


```

CCCCCCCC 000000 BBBB8888 LL      IIIIII  NN      NN      AAAAAA  GGGGGGGG  EEEEEEEEEE
CCCCCCCC 000000 88888888 LL      IIIIII  NN      NN      AAAAAA  GGGGGGGG  EEEEEEEEEE
CC        00      00  BB      BB  LL      II      NN      NN      AA      AA  GG      GG      EE
CC        00      00  BB      BB  LL      II      NN      NN      AA      AA  GG      GG      EE
CC        00      00  BB      BB  LL      II      NNNN     NN      NN      AA      AA  GG      GG      EE
CC        00      00  BB      BB  LL      II      NNNN     NN      NN      AA      AA  GG      GG      EE
CC        00      00  88888888 LL      II      NN      NN      NN      NN      AA      AA  GG      GG      EEEEEEEE
CC        00      00  88888888 LL      II      NN      NN      NN      NN      AA      AA  GG      GG      EEEEEEEE
CC        00      00  BB      BB  LL      II      NN      NN      NN      NN      AA      AA  GG      GG      EE
CC        00      00  BB      BB  LL      II      NN      NN      NN      NN      AA      AA  GG      GG      EE
CC        00      00  BB      BB  LL      II      NN      NN      NN      NN      AA      AA  GG      GG      EE
CC        00      00  BB      BB  LL      II      NN      NN      NN      NN      AA      AA  GG      GG      EE
CCCCCCCC 000000 88888888 LLLLLLLLLL IIIIII  NN      NN      AA      AA  GG      GG      EEEEEEEEEE
CCCCCCCC 000000 88888888 LLLLLLLLLL IIIIII  NN      NN      AA      AA  GG      GG      EEEEEEEEEE

```

```

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 COBSLINAGE(
2 0002 0 IDENT = '1-016' ! file COBLINAGE.B32 EDIT:LB1016
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 ++
31 0031 1
32 0032 1 FACILITY: COBOL SUPPORT
33 0033 1
34 0034 1 ABSTRACT:
35 0035 1
36 0036 1 This module contains the RTL routines to handle LINAGE files
37 0037 1 for VAX-11 COBOL.
38 0038 1
39 0039 1 ENVIRONMENT: VAX/VMS user mode
40 0040 1
41 0041 1 AUTHOR: Wm P. Storey, CREATION DATE: 18-July-1979
42 0042 1
43 0043 1 MODIFIED BY:
44 0044 1
45 0045 1 1-001 - Original version. WPS 18-JULY-79
46 0046 1
47 0047 1 1-002 - Modified to insure TOP lines on first logical page.
48 0048 1 WPS 14-AUG-79
49 0049 1
50 0050 1 1-003 - Changed context fields (CTX) from an area pointed to by
51 0051 1 RABSL CTX to an area that precedes in storage the RAB
52 0052 1 and that is specified by negative offsets from the RAB.
53 0053 1 WPS 13-SEPT-79
54 0054 1
55 0055 1 1-004 - Fix problem with end-of-page processing for WRITES that
56 0056 1 overflow page-body.
57 0057 1 WPS 17-SEPT-79

```

```

: 58      0058 1 |
: 59      0059 1 | 1-005 - Change references to REQUIRE files to make compatible with
: 60      0060 1 |         rest of system.   RKR 18-SEPT-79
: 61      0061 1 |
: 62      0062 1 | 1-006 - Fix problem with CALLing generated-code init-linage
: 63      0063 1 |         routine for case of LINKAGE SECTION linage parameters.
: 64      0064 1 |         WPS 18-SEPT-79
: 65      0065 1 |
: 66      0066 1 | 1-007 - Handle special case of "BEFORE ADVANCING identifier", where
: 67      0067 1 |         identifier is equal to zero, for which the generated code
: 68      0068 1 |         will produce a prefix-postfix of <CR><CR>. WPS 24-SEPT-79
: 69      0069 1 |
: 70      0070 1 | 1-008 - Signal filename for invalid LINAGE parameters. WPS 25-SEPT-79
: 71      0071 1 |
: 72      0072 1 | 1-009 - Cosmetic changes.   RKR 26-SEPT-79
: 73      0073 1 |
: 74      0074 1 | 1-010 - Correct problem with ADVANCING PAGE that caused loss of 1 line
: 75      0075 1 |         per page advanced (forgot to add 1). WPS 1-Oct-1979
: 76      0076 1 |
: 77      0077 1 | 1-011 - Correct problem with FOOTING that did not distinguish between
: 78      0078 1 |         "footing not specified" and "footing specified but with value 0".
: 79      0079 1 |         WPS 5-Oct-1979
: 80      0080 1 |
: 81      0081 1 | 1-012 - Addition of comments as per code review. WPS 16-Oct-1979
: 82      0082 1 | 1-013 - Further minor cosmetic changes. RKR 20-OCT-1979
: 83      0083 1 | 1-014 - Change references to OTSS_FATINTERR to COBS_FATINTERR.
: 84      0084 1 |         RKR 29-OCT-79
: 85      0085 1 | 1-015 - Addition of routine COB$TERM_LINAGE. WPS 31-Oct-1979
: 86      0086 1 | 1-016 - Added LIB$STOP as external routine. Also added EDIT field
: 87      0087 1 |         for checkin's use and updated copyright notice. LB 30-NOV-1981
: 88      0088 1 |
: 89      0089 1 | --
: 90      0090 1 | <BLF/PAGE>

```

```

: 92      0091 1  !
: 93      0092 1  SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE ) ;
: 94      0093 1
: 95      0094 1
: 96      0095 1  ! LINKAGES
: 97      0096 1
: 98      0097 1      NONE
: 99      0098 1
100     0099 1  ! TABLE OF CONTENTS
101     0100 1
102     0101 1 FORWARD ROUTINE
103     0102 1      COBSLINAGE,
104     0103 1      COBSINIT_LINAGE : NOVALUE,
105     0104 1      COB$TERM_LINAGE : NOVALUE;
106     0105 1
107     0106 1  ! INCLUDE FILES:
108     0107 1
109     0108 1 LIBRARY 'RTLSTARLE';
110     0109 1 REQUIRE 'RTLIN:COBDEF';
111     0551 1 REQUIRE 'RTLIN:RTLPSECT';
112     0646 1
113     0647 1  ! MACROS
114     0648 1
115     0649 1      MACRO
116     0650 1          PREFIX = 0,0,8,0%;
117     0651 1          POSTFIX = 0,8,8,0%;
118     0652 1          CONTROL = PREFIX%;
119     0653 1          SIGN(A,B,C,D) = A,B+7,1,D%;
120     0654 1
121     0655 1
122     0656 1  ! EQUATED SYMBOLS
123     0657 1
124     0658 1      NONE
125     0659 1
126     0660 1  ! PSECT DECLARATIONS
127     0661 1
128     0662 1 DECLARE_PSECTS (COB) ;          ! Declare PSECTS for COB$ facility
129     0663 1
130     0664 1  ! OWN STORAGE
131     0665 1
132     0666 1      NONE
133     0667 1
134     0668 1  ! EXTERNAL REFERENCES
135     0669 1
136     0670 1 EXTERNAL LITERAL
137     0671 1      COB$_ERRON_FIL,
138     0672 1      COB$_INVLINVAL,
139     0673 1      COB$_FATINTERR;
: 140     0674 1  !

```

```
142 0675 1 GLOBAL ROUTINE COBSLINAGE(RAB,INIT_LNG_ROUTIN) =
143 0676 1
144 0677 1 ++
145 0678 1
146 0679 1 FUNCTIONAL DESCRIPTION:
147 0680 1
148 0681 1 This routine is called to perform lineage functions prior to calling
149 0682 1 SYSSPUT for LINAGE files.
150 0683 1
151 0684 1 FORMAL PARAMETERS:
152 0685 1
153 0686 1 RAB.ra.v is the address of the RAB of the associated
154 0687 1 LINAGE file.
155 0688 1
156 0689 1 INIT_LNG_ROUTIN.czem.vp is the address of the entry mask of the CALLable
157 0690 1 routine generated in-line by VAX-11 COBOL to
158 0691 1 convert the lineage parameters to longwords in
159 0692 1 the context field.
160 0693 1
161 0694 1 IMPLICIT INPUTS:
162 0695 1
163 0696 1 COBSV_CTX_OPENL TRUE if LINAGE file has been opened but not yet
164 0697 1 written to. This bit is needed to make sure
165 0698 1 that TOP lines are positioned on the first
166 0699 1 logical page.
167 0700 1
168 0701 1 COBSL_CTX_LINAG specifies the number of lines in the page body;
169 0702 1 i.e., it specifies the number of lines on the
170 0703 1 logical page that can be written.
171 0704 1
172 0705 1 COBSL_CTX_FOOTI specifies the line number within the page body
173 0706 1 at which the footing area begins. The footing
174 0707 1 area is the "danger zone" area at the end of the
175 0708 1 page body in which the end-of-page condition is
176 0709 1 true, but printing/spacing is still allowed.
177 0710 1
178 0711 1 COBSL_CTX_TOP specifies the number of lines that comprise the
179 0712 1 top margin on the logical page. The top margin
180 0713 1 is a NO-print area that precedes the page body.
181 0714 1
182 0715 1 COBSL_CTX_BOTTO specifies the number of lines that comprise the
183 0716 1 bottom margin on the logical page. The bottom
184 0717 1 margin is a NO-print area that follows the page
185 0718 1 body.
186 0719 1
187 0720 1 COBSL_CTX_COUNT specifies the line number at which the device is
188 0721 1 positioned within the current page body. It is
189 0722 1 the COBOL special register LINAGE-COUNTER.
190 0723 1
191 0724 1 RABSL_RHB is the address of the two byte control area
192 0725 1 containing the print file information. The
193 0726 1 first byte is the "prefix" area, and the second
194 0727 1 byte is the "postfix" area, specifying carriage
195 0728 1 control to be performed before and after the
196 0729 1 record, respectively. One control byte will
197 0730 1 contain either an integer n in the range of 0
198 0731 1 thru 127 (indicating ADVANCING n LINES) or a
```

```

199 0732 1
200 0733 1
201 0734 1
202 0735 1
203 0736 1
204 0737 1
205 0738 1
206 0739 1
207 0740 1
208 0741 1
209 0742 1
210 0743 1
211 0744 1
212 0745 1
213 0746 1
214 0747 1
215 0748 1
216 0749 1
217 0750 1
218 0751 1
219 0752 1
220 0753 1
221 0754 1
222 0755 1
223 0756 1
224 0757 2
225 0758 2
226 0759 2
227 0760 2
228 0761 2
229 0762 2
230 0763 2
231 0764 2
232 0765 2
233 0766 2
234 0767 2
235 0768 2
236 0769 2
237 0770 2
238 0771 2
239 0772 2
240 0773 2
241 0774 2
242 0775 2
243 0776 2
244 0777 2
245 0778 2
246 0779 2
247 0780 2
248 0781 2
249 0782 2
250 0783 2
251 0784 2
252 0785 2
253 0786 2
254 0787 2
255 0788 2

```

hex 8C (indicating ADVANCING PAGE); and the other control byte will contain a hex 8D (indicating no vertical carriage control; note that <CR> is used to indicate the absence of vertical movement; this is necessary to insure proper over-printing in print files). The generated code special cases BEFORE ADVANCING 0 LINES by putting out <CR><CR> in the prefix-postfix control bytes (this is necessary to insure proper overprinting for "AFTER ADVANCING 0 LINES", BEFORE ADVANCING 0 LINES" sequence.

IMPLICIT OUTPUTS:

NONE

ROUTINE VALUE:

If End-Of-Page condition then TRUE else FALSE.

SIDE EFFECTS:

NONE

BEGIN

LITERAL

CR = %X'8D';
FF = %X'8C';
TRUE = 1;
FALSE = 0;

BUILTIN

CALLG,
FP,
TESTBITSC;

MAP

RAB : REF BLOCK [,BYTE],
FP : REF BLOCK [,BYTE];

LOCAL

RHB : REF BLOCK,
RETURN_VALUE,
TEMP;

COB\$LINAGE should not cause access violations. Since COB\$LINAGE is called before the associated SY\$SPUT, the RAB may be invalid. Validate the RAB by checking that RAB\$W_ISI is non-zero. Note that this DOES recognize the case of a CLOSED file that was previously OPEN.

IF .RAB[RAB\$W_ISI] EQL 0
THEN
RETURN FALSE;

```
256 0789 2
257 0790 2
258 0791 2
259 0792 2
260 0793 2
261 0794 2
262 0795 2
263 0796 2
264 0797 2
265 0798 2
266 0799 2
267 0800 2
268 0801 2
269 0802 2
270 0803 2
271 0804 2
272 0805 2
273 0806 2
274 0807 2
275 0808 2
276 0809 2
277 0810 2
278 0811 2
279 0812 2
280 0813 2
281 0814 2
282 0815 2
283 0816 2
284 0817 2
285 0818 2
286 0819 2
287 0820 2
288 0821 2
289 0822 2
290 0823 4
291 0824 4
292 0825 4
293 0826 4
294 0827 4
295 0828 4
296 0829 4
297 0830 4
298 0831 4
299 0832 5
300 0833 4
301 0834 4
302 0835 4
303 0836 4
304 0837 4
305 0838 3
306 0839 3
307 0840 3
308 0841 4
309 0842 4
310 0843 4
311 0844 4
312 0845 4
```

```
RHB = .RAB[RAB$L_RHB];
```

```
! Determine whether ADVANCING lines or ADVANCING PAGE,  
! and adjust RHB to point to meaningful information byte  
! within RAB$L_RHB. Once RHB has been adjusted, further  
! references are to RHB[CONTROL].
```

```
IF .RHB[SIGN(PREFIX)]
```

```
THEN
```

```
IF NOT .RHB[SIGN(POSTFIX)]
```

```
THEN
```

```
RHB = .RHB + 1
```

```
ELSE
```

```
BEGIN
```

```
! Neither is ADVANCING lines, so either one is ADVANCING PAGE  
! (very high probability) or both are <CR> (very low  
! probability). Let us assume that this is ADVANCING PAGE.
```

```
! LINAGE files may not use form-feed for ADVANCING PAGE,  
! so calculate number of new lines needed. LINAGE will  
! always be greater than LINAGE-COUNTER for ADVANCING  
! PAGE. Note that the lines at TOP are for the next logical page.
```

```
TEMP = .RAB[COB$L_CTX_LINAG] - .RAB[COB$L_CTX_COUNT] + 1  
+ .RAB[COB$L_CTX_BOTTO];
```

```
! Determine which control field has <FF>.
```

```
IF .RHB[POSTFIX] EQL FF
```

```
THEN
```

```
BEGIN
```

```
IF TESTBITSC(RAB[COB$V_CTX_OPENL])
```

```
THEN
```

```
! WRITE rec BEFORE ADVANCING PAGE  
! and this is first WRITE to file  
! Advance TOP lines prior as well.
```

```
(IF .RAB[COB$L_CTX_TOP] NEQ 0  
THEN RHB[PREFIX] = .RAB[COB$L_CTX_TOP]);
```

```
RHB = .RHB + 1
```

```
END
```

```
ELSE
```

```
IF .RHB[PREFIX] EQL FF
```

```
THEN
```

```
(IF TESTBITSC(RAB[COB$V_CTX_OPENL])  
THEN
```

```
! WRITE rec AFTER ADVANCING PAGE  
! and this is first WRITE to file.
```

```

313 0846 4
314 0847 4
315 0848 4
316 0849 4
317 0850 4
318 0851 4
319 0852 4
320 0853 4
321 0854 4
322 0855 4
323 0856 4
324 0857 4
325 0858 4
326 0859 4
327 0860 4
328 0861 4
329 0862 4
330 0863 4
331 0864 4
332 0865 4
333 0866 4
334 0867 4
335 0868 4
336 0869 4
337 0870 4
338 0871 4
339 0872 4
340 0873 4
341 0874 4
342 0875 4
343 0876 4
344 0877 4
345 0878 4
346 0879 4
347 0880 4
348 0881 4
349 0882 4
350 0883 4
351 0884 4
352 0885 4
353 0886 4
354 0887 4
355 0888 4
356 0889 4
357 0890 4
358 0891 4
359 0892 4
360 0893 4
361 0894 4
362 0895 4
363 0896 4
364 0897 4
365 0898 4
366 0899 4
367 0900 4
368 0901 4
369 0902 4

```

```

! Include the TOP portion of first logical page.
TEMP = .TEMP + .RAB[COB$LINAGE_TOP]
ELSE
! The sign bit is set for both control bytes, yet neither
! is a <FF>. Consequently, this must be case of <CR><CR>,
! which means WRITE rec BEFORE ADVANCING 0 LINES. Hence,
! the LINAGE-COUNTER is not affected.
RETURN FALSE;

! Call the 'initialize-linage-routine' which will convert the
! LINAGE parameters to longwords in the context area of the RAB.
! This 'initialize-linage-routine' is generated in-line at
! compile-time. It is necessary to call it with the AP of the
! COBOL program in which it resides to handle linage parameters
! in LINKAGE SECTION.
CALLG(.FP[SFS$SAVE_AP],(.INIT_LNG_ROUTIN));
RHB[CONTROL] = .TEMP + .RAB[COB$LINAGE_TOP];
RETURN TRUE
END;

At this point, we are advancing lines and RHB has address of the chosen
prefix/postfix lines that are to be advanced. Update LINAGE-COUNTER.
TEMP = .RAB[COB$LINAGE_COUNT];
RAB[COB$LINAGE_COUNT] = .RAB[COB$LINAGE_COUNT] + .RHB[CONTROL];

Determine if over page body or in FOOTING area.
IF .RAB[COB$LINAGE_COUNT] GTR .RAB[COB$LINAGE_LINAG]
THEN
BEGIN
! The write about to be performed will put us over the page body.
! Re-calculate the number of lines to advance to include BOTTOM
! and TOP. Note that the lines at TOP are for the next logical page,
! therefore, add lines at TOP after CALL to initialize-routine.
RHB[CONTROL] = .RAB[COB$LINAGE_LINAG] - .TEMP + 1
+ .RAB[COB$LINAGE_BOTTO];

! Call the 'initialize-linage-routine' which will convert the
! LINAGE parameters to longwords in the context area of the RAB.
! This 'initialize-linage-routine' is generated in-line at
! compile-time. It is necessary to call it with the AP of the
! COBOL program in which it resides to handle linage parameters

```

```

: 370
: 371
: 372
: 373
: 374
: 375
: 376
: 377
: 378
: 379
: 380
: 381
: 382
: 383
: 384
: 385
: 386
: 387
: 388
: 389
: 390
: 391
: 392
: 393
: 394
: 395
: 396
: 397
: 398
: 399
: 400
: 401
: 402
: 403
: 404
: 405
: 406
: 407
: 408
: 409
: 410
: 411
: 412
: 413
: 414
: 415

```

```

0903      ! in LINKAGE SECTION.
0904      !
0905      CALLG(.FP[SFSL_SAVE_AP],(.INIT_LNG_ROUTIN));
0906
0907
0908      RHB[CONTROL] = .RHB[CONTROL] + .RAB[COB$L_CTX_TOP];
0909
0910      RETURN_VALUE = TRUE
0911      END
0912 ELSE
0913
0914      ! The write to be performed will not put us over the page body.
0915      ! If we are now within the footing area, return TRUE to indicate
0916      ! end-of-page condition.
0917
0918      IF .RAB[COB$L_CTX_COUNT] GEQ .RAB[COB$L_CTX_FOOTI]
0919      THEN
0920          RETURN_VALUE = TRUE
0921      ELSE
0922          RETURN_VALUE = FALSE;
0923
0924
0925
0926      Normally, lines at TOP are positioned during end-of-page processing.
0927      It is necessary to special-case the first WRITE on the first logical
0928      page after a file has been OPENed.
0929
0930      IF TESTBITSC(RAB[COB$V_CTX_OPENL])
0931      THEN
0932          BEGIN
0933              LOCAL
0934                  PREPTR: REF BLOCK;
0935                  PREPTR = .RAB[RAB$L_RHB];
0936                  IF .PREPTR[PREFIX] EQL CR
0937                  THEN
0938                      !
0939                      ! WRITE rec BEFORE ADVANCING n LINES
0940                      ! (IF .RAB[COB$L_CTX_TOP] NEQ 0
0941                      THEN
0942                          PREPTR[PREFIX] = .RAB[COB$L_CTX_TOP]
0943                      ELSE
0944                          PREPTR[PREFIX] = .PREPTR[PREFIX] + .RAB[COB$L_CTX_TOP]
0945                      END;
0946
0947      RETURN .RETURN_VALUE
0948      END;

```

```

.TITLE COB$LINAGE
.IDENT \1-016\

.EXTRN COB$_ERRON_FIL, COB$_INVLINVAL
.EXTRN COB$_FATINTERR

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

.ENTRY COB$LINAGE, Save R2,R3

```

000C 00000

: 0675

		50	04	AC	D0	00002		MOVL	RAB, R0	0785
			02	A0	B5	00006		TSTW	2(R0)	
				03	12	00009		BNEQ	2\$	
				00BC	31	0000B	1\$:	BRW	14\$	
		53	2C	A0	D0	0000E	2\$:	MOVL	44(R0), RHB	0790
				63	95	00012		TSTB	(RHB)	0798
				4E	18	00014		BGEQ	7\$	
				63	B5	00016		TSTW	(RHB)	0800
				04	19	00018		BLSS	3\$	
				53	D6	0001A		INCL	RHB	0802
				46	11	0001C		BRB	7\$	
52	F8	A0	E8	A0	C3	0001E	3\$:	SUBL3	-24(R0), -8(R0), R2	0815
		52	EC	A0	C0	00024		ADDL2	-20(R0), R2	0816
				52	D6	00028		INCL	TEMP	
	8C	8F	01	A3	91	0002A		CMPB	1(RHB), #140	0821
				12	12	0002F		BNEQ	5\$	
09	FC	A0		00	E5	00031		BBCC	#0, -4(R0), 4\$	0825
			F0	A0	D5	00036		TSTL	-16(R0)	0832
				04	13	00039		BEQL	4\$	
		63	F0	A0	90	0003B		MOVB	-16(R0), (RHB)	0833
				53	D6	0003F	4\$:	INCL	RHB	0835
				0F	11	00041		BRB	6\$	
	8C	8F		63	91	00043	5\$:	CMPB	(RHB), #140	0839
				C2	12	00047		BNEQ	1\$	
04	FC	A0		00	E5	00049		BBCC	#0, -4(R0), 6\$	0841
		52	F0	A0	C0	0004E		ADDL2	-16(R0), TEMP	0848
	08	BC	08	BE	FA	00052	6\$:	CALLG	@8(FP), @INIT_LNG_ROUTIN	0867
		50	04	AC	D0	00057		MOVL	RAB, R0	0869
63		52	F0	A0	81	0005B		ADDB3	-16(R0), TEMP, (RHB)	
		50		01	D0	00060		MOVL	#1, R0	0871
					04	00063		RET		
		50	04	AC	D0	00064	7\$:	MOVL	RAB, R0	0878
		52	E8	A0	D0	00068		MOVL	-24(R0), TEMP	
		51		63	9A	0006C		MOVZBL	(RHB), R1	0879
	E8	A0		51	C0	0006F		ADDL2	R1, -24(R0)	
	F8	A0	E8	A0	D1	00073		CMPL	-24(R0), -8(R0)	0885
				1C	15	00078		BLEQ	8\$	
51	F8	A0		52	C3	0007A		SUBL3	TEMP, -8(R0), R1	0894
		51	EC	A0	C0	0007F		ADDL2	-20(R0), R1	0895
63		51		01	81	00083		ADDB3	#1, R1, (RHB)	
	08	BC	08	BE	FA	00087		CALLG	@8(FP), @INIT_LNG_ROUTIN	0905
		50	04	AC	D0	0008C		MOVL	RAB, R0	0908
		63	F0	A0	80	00090		ADDB2	-16(R0), (RHB)	
				07	11	00094		BRB	9\$	0910
	F4	A0	E8	A0	D1	00096	8\$:	CMPL	-24(R0), -12(R0)	0918
				05	19	0009B		BLSS	10\$	
		52		01	D0	0009D	9\$:	MOVL	#1, RETURN_VALUE	0920
				02	11	000A0		BRB	11\$	
				52	D4	000A2	10\$:	CLRL	RETURN VALUE	0922
		50	04	AC	D0	000A4	11\$:	MOVL	RAB, R0	0930
19	FC	A0		00	E5	000A8		BBCC	#0, -4(R0), 13\$	
		51	2C	A0	D0	000AD		MOVL	44(R0), PREPTR	0935
	8D	8F		61	91	000B1		CMPB	(PREPTR), #141	0936
				0B	12	000B5		BNEQ	12\$	
			F0	A0	D5	000B7		TSTL	-16(R0)	0940
				0A	13	000BA		BEQL	13\$	
		61	F0	A0	90	000BC		MOVB	-16(R0), (PREPTR)	0942

COBSLINAGE
1-016

G 14
16-Sep-1984 00:11:37
14-Sep-1984 12:10:49

VAX-11 Bliss-32 V4.0-742
[COBRTL.SRC]COBLINAGE.B32;1

Page 10
(3)

61		04	11	000C0		BRB	13\$	
50	F0	A0	80	000C2	12\$:	ADDB2	-16(R0), (PREPTR)	
		52	D0	000C6	13\$:	MOVL	RETURN_VALUE, R0	
			04	000C9		RET		
		50	D4	000CA	14\$:	CLRL	R0	
			04	000CC		RET		

: 0940
: 0944
: 0947
: 0948
:

; Routine Size: 205 bytes, Routine Base: _COB\$CODE + 0000

```

: 417 0949 1 GLOBAL ROUTINE COBSINIT_LINAGE(RAB) : NOVALUE =
: 418 0950 1
: 419 0951 1 ++
: 420 0952 1
: 421 0953 1 FUNCTIONAL DESCRIPTION:
: 422 0954 1
: 423 0955 1 This routine is called by the generated code in-line "initialize-
: 424 0956 1 lineage-routine" to consistency check the lineage parameters as per
: 425 0957 1 the COBOL standard and to set the value of LINAGE-COUNTER to 1.
: 426 0958 1
: 427 0959 1 FORMAL PARAMETERS:
: 428 0960 1
: 429 0961 1 RAB.ra.v is the address of the RAB of the associated
: 430 0962 1 LINAGE file.
: 431 0963 1
: 432 0964 1 IMPLICIT INPUTS:
: 433 0965 1
: 434 0966 1 COBSL_CTX_LINAG specifies the number of lines in the page body;
: 435 0967 1 i.e., it specifies the number of lines on the
: 436 0968 1 logical page that can be written.
: 437 0969 1
: 438 0970 1 COBSL_CTX_FOOTI specifies the line number within the page body
: 439 0971 1 at which the footing area begins. The footing
: 440 0972 1 area is the "danger zone" area at the end of the
: 441 0973 1 page body in which the end-of-page condition is
: 442 0974 1 true, but printing/spacing is still allowed.
: 443 0975 1
: 444 0976 1 COBSL_CTX_TOP specifies the number of lines that comprise the
: 445 0977 1 top margin on the logical page. The top margin
: 446 0978 1 is a NO-print area that precedes the page body.
: 447 0979 1
: 448 0980 1 COBSL_CTX_BOTTO specifies the number of lines that comprise the
: 449 0981 1 bottom margin on the logical page. The bottom
: 450 0982 1 margin is a NO-print area that follows the page
: 451 0983 1 body.
: 452 0984 1
: 453 0985 1 COBSL_CTX_COUNT specifies the line number at which the device is
: 454 0986 1 positioned within the current page body. It is
: 455 0987 1 the COBOL special register LINAGE-COUNTER.
: 456 0988 1
: 457 0989 1
: 458 0990 1 IMPLICIT OUTPUTS:
: 459 0991 1
: 460 0992 1 If the consistency checks fail, "bad lineage parameters" are SIGNALed.
: 461 0993 1
: 462 0994 1 ROUTINE VALUE:
: 463 0995 1
: 464 0996 1 NONE
: 465 0997 1
: 466 0998 1 SIDE EFFECTS:
: 467 0999 1
: 468 1000 1 NONE
: 469 1001 1
: 470 1002 1 --
: 471 1003 1
: 472 1004 2 BEGIN
: 473 1005 2
```

```

474      1006      2      EXTERNAL ROUTINE
475      1007      2      LIB$STOP:NOVALUE;                                ! Signal fatal error
476      1008      2
477      1009      2      MAP RAB : REF BLOCK [,BYTE];
478      1010      2
479      1011      2      |
480      1012      2      | COB$INIT_LINAGE should not cause access violations. Since it may
481      1013      2      | be called before the associated SYSSPUT, the RAB may be invalid.
482      1014      2      | Validate the RAB by checking that RAB$W_ISI is non-zero.
483      1015      2      |
484      1016      2      | IF .RAB[RAB$W_ISI] EQL 0
485      1017      2      | THEN RETURN;
486      1018      2      |
487      1019      2      |
488      1020      2      | At the time this routine is called, the lineage parameters have
489      1021      2      | been converted to longwords in the context area. Check that
490      1022      2      | all values are valid.
491      1023      2      |
492      1024      2      | IF .RAB[COB$ _CTX_LINAG] GTR 0
493      1025      2      |     AND
494      1026      2      |     .RAB[COB$ _CTX_TOP] GEQ 0
495      1027      2      |     AND
496      1028      2      |     .RAB[COB$ _CTX_BOTTO] GEQ 0
497      1029      2      |     AND
498      1030      2      |     .RAB[COB$ _CTX_FOOTI] GTR 0
499      1031      2      |     AND
500      1032      2      |     .RAB[COB$ _CTX_FOOTI]
501      1033      2      |     LEQ
502      1034      2      |     .RAB[COB$ _CTX_LINAG]
503      1035      2      |
504      1036      2      | THEN
505      1037      2      |
506      1038      2      |     Linage parameters are valid. As this routine is called for each
507      1039      2      |     new page, set the COBOL special register LINAGE-COUNTER to 1.
508      1040      2      |
509      1041      2      |     RAB[COB$ _CTX_COUNT] = 1
510      1042      2      |
511      1043      2      | ELSE
512      1044      2      | BEGIN
513      1045      2      |
514      1046      2      |     Linage parameters are invalid. Signal "invalid lineage value"
515      1047      2      |     and indicate what file this occurred on.
516      1048      2      |
517      1049      2      | LOCAL
518      1050      2      |     FAB      : REF BLOCK [,BYTE],
519      1051      2      |     NAM      : REF BLOCK [,BYTE],
520      1052      2      |     RSADESC : VECTOR[2];
521      1053      2      |
522      1054      2      |     FAB = .RAB[RAB$ _FAB];
523      1055      2      |     IF .FAB EQL 0 THEN SIGNAL_STOP(COB$ _FATINTERR);
524      1056      2      |
525      1057      2      |     NAM = .FAB[FAB$ _NAM];
526      1058      2      |     IF .NAM EQL 0 THEN SIGNAL_STOP(COB$ _FATINTERR);
527      1059      2      |
528      1060      2      |     RSADESC[1] = .NAM[NAM$ _RSA];
529      1061      2      |     RSADESC[0] = .NAM[NAM$ _RSL];
530      1062      2      |

```

```

: 531      1063      3
: 532      1064      3
: 533      1065      3
: 534      1066      4
: 535      1067      4
: 536      1068      4
: 537      1069      4
: 538      1070      3
: 539      1071      3
: 540      1072      3
: 541      1073      1

```

```

IF .RSADESC[0] EQL 0 THEN RSADESC[0] = .NAM[NAM$B_ESL];
IF .RSADESC[0] EQL 0
  THEN
    BEGIN
      RSADESC[0] = .FAB[FAB$B_FNS];
      RSADESC[1] = .FAB[FAB$L_FNA];
    END;
LIB$STOP(COBS_INVLINVAL,0,COBS_ERRON_FIL,1,RSADESC)
END
END;

```

				.EXTRN	LIB\$STOP	
			003C 00000	.ENTRY	COBSINIT LINAGE, Save R2,R3,R4,R5	: 0949
55	00000000G	8F	D0 00002	MOVL	#COBS FATINTERR, R5	
54	00000000G	00	9E 00009	MOVAB	LIB\$STOP, R4	
5E		08	C2 00010	SUBL2	#8, SP	
50	04	AC	D0 00013	MOVL	RAB, R0	: 1016
	02	A0	B5 00017	TSTW	2(R0)	
		65	13 0001A	BEQL	5\$	
	F8	A0	D5 0001C	TSTL	-8(R0)	: 1024
		1B	15 0001F	BLEQ	1\$	
	F0	A0	D5 00021	TSTL	-16(R0)	: 1026
		16	19 00024	BLSS	1\$	
	EC	A0	D5 00026	TSTL	-20(R0)	: 1028
		11	19 00029	BLSS	1\$	
	F4	A0	D5 0002B	TSTL	-12(R0)	: 1030
		0C	15 0002E	BLEQ	1\$	
F8	A0	F4	A0 D1 00030	CMPL	-12(R0), -8(R0)	: 1034
		05	14 00035	BGTR	1\$	
E8	A0		01 D0 00037	MOVL	#1, -24(R0)	: 1041
			04 0003B	RET		
	53	3C	A0 D0 0003C 1\$:	MOVL	60(R0), FAB	: 1054
			05 12 00040	BNEQ	2\$: 1055
			55 DD 00042	PUSHL	R5	
64			01 FB 00044	CALLS	#1, LIB\$STOP	
52	28	A3	D0 00047 2\$:	MOVL	40(FAB), NAM	: 1057
			05 12 0004B	BNEQ	3\$: 1058
			55 DD 0004D	PUSHL	R5	
04	64		01 FB 0004F	CALLS	#1, LIB\$STOP	
	AE	04	A2 D0 00052 3\$:	MOVL	4(NAM), RSADESC+4	: 1060
	6E	03	A2 9A 00057	MOVZBL	3(NAM), RSADESC	: 1061
			0F 12 0005B	BNEQ	4\$: 1063
	6E	0B	A2 9A 0005D	MOVZBL	11(NAM), RSADESC	
			09 12 00061	BNEQ	4\$: 1064
	6E	34	A3 9A 00063	MOVZBL	52(FAB), RSADESC	: 1067
04	AE	2C	A3 D0 00067	MOVL	44(FAB), RSADESC+4	: 1068
			5E DD 0006C 4\$:	PUSHL	SP	: 1071
			01 DD 0006E	PUSHL	#1	
	00000000G	8F	DD 00070	PUSHL	#COBS_ERRON_FIL	
		7E	D4 00076	CLRL	-(SP)	
	00000000G	8F	DD 00078	PUSHL	#COBS_INVLINVAL	
64		05	FB 0007E	CALLS	#5, LIB\$STOP	
			04 00081 5\$:	RET		: 1073

COBSLINAGE
1-016

K 14
16-Sep-1984 00:11:37
14-Sep-1984 12:10:49

VAX-11 Bliss-32 V4.0-742
[COBRTL.SRC]COBLINAGE.B32;1

Page 14
(4)

; Routine Size: 130 bytes, Routine Base: _COB\$CODE + 00CD

```

: 543      1074 1 GLOBAL ROUTINE COB$TERM_LINAGE(RAB) : NOVALUE =
: 544      1075 1
: 545      1076 1 ++
: 546      1077 1
: 547      1078 1 FUNCTIONAL DESCRIPTION:
: 548      1079 1
: 549      1080 1     This routine is called to advance the number of lines needed to
: 550      1081 1     finish out the logical page before the actual CLOSE is done.
: 551      1082 1
: 552      1083 1
: 553      1084 1 FORMAL PARAMETERS:
: 554      1085 1
: 555      1086 1     RAB.ra.v           is the address of the RAB of the associated
: 556      1087 1                      LINAGE file.
: 557      1088 1
: 558      1089 1 IMPLICIT INPUTS:
: 559      1090 1
: 560      1091 1     COB$V_CTX_OPENL    TRUE if LINAGE file has been opened but not yet
: 561      1092 1                      written to.
: 562      1093 1
: 563      1094 1     COB$L_CTX_LINAG    specifies the number of lines in the page body;
: 564      1095 1                      i.e., it specifies the number of lines on the
: 565      1096 1                      logical page that can be written.
: 566      1097 1
: 567      1098 1     COB$L_CTX_FOOTI    specifies the line number within the page body
: 568      1099 1                      at which the footing area begins. The footing
: 569      1100 1                      area is the "danger zone" area at the end of the
: 570      1101 1                      page body in which the end-of-page condition is
: 571      1102 1                      true, but printing/spacing is still allowed.
: 572      1103 1
: 573      1104 1     COB$L_CTX_TOP      specifies the number of lines that comprise the
: 574      1105 1                      top margin on the logical page. The top margin
: 575      1106 1                      is a NO-print area that precedes the page body.
: 576      1107 1
: 577      1108 1     COB$L_CTX_BOTTO    specifies the number of lines that comprise the
: 578      1109 1                      bottom margin on the logical page. The bottom
: 579      1110 1                      margin is a NO-print area that follows the page
: 580      1111 1                      body.
: 581      1112 1
: 582      1113 1     COB$L_CTX_COUNT    specifies the line number at which the device is
: 583      1114 1                      positioned within the current page body. It is
: 584      1115 1                      the COBOL special register LINAGE-COUNTER.
: 585      1116 1
: 586      1117 1
: 587      1118 1 IMPLICIT OUTPUTS:
: 588      1119 1
: 589      1120 1     A PUT to the lineage file is performed; any error status caused by the
: 590      1121 1     'write' is ignored.
: 591      1122 1
: 592      1123 1 ROUTINE VALUE:
: 593      1124 1
: 594      1125 1     NONE
: 595      1126 1
: 596      1127 1 SIDE EFFECTS:
: 597      1128 1
: 598      1129 1     NONE
: 599      1130 1

```

```

: 600      1131  1  !--
: 601      1132
: 602      1133      BEGIN
: 603      1134
: 604      1135      EXTERNAL ROUTINE
: 605      1136          SYSSPUT;
: 606      1137
: 607      1138      MAP
: 608      1139          RAB : REF BLOCK [,BYTE];
: 609      1140
: 610      1141      LOCAL
: 611      1142          RHB : REF BLOCK [,BYTE];
: 612      1143
: 613      1144      |
: 614      1145      | COB$TERM LINAGE should not cause access violations. Since it WILL
: 615      1146      | be called before the associated SYSSCLOSE, the RAB may be invalid.
: 616      1147      | Validate the RAB by checking that RAB$W_ISI is non-zero.
: 617      1148      |
: 618      1149      | IF .RAB[RAB$W_ISI] EQL 0
: 619      1150      | THEN RETURN;
: 620      1151      |
: 621      1152      |
: 622      1153      | If no WRITE has ever been done for this file, then no adjustment
: 623      1154      | need be done at CLOSE time. Note that the flag bit is checked
: 624      1155      | but not cleared; if it is set, we will not be doing a WRITE either.
: 625      1156      |
: 626      1157      | IF .RAB[COB$V_CTX_OPENL]
: 627      1158      | THEN
: 628      1159      | RETURN;
: 629      1160
: 630      1161      RHB = .RAB[RAB$L_RHB];
: 631      1162
: 632      1163      RHB[PREFIX] = .RAB[COB$L_CTX_LINAG] - .RAB[COB$L_CTX_COUNT] + 1
: 633      1164      + .RAB[COB$L_CTX_BOTTO];
: 634      1165
: 635      1166      |
: 636      1167      | Make sure that there is something to advance.
: 637      1168      |
: 638      1169      | IF .RHB[PREFIX] EQL 0
: 639      1170      | THEN
: 640      1171      | RETURN;
: 641      1172
: 642      1173      RHB[POSTFIX] = 0;
: 643      1174
: 644      1175      |
: 645      1176      | The actual WRITE is done by PUTing a record of 0 length with appropriate
: 646      1177      | advance in the PRN control fields.
: 647      1178      |
: 648      1179      RAB[RAB$W_RSZ] = 0;
: 649      1180
: 650      1181      SYSSPUT(.RAB)
: 651      1182      END;

```

.EXTRN SYSSPUT

COBSLINAGE
1-016

N 14
16-Sep-1984 00:11:37
14-Sep-1984 12:10:49

VAX-11 Bliss-32 V4.0-742
[COBRTL.SRC]COBLINAGE.B32:1

Page 17
(5)

Address	Op Code	Op Name	Operand 1	Operand 2	Instruction	Address
50	04	AC	D0	00002	.ENTRY COB\$TERM_LINAGE, Save R2	1074
	02	A0	B5	00006	MOVL RAB, R0	1149
	27	13	00009		TSTW 2(R0)	
	FC	A0	E8	0000B	BEQL 1\$	1157
51	2C	A0	D0	0000F	BLBS -4(R0), 1\$	1161
	F8	A0	C3	00013	MOVL 44(R0), RHB	1163
	51	A0	C0	00019	SUBL3 -24(R0), -8(R0), R1	1164
62	EC	A0	81	0001D	ADDL2 -20(R0), R1	
		01	81	0001D	ADDB3 #1, R1, (RHB)	
		OF	13	00021	BEQL 1\$	1169
	01	A2	94	00023	CLRB 1(RHB)	1173
	22	A0	B4	00026	CLRW 34(R0)	1179
		50	DD	00029	PUSHL R0	1181
		01	FB	0002B	CALLS #1, SYSS\$PUT	
		04	00032	1\$:	RET	1182

: Routine Size: 51 bytes, Routine Base: _COB\$CODE + 014F

: 652 1183 1 END
: 653 1184 0 ELUDOM

.EXTRN LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
_COB\$CODE	386	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	11	0	581	00:00.7

COMMAND QUALIFIERS

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

: Size: 386 code + 0 data bytes
: Run Time: 00:09.3
: Elapsed Time: 00:40.4

COBSLINAGE
1-016

B 15
16-Sep-1984 00:11:37

VAX-11 BLISS-32 V4.0-742

Page 18

: Lines/CPU Min: 7605
: Lexemes/CPU-Min: 31856
: Memory Used: 121 pages
: Compilation Complete

0063 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

This image displays a grid of 144 small, faded screenshots of COBOL programs, arranged in 12 rows and 12 columns. Each screenshot shows a snippet of code or a program header. The titles of the programs, visible in the top-left corner of each screenshot, are as follows:

- Row 1: COBPAUSE LIS
- Row 2: COBMSG LIS
- Row 3: COBHANDLE LIS
- Row 4: COBINTAR LIS
- Row 5: COBINTER LIS
- Row 6: COBMILQ LIS, COBPOSERA LIS
- Row 7: COBIOEXCE LIS
- Row 8: COBIMAGE LIS
- Row 9: COBKEY LIS
- Row 10: COBINUSE LIS
- Row 11: (Titles are mostly illegible)
- Row 12: (Titles are mostly illegible)