCRIPTOR;
181 1372 2
182 1373 2 RETURN STR$LEFT_R8 (DEST_DESC [0,0,0,0],
183 1374 2     SRC_DESC [0,0,0,0],
184 1375 2     ..END_POS);
185 1376 1 END;                                     !End of STR$LEFT
```

STR\$LEFT
1-011

B 12
16-Sep-1984 01:40:48
14-Sep-1984 12:40:07

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

Page 6
(3)

.TITLE STR\$LEFT
.IDENT \1-011\

.EXTRN STR\$COPY_R R8, STR\$_NORMAL
.EXTRN STR\$_ILLSTRPOS

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

.ENTRY STR\$LEFT, Save R2,R3,R4,R5,R6,R7,R8
MOVL @END_POS, R2
MOVQ DEST_DESC, R0
BSBW STR\$LEFT_R8
RET

; 1320
; 1374
;
; 1376

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

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


```
187 1377 1 GLOBAL ROUTINE STR$LEFT_R8 (      ! extract the left substring
188 1378 1
189 1379 1     DEST_DESC,      ! Pointer to destination descriptor
190 1380 1     SRC_DESC,      ! Pointer to source descriptor
191 1381 1     END_POS      ! Last character to be included
192 1382 1
193 1383 1     ) : STR$JSB_LEFT =
194 1384 1
195 1385 1 ++
196 1386 1 FUNCTIONAL DESCRIPTION:
197 1387 1
198 1388 1     This routine extracts the characters starting at the leftmost
199 1389 1     character (character position 1) and continuing through the
200 1390 1     character position specified by the input and copies that
201 1391 1     substring to the destination string (by JSB to STR$COPY_R_R8)
202 1392 1     according to the syntax of the class of the destination string.
203 1393 1     If the input character position is > the length of the input
204 1394 1     string, then the length of the input string is used. If the
205 1395 1     input character position is < 1, the destination becomes a null
206 1396 1     string.
207 1397 1
208 1398 1 FORMAL PARAMETERS:
209 1399 1
210 1400 1     DEST_DESC.wt.dx      pointer to destination string descriptor
211 1401 1     SRC_DESC.rt.dx      pointer to source string descriptor
212 1402 1     END_POS.rl.v      value of last character position to
213 1403 1                     include
214 1404 1
215 1405 1 IMPLICIT INPUTS:
216 1406 1
217 1407 1     NONE
218 1408 1
219 1409 1 IMPLICIT OUTPUTS:
220 1410 1
221 1411 1     NONE
222 1412 1
223 1413 1 COMPLETION CODES:
224 1414 1
225 1415 1     SSS NORMAL          Success
226 1416 1     STR$_ILLSTRPOS      Character position reference outside of string
227 1417 1     STR$_ILLSTRSPE      End pos < start pos, or length too long
228 1418 1     STR$_NEGSTRLEN      Negative length supplied, 0 used
229 1419 1     STR$_TRU           Truncation occurred in copying to destination
230 1420 1                     (Warning)
231 1421 1
232 1422 1 SIDE EFFECTS:
233 1423 1
234 1424 1     May signal:
235 1425 1     STR$_FATINTERR      Fatal internal error
236 1426 1     STR$_ILLSTRCLA      Illegal (or unsupported) string class
237 1427 1     STR$_INSVIRMEM      Insufficient virtual memory for
238 1428 1                       reallocation of dynamic string
239 1429 1
240 1430 1 --
241 1431 1
242 1432 2 BEGIN
243 1433 2
```

```

244      1434 2 LOCAL
245      1435 2     IN_LEN,           ! length of source string
246      1436 2     IN_ADDR,         ! addr of 1st byte of source data
247      1437 2     COPY_STATUS,      ! status from STR$COPY_R_R8
248      1438 2     RETURN_STATUS,    ! keep track of return status
249      1439 2     COPY_LENGTH;       ! length to be copied
250      1440 2
251      1441 2 MAP
252      1442 2     SRC_DESC : REF $STR$DESCRIPTOR,
253      1443 2     DEST_DESC : REF $STR$DESCRIPTOR;
254      1444 2
255      1445 2     RETURN_STATUS = 1 ;           ! assume success to follow
256      1446 2
257      1447 2 !+
258      1448 2 !- Calculate the length and address of first byte of source string.
259      1449 2 !-
260      1450 2     $STR$GET_LEN_ADDR ( SRC_DESC, IN_LEN, IN_ADDR ) ;
261      1451 2
262      1452 2 !+
263      1453 2 !- Compute the correct substring and use the length and address of
264      1454 2 !- the substring to copy to dest. The length must be greater than or
265      1455 2 !- equal to 0 and is the smaller of the length specified by END_POS which
266      1456 2 !- was input and the source string length. Return STR$_ILLSTRPOS if
267      1457 2 !- END_POS does not meet those criteria.
268      1458 2 !-
269      1459 2
270      1460 2     COPY_LENGTH =           ! compute the length
271      1461 3     BEGIN
272      1462 3     IF .IN_LEN LSS .END_POS
273      1463 3     THEN
274      1464 4     BEGIN
275      1465 4     RETURN_STATUS = STR$_ILLSTRPOS; ! input length is too long
276      1466 4
277      1467 4     .IN_LEN           ! use srclen and remember
278      1468 4                     ! error
279      1469 4     END
280      1470 4
281      1471 3     ELSE
282      1472 3     .END_POS           ! use specified end position
283      1473 3     END;
284      1474 2
285      1475 2     IF .COPY_LENGTH LSS 0 THEN
286      1476 3     BEGIN
287      1477 3     COPY_LENGTH = 0;           ! input length is negative
288      1478 3     RETURN_STATUS = STR$_ILLSTRPOS; ! use 0 and remember error
289      1479 3     END;
290      1480 2
291      1481 2     COPY_STATUS = STR$COPY_R_R8 ( .DEST_DESC, .COPY_LENGTH, .IN_ADDR );
292      1482 2
293      1483 2     IF .COPY_STATUS NEQ SS$ NORMAL
294      1484 2     THEN RETURN_STATUS = .COPY_STATUS; ! copy truncated, or severe
295      1485 2                                     ! error, return new status
296      1486 2                                     ! instead of previous value
297      1487 2
298      1488 2     $STR$SIGNAL FATAL (RETURN_STATUS); ! if fatal error, signal
299      1489 2     RETURN .RETURN_STATUS;
300      1490 1     END;           !End of STR$LEFT
```


				.EXTRN	STR\$ANALYZE_SDESC_R1	
				.EXTRN	LIB\$STOP	
54		50	D0 00000	STR\$LEFT R8::		
		01	DD 00003	MOVL	R0, R4	: 1377
02	03	A1	91 00005	PUSHL	#1	: 1445
		09	1A 00009	CMPB	3(SRC_DESC), #2	: 1450
50		61	3C 0000B	BGTRU	1\$	
53	04	A1	D0 0000E	MOVZWL	(SRC_DESC), IN_LEN	
		0C	11 00012	MOVL	4(SRC_DESC), IN_ADDR	
50		51	D0 00014	BRB	2\$	
	00000000G	00	16 00017	MOVL	SRC_DESC, R0	
53		51	D0 0001D	JSB	STR\$ANALYZE_SDESC_R1	
52		50	D1 00020	MOVL	R1, R3	
		0C	18 00023	CML	IN_LEN, END_POS	: 1462
6E	00000000G	8F	D0 00025	BGEQ	3\$	
51		50	D0 0002C	MOVL	#STR\$ ILLSTRPOS, RETURN_STATUS	: 1465
		03	11 0002F	MOVL	IN_LEN, COPY_LENGTH	: 1467
51		52	D0 00031	BRB	4\$	
		09	18 00034	MOVL	END_POS, COPY_LENGTH	: 1472
		51	D4 00036	BGEQ	5\$	: 1475
6E	00000000G	8F	D0 00038	CLRL	COPY_LENGTH	: 1477
52		53	D0 0003F	MOVL	#STR\$ ILLSTRPOS, RETURN_STATUS	: 1478
50		54	D0 00042	MOVL	IN_ADDR, R2	: 1481
	00000000G	00	16 00045	MOVL	DEST_DESC, R0	
01		50	D1 0004B	JSB	STR\$COPY R R8	
		03	13 0004E	CML	COPY_STATUS, #1	: 1483
6E		50	D0 00050	BEQL	6\$	
10		6E	E8 00053	MOVL	COPY_STATUS, RETURN_STATUS	: 1484
03		00	ED 00056	BLBS	RETURN_STATUS, 7\$	: 1488
		09	12 0005B	CMPZV	#0, #3, RETURN_STATUS, #4	
		6E	DD 0005D	BNEQ	7\$	
	00000000G	00	01 FB 0005F	PUSHL	RETURN_STATUS	
50		8E	D0 00066	CALLS	#1, LIB\$STOP	
		05	00069	MOVL	RETURN_STATUS, R0	: 1489
				RSB		: 1490

; Routine Size: 106 bytes, Routine Base: _STR\$CODE + 000E

:	301	1491	1
:	302	1492	1 END
:	303	1493	1
:	304	1494	0 ELUDOM

!End of module

PSECT SUMMARY

Name	Bytes	Attributes
------	-------	------------

STR\$LEFT
1-011

F 12
16-Sep-1984 01:40:48
14-Sep-1984 12:40:07

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

Page 10
(4)

: _STR\$CODE

120 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	00:00.8

COMMAND QUALIFIERS

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

: Size: 120 code + 0 data bytes
: Run Time: 00:05.6
: Elapsed Time: 00:27.9
: Lines/CPU Min: 16151
: Lexemes/CPU-Min: 34929
: Memory Used: 80 pages
: Compilation Complete

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

