

[illegible]

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLL IIIII
LLLLLLLLLL IIIII
SSSSSSSS
SSSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
SSSSSSSS
SSSSSSSS

```



```

1 0001 0 MODULE FOR$$UDF_RU (
2 0002 0 IDENT = '1-013'
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1 ++
30 0030 1 FACILITY: FORTRAN Support Library - not user callable
31 0031 1
32 0032 1 ABSTRACT:
33 0033 1
34 0034 1 This module implements FORTRAN Read unformatted I/O
35 0035 1 statements (sequential access - S, direct access - D,
36 0036 1 at the User data Formatter level of
37 0037 1 abstraction (UDF level is 2nd level). This module
38 0038 1 calls the appropriate Read record
39 0039 1 routine at the record handling level of abstraction (REC
40 0040 1 level is 3rd level) to Read a record.
41 0041 1
42 0042 1 ENVIRONMENT: User access mode; reentrant AST level or not.
43 0043 1
44 0044 1 AUTHOR: Thomas N. Hastings; CREATION DATE: 31-Aug-77
45 0045 1
46 0046 1 MODIFIED BY:
47 0047 1 Thomas N. Hastings, 31-Aug-77: Version 01
48 0048 1 01 - original
49 0049 1 0-3 - Removed parameters to record level routines JMT 17-OCT-77
50 0050 1 0-5 - Global register CCB. JMT 8-Apr-78
51 0051 1 0-06 - Change REQUIRE files for VAX system build. DGP 28-Apr-78
52 0052 1 0-07 - Use JSB linkages. TNH 22-May-78
53 0053 1 0-08 - PIC dispatch tables. TNH 7-June-78
54 0054 1 0-09 - Change file name to FORUDFRU.B32, and change the names of
55 0055 1 the REQUIRE files similarly. JBS 14-NOV-78
56 0056 1 1-001 - Update the version number and copyright notice. JBS 16-NOV-78
57 0057 1 1-002 - Change the REQUIRE file names from FOR... to OTS... JBS 06-DEC-78

```

! FORTRAN Read unformatted UDF
! File: FORUDFRU.B32 Edit: JAW1013

```
: 58      0058 1 1-003 - Change references to I/O dispatch tables. DGP 08-Dec-78
: 59      0059 1 1-004 - Change dispatch table references to longwords. DGP 11-Dec-78
: 60      0060 1 1-005 - Change ISB$A_BUF_BEG, BUF_PTR, BUF_END, BUF_HIGH to LUB. DGP 08-Jan-79
: 61      0061 1 1-006 - Use 32-bit addresses for externals. JBS 27-JAN-1979
: 62      0062 1 1-007 - Use language-specific dispatch tables. JBS 26-JUN-1979
: 63      0063 1 1-008 - Use ISB dispatch offsets. SBL 12-July-1979
: 64      0064 1 1-009 - Give error on requesting too much data if not segmented.
: 65      0065 1 SBL 18-Sept-1979
: 66      0066 1 1-010 - Implement unbuffered transfers for single-element lists: defer
: 67      0067 1 REC-level initialization until first element is requested;
: 68      0068 1 then if recordtype is not segmented and record size permits,
: 69      0069 1 set up RAB so record will be read directly into element, and
: 70      0070 1 do not copy any data from buffer to element. JAW 06-May-1981
: 71      0071 1 1-011 - Continuation of 1-010. Do not attempt an unbuffered transfer
: 72      0072 1 for a keyed read. Set LUB$A_BUF_PTR when deferring REC-level
: 73      0073 1 initialization. Support recordtype=variable. JAW 02-Jun-1981
: 74      0074 1 1-012 - Continuation of 1-010. Handle sequential read correctly when
: 75      0075 1 access=keyed. Add minor optimizations. JAW 06-Jun-1981
: 76      0076 1 1-013 - Add test for ISB$V_NEED_INIT in FOR$$UDF_RU1. JAW 06-Jun-1981
: 77      0077 1 --
: 78      0078 1
: 79      0079 1 !<BLF/PAGE>
```



```

81 0080 1 |
82 0081 1 | SWITCHES:
83 0082 1 |
84 0083 1 |
85 0084 1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
86 0085 1 |
87 0086 1 |
88 0087 1 | LINKAGES:
89 0088 1 |
90 0089 1 |
91 0090 1 REQUIRE 'RTLIN:OTSLNK'; ! Define all linkages
92 0519 1 |
93 0520 1 |
94 0521 1 | TABLE OF CONTENTS:
95 0522 1 |
96 0523 1 |
97 0524 1 FORWARD ROUTINE
98 0525 1     FOR$$UDF_RU0 : JSB_UDF0 NOVALUE, ! initialization
99 0526 1     FOR$$UDF_RU1 : CAL_CCB NOVALUE, ! format one user I/O list element
100 0527 1     FOR$$UDF_RU9 : JSB_UDF9 NOVALUE; ! end of user I/O list - finish
101 0528 1 |
102 0529 1 |
103 0530 1 | INCLUDE FILES:
104 0531 1 |
105 0532 1 |
106 0533 1 REQUIRE 'RTLML:FORERR'; ! FORTRAN error codes
107 0601 1 REQUIRE 'RTLML:OTSLUB'; ! Logical unit block (LUB) offsets
108 0741 1 |
109 0742 1 REQUIRE 'RTLML:OTSISB'; ! I/O statement block (ISB) offsets
110 0910 1 |
111 0911 1 REQUIRE 'RTLIN:OTSMAC'; ! Macros
112 1105 1 |
113 1106 1 REQUIRE 'RTLIN:RTLPSPECT'; ! Define DECLARE_PSECTS macro
114 1201 1 |
115 1202 1 REQUIRE 'RTLIN:RTLDBG'; ! RTL debugging macros
116 1311 1 |
117 1312 1 LIBRARY 'RTLSTARLE';
118 1313 1 |
119 1314 1 |
120 1315 1 | MACROS:
121 1316 1 |
122 1317 1 |
123 1318 1 | EQUATED SYMBOLS:
124 1319 1 |
125 1320 1 |
126 1321 1 | PSECT DECLARATIONS:
127 1322 1 |
128 1323 1 DECLARE_PSECTS (FOR); ! declare PSECTS for FOR$ facility
129 1324 1 |
130 1325 1 | OWN STORAGE:
131 1326 1 |
132 1327 1 |     None
133 1328 1 |
134 1329 1 | EXTERNAL REFERENCES:
135 1330 1 |
136 1331 1 |
137 1332 1 EXTERNAL
```

FOR\$\$UDF_RU
1-013

M 10
16-Sep-1984 00:50:12 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:32:52 [FORRTL.SRC]FORUDFRU.B32;1

Page 4
(2)

```
: 138      1333 1    FOR$$AA_REC_PRO : VECTOR,  
: 139      1334 1  
: 140      1335 1  
: 141      1336 1  
: 142      1337 1    FOR$$AA_REC_PR1 : VECTOR,  
: 143      1338 1  
: 144      1339 1  
: 145      1340 1  
: 146      1341 1    FOR$$AA_REC_PR9 : VECTOR;  
: 147      1342 1  
: 148      1343 1  
: 149      1344 1  
: 150      1345 1  
: 151      1346 1  
: 152      1347 1  EXTERNAL ROUTINE  
: 153      1348 1    FOR$$SIGNAL_STO : NOVALUE;
```

```
: PIC array of record processor  
: procedure-initializations in REC  
: level of abstraction. Indexed by  
: I/O statement type (ISB$B_STTM_TYPE)  
: PIC array of record processor procedures  
: Read a record in REC level of  
: abstraction. Indexed by I/O statement  
: type (ISB$B_STTM_TYPE)  
: PIC array of record processor procedures  
  
: Read last record in REC level of  
: abstraction. Indexed by I/O  
: statement type (ISB$B_STTM_TYPE)  
  
: Signal fatal error
```

FOR
2-0


```
155 1349 1 GLOBAL ROUTINE FOR$$UDF_RU0 ! Read unformatted UDF initialization
156 1350 1 : JSB_UDFO NOVALUE =
157 1351 1
158 1352 1 ++
159 1353 1 FUNCTIONAL DESCRIPTION:
160 1354 1
161 1355 1 Initialize Read unformatted User data formatter (UDF)
162 1356 1
163 1357 1 CALLING SEQUENCE:
164 1358 1
165 1359 1 JSB FOR$$UDF_RU0 ()
166 1360 1
167 1361 1 FORMAL PARAMETERS:
168 1362 1
169 1363 1 None.
170 1364 1
171 1365 1 IMPLICIT INPUTS:
172 1366 1
173 1367 1 CCB Pointer to current logical unit block
174 1368 1 ISB$B_STTM_TYPE I/O statement type code - set by
175 1369 1 each I/O statement initialization
176 1370 1
177 1371 1 IMPLICIT OUTPUTS:
178 1372 1
179 1373 1 LUB$A_BUF_PTR Adr. of next byte of input
180 1374 1 ISB$A_BUF_END Adr. of end+1 byte of input buffer
181 1375 1 data buffer
182 1376 1 ISB$V_NEED_INIT Flag indicating initialization needed
183 1377 1
184 1378 1 ROUTINE VALUE:
185 1379 1 COMPLETION CODES:
186 1380 1
187 1381 1 NONE
188 1382 1
189 1383 1 SIDE EFFECTS:
190 1384 1
191 1385 1 NONE
192 1386 1
193 1387 1 --
194 1388 1
195 1389 2 BEGIN
196 1390 2
197 1391 2 EXTERNAL REGISTER
198 1392 2 CCB : REF BLOCK [, BYTE];
199 1393 2
200 1394 2
201 1395 2 + Initialize Record processing level of abstraction.
202 1396 2 Set pointer to current (LUB$A_BUF_PTR) and last+1
203 1397 2 (LUB$A_BUF_END) character position for user data in
204 1398 2 input buffer
205 1399 2 -
206 1400 2
207 1401 2 +
208 1402 2 Optimization:
209 1403 2 For an unformatted READ (other than keyed), defer REC-level
210 1404 2 initialization until FOR$$UDF_RU1, in case the I/O list is
211 1405 2 single-element and the record can be read directly into the
```



```

: 212      1406  2  | element. Also, set LUB$A_BUF_PTR to LUB$A_BUF_END to indicate to
: 213      1407  2  | certain of the element transmitters that no data has yet been
: 214      1408  2  | read.
: 215      1409  2  |
: 216      1410  2  | REC-level initialization is not deferred for a keyed READ because
: 217      1411  2  | the key is unavailable once FOR$$IO_BEG returns to the user.
: 218      1412  2  |
: 219      1413  2  |
: 220      1414  2  | IF .CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_RKU THEN
: 221      1415  2  |   JSB_RECO (FOR$$AA_REC_PRO + .FOR$$AA_REC_PRO [.CCB [ISB$B_STTM_TYPE] -
: 222      1416  2  |   ISB$K_FORSTTYLO + 1])
: 223      1417  2  | ELSE
: 224      1418  3  |   BEGIN
: 225      1419  3  |     CCB [ISB$V_NEED_INIT] = 1;
: 226      1420  3  |     CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_END];
: 227      1421  2  |   END;
: 228      1422  2  |
: 229      1423  2  | RETURN;
: 230      1424  1  | END;

```

! End of FOR\$\$UDF_RU0 routine

```

.TITLE FOR$$UDF_RU
.IDENT \1-013\

.EXTRN FOR$$AA_REC_PRO
.EXTRN FOR$$AA_REC_PRO1
.EXTRN FOR$$AA_REC_PRO9
.EXTRN FOR$$SIGNAL_STO

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

```

```

      10      FF71  CB  91 00000 FOR$$UDF RU0::
      14      12 00005      CMPB      -143(CCB), #16      : 1414
      50      FF71  CB  9A 00007      BNEQ      1$
      50 00000000G0040 D0 0000C      MOVZBL   -143(CCB), R0      : 1416
      00000000G0040 D0 0000C      MOVL      FOR$$AA_REC_PRO[R0], R0      : 1415
      97 AB      20 88 0001B 1$:      JMP      FOR$$AA_REC_PRO[R0]
      B0 AB      B4 AB D0 0001F      BISB2    #32, -105(CCB)      : 1419
      05 00024      RSB      -76(CCB), -80(CCB)      : 1420
      : 1424

```

; Routine Size: 37 bytes, Routine Base: _FOR\$CODE + 0000

; 231 1425 1


```
290 1483 1 |
291 1484 1 | FUNCTIONAL VALUE:
292 1485 1 |
293 1486 1 |     NONE
294 1487 1 |
295 1488 1 | SIDE EFFECTS:
296 1489 1 |
297 1490 1 |     FOR$_INPSTAREQ if too much data requested.
298 1491 1 |
299 1492 1 |
300 1493 2 | BEGIN
301 1494 2 |
302 1495 2 | EXTERNAL REGISTER
303 1496 2 |     CCB : REF BLOCK [, BYTE];
304 1497 2 |
305 1498 2 |
306 1499 2 | +
307 1500 2 | | If this is a single-element list, check to see if the conditions
308 1501 2 | | for an unbuffered transfer are met. If so, set RAB$L_UBF and
309 1502 2 | | RAB$W_USZ to point directly to the element. Then perform REC-
310 1503 2 | | level initialization, which was deferred in anticipation of a
311 1504 2 | | possible single-element list, and which accomplishes the read.
312 1505 2 | | Afterward, reset RAB$L_UBF, RAB$W_USZ and RAB$V_LOC.
313 1506 2 | |
314 1507 2 | IF .CCB [ISB$V_SNGL_ELEM] AND .CCB [ISB$V_NEED_INIT]
315 1508 2 | THEN
316 1509 2 |     IF NOT .CCB [LUB$V_SEGMENTED] AND .ELEM_SIZE LEQU 65535
317 1510 2 |     THEN
318 1511 3 |         BEGIN
319 1512 3 |             CCB [RAB$L_UBF] = .ELEM_ADR;
320 1513 3 |             CCB [RAB$W_USZ] = .ELEM_SIZE;
321 1514 3 |             CCB [RAB$V_LOC] = 0;
322 1515 3 |             JSB_RECO (FOR$$AA REC PRO +
323 1516 3 |                 .FOR$$AA REC PRO [.CCB [ISB$B_STTM_TYPE] -
324 1517 3 |                     ISB$R_FORSTTYLO + 1]);
325 1518 3 |             IF .CCB [RAB$W_RSZ] LSSU .CCB [RAB$W_USZ]
326 1519 3 |             THEN
327 1520 3 |                 FOR$$SIGNAL_STO (FOR$K_INPSTAREQ);
328 1521 3 |                 CCB [RAB$L_UBF] = .CCB [LUB$A_RBUF_ADR];
329 1522 3 |                 CCB [RAB$W_USZ] = .CCB [LUB$W_RBUF_SIZE];
330 1523 3 |                 CCB [RAB$V_LOC] = 1;
331 1524 3 |                 CCB [ISB$V_NEED_INIT] = 0;
332 1525 3 |                 RETURN;
333 1526 3 |             END
334 1527 2 |         ELSE
335 1528 2 |             CCB [ISB$V_SNGL_ELEM] = 0;
336 1529 2 |
337 1530 2 | +
338 1531 2 | | If REC-level initialization has not yet been done, do it.
339 1532 2 | |
340 1533 2 |
341 1534 2 | IF .CCB [ISB$V_NEED_INIT]
342 1535 2 | THEN
343 1536 3 |     BEGIN
344 1537 3 |         JSB_RECO (FOR$$AA REC PRO +
345 1538 3 |             .FOR$$AA REC PRO [.CCB [ISB$B_STTM_TYPE] -
346 1539 3 |                 ISB$R_FORSTTYLO + 1]);
```



```
347 1540 3      CCB [ISB$V_NEED_INIT] = 0;
348 1541 2      END;
349 1542 2
350 1543 3      BEGIN
351 1544 3      LOCAL
352 1545 3      TMP_ELEM_SIZE,      ! temp no. ob bytes left in user element to copy
353 1546 3      TMP_ELEM_ADR,      ! temp adr. of rest of user element to copy
354 1547 3      TMP_DIFF;      ! temp no. of bytes left to move to user element
355 1548 3
356 1549 3
357 1550 3      !+ Copy as much of input buffer as will fit into user element. If
358 1551 3      ! file is segmented, continue until request is fulfilled. If not,
359 1552 3      ! and variable wants too much data, signal stop FOR$_INPSTAREQ.
360 1553 3      !-
361 1554 3
362 1555 3      TMP_ELEM_SIZE = .ELEM_SIZE;
363 1556 3      TMP_ELEM_ADR = .ELEM_ADR;
364 1557 3
365 1558 3      WHILE .CCB [LUB$A_BUF_PTR] + .TMP_ELEM_SIZE GTRA .CCB [LUB$A_BUF_END] DO
366 1559 4      BEGIN
367 1560 4      TMP_DIFF = .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR];
368 1561 4      TMP_ELEM_ADR = CH$MOVE(.TMP_DIFF, .CCB [LUB$A_BUF_PTR], .TMP_ELEM_ADR);
369 1562 4      TMP_ELEM_SIZE = .TMP_ELEM_SIZE - .TMP_DIFF;
370 1563 4      CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] + .TMP_DIFF;
371 1564 4
372 1565 4      JSB_REC1 (FOR$$AA_REC_PR1 + .FOR$$AA_REC_PR1 [.CCB [ISB$B_STTM_TYPE] -
373 1566 4      ! SB$K_FORSTTYLO + 1]);
374 1567 4      IF NOT .CCB [LUB$V_SEGMENTED]
375 1568 4      THEN
376 1569 4      FOR$$SIGNAL_STO (FOR$K_INPSTAREQ);
377 1570 3      END;
378 1571 3
379 1572 3      !+
380 1573 3      ! Copy from input buffer to the remainder of the user element.
381 1574 3      ! Update buffer pointer (LUB$A_BUF_PTR) to point to last byte+1 moved.
382 1575 3      !-
383 1576 3
384 1577 3      CH$MOVE (.TMP_ELEM_SIZE, .CCB [LUB$A_BUF_PTR], .TMP_ELEM_ADR);
385 1578 3      CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] + .TMP_ELEM_SIZE;
386 1579 3      RETURN;      ! Return from FOR$$UDF_WF1 routine
387 1580 2      END;
388 1581 1 END;      ! End of FOR$$UDF_RU1
```

			07FC 00000	.ENTRY	FOR\$\$UDF_RU1, Save R2,R3,R4,R5,R6,R7,R8,R9,-;	1426
					R10	
		5A	00000000G	MOVAB	FOR\$\$AA_REC_PR1, R10	
		59	00000000G	MOVAB	FOR\$\$SIGNAL_STO, R9	
		58	00000000G	MOVAB	FOR\$\$AA_REC_PRO, R8	
53	97	AB		BBC	#4, -105(CCB), 3\$	1507
63	97	AB		BBC	#5, -105(CCB), 4\$	
45	FD	AB		BBS	#3, -3(CCB), 2\$	1509
	0000FFFF	8F	08	CMPL	ELEM_SIZE, #65535	
			3B	BGTRU	2\$	

24	AB	0C	AC	D0	00030	MOVL	ELEM_ADR, 36(CCB)	1512	
20	AB	08	AC	B0	00035	MOVW	ELEM_SIZE, 32(CCB)	1513	
06	AB		01	8A	0003A	BICB2	#1, 6(CCB)	1514	
50		FF71	CB	9A	0003E	MOVZBL	-143(CCB), R0	1517	
50			6840	D0	00043	MOVL	FOR\$\$AA_REC_PRO[R0], R0	1516	
			6840	16	00047	JSB	FOR\$\$AA_REC_PRO[R0]	1515	
20	AB	22	AB	B1	0004A	CMPL	34(CCB), 32(CCB)	1518	
			07	1E	0004F	BGEQU	1\$		
	7E	43	8F	9A	00051	MOVZBL	#67, -(SP)	1520	
69			01	FB	00055	CALLS	#1, FOR\$\$SIGNAL_STO		
24	AB	EC	AB	D0	00058 1\$:	MOVL	-20(CCB), 36(CCB)	1521	
20	AB	D2	AB	B0	0005D	MOVW	-46(CCB), 32(CCB)	1522	
06	AB		01	88	00062	BISB2	#1, 6(CCB)	1523	
97	AB		20	8A	00066	BICB2	#32, -105(CCB)	1524	
				04	0006A	RET		1511	
97	AB		10	8A	0006B 2\$:	BICB2	#16, -105(CCB)	1528	
97	AB		05	E1	0006F 3\$:	BBC	#5, -105(CCB), 4\$	1534	
50		FF71	CB	9A	00074	MOVZBL	-143(CCB), R0	1539	
50			6840	D0	00079	MOVL	FOR\$\$AA_REC_PRO[R0], R0	1538	
			6840	16	0007D	JSB	FOR\$\$AA_REC_PRO[R0]	1537	
97	AB		20	8A	00080	BICB2	#32, -105(CCB)	1540	
56		08	AC	D0	00084 4\$:	MOVL	ELEM_SIZE, TMP_ELEM_SIZE	1555	
53		0C	AC	D0	00088	MOVL	ELEM_ADR, TMP_ELEM_ADR	1556	
50	56	B0	AB	C1	0008C 5\$:	ADDL3	-80(CCB), TMP_ELEM_SIZE, R0	1558	
B4	AB		50	D1	00091	CMPL	R0, -76(CCB)		
			2C	1B	00095	BLEQU	6\$		
57	B4	AB	B0	AB	C3	00097	SUBL3	-80(CCB), -76(CCB), TMP_DIFF	1560
63	B0	BB	57	28	0009D	MOVCL	TMP_DIFF, @-80(CCB), (TMP_ELEM_ADR)	1561	
	56		57	C2	000A2	SUBL2	TMP_DIFF, TMP_ELEM_SIZE	1562	
B0	AB		57	C0	000A5	ADDL2	TMP_DIFF, -80(CCB)	1563	
50		FF71	CB	9A	000A9	MOVZBL	-143(CCB), R0	1566	
50			6A40	D0	000AE	MOVL	FOR\$\$AA_REC_PR1[R0], R0	1565	
			6A40	16	000B2	JSB	FOR\$\$AA_REC_PR1[R0]		
D2	FD	AB	03	E0	000B5	BBS	#3, -3(CCB), 5\$	1567	
	7E	43	8F	9A	000BA	MOVZBL	#67, -(SP)	1569	
69			01	FB	000BE	CALLS	#1, FOR\$\$SIGNAL_STO		
			C9	11	000C1	BRB	5\$	1558	
63	B0	BB	56	28	000C3 6\$:	MOVCL	TMP_ELEM_SIZE, @-80(CCB), (TMP_ELEM_ADR)	1577	
	B0	AB	56	C0	000C8	ADDL2	TMP_ELEM_SIZE, -80(CCB)	1578	
			04	000CC		RET		1581	

; Routine Size: 205 bytes, Routine Base: _FOR\$CODE + 0025

; 389 1582 1


```
391 1583 1 GLOBAL ROUTINE FOR$$UDF_RU9          ! unformatted input - end of I/O list call
392 1584 1   : JSB_UDF9 NOVALUE =
393 1585 1
394 1586 1   ++
395 1587 1   FUNCTIONAL DESCRIPTION:
396 1588 1
397 1589 1       FOR$$UDF_RU9 performs end of I/O list input formatting.
398 1590 1
399 1591 1   CALLING SEQUENCE:
400 1592 1
401 1593 1       JSB FOR$$UDF_RU9 ( )
402 1594 1
403 1595 1   FORMAL PARAMETERS:
404 1596 1
405 1597 1       NONE
406 1598 1
407 1599 1   IMPLICIT INPUTS:
408 1600 1
409 1601 1       See FOR$$UDF_RU1
410 1602 1
411 1603 1   IMPLICIT OUTPUTS:
412 1604 1
413 1605 1       See FOR$$UDF_RU1
414 1606 1
415 1607 1   FUNCTION VALUE:
416 1608 1
417 1609 1       NONE
418 1610 1
419 1611 1   SIDE EFFECTS:
420 1612 1
421 1613 1       See FOR$$UDF_RU1
422 1614 1
423 1615 1   --
424 1616 1
425 1617 2   BEGIN
426 1618 2
427 1619 2   EXTERNAL REGISTER
428 1620 2       CCB : REF BLOCK [, BYTE];
429 1621 2
430 1622 2   !+
431 1623 2   ! If REC-level initialization has not yet been done (the I/O list
432 1624 2   ! was empty), do it.
433 1625 2   !-
434 1626 2
435 1627 2   IF .CCB [ISB$V_NEED_INIT]
436 1628 2   THEN
437 1629 2       JSB_RECO (FOR$$AA_REC_PRO +
438 1630 2       .FOR$$AA_REC_PRO [.CCB [ISB$B_STTM_TYPE] - ISB$K_FORSTTYLO + 1]);
439 1631 2
440 1632 2   !+
441 1633 2   ! Call record level of abstraction to input buffer from beginning up to but not including LUB$A_BUF_PTR
442 1634 2   !-
443 1635 2
444 1636 2   JSB_REC9 (FOR$$AA_REC_PR9 + .FOR$$AA_REC_PR9 [.CCB [ISB$B_STTM_TYPE] -
445 1637 2       ISB$K_FORSTTYLO + 1]);
446 1638 2
447 1639 2   RETURN;
```

: 448 1640 1 END;

! End of FOR\$\$UDF_RU9 Routine

14	97	AB	05	E1 00000	FOR\$\$UDF_RU9::			
					BBC	#5, -105(CCB), 1\$	: 1627	
		50	FF71	CB	9A 00005	MOVZBL	-143(CCB), R0	: 1630
		50	00000000G0040	D0 0000A	MOVL	FOR\$\$AA_REC_PRO[R0], R0		
			00000000G0040	16 00012	JSB	FOR\$\$AA_REC_PRO[R0]		: 1629
		50	FF71	CB	9A 00019	MOVZBL	-143(CCB), R0	: 1637
		50	00000000G0040	D0 0001E	MOVL	FOR\$\$AA_REC_PR9[R0], R0		: 1636
			00000000G0040	17 00026	JMP	FOR\$\$AA_REC_PR9[R0]		

: Routine Size: 45 bytes, Routine Base: _FOR\$CODE + 00F2

: 449 1641 1
: 450 1642 1 END
: 451 1643 1
: 452 1644 0 ELUDOM

! End of FOR\$\$UDF_RU Module

PSECT SUMMARY

Name	Bytes	Attributes
_FOR\$CODE	287	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	4	0	581	00:01.1

COMMAND QUALIFIERS

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

: Size: 287 code + 0 data bytes
: Run Time: 00:15.1
: Elapsed Time: 00:46.9

FOR\$\$UDF_RU
1-013

I 11
16-Sep-1984 00:50:12

VAX-11 Bliss-32 V4.0-742

Page 13

: Lines/CPU Min: 6532
: Lexemes/CPU-Min: 36667
: Memory Used: 179 pages
: Compilation Complete

FOR
2-0

0184 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY