

MMM	MMM	AAAAAAAAA	CCCCCCCCCCCCC	RRRRRRRRRRRRR	000000000			
MMM	MMM	AAAAAAAAA	CCCCCCCCCCCCC	RRRRRRRRRRRRR	000000000			
MMM	MMM	AAAAAAAAA	CCCCCCCCCCCCC	RRRRRRRRRRRRR	000000000			
MMMMMM	MMMMMM	AAA	AAA	CCC	RRR	RRR	000	000
MMMMMM	MMMMMM	AAA	AAA	CCC	RRR	RRR	000	000
MMMMMM	MMMMMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRRRRRRRRRRRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRRRRRRRRRRRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRRRRRRRRRRRR	RRR	000	000
MMM	MMM	AAAAAAAAAAAAAAAAA	CCC	RRR	RRR	RRR	000	000
MMM	MMM	AAAAAAAAAAAAAAAAA	CCC	RRR	RRR	RRR	000	000
MMM	MMM	AAAAAAAAAAAAAAAAA	CCC	RRR	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCCCCCCCCCCCC	RRR	RRR	000000000	000000000
MMM	MMM	AAA	AAA	CCCCCCCCCCCCC	RRR	RRR	000000000	000000000
MMM	MMM	AAA	AAA	CCCCCCCCCCCCC	RRR	RRR	000000000	000000000

```
IIIIII  NN  NN  TTTTTTTTTT  000000  UU  UU  TTTTTTTTTT
IIIIII  NN  NN  TTTTTTTTTT  000000  UU  UU  TTTTTTTTTT
II      NN  NN  TT          00      00  UU  UU  TT
II      NN  NN  TT          00      00  UU  UU  TT
II      NNNN NN  TT          00      00  UU  UU  TT
II      NNNN NN  TT          00      00  UU  UU  TT
II      NN  NN  TT          00      00  UU  UU  TT
II      NN  NN  TT          00      00  UU  UU  TT
II      NN  NN  TT          00      00  UU  UU  TT
II      NN  NN  TT          00      00  UU  UU  TT
II      NN  NN  TT          00      00  UU  UU  TT
II      NN  NN  TT          00      00  UU  UU  TT
IIIIII  NN  NN  TT          000000  UUUUUUUUUU  TT
IIIIII  NN  NN  TT          000000  UUUUUUUUUU  TT
```

```
....
....
....
....
```

```
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
```


(2)	48
(3)	62
(4)	123
(5)	175
(6)	200
(7)	223

DECLARATIONS
MAC\$INTOUT_LW WRITE TO INT. BUFFER IN LONGWORDS
MAC\$INTOUT_WD WRITE WORD (2 BYTES) TO INT. FILE
MAC\$INTOUT_X WRITE ONLY ACTION CODE TO INT. FILE
MAC\$INTOUT_N PREPARE TO WRITE 'N' BYTES TO INT. FILE
COPY FRAME FROM INT. BUFFER TO VIRT. MEMORY

```
0000 1 .TITLE MAC$INTOUT WRITE CODE TO INTERMEDIATE BUFFER
0000 2 .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 *
0000 25 *
0000 26 *****
0000 27
0000 28
0000 29 ++
0000 30 FACILITY: VAX MACRO ASSEMBLER OBJECT LIBRARY
0000 31
0000 32 ABSTRACT:
0000 33
0000 34 The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000 35 modules for input to the VAX-11 LINKER.
0000 36
0000 37 ENVIRONMENT: USER MODE
0000 38
0000 39 AUTHOR: Benn Schreiber, CREATION DATE: 20-AUG-78
0000 40
0000 41 MODIFIED BY:
0000 42
0000 43 V01.02 RN0005 R. Newland 27-Aug-1979
0000 44 Remove .ALIGN LONG statements
0000 45
0000 46 --
```



```

0000 48      .SBTTL  DECLARATIONS
0000 49  :
0000 50  : INCLUDE FILES:
0000 51  :
0000 52  :
0000 53  :
0000 54  : MACROS:
0000 55  :
0000 56      $MAC_GENVALDEF      ;COMMON VAX-11 MACRO SYMBOLS
0000 57      $MAC_INTCODDEF     ;DEFINE INT. FILE CODES
0000 58      $MAC_CTLFLGDEF     ;FLAG BITS POINTED TO BY R11
0000 59
00000000 60      .PSECT  MAC$RO_CODE_P1,NOWRT,GBL, LONG

```



```
0000 62 .SBTTL MAC$INTOUT_LW WRITE TO INT. BUFFER IN LONGWORDS
0000 63
0000 64 :++
0000 65 : FUNCTIONAL DESCRIPTION:
0000 66
0000 67 : THIS ROUTINE IS CALLED TO PUT LONGWORDS FROM THE STACK INTO
0000 68 : THE INTERMEDIATE FILE.
0000 69
0000 70 : CALLING SEQUENCE:
0000 71
0000 72 : PUSHL LWN
0000 73 : PUSHL LWN-1
0000 74 :
0000 75 :
0000 76 : PUSHL LW1
0000 77 : BSBW MAC$INTOUT_N_LW
0000 78
0000 79 : INPUTS:
0000 80
0000 81 : LONGWORDS ON THE STACK
0000 82
0000 83 : OUTPUTS:
0000 84
0000 85 : LONGWORDS POPPED FROM STACK AND PUT IN INTER. BUFFER.
0000 86
0000 87 :--
0000 88
0000 89 MAC$INTERR 3_LW::
00 6B 07 E5 0000 90 BBCC #FLG$V_EXPOPT,(R11),.+1 ;DO NOT ALLOW EXPRESSION OPT.
0004 91 MAC$INTOUT 3_LW::
51 0E D0 0004 92 MOVL #14,R1 ;BYTE COUNT TOTAL
15 10 0007 93 BSBB OUT_LW ;JOIN COMMON CODE
03 0009 94 .BYTE 3 ;TO STORE 3 LONGWORDS
000A 95
000A 96 MAC$INTERR 2_LW::
00 6B 07 E5 000A 97 BBCC #FLG$V_EXPOPT,(R11),.+1 ;DO NOT ALLOW EXPRESSION OPT.
000E 98 MAC$INTOUT 2_LW::
51 0A D0 000E 99 MOVL #10,R1 ;BYTE COUNT TOTAL
0B 10 0011 100 BSBB OUT_LW ;GO TO COMMON CODE
02 0013 101 .BYTE 2 ;TO STORE 2 LONGWORDS
0014 102
0014 103 MAC$INTERR 1_LW::
00 6B 07 E5 0014 104 BBCC #FLG$V_EXPOPT,(R11),.+1 ;DO NOT ALLOW EXPRESSION OPT.
0018 105 MAC$INTOUT 1_LW::
51 06 D0 0018 106 MOVL #6,R1 ;BYTE COUNT TOTAL
01 10 001B 107 BSBB OUT_LW ;GO TO COMMON CODE
01 001D 108 .BYTE 1 ;TO STORE 1 BYTE
001E 109 OUT_LW:
52 0000'CF 51 C3 001E 110 SUBL3 R1,W*MAC$GL_INTCNT,R2 ;SEE IF ROOM IN BUFFER
09 14 0024 111 BGTR 10$ ;IF GTR YES
0096 30 0026 112 BSBW MAC$OUTFRAME ;NO--WRITE THE BUFFER
52 51 C3 0029 113 SUBL3 R1,W*MAC$GL_INTCNT,R2 ;UPDATE COUNT IN R2
0000'CF 52 D0 002F 114 10$: MOVL R2,W*MAC$GL_INTCNT ;UPDATE INT. COUNT
89 51 90 0034 115 MOVB R1,(R9)+ ;STORE COUNT BYTE
89 50 90 0037 116 MOVB R0,(R9)+ ;STORE ACTION BYTE
51 9E 9A 003A 117 MOVZBL @($P)+,R1 ;GET LONGWORD COUNT AND CLEAR IMMEDIATE CALL
50 8ED0 003D 118 POPL R0 ;POP OFF RETURN ADDRESS
```


MAC\$INTOUT
V04-000

WRITE CODE TO INTERMEDIATE BUFFER L 1
MAC\$INTOUT_LW WRITE TO INT. BUFFER IN LO

16-SEP-1984 02:06:09 VAX/VMS Macro V04-00
5-SEP-1984 01:48:36 [MACRO.SRC]INTOUT.MAR;1

Page 4
(3)

FA	89	8ED0	004C	119	20\$:	POPL	(R9)+		
	51	F5	0043	120		SOBGTR	R1,20\$		
	60	17	0046	121		JMP	(R0)		

;GET NEXT LONGWORD
;LOOP FOR ALL
;RETURN TO CALLER

```
0048 123 .SBTTL MAC$INTOUT_WD WRITE WORD (2 BYTES) TO INT. FILE
0048 124
0048 125 :++
0048 126 : FUNCTIONAL DESCRIPTION:
0048 127 :
0048 128 : THIS ROUTINE IS CALLED TO OUTPUT ONE WORD (2 BYTES) TO
0048 129 : THE INTERMEDIATE FILE.
0048 130 :
0048 131 : CALLING SEQUENCE:
0048 132 :
0048 133 : MOVZBL #ACTION,R0
0048 134 : MOVZWL DATA,R1
0048 135 : BSBW MAC$INTOUT_WD
0048 136 :
0048 137 :--
0048 138
0048 139 MAC$INTOUT_WD::
52 0000'CF 04 C3 0048 140 SUBL3 #4,W^MAC$GL_INTCNT,R2 ;SEE IF ROOM IN BUFFER
08 14 004E 141 BGTR 10$ ;IF GTR YES
6D 10 0050 142 BSBB MAC$OUTFRAME ;NO--WRITE THE BUFFER
52 0000'CF 04 C3 0052 143 SUBL3 #4,W^MAC$GL_INTCNT,R2 ;UPDATE R2
0000'CF 52 D0 0058 144 10$: MOVL R2,W^MAC$GL_INTCNT ;UPDATE INT. COUNT
89 04 90 005D 145 MOVB #4,(R9)+ ;COUNT IS 4 BYTES
89 50 90 0060 146 MOVB R0,(R9)+ ;STORE ACTION BYTE
89 51 B0 0063 147 MOVW R1,(R9)+ ;STORE DATA WORD
05 0066 148 RSB
0067 149
0067 150 :++
0067 151 : FUNCTIONAL DESCRIPTION:
0067 152 :
0067 153 : THIS ROUTINE IS CALLED TO OUTPUT A 'STORE IMMEDIATE' BYTE
0067 154 : COMMAND TO THE INTERMEDIATE FILE, AND ONLY STORE ONE BYTE
0067 155 : RATHER THAN A WHOLE LONGWORD. NOTE THAT THE PASS 2 PROCESSOR
0067 156 : MUST NOT CARE THAT THERE IS ONLY A BYTE.
0067 157 :
0067 158 : INPUTS:
0067 159 :
0067 160 : R0 BYTE TO STORE
0067 161 :
0067 162 :--
0067 163
0067 164 MAC$INTOUT_BY::
7E 0000'CF 03 C3 0067 165 SUBL3 #3,W^MAC$GL_INTCNT,-(SP) ;SEE IF ROOM
08 14 006D 166 BGTR 10$ ;IF GTR YES
4E 10 006F 167 BSBB MAC$OUTFRAME ;NO--WRITE BUFFER
6E 0000'CF 03 C3 0071 168 SUBL3 #3,W^MAC$GL_INTCNT,(SP) ;UPDATE STACK
0000'CF 8E D0 0077 169 10$: MOVL (SP)+,W^MAC$GL_INTCNT ;UPDATE COUNT
89 03 90 007C 170 MOVB #3,(R9)+ ;STORE BYTE COUNT
89 26 90 007F 171 MOVB #INT$ STIB,(R9)+ ;STORE ACTION CODE
89 50 90 0082 172 MOVB R0,(R9)+ ;STORE DATA BYTE
05 0085 173 RSB
```



```

0086 175 .SBTTL MAC$INTOUT_X WRITE ONLY ACTION CODE TO INT. FILE
0086 176
0086 177 :++
0086 178 : FUNCTIONAL DESCRIPTION:
0086 179
0086 180 : THIS ROUTINE IS CALLED TO OUTPUT ONLY AN ACTION CODE. IT
0086 181 : IS USED WHEN AN ACTION CODE TAKES NO OTHER PARAMETERS
0086 182 : (FOR INSTANCE, INT$_CHKL).
0086 183
0086 184 : INPUT PARAMETERS:
0086 185
0086 186 : R0 ACTION CODE
0086 187
0086 188 :--
0086 189
0086 190 MAC$INTOUT_X::
7E 0000'CF 02 C3 0086 191 SUBL3 #2,W^MAC$GL_INTCNT,-(SP) :SEE IF ROOM IN BUFFER
08 14 008C 192 BGTR 10$ :IF GTR YES
2F 10 008E 193 BSBB MAC$OUTFRAME :NO--OUTPUT CURRENT BUFFER
6E 0000'CF 02 C3 0090 194 SUBL3 #2,W^MAC$GL_INTCNT,(SP) :ADJUST VALUE ON STACK
0000'CF 8E D0 0096 195 10$: MOVL (SP)+,W^MAC$GL_INTCNT :SET NEW BUFFER COUNT
89 02 90 009B 196 MOVB #2,(R9)+ :SET BYTE COUNT IN BUFFER
89 50 90 009E 197 MOVB R0,(R9)+ :SET ACTION INTO BUFFER
05 00A1 198 RSB

```

```
00A2 200 .SBTTL MAC$INTOUT_N PREPARE TO WRITE 'N' BYTES TO INT. FILE
00A2 201
00A2 202 :++
00A2 203 : FUNCTIONAL DESCRIPTION:
00A2 204 :
00A2 205 : THIS ROUTINE PREPARES FOR SOMEONE TO EMIT 'N' BYTES TO THE
00A2 206 : INTERMEDIATE FILE.
00A2 207 :
00A2 208 : INPUT PARAMETERS:
00A2 209 :
00A2 210 : R0 NUMBER OF BYTES IN FRAME, INCLUDING COUNT BYTE
00A2 211 :
00A2 212 :--
00A2 213
00A2 214 MAC$INTOUT N::
7E 0000'CF 50 C3 00A2 215 SUBL3 R0,W^MAC$GL_INTCNT,-(SP);IS THERE ENOUGH ROOM IN THE BUFFER?
08 14 00A8 216 BGTR 10$ ;IF GTR YES
13 10 00AA 217 BSBB MAC$OUTFRAME ;NO--WRITE BUFFER TO VIRT. MEMORY
6E 0000'CF 50 C3 00AC 218 SUBL3 R0,W^MAC$GL_INTCNT,(SP) ;ADJUST THE COUNT ON THE STACK
0000'CF 8E D0 00B2 219 10$: MOVL (SP)+,W^MAC$GL_INTCNT ;UPDATE THE BUFFER COUNT
89 50 90 00B7 220 MOVB R0,(R9)+ ;STORE THE COUNT BYTE IN THE BUFFER
05 00BA 221 RSB
```



```
00BB 223 .SBTTL COPY FRAME FROM INT. BUFFER TO VIRT. MEMORY
00BB 224
00BB 225 :++
00BB 226 : FUNCTIONAL DESCRIPTION:
00BB 227 :
00BB 228 : THIS ROUTINE IS CALLED WHEN THE INT. BUFFER DOES NOT HAVE ENOUGH
00BB 229 : ROOM FOR THE NEXT FRAME. THE COUNT IS FIXED IN THE CURRENT BLOCK,
00BB 230 : A NEW BLOCK OF MEMORY IS ALLOCATED, AND THE POINTER AND COUNTER
00BB 231 : ARE SET UP FOR IT.
00BB 232 :
00BB 233 :--
00BB 234
00BB 235 MAC$SETFRAME::
03 BB 00BB 236 PUSH R0,R1> ;SAVE REGISTERS
04 11 00BD 237 BRB FRAM_0 ;JUST ALLOCATE THE BLOCK AND SETUP
00BF 238
00BF 239 MAC$OUTFRAME::
03 BB 00BF 240 PUSH R0,R1> ;PRESERVE ALL REGISTERS USED
48 10 00C1 241 BSBB MAC$FIXFRAME ;FIX BYTE COUNT IN CURRENT BLOCK
00C3 242 FRAM_0:
50 000013F4 8F DO 00C3 243 MOVL #INT$K_BUFSIZ,R0 ;GET THE SIZE OF A BUFFER
0000'CF 50 DO 00CA 244 MOVL R0,W^MAC$GL_INTCNT ;RESET THE COUNTER
OC A0 9F 00CF 245 PUSHAB 12(R0) ;STACK # BYTES WE REALLY NEED
50 5E DO 00D2 246 MOVL SP,R0 ;SAVE PTR TO COUNT
0000'CF DF 00D5 247 PUSHAL W^MAC$GL_INTFRMPT ;STACK PTR TO RESULT LONGWORD
50 DD 00D9 248 PUSH R0 ;STACK ADDRESS OF SIZE
00000000'GF 02 FB 00DB 249 CALLS #2,G^LIB$GET_VM ;ALLOCATE VM FOR NEXT BLOCK
23 50 E9 00E2 250 BLBC R0,20$ ;BRANCH IF ALLOC ERROR
8E D5 00E5 251 TSTL (SP)+ ;CLEAR COUNT FROM STACK
50 0000'CF DO 00E7 252 MOVL W^MAC$GL_INTFRMPT,R0 ;GET PTR TO THE BLOCK
0004'DF 60 OE 00EC 253 INSQUE (R0),@W^MAC$GL_INTQUE+4 ;INSERT AT TAIL OF QUEUE
59 08 A0 9E 00F1 254 MOVAB 8(R0),R9 ;POINT TO THE COUNT WORD
89 D4 00F5 255 CLRL (R9)+ ;CLEAR IT AND SET UP R9 AS POINTER
0000'CF 59 00001390 8F C1 00F7 256 ADDL3 #INT$K_BUFWRN,R9,W^MAC$GL_INTWRNPT ;SET UP WARNING LIMIT
03 BA 0101 257 POP R0,R1> ;RESTORE REGISTERS
00 6B 07 E5 0103 258 BBCC #FLG$V_EXPOPT,(R11),.+1 ;DO NOT ALLOW EXPRESSION OPT.
FEF5' 31 05 0107 259 RSB ;ALL DONE
20$: 0108 260 BRW MAC$ERR_NOMEM ;REPORT NO MEMORY TODAY AND QUIT
010B 261
010B 262 :++
010B 263 : FUNCTIONAL DESCRIPTION:
010B 264 :
010B 265 : THIS ROUTINE UPDATES THE BYTE COUNT IN THE CURRENT FRAME OF
010B 266 : THE INT. BUFFER.
010B 267 :
010B 268 :--
010B 269
010B 270 MAC$FIXFRAME::
50 0000'CF DO 010B 271 MOVL W^MAC$GL_INTFRMPT,R0 ;POINT TO THE CURRENT BLOCK
51 59 50 C3 0110 272 SUBL3 R0,R9,R1 ;FIGURE # BYTES USED IN IT
08 A0 51 OC C3 0114 273 SUBL3 #12,R1,8(R0) ;DON'T COUNT THE OVERHEAD BYTES
0000'CF 08 A0 C0 0119 274 ; AND STORE COUNT IN OVH COUNT WORD
05 011F 275 ADDL2 8(R0),W^MAC$GL_INTPAGRQ ;UPDATE COUNT OF BYTES USED
0120 276 RSB
0120 277
0120 278 .END
```


MAC\$INTOUT
Symbol table

WRITE CODE TO INTERMEDIATE BUFFER

D 2

16-SEP-1984 02:06:09
5-SEP-1984 01:48:36

VAX/VMS Macro V04-00
[MACRO.SRC]INTOUT.MAR;1

Page 9
(7)

\$COUNT = 0000003B
ARG\$K_SIZE = 000003E8
AUD\$K_SIZE = 00000010
BLNK = 00000020
CHRS\$M_COMMA_CR = 00000020
CHRS\$M_ILL_CHR = 00000040
CHRS\$M_NUM_BER = 00000010
CHRS\$M_SPA_MSK = 00000001
CHRS\$M_SYM_CH1 = 00000008
CHRS\$M_SYM_CHR = 00000004
CHRS\$M_SYM_DLM = 00000002
CHRS\$V_COMMA_CR = 00000005
CHRS\$V_CVTLWC = 00000061
CHRS\$V_ILL_CHR = 00000006
CHRS\$V_NOCVT = 0000007F
CHRS\$V_NUM_BER = 00000004
CHRS\$V_SPA_MSK = 00000000
CHRS\$V_SYM_CH1 = 00000003
CHRS\$V_SYM_CHR = 00000002
CHRS\$V_SYM_DLM = 00000001
CR = 0000000D
FF = 0000000C
FLG\$M_ALLCHR = 00000001
FLG\$M_BOL = 00000002
FLG\$M_CHKLPND = 00100000
FLG\$M_COMPEXPR = 00000004
FLG\$M_CONT = 00000008
FLG\$M_CRF = 40000000
FLG\$M_CRSEEN = 00000001
FLG\$M_DATRPT = 00000010
FLG\$M_DBGOUT = 00004000
FLG\$M_DLMSTR = 00008000
FLG\$M_ENDMCH = 00000020
FLG\$M_EVALEXPR = 00000040
FLG\$M_EXPOPT = 00000080
FLG\$M_EXTERR = 00010000
FLG\$M_EXTWRN = 00020000
FLG\$M_FIRSTLN = 00000200
FLG\$M_IFSTAT = 00800000
FLG\$M_IIF = 00400000
FLG\$M_INSERT = 00000100
FLG\$M_IRPC = 20000000
FLG\$M_LEXOP = 00000002
FLG\$M_LSTXST = 00000200
FLG\$M_MAC2COL = 00000800
FLG\$M_MACL = 00000800
FLG\$M_MACLTB = 08000000
FLG\$M_MACTXT = 00010000
FLG\$M_MEBLST = 00001000
FLG\$M_MOREARG = 00002000
FLG\$M_MOREINP = 00000008
FLG\$M_NEWPND = 00000400
FLG\$M_NOREF = 01000000
FLG\$M_NTTYPEPC = 00000020
FLG\$M_NULCHR = 00040000
FLG\$M_OBJXST = 00200000
FLG\$M_OPNDCHK = 00000100

FLG\$M_OPRND = 00002000
FLG\$M_OPTVFLIDX = 00001000
FLG\$M_ORDLST = 00020000
FLG\$M_P2 = 00004000
FLG\$M_RPTIRP = 10000000
FLG\$M_SEQFIL = 02000000
FLG\$M_SKAN = 00008000
FLG\$M_SPECOP = 00000004
FLG\$M_SPLALL = 04000000
FLG\$M_STOIMF = 00040000
FLG\$M_SYM2COL = 00000400
FLG\$M_TOCFLG = 00080000
FLG\$M_UPAFLG = 00000010
FLG\$M_UPDFIL = 00000080
FLG\$M_UPMARG = 00000040
FLG\$M_XCRF = 80000000
FLG\$V_ALLCHR = 00000000
FLG\$V_BOL = 00000001
FLG\$V_CHKLPND = 00000014
FLG\$V_COMPEXPR = 00000002
FLG\$V_CONT = 00000003
FLG\$V_CRF = 0000001E
FLG\$V_CRSEEN = 00000020
FLG\$V_DATRPT = 00000004
FLG\$V_DBGOUT = 0000002E
FLG\$V_DLMSTR = 0000002F
FLG\$V_ENDMCH = 00000005
FLG\$V_EVALEXPR = 00000006
FLG\$V_EXPOPT = 00000007
FLG\$V_EXTERR = 00000030
FLG\$V_EXTWRN = 00000031
FLG\$V_FIRSTLN = 00000029
FLG\$V_IFSTAT = 00000017
FLG\$V_IIF = 00000016
FLG\$V_INSERT = 00000008
FLG\$V_IRPC = 0000001D
FLG\$V_LEXOP = 00000021
FLG\$V_LSTXST = 00000009
FLG\$V_MAC2COL = 0000002B
FLG\$V_MACL = 0000000B
FLG\$V_MACLTB = 0000001B
FLG\$V_MACTXT = 00000010
FLG\$V_MEBLST = 0000000C
FLG\$V_MOREARG = 0000002D
FLG\$V_MOREINP = 00000023
FLG\$V_NEWPND = 0000000A
FLG\$V_NOREF = 00000018
FLG\$V_NTTYPEPC = 00000025
FLG\$V_NULCHR = 00000032
FLG\$V_OBJXST = 00000015
FLG\$V_OPNDCHK = 00000028
FLG\$V_OPRND = 0000000D
FLG\$V_OPTVFLIDX = 0000002C
FLG\$V_ORDLST = 00000011
FLG\$V_P2 = 0000000E
FLG\$V_RPTIRP = 0000001C
FLG\$V_SEQFIL = 00000019

FLG\$V_SKAN = 0000000F
FLG\$V_SPECOP = 00000022
FLG\$V_SPLALL = 0000001A
FLG\$V_STOIMF = 00000012
FLG\$V_SYM2COL = 0000002A
FLG\$V_TOCFLG = 00000013
FLG\$V_UPAFLG = 00000024
FLG\$V_UPDFIL = 00000027
FLG\$V_UPMARG = 00000026
FLG\$V_XCRF = 0000001F
FRAM_0 = 000000C3 R 01
HASHSZ = 0000007F
HYPHEN = 0000002D
INPSK_BUFSIZ = 000003E8
INTSK_BUFSIZ = 000013F4
INTSK_BUFWRN = 00001390
INT\$_ADD = 00000001
INT\$_AND = 00000002
INT\$_ASH = 00000003
INT\$_ASN = 0000000C
INT\$_AUGPC = 0000000D
INT\$_BDST = 0000000E
INT\$_CHKL = 0000000F
INT\$_DIV = 00000004
INT\$_END = 00000010
INT\$_EPT = 00000011
INT\$_ERR = 00000012
INT\$_ETX = 00000013
INT\$_FNEWL = 00000014
INT\$_ILG = 00000000
INT\$_INFO = 0000003A
INT\$_LGLAB = 00000015
INT\$_MACL = 00000016
INT\$_MUL = 00000005
INT\$_NEG = 00000006
INT\$_NEWL = 00000017
INT\$_NEWP = 00000018
INT\$_NOT = 00000007
INT\$_OP = 00000019
INT\$_OR = 00000008
INT\$_PRIL = 0000001A
INT\$_PRT = 0000001B
INT\$_PSECT = 0000001C
INT\$_REDEF = 0000001D
INT\$_REF = 0000001E
INT\$_REST = 0000001F
INT\$_SAME = 00000009
INT\$_SAVE = 00000020
INT\$_SBTTL = 00000021
INT\$_SETFLAG = 00000022
INT\$_SETLONG = 00000023
INT\$_SPIC = 00000024
INT\$_SPID = 00000025
INT\$_STIB = 00000026
INT\$_STIL = 00000028
INT\$_STIW = 00000027
INT\$_STKEPT = 00000029

MAC\$INTOUT
Symbol table

WRITE CODE TO INTERMEDIATE BUFFER E 2

16-SEP-1984 02:06:09 VAX/VMS Macro V04-00
5-SEP-1984 01:48:36 [MACRO.SRC]INTOUT.MAR;1

Page 10
(7)

INT\$-STKG = 0000002A
INT\$-STKL = 0000002B
INT\$-STKPC = 0000002C
INT\$-STKS = 0000002D
INT\$-STOB = 00000034
INT\$-STOL = 0000002E
INT\$-STOW = 00000035
INT\$-STRB = 0000002F
INT\$-STRL = 00000031
INT\$-STRSB = 00000032
INT\$-STRSW = 00000033
INT\$-STRW = 00000030
INT\$-STSB = 00000036
INT\$-STSW = 00000037
INT\$-SUB = 0000000A
INT\$-SUME = 00000039
INT\$-WRN = 00000038
INT\$-XOR = 0000000B
LIB\$GET VM ***** X 01
LST\$K_BUF\$IZ = 00000086
LST\$K_L_P_PAGE = 0000003C
LST\$K_TITLE\$IZ = 00000028
MAC\$ERR_NOMEM ***** X 01
MAC\$FIXFRAME 0000010B RG 01
MAC\$GL_INT\$CNT ***** X 01
MAC\$GL_INT\$FRMPT ***** X 01
MAC\$GL_INT\$PAGRQ ***** X 01
MAC\$GL_INT\$QUE ***** X 01
MAC\$GL_INT\$WRNPT ***** X 01
MAC\$INTERR_1_LW 00000014 RG 01
MAC\$INTERR_2_LW 0000000A RG 01
MAC\$INTERR_3_LW 00000000 RG 01
MAC\$INTOUT_1_LW 00000018 RG 01
MAC\$INTOUT_2_LW 0000000E RG 01
MAC\$INTOUT_3_LW 00000004 RG 01
MAC\$INTOUT_BY 00000067 RG 01
MAC\$INTOUT_N 000000A2 RG 01
MAC\$INTOUT_WD 00000048 RG 01
MAC\$INTOUT_X 00000086 RG 01
MAC\$OUTFRAME 000000BF RG 01
MAC\$SETFRAME 000000BB RG 01
MAC_SUBSYS = 0000007D
OBJ\$K_BUF\$IZ = 00000200
OPF\$M_LASTOPR = 00002000
OPF\$M_OPTEXP = 00001000
OPF\$V_LASTOPR = 0000000D
OPF\$V_OPTEXP = 0000000C
OUT_LW 0000001E R 01
RDX\$V_BINARY = 00000000
RDX\$V_DECIMAL = 00000002
RDX\$V_DOUBLE = 00000005
RDX\$V_FLOAT = 00000004
RDX\$V_GFLOAT = 00000006
RDX\$V_HEX = 00000003
RDX\$V_HFLOAT = 00000007
RDX\$V_OCTAL = 00000001
REG\$_PC = 0000000F

SEMI = 0000003B
STB\$K_PG_MISS = 0000000A
SYM\$K_MAXLEN = 0000001F
SYM\$K_TWOCOL = 00000010
TAB = 00000009
X1 = 00000033
X2 = 00080000

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 (0.)	00 (0.)	NOPIC USR
MAC\$RO_CODE_P1	00000120 (288.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	36	00:00:00.03	00:00:01.03
Command processing	126	00:00:00.33	00:00:02.51
Pass 1	142	00:00:01.59	00:00:06.12
Symbol table sort	0	00:00:00.20	00:00:01.55
Pass 2	63	00:00:00.52	00:00:02.42
Symbol table output	20	00:00:00.11	00:00:00.11
Psect synopsis output	1	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	390	00:00:02.80	00:00:13.76

The working set limit was 1050 pages.

16236 bytes (32 pages) of virtual memory were used to buffer the intermediate code.

There were 20 pages of symbol table space allocated to hold 235 non-local and 7 local symbols.

278 source lines were read in Pass 1, producing 12 object records in Pass 2.

0 pages of virtual memory were used to define 0 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[MACRO.OBJ]MACRO.MLB;1	3
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	0
TOTALS (all libraries)	3

205 GETS were required to define 3 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:INTOUT/OBJ=OBJ\$:INTOUT MSRC\$:INTOUT/UPDATE=(ENH\$:INTOUT)+LIB\$:MACRO/LIB

0226 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

