


```
BBBBBBBBB      AAAAAA      SSSSSSSS      MM      MM      AAAAAA      RRRRRRRR      GGGGGGGG      IIIIII      NN      NN
BBBBBBBBB      AAAAAA      SSSSSSSS      MM      MM      AAAAAA      RRRRRRRR      GGGGGGGG      IIIIII      NN      NN
BB      BB      AA      AA      SS      SSSSSSSS      MMMM      MMMM      AA      AA      RR      RR      GG      GG      III      NN      NN
BB      BB      AA      AA      SS      SSSSSSSS      MMMM      MMMM      AA      AA      RR      RR      GG      GG      III      NN      NN
BB      BB      AA      AA      SS      SSSSSSSS      MM      MM      AA      AA      RR      RR      GG      GG      III      NN      NN
BBBBBBBBB      AA      AA      SSSSSS      MM      MM      AA      AA      RRRRRRRR      GG      GG      III      NN      NN
BBBBBBBBB      AA      AA      SSSSSS      MM      MM      AA      AA      RRRRRRRR      GG      GG      III      NN      NN
BB      BB      AAAAAAAAAA      SS      MM      MM      AAAAAAAAAA      RR      RR      GG      GGGGGG      III      NN      NN
BB      BB      AAAAAAAAAA      SS      MM      MM      AAAAAAAAAA      RR      RR      GG      GGGGGG      III      NN      NN
BB      BB      AA      AA      SS      MM      MM      AA      AA      RR      RR      GG      GG      III      NN      NN
BB      BB      AA      AA      SS      MM      MM      AA      AA      RR      RR      GG      GG      III      NN      NN
BBBBBBBBB      AA      AA      SSSSSSSS      MM      MM      AA      AA      RR      RR      GGGGGG      IIIIII      NN      NN
BBBBBBBBB      AA      AA      SSSSSSSS      MM      MM      AA      AA      RR      RR      GGGGGG      IIIIII      NN      NN
                                     ....
                                     ....
                                     ....
                                     ....

LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
LL      II      SS      SSSSSSSS
LL      II      SS      SSSSSSSS
LL      II      SS      SSSSSS
LL      II      SS      SSSSSS
LL      II      SS      SSSSSS
LL      II      SS      SSSSSS
LL      II      SS      SSSSSS
LL      II      SS      SSSSSS
LLLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLLL      IIIIII      SSSSSSSS
```



```

1 0001 0 MODULE BASSMARGIN (
2 0002 0 IDENT = '1-0013'
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 FACILITY: BASIC-PLUS-2 Miscellaneous I/O
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains the BASIC MARGIN function, which returns
36 0036 1 the margin of the file open on a channel. The margin setting
37 0037 1 can also be changed.
38 0038 1
39 0039 1 ENVIRONMENT: VAX-11 User Mode
40 0040 1
41 0041 1 AUTHOR: John Sauter, CREATION DATE: 08-MAY-1979
42 0042 1
43 0043 1 MODIFIED BY:
44 0044 1
45 0045 1 1-001 - Original.
46 0046 1 1-002 - Allow MARX on channel 0 before it is open. JBS 23-MAY-1979
47 0047 1 1-003 - Make the margin 16 bits. JBS 30-MAY-1979
48 0048 1 1-004 - If the new margin is larger than the record size (on unit 0
49 0049 1 only) increase the record size. JBS 31-MAY-1979
50 0050 1 1-005 - Update RAB$L_UBF and RAB$W_USZ when increasing the record
51 0051 1 size. JBS 04-JUN-1979
52 0052 1 1-006 - Add BASS$NOMARGIN. JBS 13-JUL-1979
53 0053 1 1-007 - Set up ISB$A_USER_FP. JBS 25-JUL-1979
54 0054 1 1-008 - Correct a coding error in the call to LIB$FREE_VM.
55 0055 1 QAR N11-03250. JBS 29-NOV-1979
56 0056 1 1-009 - When relocating LUB fields, dont forget LUB$A_UBF. JBS 29-NOV-1979
57 0057 1 1-0010- BASS$NOMARGIN needs to do a BASS$CB_POP before return. FM 9-FEB-81

```

! File: BASMARGIN.B32 EDIT:PL10013

BASSMARGIN
1-0013

M 13
16-Sep-1984 00:42:55
14-Sep-1984 11:55:14

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASMARGIN.B32;1

Page 2
(1)

```

: 58      0058 1 : 1-0011- MARGIN should operate only on terminal format files.
: 59      0059 1 : 1-0012- NOMARGIN should also operate only on terminal format
: 60      0060 1 : files. PLL 05-JUN-81
: 61      0061 1 : 1-0013- LIB$STOP should be declared EXTERNAL. PLL 20-Nov-81
: 62      0062 1 : --
: 63      0063 1 :
: 64      0064 1 : <BLF/PAGE>
```

PLL 12-May-81


```

66      0065 1 |
67      0066 1 | SWITCHES:
68      0067 1 |
69      0068 1 |
70      0069 1 | SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
71      0070 1 |
72      0071 1 |
73      0072 1 | LINKAGES:
74      0073 1 |
75      0074 1 |
76      0075 1 | REQUIRE 'RTLIN:OTSLNK';           ! Define linkages
77      0504 1 |
78      0505 1 |
79      0506 1 | TABLE OF CONTENTS:
80      0507 1 |
81      0508 1 |
82      0509 1 | FORWARD ROUTINE
83      0510 1 |     BASMARGIN,           ! Return right margin
84      0511 1 |     BAS$NOMARGIN : NOVALUE;      ! Turn off right margin
85      0512 1 |
86      0513 1 |
87      0514 1 | INCLUDE FILES:
88      0515 1 |
89      0516 1 |
90      0517 1 | REQUIRE 'RTLML:OTSLUB';           ! Get LUB definitions
91      0657 1 |
92      0658 1 | REQUIRE 'RTLML:OTSISB';           ! Get ISB definitions
93      0826 1 |
94      0827 1 | REQUIRE 'RTLIN:RTLPSECT';         ! Macros for defining psects
95      0922 1 |
96      0923 1 | LIBRARY 'RTLSTARLE';             ! System symbols
97      0924 1 |
98      0925 1 |
99      0926 1 | MACROS:
100     0927 1 |
101     0928 1 |     NONE
102     0929 1 |
103     0930 1 | EQUATED SYMBOLS:
104     0931 1 |
105     0932 1 |     NONE
106     0933 1 |
107     0934 1 | PSECTS:
108     0935 1 |
109     0936 1 | DECLARE_PSECTS (BAS);             ! Declare psects for BAS$ facility
110     0937 1 |
111     0938 1 | OWN STORAGE:
112     0939 1 |
113     0940 1 |     NONE
114     0941 1 |
115     0942 1 | EXTERNAL REFERENCES:
116     0943 1 |
117     0944 1 |
118     0945 1 | EXTERNAL ROUTINE
119     0946 1 |     LIB$STOP : NOVALUE,           ! Signal fatal error
120     0947 1 |     LIB$FREE VM,                  ! Deallocate storage
121     0948 1 |     LIB$GET_VM,                   ! Allocate storage
122     0949 1 |     BAS$$CB_PUSH : JSB_CB_PUSH NOVALUE, ! Load register CCB

```


BASMARGIN
1-0013

B 14
16-Sep-1984 00:42:55 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:55:14 [BASRTL.SRC]BASMARGIN.B32;1

Page 4
(2)

```
: 123      0950 1      BAS$$CB_POP : JSB CB_POP NOVALUE,      ! Done with register CCB
: 124      0951 1      BAS$$STOP_IO : NOVALUE,              ! Signal fatal I/O error
: 125      0952 1      BAS$$STOP : NOVALUE,                  ! Signal fatal error
: 126      0953 1      BAS$$OPEN_ZERO : CALL_CCB NOVALUE;    ! Open channel zero.
: 127      0954 1
: 128      0955 1      !+
: 129      0956 1      !- The following are the error codes used in this module.
: 130      0957 1      !-
: 131      0958 1
: 132      0959 1      EXTERNAL LITERAL
: 133      0960 1      BAS$K_ILLIO_CHA : UNSIGNED (8),        ! Illegal I/O channel
: 134      0961 1      BAS$K_IO_CHANOT : UNSIGNED (8),        ! I/O channel not open
: 135      0962 1      BAS$K_ILCOPE : UNSIGNED (8),           ! Illegal operation
: 136      0963 1      BAS$K_MAXMEMEXC : UNSIGNED (8),        ! Maximum memory exceeded
: 137      0964 1      OTSS_FATINTERR;                          ! Fatal internal error
: 138      0965 1
```



```
140 0966 1 GLOBAL ROUTINE BASSMARGIN (      ! Return right margin
141 0967 1     CHAN,                        ! Channel whose margin to return
142 0968 1     NEW_MARGIN                    ! Optional new setting
143 0969 1     ) =
144 0970 1
145 0971 1 ++
146 0972 1 FUNCTIONAL DESCRIPTION:
147 0973 1
148 0974 1     Returns the right margin of the specified channel. Optionally,
149 0975 1     a new right margin can be specified. If the right margin of the
150 0976 1     user's terminal (channel 0) is being extended, this routine will
151 0977 1     reallocate the channel buffer if necessary so that there is
152 0978 1     enough room for a line out to the margin.
153 0979 1
154 0980 1 FORMAL PARAMETERS:
155 0981 1
156 0982 1     CHAN.rl.v      The channel whose margin to return.
157 0983 1     NEW_MARGIN.rl.v  Optionally, the new right margin. 0 means
158 0984 1                     return to default.
159 0985 1
160 0986 1 IMPLICIT INPUTS:
161 0987 1
162 0988 1     LUB$W_R_MARGIN  The channel's current right margin
163 0989 1     LUB$W_D_MARGIN  The channel's default right margin
164 0990 1     LUB$W_RBUF_SIZE The length of the record buffer.
165 0991 1
166 0992 1 IMPLICIT OUTPUTS:
167 0993 1
168 0994 1     LUB$W_R_MARGIN  The channel's current right margin
169 0995 1     LUB$W_RBUF_SIZE The length of the record buffer
170 0996 1     LUB$A_UBF       The address of the record buffer
171 0997 1     Various other fields in the LUB and RAB that point into the
172 0998 1     record buffer, which get relocated.
173 0999 1
174 1000 1 ROUTINE VALUE:
175 1001 1
176 1002 1     The right margin of the channel, before any changes made
177 1003 1     by this routine.
178 1004 1
179 1005 1 SIDE EFFECTS:
180 1006 1
181 1007 1     May change the right margin of this channel.
182 1008 1     If channel 0 is not open, opens it.
183 1009 1     BASS$CB_PUSH will signal if the channel number is invalid.
184 1010 1     If it must expand the channel buffer, will disable ASTs while
185 1011 1     doing so, and relocate some LUB and RAB fields.
186 1012 1
187 1013 1 --
188 1014 1
189 1015 2 BEGIN
190 1016 2
191 1017 2 BUILTIN
192 1018 2     FP;
193 1019 2
194 1020 2 GLOBAL REGISTER
195 1021 2     CCB = K_CCB_REG : REF BLOCK [, BYTE];
196 1022 2
```



```
197 1023 2 BUILTIN
198 1024 2 ACTUALCOUNT;
199 1025 2
200 1026 2 LOCAL
201 1027 2 FMP : REF BLOCK [, BYTE],
202 1028 2 RIGHT_MARGIN,
203 1029 2 LOG_UNIT;
204 1030 2
205 1031 2 FMP = .FP;
206 1032 2
207 1033 2 !+ Get the LUB for the channel. Signal if the channel number is invalid.
208 1034 2 !-
209 1035 2 LOG_UNIT =
210 1036 2 BEGIN
211 1037 2
212 1038 2 IF (.CHAN GTR 0)
213 1039 2 THEN
214 1040 2 .CHAN
215 1041 2 ELSE
216 1042 2
217 1043 2 IF (.CHAN EQL 0)
218 1044 2 THEN
219 1045 2 LUB$K_LUN_BPRI
220 1046 2 ELSE
221 1047 2 BEGIN
222 1048 2 BASS$STOP (BASS$ILLIO_CHA);
223 1049 2 0
224 1050 2 END
225 1051 2
226 1052 2 END;
227 1053 2 BASS$CB_PUSH (.LOG_UNIT, LUB$K_LUN_BPRI);
228 1054 2 CCB [ISB$A_USER_FP] = .FMP [SF$L_SAVE_FP];
229 1055 2
230 1056 2 !+ If the channel is zero and not open, open it.
231 1057 2 !-
232 1058 2
233 1059 2 IF (.CHAN EQL 0)
234 1060 2 THEN
235 1061 2 BEGIN
236 1062 2
237 1063 2 IF ( NOT .CCB [LUB$V_OPENED]) THEN BASS$OPEN_ZERO (.FMP [SF$L_SAVE_FP]);
238 1064 2
239 1065 2 END;
240 1066 2
241 1067 2 !+ If the channel is not now open, we have an error.
242 1068 2 !-
243 1069 2
244 1070 2
245 1071 2 IF ( NOT .CCB [LUB$V_OPENED]) THEN BASS$STOP_IO (BASS$IO_CHANOT);
246 1072 2
247 1073 2 !+ If this is anything but a terminal format file, the MARGIN state-
248 1074 2 !- ment is unreasonable.
249 1075 2
250 1076 2
251 1077 2
252 1078 2 IF (.CCB [LUB$B_ORGAN] NEQ LUB$K_ORG_TERM1) THEN BASS$STOP_IO (BASS$ILLOPE);
253 1079 2
```



```
254 1080 2 1+
255 1081 2 1- Fetch the current margin, so we can return it.
256 1082 2 1-
257 1083 2 RIGHT_MARGIN = .CCB [LUB$W_R_MARGIN];
258 1084 2 1+
259 1085 2 1- If a new margin is specified, change to it. If the new margin is
260 1086 2 1- zero, change instead to the default margin.
261 1087 2 1-
262 1088 2 1-
263 1089 3 IF (ACTUALCOUNT () GEQ 2)
264 1090 2 THEN
265 1091 3 BEGIN
266 1092 3 CCB [LUB$W_R_MARGIN] =
267 1093 4 BEGIN
268 1094 4
269 1095 4 IF (.NEW_MARGIN EQL 0) THEN .CCB [LUB$W_D_MARGIN] ELSE .NEW_MARGIN
270 1096 4
271 1097 3 END;
272 1098 3 1+
273 1099 3 1- If the margin is larger than the buffer, and this is channel 0
274 1100 3 1- (which means the user has no control over the size of the buffer
275 1101 3 1- since he cannot open channel 0 himself), increase the buffer
276 1102 3 1- size to match the new margin.
277 1103 3 1-
278 1104 3 1-
279 1105 5 IF ((.CCB [LUB$W_R_MARGIN] GTR .CCB [LUB$W_RBUF_SIZE]) AND .CCB [LUB$V_UNIT_0] AND ( NOT .CCB [
280 1106 4 LUB$V_USER_RBUF]))
281 1107 3 THEN
282 1108 4 BEGIN
283 1109 4
284 1110 4 LOCAL
285 1111 4 NEW_BUFFER,
286 1112 4 GET_VM_STATUS,
287 1113 4 FREE_VM_STATUS,
288 1114 4 AST_STATUS,
289 1115 4 OLD_BUFFER,
290 1116 4 OLD_SIZE,
291 1117 4 NEW_SIZE;
292 1118 4
293 1119 4 1+
294 1120 4 1- We must be sure that an AST does not fool with the buffer as we
295 1121 4 1- are reallocating it.
296 1122 4 1-
297 1123 4 AST_STATUS = $SETAST (ENBFLG = 0);
298 1124 4 1+
299 1125 4 1- Allocate a new buffer.
300 1126 4 1-
301 1127 4 NEW_SIZE = MAX (.CCB [LUB$W_RBUF_SIZE], .CCB [LUB$W_D_MARGIN], .CCB [LUB$W_R_MARGIN]);
302 1128 4 GET_VM_STATUS = LIB$GET_VM (NEW_SIZE, NEW_BUFFER);
303 1129 4
304 1130 4 IF ( NOT .GET_VM_STATUS) THEN BASS$STOP_IO (BASS$K_MAXMEMEXC);
305 1131 4
306 1132 4 1+
307 1133 4 1- Remember the old buffer address and length for later.
308 1134 4 1-
309 1135 4 OLD_BUFFER = .CCB [LUB$A_UBF];
310 1136 4 OLD_SIZE = .CCB [LUB$W_RBUF_SIZE];
```



```
311 1137 4 !+
312 1138 4 ! Point the LUB to the new buffer.
313 1139 4 !-
314 1140 4 CCB [LUB$A_UBF] = .NEW_BUFFER;
315 1141 4 CCB [LUB$W_RBUF_SIZE] = .NEW_SIZE;
316 1142 4 CCB [RAB$W_USZ] = .NEW_SIZE;
317 1143 4 !+
318 1144 4 ! Now relocate all of the LUB fields that point into the buffer.
319 1145 4 !-
320 1146 4 CCB [LUB$A_RBUF_ADR] = .CCB [LUB$A_RBUF_ADR] - .OLD_BUFFER + .NEW_BUFFER;
321 1147 4 CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] - .OLD_BUFFER + .NEW_BUFFER;
322 1148 4 CCB [LUB$A_BUF_END] = .CCB [LUB$A_BUF_END] - .OLD_BUFFER + .NEW_BUFFER;
323 1149 4 CCB [LUB$A_BUF_BEG] = .CCB [LUB$A_BUF_BEG] - .OLD_BUFFER + .NEW_BUFFER;
324 1150 4 CCB [LUB$A_BUF_HIGH] = .CCB [LUB$A_BUF_HIGH] - .OLD_BUFFER + .NEW_BUFFER;
325 1151 4 CCB [RAB$L_UBF] = .CCB [RAB$L_UBF] - .OLD_BUFFER + .NEW_BUFFER;
326 1152 4 !+
327 1153 4 ! Copy the data from the old buffer to the new.
328 1154 4 !-
329 1155 4 CH$COPY (.OLD_SIZE, .OLD_BUFFER, 0, .NEW_SIZE, .NEW_BUFFER);
330 1156 4 !+
331 1157 4 ! Now that the CCB is updated, we can turn ASTs back on.
332 1158 4 !-
333 1159 4
334 1160 4 IF (.AST_STATUS EQL SSS_WASSET) THEN $SETAST (ENBFLG = 1);
335 1161 4
336 1162 4 !+
337 1163 4 ! Free the old buffer.
338 1164 4 !-
339 1165 4 FREE_VM_STATUS = LIB$FREE_VM (OLD_SIZE, OLD_BUFFER);
340 1166 4
341 1167 4 IF ( NOT .FREE_VM_STATUS) THEN LIB$STOP (OTSS_FATINTERR);
342 1168 4
343 1169 4 END;
344 1170 3
345 1171 3 !+
346 1172 3 ! Since we now have a right margin, make sure the NOMARGIN bit
347 1173 3 ! is clear.
348 1174 3 !-
349 1175 3 CCB [LUB$V_NOMARGIN] = 0;
350 1176 3 END;
351 1177 2
352 1178 2 !+
353 1179 2 ! We are done with register CCB.
354 1180 2 !-
355 1181 2 BASS$CB_POP ();
356 1182 2 !+
357 1183 2 ! Return the previous (or unchanged) right margin.
358 1184 2 !-
359 1185 2 RETURN (.RIGHT_MARGIN);
360 1186 1 END;
```

! end of BASSMARGIN

.TITLE BASSMARGIN
.IDENT \1-0013\.EXTRN LIB\$STOP, LIB\$FREE_VM
.EXTRN LIB\$GET_VM, BASS\$CB_PUSH

			69	7E	D4	000A1	CLRL	-(SP)	1123
			56	01	FB	000A3	CALLS	#1, SYSS\$SETAST	
			50	50	D0	000A6	MOVL	R0, AST_STATUS	
			50	D2	AB	3C 000A9	MOVZWL	-46(CCB), R0	1127
			50	D6	AB	B1 000AD	CMPW	-42(CCB), R0	
				04	1B	000B1	BLEQU	12\$	
			50	D6	AB	3C 000B3	MOVZWL	-42(CCB), R0	
			50	D4	AB	B1 000B7	CMPW	-44(CCB), R0	
				04	1B	000BB	BLEQU	13\$	
			50	D4	AB	3C 000BD	MOVZWL	-44(CCB), R0	
	04		AE	50	D0	000C1	MOVL	R0, NEW_SIZE	
				5E	DD	000C5	PUSHL	SP	1128
				08	AE	9F 000C7	PUSHAB	NEW_SIZE	
	00000000G	00		02	FB	000CA	CALLS	#2, LIB\$GET VM	
		07		50	E8	000D1	BLBS	GET VM STATUS, 14\$	1130
		7E		00G	8F	9A 000D4	MOVZBL	#BAS\$K-MAXMEMEXC, -(SP)	
		68		01	FB	000D8	CALLS	#1, BAS\$\$STOP IO	
	08	AE		9C	AB	D0 000DB	MOVL	-100(CCB), OLD_BUFFER	1135
	0C	AE		D2	AB	3C 000E0	MOVZWL	-46(CCB), OLD_SIZE	1136
		51		6E	D0	000E5	MOVL	NEW_BUFFER, RT	1140
	9C	AB		51	D0	000E8	MOVL	R1, -100(CCB)	
	D2	AB		04	AE	B0 000EC	MOVW	NEW_SIZE, -46(CCB)	1141
	20	AB		04	AE	B0 000F1	MOVW	NEW_SIZE, 32(CCB)	1142
		50		08	AE	D0 000F6	MOVL	OLD_BUFFER, R0	1146
	EC	AB		50	C3	000FA	SUBL3	R0, -20(CCB), R2	
EC	52	AB		51	C1	000FF	ADDL3	R1, R2, -20(CCB)	
B0	52	AB		50	C3	00104	SUBL3	R0, -80(CCB), R2	1147
	52	AB		51	C1	00109	ADDL3	R1, R2, -80(CCB)	
B4	52	AB		50	C3	0010E	SUBL3	R0, -76(CCB), R2	1148
	52	AB		51	C1	00113	ADDL3	R1, R2, -76(CCB)	
BC	52	AB		50	C3	00118	SUBL3	R0, -68(CCB), R2	1149
	52	AB		51	C1	0011D	ADDL3	R1, R2, -68(CCB)	
C0	52	AB		50	C3	00122	SUBL3	R0, -64(CCB), R2	1150
	52	AB		51	C1	00127	ADDL3	R1, R2, -64(CCB)	
24	52	AB		50	C3	0012C	SUBL3	R0, 36(CCB), R2	1151
	52	AB		51	C1	00131	ADDL3	R1, R2, 36(CCB)	
04	AE	00		0C	AE	2C 00136	MOVCS	OLD_SIZE, (R0), #0, NEW_SIZE, (R1)	1155
		60		61		0013D			
		09		56	D1	0013E	CMPL	AST_STATUS, #9	1160
				05	12	00141	BNEQ	15\$	
				01	DD	00143	PUSHL	#1	
		69		01	FB	00145	CALLS	#1, SYSS\$SETAST	
				08	AE	9F 00148	PUSHAB	OLD_BUFFER	1165
				10	AE	9F 0014B	PUSHAB	OLD_SIZE	
	00000000G	00		02	FB	0014E	CALLS	#2, LIB\$FREE VM	
		0D		50	E8	00155	BLBS	FREE VM STATUS, 16\$	1167
				8F	DD	00158	PUSHL	#OTS\$ FATINTERR	
	00000000G	00		01	FB	0015E	CALLS	#1, LIB\$STOP	
	A1	AB		02	8A	00165	BICB2	#2, -95(CCB)	1175
				00	16	00169	JSB	BAS\$\$CB POP	1181
		50		57	D0	0016F	MOVL	RIGHT_MARGIN, R0	1185
				04		00172	RET		1186

; Routine Size: 371 bytes, Routine Base: _BAS\$CODE + 0000

; 361 1187 1

```
363 1188 1 GLOBAL ROUTINE BAS$NOMARGIN (          ! Disable right margin
364 1189 1     CHAN                                ! Channel whose margin to disable
365 1190 1     ) : NOVALUE =
366 1191 1
367 1192 1 ++
368 1193 1 FUNCTIONAL DESCRIPTION:
369 1194 1
370 1195 1     Disable the right margin feature of a channel.
371 1196 1
372 1197 1 FORMAL PARAMETERS:
373 1198 1
374 1199 1     CHAN.rl.v      The channel whose margin to disable.
375 1200 1
376 1201 1 IMPLICIT INPUTS:
377 1202 1
378 1203 1     NONE
379 1204 1
380 1205 1 IMPLICIT OUTPUTS:
381 1206 1
382 1207 1     LUB$V_NOMARGIN Set to 1. This causes the formatting
383 1208 1                  routines to not enforce any right
384 1209 1                  margin.
385 1210 1
386 1211 1 ROUTINE VALUE:
387 1212 1 COMPLETION CODES:
388 1213 1
389 1214 1     NONE
390 1215 1
391 1216 1 SIDE EFFECTS:
392 1217 1
393 1218 1     If channel 0 is not open, opens it.
394 1219 1     BAS$$CB_PUSH will signal if the channel number is invalid.
395 1220 1
396 1221 1 --
397 1222 1
398 1223 2 BEGIN
399 1224 2
400 1225 2 BUILTIN
401 1226 2     FP;
402 1227 2
403 1228 2 GLOBAL REGISTER
404 1229 2     CCB = K_CCB_REG : REF BLOCK [, BYTE];
405 1230 2
406 1231 2 LOCAL
407 1232 2     FMP : REF BLOCK [, BYTE],
408 1233 2     LOG_UNIT;
409 1234 2
410 1235 2     FMP = .FP;
411 1236 2 ++
412 1237 2 Get the LUB for the channel. Signal if the channel number is invalid.
413 1238 2 --
414 1239 2     LOG_UNIT =
415 1240 3 BEGIN
416 1241 3
417 1242 4     IF (.CHAN GTR 0)
418 1243 3     THEN
419 1244 3         .CHAN
```



```

: 420      1245 3      ELSE
: 421      1246 3
: 422      1247 4      IF (.CHAN EQL 0)
: 423      1248 3      THEN
: 424      1249 3          LUB$K_LUN_BPRI
: 425      1250 3      ELSE
: 426      1251 4          BEGIN
: 427      1252 4          BASS$STOP (BASS$ILLIO_CHA);
: 428      1253 4          0
: 429      1254 4          END
: 430      1255 4
: 431      1256 2      END;
: 432      1257 2      BASS$CB_PUSH (.LOG_UNIT, LUB$K_LUN_BPRI);
: 433      1258 2      CCB [ISB$A_USER_FP] = .FMP [SF$L_SAVE_FP];
: 434      1259 2      +
: 435      1260 2      :- If the channel is zero and not open, open it.
: 436      1261 2
: 437      1262 2
: 438      1263 3      IF (.CHAN EQL 0)
: 439      1264 3      THEN
: 440      1265 3          BEGIN
: 441      1266 3
: 442      1267 3          IF ( NOT .CCB [LUB$V_OPENED]) THEN BASS$OPEN_ZERO (.FMP [SF$L_SAVE_FP]);
: 443      1268 3
: 444      1269 3          END;
: 445      1270 2
: 446      1271 2      +
: 447      1272 2      :- If the channel is not now open, we have an error.
: 448      1273 2
: 449      1274 2
: 450      1275 2      IF ( NOT .CCB [LUB$V_OPENED]) THEN BASS$STOP_IO (BASS$IO_CHANOT);
: 451      1276 2
: 452      1277 2      +
: 453      1278 2      :- If this is anything but a terminal format file, the NOMARGIN statement is unreasonable.
: 454      1279 2
: 455      1280 2
: 456      1281 2      IF (.CCB [LUB$B_ORGAN] NEQ LUB$K_ORG_TERMI) THEN BASS$STOP_IO (BASS$ILLOPE);
: 457      1282 2
: 458      1283 2      +
: 459      1284 2      :- Set the NOMARGIN bit, so the margin will not be enforced.
: 460      1285 2
: 461      1286 2      CCB [LUB$V_NOMARGIN] = 1;
: 462      1287 2      +
: 463      1288 2      :- Set the margin cell to 0, so the margin function will
: 464      1289 2      :- return a zero.
: 465      1290 2
: 466      1291 2      CCB [LUB$W_R_MARGIN] = 0;
: 467      1292 2      +
: 468      1293 2      :- Pop the channel information.
: 469      1294 2
: 470      1295 2      BASS$CB_POP ();
: 471      1296 2      RETURN;
: 472      1297 1      END;

```

! end of BASS\$NOMARGIN

BASSMARGIN
1-0013

K 14
16-Sep-1984 00:42:55
14-Sep-1984 11:55:14

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASMARGIN.B32;1

Page 13
(4)

			083C 00000	.ENTRY	BASSNOMARGIN, Save R2,R3,R4,R5,R11	: 1188
55	00000000G	00	9E 00002	MOVAB	BASS\$STOP_IO, R5	: 1235
53		5D	D0 00009	MOVL	FP, FMP	: 1242
54	04	AC	D0 0000C	MOVL	CHAN, R4	: 1244
		05	15 00010	BLEQ	1\$	: 1247
52		54	D0 00012	MOVL	R4, LOG_UNIT	: 1252
		14	11 00015	BRB	3\$	: 1251
		05	12 00017	BNEQ	2\$	: 1257
52		08	CE 00019	MNEGL	#8, LOG_UNIT	: 1258
		0D	11 0001C	BRB	3\$	: 1263
00000000G	7E	00G	8F 9A 0001E	MOVZBL	#BASSK_ILLIO CHA, -(SP)	: 1267
	00		01 FB 00022	CALLS	#1, BASS\$STOP	: 1275
		52	D4 00029	CLRL	LOG_UNIT	: 1281
50		08	CE 0002B	MNEGL	#8, R0	: 1286
	00000000G	00	16 0002E	JSB	BASS\$CB_PUSH	: 1291
FF4C	CB	0C	A3 D0 00034	MOVL	12(FMP), -180(CCB)	: 1295
		54	D5 0003A	TSTL	R4	: 1297
		0E	12 0003C	BNEQ	4\$	
15	FC	AB	E8 0003E	BLBS	-4(CCB), 5\$	
	0C	A3	DD 00042	PUSHL	12(FMP)	
00000000G	00	01	FB 00045	CALLS	#1, BASS\$OPEN_ZERO	
	07	FC	AB E8 0004C	BLBS	-4(CCB), 5\$	
	7E	00G	8F 9A 00050	MOVZBL	#BASSK_IO CHANOT, -(SP)	
	65	01	FB 00054	CALLS	#1, BASS\$STOP_IO	
	04	C4	AB 91 00057	CMPB	-60(CCB), #4	
		07	13 0005B	BEQL	6\$	
	7E	00G	8F 9A 0005D	MOVZBL	#BASSK_ILLOPE, -(SP)	
	05	01	FB 00061	CALLS	#1, BASS\$STOP_IO	
A1	AB	02	88 00064	BISB2	#2, -95(CCB)	
		D4	AB B4 00068	CLRW	-44(CCB)	
	00000000G	00	16 0006B	JSB	BASS\$CB_POP	
		04	00071	RET		

; Routine Size: 114 bytes, Routine Base: _BASS\$CODE + 0173

: 473 1298 1
: 474 1299 1 END
: 475 1300 1
: 476 1301 0 ELUDOM

! end of module BASSMARGIN

PSECT SUMMARY

Name	Bytes	Attributes
_BASS\$CODE	485	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

BASSMARGIN
1-0013

L 14
16-Sep-1984 00:42:55
14-Sep-1984 11:55:14

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASMARGIN.B32;1

Page 14
(4)

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
;\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	7	0	581	00:01.2

COMMAND QUALIFIERS

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

: Size: 485 code + 0 data bytes
: Run Time: 00:15.7
: Elapsed Time: 00:35.7
: Lines/CPU Min: 4971
: Lexemes/CPU-Min: 32044
: Memory Used: 192 pages
: Compilation Complete

0024 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY