

BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLLLLLL

BBBBBBBB	AAAAAA	SSSSSSSS	TTTTTTTTTT	AAAAAA	BBBBBBBB	
BBBBBBBB	AAAAAA	SSSSSSSS	TTTTTTTTTT	AAAAAA	BBBBBBBB	
BB	AA	SS	TT	AA	BB	
BB	AA	SS	TT	AA	BB	
BB	AA	SS	TT	AA	BB	
BB	AA	SS	TT	AA	BB	
BBBBBBBB	AA	SSSSSS	TT	AA	BBBBBBBB	
BBBBBBBB	AA	SSSSSS	TT	AA	BBBBBBBB	
BB	AAAAAAAAA	SS	TT	AAAAAAAAA	BB	
BB	AAAAAAAAA	SS	TT	AAAAAAAAA	BB	
BB	AA	SS	TT	AA	BB	
BB	AA	SS	TT	AA	BB	
BBBBBBBB	AA	SSSSSSSS	TT	AA	BBBBBBBB	
BBBBBBBB	AA	SSSSSSSS	TT	AA	BBBBBBBB	

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
LL	II	SS
LLLLLLLLLL	IIIIII	SSSSSSSS
LLLLLLLLLL	IIIIII	SSSSSSSS


```
1 0001 0 MODULE BAS$TAB (
2 0002 0 IDENT = '1-007'
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 TAB function,
36 0036 1 which produces enough spaces to reach a specified position.
37 0037 1
38 0038 1 ENVIRONMENT: VAX-11 User Mode
39 0039 1
40 0040 1 AUTHOR: John Sauter, CREATION DATE: 01-MAY-1979
41 0041 1
42 0042 1 MODIFIED BY:
43 0043 1
44 0044 1 1-001 - Original.
45 0045 1 1-002 - Call BAS$$CB GET, so this module does not have to be in the
46 0046 1 sharable library. JBS 22-AUG-1979
47 0047 1 1-003 - Convert BAS$STRING to STR$DUPL_CHAR. JBS 08-NOV-1979
48 0048 1 1-004 - Add BAS$ANSI_TAB entry point. PLL 22-Jun-81
49 0049 1 1-005 - BAS$ANSI_TAB should check for zero argument, and supply 1
50 0050 1 if necessary. PLL 23-Jul-1982
51 0051 1 1-006 - fix always tabbing 1 in BAS$ANSI_TAB. Also, in ANSI tab,
52 0052 1 the value passed is the column where we want to start the
53 0053 1 next print field, which implies that the amount we want to
54 0054 1 tab is one less than that. MDL 18-Nov-1982
55 0055 1 1-007 - correct a couple of minor ANSI tab semantic bugs. MDL 27-Jun-1983
56 0056 1 --
57 0057 1
```

BASSTAB
1-007

F 5
16-Sep-1984 01:18:05
14-Sep-1984 11:56:41

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

Page 2
(1)

; 58

0058 1 !<BLF/PAGE>


```
60 0059 1 |
61 0060 1 | SWITCHES:
62 0061 1 |
63 0062 1 |
64 0063 1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
65 0064 1 |
66 0065 1 |
67 0066 1 | LINKAGES:
68 0067 1 |
69 0068 1 |
70 0069 1 REQUIRE 'RTLIN:OTSLNK'; ! Define linkages
71 0498 1 |
72 0499 1 |
73 0500 1 | TABLE OF CONTENTS:
74 0501 1 |
75 0502 1 |
76 0503 1 FORWARD ROUTINE
77 0504 1 BAS$TAB, ! Produce spaces to reach a position
78 0505 1 BAS$ANSI_TAB; ! Same as BAS$TAB but conforms to Min ANSI
79 0506 1 |
80 0507 1 |
81 0508 1 | INCLUDE FILES:
82 0509 1 |
83 0510 1 |
84 0511 1 REQUIRE 'RTLML:BASPAR'; ! Intermodule BASIC parameters and constants
85 0533 1 |
86 0534 1 REQUIRE 'RTLML:OTSLUB'; ! Get logical unit block definitions
87 0674 1 |
88 0675 1 REQUIRE 'RTLIN:RTLPSECT'; ! Macros for defining psects
89 0770 1 |
90 0771 1 LIBRARY 'RTLSTARLE'; ! System symbols
91 0772 1 |
92 0773 1 |
93 0774 1 | MACROS:
94 0775 1 |
95 0776 1 | NONE
96 0777 1 |
97 0778 1 | EQUATED SYMBOLS:
98 0779 1 |
99 0780 1 | NONE
100 0781 1 |
101 0782 1 | PSECTS:
102 0783 1 |
103 0784 1 DECLARE_PSECTS (BAS); ! Declare psects for BAS$ facility
104 0785 1 |
105 0786 1 | OWN STORAGE:
106 0787 1 |
107 0788 1 | NONE
108 0789 1 |
109 0790 1 | EXTERNAL REFERENCES:
110 0791 1 |
111 0792 1 |
112 0793 1 EXTERNAL ROUTINE
113 0794 1 BAS$CCPOS, ! Get current position
114 0795 1 STR$DUPL_CHAR, ! Produce spaces
115 0796 1 BAS$CB_GET : JSB CB GET NOVALUE, ! Load current CCB
116 0797 1 BAS$REC_WSL1 : JSB_REC_WSL1 NOVALUE, ! Write a record
```

BASSTAB
1-007

H 5
16-Sep-1984 01:18:05 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:56:41 [BASRTL.SRC]BASTAB.B32;1

Page 4
(2)

```
: 117      0798 1      BAS$$SIGNAL : NOVALUE,          ! Signal an error
: 118      0799 1      BAS$$STOP : NOVALUE;           ! Signal fatal error
: 119      0800 1
: 120      0801 1      !+
: 121      0802 1      !- The following are the error codes used in this module.
: 122      0803 1
: 123      0804 1
: 124      0805 1      EXTERNAL LITERAL
: 125      0806 1      BAS$K_SYNERR : UNSIGNED (8),      ! Syntax error
: 126      0807 1      BAS$K_NEGZERTAB : UNSIGNED (8);    ! Negative or zero TAB
: 127      0808 1
```



```
129 0809 1 GLOBAL ROUTINE BAS$TAB (
130 0810 1     RESULT,
131 0811 1     POSITION
132 0812 1 ) =
133 0813 1
134 0814 1 !++
135 0815 1 ! FUNCTIONAL DESCRIPTION:
136 0816 1
137 0817 1     Produce enough spaces that, if they were printed, we would
138 0818 1     advance to the specified position. If we are beyond the
139 0819 1     specified position, return the null string.
140 0820 1
141 0821 1     Note that the compiler will convert calls to the TAB function
142 0822 1     to calls to STR$DUPL_CHAR if the call is not lexically inside
143 0823 1     an I/O list, so this routine need only worry about the
144 0824 1     "true" TAB function. With the current structure of the RTL
145 0825 1     only the compiler knows which kind of TAB function is being
146 0826 1     used: consider a function call in an I/O list, with the
147 0827 1     function calling TAB.
148 0828 1
149 0829 1 ! FORMAL PARAMETERS:
150 0830 1
151 0831 1     RESULT.wt.dx    A string containing the number of spaces
152 0832 1                   required to reach the specified position.
153 0833 1     POSITION.rl.v    The target position.
154 0834 1
155 0835 1 ! IMPLICIT INPUTS:
156 0836 1
157 0837 1     OTS$$A_CUR_LUB.ra    The LUB of the current I/O list
158 0838 1                       We get from it the current position.
159 0839 1
160 0840 1 ! IMPLICIT OUTPUTS:
161 0841 1
162 0842 1     NONE
163 0843 1
164 0844 1 ! COMPLETION CODES:
165 0845 1
166 0846 1     Same as STR$DUPL_CHAR
167 0847 1
168 0848 1 ! SIDE EFFECTS:
169 0849 1
170 0850 1     Signals if an error is encountered.
171 0851 1     BAS$K_SYNERR means that the TAB function has been called
172 0852 1     not in an I/O list.
173 0853 1
174 0854 1 !--
175 0855 1
176 0856 2     BEGIN
177 0857 2
178 0858 2     GLOBAL REGISTER
179 0859 2     CCB = K_CCB_REG : REF BLOCK [, BYTE];
180 0860 2
181 0861 2     LOCAL
182 0862 2     CURRENT_POS;
183 0863 2
184 0864 2 !+
185 0865 2 ! Load register 11 so we can get the unit number.
```

```
186 0866 2 !-
187 0867 BAS$$CB_GET ();
188 0868
189 0869 IF (.CCB EQLA 0) THEN BAS$$STOP (BAS$K_SYNERR);
190 0870
191 0871 CURRENT_POS = BAS$CCPOS ((IF (.CCB [LUB$V_UNIT_0]) THEN 0 ELSE .CCB [LUB$W_LUN]));
192 0872
193 0873 IF (.CURRENT_POS GEQ .POSITION)
194 0874 THEN
195 0875 BEGIN
196 0876 !+
197 0877 We have gone too far, return the null string.
198 0878 !-
199 0879 RETURN (STR$DUPL_CHAR (.RESULT, %REF (0)));
200 0880 END
201 0881 ELSE
202 0882 BEGIN
203 0883 !+
204 0884 Return enough spaces to get to the target position.
205 0885 !-
206 0886 RETURN (STR$DUPL_CHAR (.RESULT, %REF (.POSITION - .CURRENT_POS)));
207 0887 END;
208 0888 ! end of BAS$TAB
```

```
.TITLE BAS$TAB
.IDENT \1-007\

.EXTRN BAS$CCPOS, STR$DUPL_CHAR
.EXTRN BAS$$CB_GET, BAS$REC_WSL1
.EXTRN BAS$SIGNAL, BAS$STOP
.EXTRN BAS$K_SYNERR, BAS$K_NEGZERTAB

.PSECT _BAS$CODE, NOWRT, SHR, PIC, 2
```

			0800 00000	.ENTRY BAS\$TAB, Save R11	0809
				SUBL2 #4, SP	
	5E	00000000G	04 C2 00002	JSB BAS\$\$CB_GET	0867
			00 16 00005	TSTL CCB	0869
			5B D5 0000B	BNEQ 1\$	
			0B 12 0000D	MOVZBL #BAS\$K_SYNERR, -(SP)	
	7E	00G	8F 9A 0000F	CALLS #1, BAS\$\$STOP	
00000000G	00		01 FB 00013	TSTB -2(CCB)	0871
		FE	AB 95 0001A 1\$:	BGEQ 2\$	
			04 18 0001D	CLRL -(SP)	
			7E D4 0001F	BRB 3\$	
			04 11 00021	CVTBL -58(CCB), -(SP)	
	7E	C6	AB 32 00023 2\$:	CALLS #1, BAS\$CCPOS	
00000000G	00		01 FB 00027 3\$:	CMPD CURRENT_POS, POSITION	0873
08	AC		50 D1 0002E	BLSS 4\$	
			04 19 00032	CLRL (SP)	0879
			6E D4 00034	BRB 5\$	
			05 11 00036	SUBL3 CURRENT_POS, POSITION, (SP)	0886
6E	08	AC	50 C3 00038 4\$:	PUSHL SP	
			5E DD 0003D 5\$:	PUSHL RESULT	
			AC DD 0003F	CALLS #2, STR\$DUPL_CHAR	
00000000G	00	04	02 FB 00042	RET	0888
			04 00049		

BAS\$TAB
1-007

K 5
16-Sep-1984 01:18:05
14-Sep-1984 11:56:41

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

Page 7
(3)

; Routine Size: 74 bytes, Routine Base: _BAS\$CODE + 0000

```
210 0889 1 GLOBAL ROUTINE BASSANSI_TAB (      ! TAB by ANSI standards
211 0890 1     RESULT,                          ! Descriptor of resultant spaces
212 0891 1     POSITION                          ! Where to go
213 0892 1 ) =
214 0893 1
215 0894 1 ++
216 0895 1 FUNCTIONAL DESCRIPTION:
217 0896 1
218 0897 1     Produce enough spaces that, if they were printed, we would
219 0898 1     advance to the specified position. Similar to BAS$TAB,
220 0899 1     but, to satisfy Minimal ANSI standards, includes extra
221 0900 1     checks on the input argument POSITION.
222 0901 1
223 0902 1 FORMAL PARAMETERS:
224 0903 1
225 0904 1     RESULT.wt.dx      A string containing the number of spaces
226 0905 1                     required to reach the specified position.
227 0906 1     POSITION.rl.v      The target position
228 0907 1
229 0908 1 IMPLICIT INPUTS:
230 0909 1
231 0910 1     OTSS$A_CUR_LUB.ra      The LUB of the current I/O list
232 0911 1                     We get it from the current position.
233 0912 1
234 0913 1 IMPLICIT OUTPUTS:
235 0914 1
236 0915 1     NONE
237 0916 1
238 0917 1 COMPLETION CODES:
239 0918 1
240 0919 1     Same as STR$DUPL_CHAR
241 0920 1
242 0921 1 SIDE EFFECTS:
243 0922 1
244 0923 1     Signals NEGZERTAB if POSITION arg is negative or zero
245 0924 1     BASSK_SYNERR means that the TAB function has been called
246 0925 1     not in an I/O list.
247 0926 1
248 0927 1 --
249 0928 1
250 0929 2 BEGIN
251 0930 2
252 0931 2 GLOBAL REGISTER
253 0932 2     CCB = K_CCB_REG : REF BLOCK [, BYTE];
254 0933 2
255 0934 2 LOCAL
256 0935 2     CURRENT_POS;
257 0936 2
258 0937 2 ++
259 0938 2 ! If a negative tab is requested, signal an info error and supply 1.
260 0939 2 --
261 0940 2
262 0941 3 IF (.POSITION LEQ 0)
263 0942 3 THEN
264 0943 3     BEGIN
265 0944 3     BASS$SIGNAL (BASSK_NEGZERTAB);
266 0945 3     POSITION = 1;
```



```
267 0946 2 END;
268 0947
269 0948
270 0949 + adjust POSITION - for ANSI tab, a tab of 1 really means tab 0 and
271 0950 start printing in column 1, whereas with a regular tab it would
272 0951 mean tab 1 and start printing in column 2.
273 0952 (see comment for edit 1-006 in edit history).
274 0953
275 0954 - POSITION = .POSITION - 1;
276 0955
277 0956 +
278 0957 Load register 11 so we can get the unit number.
279 0958 -
280 0959
281 0960 BAS$$CB_GET ();
282 0961
283 0962 If (.CCB EQLA 0) THEN BAS$$STOP (BAS$K_SYNNERR);
284 0963
285 0964 CURRENT_POS = BAS$CCPOS ((IF (.CCB [LUB$V_UNIT_0]) THEN 0 ELSE .CCB [LUB$W_LUN]));
286 0965
287 0966 +
288 0967 POSITION should be modulo margin.
289 0968 -
290 0969
291 0970 POSITION = .POSITION MOD .CCB [LUB$W_R_MARGIN];
292 0971
293 0972 +
294 0973 If we are already beyond the requested position, start a new line and space
295 0974 POSITION spaces.
296 0975 -
297 0976
298 0977 IF (.CURRENT_POS GTR .POSITION)
299 0978 THEN
300 0979 BEGIN
301 0980 BAS$$REC WSL1 (BAS$K_MAR_EXC);
302 0981 CCB [LUB$V_FORM_CHAR] = 0;
303 0982 CCB [LUB$L_PRINT_POS] = 0;
304 0983 POSITION = MAX (0, .POSITION);
305 0984 CURRENT_POS = 0;
306 0985 END;
307 0986
308 0987 +
309 0988 If the requested position equals the current position, do nothing.
310 0989 -
311 0990
312 0991 IF (.CURRENT_POS EQL .POSITION)
313 0992 THEN
314 0993 BEGIN
315 0994 RETURN (STR$DUPL_CHAR (.RESULT, %REF(0)));
316 0995 END
317 0996 ELSE
318 0997 BEGIN
319 0998 +
320 0999 Return enough spaces to get to the target position.
321 1000 -
322 1001
323 1002 RETURN (STR$DUPL_CHAR (.RESULT, %REF (.POSITION - .CURRENT_POS)));
```

BAS\$TAB
1-007

N 5
16-Sep-1984 01:18:05
14-Sep-1984 11:56:41

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

Page 10
(4)

: 324
: 325
: 326
1003 2
1004 2
1005 1
END;
END;

! end of BAS\$ANSI_TAB

				083C 00000	.ENTRY	BAS\$ANSI_TAB, Save R2,R3,R4,R5,R11	: 0889
	5E		08	04 C2 00002	SUBL2	#4, SP	: 0941
			08	AC D5 00005	TSTL	POSITION	: 0944
			08	0F 14 00008	BGTR	1\$	: 0945
	7E	00G	08	8F 9A 0000A	MOVZBL	#BAS\$K_NEGZERTAB, -(SP)	: 0954
00000000G	00		08	01 FB 0000E	CALLS	#1, BAS\$SIGNAL	: 0960
	08	AC	08	01 D0 00015	MOVL	#1, POSITION	: 0962
			08	AC D7 00019	DECL	POSITION	: 0964
		00000000G	08	00 16 0001C	JSB	BAS\$CB_GET	: 0970
			08	5B D5 00022	TSTL	CCB	: 0977
			08	0B 12 00024	BNEQ	2\$	: 0980
	7E	00G	08	8F 9A 00026	MOVZBL	#BAS\$K_SYNERR, -(SP)	: 0981
00000000G	00		FE	01 FB 0002A	CALLS	#1, BAS\$STOP	: 0982
			FE	AB 95 00031	TSTB	-2(CCB)	: 0983
			04	18 00034	BGEQ	3\$	: 0984
			7E	D4 00036	CLRL	-(SP)	: 0991
			04	11 00038	BRB	4\$	: 0994
	7E	C6	AB	32 0003A	CVTL	-58(CCB), -(SP)	: 1002
00000000G	00		01	FB 0003E	CALLS	#1, BAS\$CPOS	: 1005
	52		50	D0 00045	MOVL	R0, CURRENT_POS	
	50	D4	AB	3C 00048	MOVZWL	-44(CCB), R0	
7E	00	08	01	7A 0004C	EMUL	#1, POSITION, #0, -(SP)	
50	50	08	50	7B 00052	EDIV	R0, (SP)+, R0, R0	
	08	AC	50	D0 00057	MOVL	R0, POSITION	
	08	AC	52	D1 0005B	CMPL	CURRENT_POS, POSITION	
			1E	15 0005F	BLEQ	6\$	
	50		06	D0 00061	MOVL	#6, R0	
		00000000G	00	16 00064	JSB	BAS\$REC_WSL1	
	FE	AB	04	8A 0006A	BICB2	#4, -2(CCB)	
			AB	D4 0006E	CLRL	-56(CCB)	
	50	C8	AC	D0 00071	MOVL	POSITION, R0	
		08	02	18 00075	BGEQ	5\$	
			50	D4 00077	CLRL	R0	
	08	AC	50	D0 00079	MOVL	R0, POSITION	
			52	D4 0007D	CLRL	CURRENT_POS	
	08	AC	52	D1 0007F	CMPL	CURRENT_POS, POSITION	
			04	12 00083	BNEQ	7\$	
			6E	D4 00085	CLRL	(SP)	
			05	11 00087	BRB	8\$	
	6E	08	52	C3 00089	SUBL3	CURRENT_POS, POSITION, (SP)	
			5E	DD 0008E	PUSHL	SP	
			AC	DD 00090	PUSHL	RESULT	
00000000G	00		02	FB 00093	CALLS	#2, STR\$DUPL_CHAR	
		04	04	0009A	RET		

; Routine Size: 155 bytes, Routine Base: _BAS\$CODE + 004A

; 327 1006 1

BAS\$TAB
1-007

B 6
16-Sep-1984 01:18:05 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:56:41 [BASRTL.SRC]BASTAB.B32;1

Page 11
(4)

: 328 1007 1 END
: 329 1008 1
: 330 1009 0 ELUDOM

! end of module BAS\$TAB

PSECT SUMMARY

: Name Bytes Attributes
: _BAS\$CODE 229 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 0 0 581 00:01.1

COMMAND QUALIFIERS

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

: Size: 229 code + 0 data bytes
: Run Time: 00:09.4
: Elapsed Time: 00:29.8
: Lines/CPU Min: 6447
: Lexemes/CPU-Min: 31003
: Memory Used: 103 pages
: Compilation Complete

0032 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

