


```
CCCCCCCC HH HH MM MM FFFFFFFF IIIIII NN NN SSSSSSSS TTTTTTTTTT RRRRRRRR
CCCCCCCC HH HH MM MM FFFFFFFF IIIIII NN NN SSSSSSSS TTTTTTTTTT RRRRRRRR
CC HH HH MM MM FF FF NN NN SS SS
CC HH HH MMM MMM FF NN NN SS SS
CC HH HH MM MM FF NN NN SS SS
CC HH HH MM MM FF FFFFFFFF IIIIII NN NN SSSSSS TTTTTT RRRRRRRR
CC HH HH MM MM FFFFFFFF IIIIII NN NN SSSSSS TTTTTT RRRRRRRR
CC HH HH MM MM FF FF NN NN SSSSSS SS
CC HH HH MM MM FF FF NN NN SSSSSS SS
CC HH HH MM MM FF FF NN NN SSSSSS SS
CC HH HH MM MM FF FF NN NN SSSSSS SS
CCCCCCCC HH HH MM MM FF FF IIIIII NN NN SSSSSSSS TTTTTT RRRRRRRR
CCCCCCCC HH HH MM MM FF FF IIIIII NN NN SSSSSSSS TTTTTT RRRRRRRR

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 IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS
```



```
0001 0 %TITLE 'EDT$CHMF INSTR - search for a specific string'
0002 0 MODULE EDT$CHMF INSTR ( ! Search for a specific string
0003 0 IDENT = 'V04-000' ! File: CHMF INSTR.BLI Edit: JBS1007
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1 *****
0008 1 *
0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0011 1 * ALL RIGHTS RESERVED.
0012 1 *
0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0018 1 * TRANSFERRED.
0019 1 *
0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0022 1 * CORPORATION.
0023 1 *
0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0026 1 *
0027 1 *
0028 1 *****
0029 1
0030 1
0031 1 ++
0032 1 FACILITY: EDT -- The DEC Standard Editor
0033 1
0034 1 ABSTRACT:
0035 1
0036 1 This module searches text for a specific string.
0037 1
0038 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0039 1
0040 1 AUTHOR: Bob Kushlis, CREATION DATE: Unknown
0041 1
0042 1 MODIFIED BY:
0043 1
0044 1 1-001 - Original. DJS 04-Feb-1981. This module was created by
0045 1 extracting the routine EDT$$$STR SEACMD from module CHANGE.BLI.
0046 1 1-002 - Regularized headers. JBS 27-Feb-1981
0047 1 1-003 - Fix module name. JBS 02-Mar-1981
0048 1 1-004 - Change return value on backwards search with search end set. SMB 27-Oct-1981
0049 1 1-005 - Set a flag if control C actually aborts something. JBS 24-May-1982
0050 1 1-006 - Return a zero if string not found. SMB 17-Aug-1982
0051 1 1-007 - Remove EDT$G_LN_NO for new screen update logic. JBS 29-Sep-1982
0052 1 --
0053 1
```

```
.. 55      0054 1 %SBTTL 'Declarations'
.. 56      0055 1
.. 57      0056 1  TABLE OF CONTENTS:
.. 58      0057 1
.. 59      0058 1
.. 60      0059 1  REQUIRE 'EDT$SRC:TRAROUNAM';
.. 61      0498 1
.. 62      0499 1  FORWARD ROUTINE
.. 63      0500 1    EDT$$$STR_SEACMD;
.. 64      0501 1
.. 65      0502 1
.. 66      0503 1  INCLUDE FILES:
.. 67      0504 1
.. 68      0505 1
.. 69      0506 1  REQUIRE 'EDT$SRC:EDTREQ';
.. 70      0641 1
.. 71      0642 1
.. 72      0643 1  MACROS:
.. 73      0644 1
.. 74      0645 1    NONE
.. 75      0646 1
.. 76      0647 1  EQUATED SYMBOLS:
.. 77      0648 1
.. 78      0649 1    NONE
.. 79      0650 1
.. 80      0651 1  OWN STORAGE:
.. 81      0652 1
.. 82      0653 1    NONE
.. 83      0654 1
.. 84      0655 1  EXTERNAL REFERENCES:
.. 85      0656 1
.. 86      0657 1    In the routine
```

! Search for a specific string


```

88 0658 1 %SBTTL 'EDT$$$STR_SEACMD - search for a specific string'
89 0659 1
90 0660 1 GLOBAL ROUTINE EDT$$$STR_SEACMD (
91 0661 1     ADDR,
92 0662 1     LEN,
93 0663 1     M,
94 0664 1     DIR
95 0665 1 ) =
96 0666 1
97 0667 1 ++
98 0668 1 FUNCTIONAL DESCRIPTION:
99 0669 1
100 0670 1     This routine searches either forward or backward from the current
101 0671 1     position looking for a specific string.
102 0672 1
103 0673 1 FORMAL PARAMETERS:
104 0674 1
105 0675 1     ADDR      the address of the string to find
106 0676 1     LEN      the length of the string
107 0677 1     M        indicates whether or not we should move a character before beginning.
108 0678 1     DIR      the direction of the search.
109 0679 1
110 0680 1 IMPLICIT INPUTS:
111 0681 1
112 0682 1     EDT$$A_US_ENT
113 0683 1     EDT$$G_SEA_BEG
114 0684 1     EDT$$G_SEA_BNDD
115 0685 1     EDT$$T_LN_BUF
116 0686 1     EDT$$A_LN_PTR
117 0687 1     EDT$$A_LN_END
118 0688 1
119 0689 1 IMPLICIT OUTPUTS:
120 0690 1
121 0691 1     EDT$$A_CUR_BUF
122 0692 1     EDT$$A_LN_PTR
123 0693 1     EDT$$G_CC_DONE
124 0694 1
125 0695 1 ROUTINE VALUE:
126 0696 1
127 0697 1     1 is returned if the string is found, 0 if not, 2 if search aborted by control C.
128 0698 1
129 0699 1 SIDE EFFECTS:
130 0700 1
131 0701 1     NONE
132 0702 1
133 0703 1 --
134 0704 1
135 0705 2 BEGIN
136 0706 2
137 0707 2 EXTERNAL ROUTINE
138 0708 2     EDT$$CHK_CC,
139 0709 2     EDT$$GET_TXTLN,
140 0710 2     EDT$$CS_LEFT,
141 0711 2     EDT$$CS_RIGHT,
142 0712 2     EDT$$CS_UP,
143 0713 2     EDT$$TST_ONSTR,
144 0714 2     EDT$$RPOS : NOVALUE,

! Check to see if a CTRL/C has been typed
! Get current line in line buffer
! Move left a character
! Move right a character
! Move up a line
! Compare the current character position with a string descriptor
! Restore the saved buffer position
```

```
: 145      0715      2      EDT$$$SAV_BUFPOS : NOVALUE,      ! Save the current buffer position
: 146      0716      2      EDT$$$STR_SEA,
: 147      0717      2      EDT$$$G_CC_DONE;      ! Set to 1 if control C actually aborts something
: 148      0718
: 149      0719      2      EXTERNAL
: 150      0720      2      EDT$$$A_US_ENT : VECTOR,      ! Pointers to user defined entities
: 151      0721      2      EDT$$$G_SEA_BEG,      ! Leave search at beginning if on
: 152      0722      2      EDT$$$G_SEA_BNDD,      ! Is search bounded.
: 153      0723      2      EDT$$$A_CUR_BUF : REF TBCB_BLOCK,      ! The current buffer tbc
: 154      0724      2      EDT$$$T_LN_BUF,      ! Current line buffer
: 155      0725      2      EDT$$$A_LN_PTR,      ! Current character pointer
: 156      0726      2      EDT$$$A_LN_END;      ! End of current line pointer
: 157      0727
: 158      0728      2      LABEL
: 159      0729      2      SEARCH_LOOP;
: 160      0730
: 161      0731      2      LOCAL
: 162      0732      2      MOVE,
: 163      0733      2      INIT_LEN,
: 164      0734      2      CP,
: 165      0735      2      C,
: 166      0736      2      START_POS : POS_BLOCK,
: 167      0737      2      STS;
: 168      0738
: 169      0739      2      MOVE = .M;
: 170      0740      2      +
: 171      0741      2      Remember where we started.
: 172      0742      2      -
: 173      0743      2      EDT$$$SAV_BUFPOS (START_POS);
: 174      0744      2      +
: 175      0745      2      If searching backward, with search at end, move length of search string back
: 176      0746      2      before starting, so we won't keep finding the same string! Also, if searching
: 177      0747      2      forward with search at end, do not move before starting search.
: 178      0748      2      -
: 179      0749
: 180      0750      3      IF (.EDT$$$G_SEA_BEG EQL 0) :
: 181      0751      2      THEN
: 182      0752
: 183      0753      3      IF (.DIR EQL DIR_BACKWARD)
: 184      0754      2      THEN
: 185      0755      3      BEGIN
: 186      0756      3      INCR I FROM 1 TO .LEN DO
: 187      0757      3      IF ( NOT EDT$$$CS_LEFT () )
: 188      0758      3      THEN
: 189      0759      4      BEGIN
: 190      0760      3      +
: 191      0761      4      If cursor-position minus buffer-begin-position is less than length of
: 192      0762      4      search string, then search automatically fails. Reposition cursor
: 193      0763      4      and return not found.
: 194      0764      4      -
: 195      0765      4      EDT$$$RPOS (START_POS);
: 196      0766      4      RETURN (0);
: 197      0767      4      END;
: 198      0768      4
: 199      0769      3
: 200      0770      3
: 201      0771      3      END
```



```
202 0772 2 ELSE
203 0773 2 MOVE = 0;
204 0774 2
205 0775 2 !+
206 0776 2 Find the prefix of the search string up to the first carriage return.
207 0777 2 !-
208 0778 2 INIT_LEN = 0;
209 0779 2 CP = ".ADDR;
210 0780 2
211 0781 2 WHILE ((.INIT_LEN LSS .LEN) AND CH$RCHAR_A (CP) NEQ ASC_K_CR) DO
212 0782 2 INIT_LEN = .INIT_LEN + 1;
213 0783 2
214 0784 2 !+
215 0785 2 Now, look for the string.
216 0786 2 !-
217 0787 2 STS = 0;
218 0788 2 SEARCH_LOOP :
219 0789 2 BEGIN
220 0790 2
221 0791 3 WHILE 1 DO
222 0792 4 BEGIN
223 0793 4
224 0794 4 IF EDT$$CHK_CC ()
225 0795 4 THEN
226 0796 5 BEGIN
227 0797 5 STS = 2;
228 0798 5 EDT$$G_CC_DONE = 1;
229 0799 5 EXITLOOP;
230 0800 4 END;
231 0801 4
232 0802 4 IF .MOVE
233 0803 4 THEN
234 0804 4
235 0805 4 IF ((IF .DIR EQL DIR_BACKWARD THEN EDT$$CS_LEFT () ELSE EDT$$CS_RIGHT ()) EQL 0) THEN EXITLOOP;
236 0806 4
237 0807 4 MOVE = 1;
238 0808 4 !+
239 0809 4 Find next occurrence of the initial string.
240 0810 4 !-
241 0811 4
242 0812 5 IF (.EDT$$G_SEA_BNDD EQL 0)
243 0813 4 THEN
244 0814 4
245 0815 5 IF (.INIT_LEN NEQ 0)
246 0816 4 THEN
247 0817 5 BEGIN
248 0818 5 EDT$$A_CUR_BUF [TBCB_CHAR_POS] = CH$DIFF (.EDT$$A_LN_PTR, CH$PTR (EDT$$T_LN_BUF));
249 0819 5
250 0820 5 IF ( NOT (STS = EDT$$STR_SEA (.ADDR, .INIT_LEN, .DIR))) THEN EXITLOOP;
251 0821 5
252 0822 5 EDT$$GET_TXTLN ();
253 0823 5 END
254 0824 4 ELSE
255 0825 4 !+
256 0826 4 Look for the next carriage return.
257 0827 4 !-
258 0828 4
```


EDT\$CHMF INSTR
V04-000

EDT\$CHMF INSTR - search for a specific string
EDT\$\$STR_SEACMD - search for a specific string

C 9
15-Sep-1984 23:56:28
14-Sep-1984 12:22:31

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMF INSTR.BLI;1

Page 6
(3)

```

259      0829 4      WHILE (CH$RCHAR (.EDT$$A_LN_PTR) NEQ ASC_K_CR) DO
260      0830 4
261      0831 5      IF (.DIR EQL DIR_BACKWARD)
262      0832 4      THEN
263      0833 4
264      0834 5      IF CH$PTR_EQL (.EDT$$A_LN_PTR, CH$PTR (EDT$$T_LN_BUF))
265      0835 4      THEN
266      0836 4
267      0837 4      IF EDT$$CS_UP () THEN EDT$$A_LN_PTR = .EDT$$A_LN_END ELSE LEAVE SEARCH_LOOP
268      0838 4
269      0839 4      ELSE
270      0840 4      EDT$$A_LN_PTR = CH$PLUS (.EDT$$A_LN_PTR, -1)
271      0841 4
272      0842 4      ELSE
273      0843 4      EDT$$A_LN_PTR = CH$PLUS (.EDT$$A_LN_PTR, 1);
274      0844 4
275      0845 4      !+
276      0846 4      ! Now check to see if the entire string matches.
277      0847 4      !-
278      0848 4
279      0849 4      IF EDT$$TST_ONSTR (.ADDR, .LEN)
280      0850 4      THEN
281      0851 5      BEGIN
282      0852 5      !+
283      0853 5      ! If we have gotten here, the string matched.
284      0854 5      !-
285      0855 5      RETURN (1);
286      0856 5      END
287      0857 4      ELSE
288      0858 4      STS = 0;
289      0859 4
290      0860 4      !+
291      0861 4      ! If search is bounded, fail if we are on a page mark.
292      0862 4      !-
293      0863 4
294      0864 5      IF (.EDT$$G_SEA_BNDD NEQ 0)
295      0865 4      THEN
296      0866 4
297      0867 4      IF EDT$$TST_ONSTR (CH$PLUS (.EDT$$A_US_ENT [3], 1), CH$RCHAR (.EDT$$A_US_ENT [3])) THEN EXITLOOP
298      0868 4
299      0869 3      END;
300      0870 3
301      0871 2      END;
302      0872 2      !+
303      0873 2      ! String was not found, reposition.
304      0874 2      !-
305      0875 2      EDT$$RPOS (START_POS);
306      0876 2      RETURN (.STS);
307      0877 1      END;

```

! of routine EDT\$\$STR_SEACMD

```

.TITLE EDT$CHMF INSTR EDT$CHMF INSTR - search for a spec
       ific string
.IDENT  \V04-000\
.EXTRN EDT$$CHK_CC, EDT$$GET_TXTLN
.EXTRN EDT$$CS_LEFT, EDT$$CS_RIGHT

```


				07FC 00000			
	5A	00000000G	00	9E	00002		
	59	00000000G	00	9E	00009		
	58	00000000G	00	9E	00010		
	57	00000000G	00	9E	00017		
	56	00000000G	00	9E	0001E		
	55	00000000G	00	9E	00025		
	5E		10	C2	0002C		
	54	0C	AC	D0	0002F		
			5E	DD	00033		
00000000G	00		01	FB	00035		
		00000000G	00	D5	0003C		
			20	12	00042		
		10	AC	D5	00044		
			19	12	00047		
			52	D4	00049		
			0E	11	0004B		
	66		00	FB	0004D	1\$:	
	08		50	E8	00050		
			5E	DD	00053		
	67		01	FB	00055		
		00ED	31	00058			
ED	52		08	AC	F3	0005B	2\$:
			02	11	00060		
			54	D4	00062	3\$:	
			53	D4	00064	4\$:	
	50		04	AC	D0	00066	
08	AC		53	D1	0006A	5\$:	
			0C	18	0006E		
	51		80	9A	00070		
	0D		51	91	00073		
			04	13	00076		
			53	D6	00078		
			EE	11	0007A		
			52	D4	0007C	6\$:	
00000000G	00		00	FB	0007E	7\$:	
	0D		50	E9	00085		
	52		02	D0	00088		
00000000G	00		01	D0	0008B		
		00AA	31	00092	8\$:		
	15		54	E9	00095	9\$:	
		10	AC	D5	00098		
			05	12	0009B		
	66		00	FB	0009D		
			07	11	000A0		
00000000G	00		00	FB	000A2	10\$:	

.EXTRN	EDT\$\$CS UP, EDT\$\$TST ONSTR	
.EXTRN	EDT\$\$RPOS, EDT\$\$SAV BUFPOS	
.EXTRN	EDT\$\$STR_SEA, EDT\$\$G_CC DONE	
.EXTRN	EDT\$\$A_US_ENT, EDT\$\$G_SEA_BEG	
.EXTRN	EDT\$\$G_SEA_BNDD	
.EXTRN	EDT\$\$A_CUR_BUF, EDT\$\$T_LN_BUF	
.EXTRN	EDT\$\$A_LN_PTR, EDT\$\$A_LN_END	
.PSECT	_EDT\$CODE, NOWRT, SHR, PIC, 2	
.ENTRY	EDT\$\$STR_SEACMD, Save R2,R3,R4,R5,R6,R7,R8,-, R9,R10	0660
MOVAB	EDT\$\$TST ONSTR, R10	
MOVAB	EDT\$\$T_LN_BUF, R9	
MOVAB	EDT\$\$G_SEA_BNDD, R8	
MOVAB	EDT\$\$RPOS, R7	
MOVAB	EDT\$\$CS_LEFT, R6	
MOVAB	EDT\$\$A_LN_PTR, R5	
SUBL2	#16, SP	
MOVL	M, MOVE	0739
PUSHL	SP	0743
CALLS	#1, EDT\$\$SAV BUFPOS	
TSTL	EDT\$\$G_SEA_BEG	0750
BNEQ	4\$	
TSTL	DIR	0753
BNEQ	3\$	
CLRL	I	0757
BRB	2\$	
CALLS	#0, EDT\$\$CS_LEFT	0759
BLBS	R0, 2\$	
PUSHL	SP	0767
CALLS	#1, EDT\$\$RPOS	
BRW	21\$	0768
AOBLEQ	LEN, I, 1\$	0759
BRB	4\$	0753
CLRL	MOVE	0773
CLRL	INIT_LEN	0778
MOVL	ADDR, CP	0779
CMPL	INIT_LEN, LEN	0781
BGEQ	6\$	
MOVZBL	(CP)+, R1	
CMPB	R1, #13	
BEQL	6\$	
INCL	INIT_LEN	0782
BRB	5\$	
CLRL	STS	0787
CALLS	#0, EDT\$\$CHK_CC	0794
BLBC	R0, 9\$	
MOVL	#2, STS	0797
MOVL	#1, EDT\$\$G_CC_DONE	0798
BRW	20\$	0796
BLBC	MOVE, 12\$	0802
TSTL	DIR	0805
BNEQ	10\$	
CALLS	#0, EDT\$\$CS_LEFT	
BRB	11\$	
CALLS	#0, EDT\$\$CS_RIGHT	

		50	D5	000A9	11\$:	TSTL	R0		
		E5	13	000AB		BEQL	8\$		
	54	01	D0	000AD	12\$:	MOVL	#1, MOVE		0807
		68	D5	000B0		TSTL	EDT\$\$G_SEA_BNDD		0812
		61	12	000B2		BNEQ	16\$		
		53	D5	000B4		TSTL	INIT_LEN		0815
		2D	13	000B6		BEQL	13\$		
	50	00	D0	000B8		MOVL	EDT\$\$A_CUR_BUF, R0		0818
	51	69	9E	000BF		MOVAB	EDT\$\$T_LN_BUF, R1		
OC	A0	65	51	A3	000C2	SUBW3	R1, EDT\$\$A_LN_PTR, 12(R0)		
			AC	DD	000C7	PUSHL	DIR		0820
			53	DD	000CA	PUSHL	INIT_LEN		
			AC	DD	000CC	PUSHL	ADDR		
	00000000G	00	03	FB	000CF	CALLS	#3, EDT\$\$STR_SEA		
		52	50	D0	000D6	MOVL	R0, STS		
		63	52	E9	000D9	BLBC	STS, 20\$		
	00000000G	00	00	FB	000DC	CALLS	#0, EDT\$\$GET_TXTLN		0822
			30	11	000E3	BRB	16\$		0815
		51	65	D0	000E5	13\$:	MOVL	EDT\$\$A_LN_PTR, R1	0829
	0D		61	91	000E8		CMPB	(R1), #13	
			28	13	000EB		BEQL	16\$	
			AC	D5	000ED		TSTL	DIR	0831
			1F	12	000F0		BNEQ	15\$	
		50	69	9E	000F2		MOVAB	EDT\$\$T_LN_BUF, R0	0834
		50	51	D1	000F5		CMPL	R1, R0	
			13	12	000F8		BNEQ	14\$	
	00000000G	00	00	FB	000FA		CALLS	#0, EDT\$\$CS_UP	0837
		3B	50	E9	00101		BLBC	R0, 20\$	
	65	00000000G	00	D0	00104		MOVL	EDT\$\$A_LN_END, EDT\$\$A_LN_PTR	
			D8	11	0010B		BRB	13\$	
			65	D7	0010D	14\$:	DECL	EDT\$\$A_LN_PTR	0840
			D4	11	0010F		BRB	13\$	0834
			65	D6	00111	15\$:	INCL	EDT\$\$A_LN_PTR	0843
			D0	11	00113		BRB	13\$	0831
			AC	7D	00115	16\$:	MOVQ	ADDR, -(SP)	0849
	7E		02	FB	00119		CALLS	#2, EDT\$\$TST_ONSTR	
	6A		50	E9	0011C		BLBC	R0, 17\$	
	04		01	D0	0011F		MOVL	#1, R0	0855
	50			04	00122		RET		
			52	D4	00123	17\$:	CLRL	STS	0858
			68	D5	00125		TSTL	EDT\$\$G_SEA_BNDD	0864
			03	12	00127		BNEQ	19\$	
			FF52	31	00129	18\$:	BRW	7\$	
	50	00000000G	00	D0	0012C	19\$:	MOVL	EDT\$\$A_US_ENT+12, R0	0867
	7E		60	9A	00133		MOVZBL	(R0), =(SP)	
			A0	9F	00136		PUSHAB	1(R0)	
		01	02	FB	00139		CALLS	#2, EDT\$\$TST_ONSTR	
	6A		50	E9	0013C		BLBC	R0, 18\$	
	EA		5E	DD	0013F	20\$:	PUSHL	SP	0875
			01	FB	00141		CALLS	#1, EDT\$\$RPOS	
	67		52	D0	00144		MOVL	STS, R0	0876
	50			04	00147		RET		
			50	D4	00148	21\$:	CLRL	R0	0877
				04	0014A		RET		

; Routine Size: 331 bytes, Routine Base: _EDT\$CODE + 0000

EDT\$CHMF INSTR
V04-000

EDT\$CHMF INSTR - search for a specific string
EDT\$\$STR_SEACMD - search for a specific string

F 9
15-Sep-1984 23:56:28
14-Sep-1984 12:22:31

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMF INSTR.BLI;1

Page 9
(3)

; 308
; 309

0878 1
0879 1 !<BLF/PAGE>

: 311
: 312
: 313

0880 1 END
0881 1
0882 0 ELUDOM

! of module EDT\$CHMF INSTR

PSECT SUMMARY

Name	Bytes	Attributes
_EDT\$CODE	331	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[EDT.SRC]EDT.L32;1	377	36	9	40	00:00.2
\$255\$DUA28:[EDT.SRC]PSECTS.L32;1	2	1	50	7	00:00.1

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACEBACK/LIS=LIS\$:CHMF INSTR/OBJ=OBJ\$:CHMF INSTR MSRC\$:CHMF INSTR.BLI/UPDATE=(ENH\$:C
HMF INSTR)

: Size: 331 code + 0 data bytes
: Run Time: 00:18.4
: Elapsed Time: 00:22.7
: Lines/CPU Min: 2877
: Lexemes/CPU-Min: 8508
: Memory Used: 128 pages
: Compilation Complete

0131 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY