

DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLLLLLLLLLLLLLLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLLLLLLLLLLLLLLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLLLLLLLLLLLLLLL

DISALLOW
Table of contents

- Evaluate Disallow Expressions M 13

15-SEP-1984 23:44:21 VAX/VMS Macro V04-00

Page 0

(3)	68	Evaluate Disallow Expression
(4)	141	Dispatch to Disallow Expression Evaluator
(5)	174	Evaluate Path Expression
(7)	370	Evaluate Not Expression
(8)	395	Evaluate Any2 Expression
(9)	429	Evaluate And Expression
(10)	460	Evaluate Or Expression
(11)	491	Evaluate Xor Expression
(12)	526	Evaluate Neg Path Expression

```
0000 1 .TITLE DISALLOW - Evaluate Disallow Expressions
0000 2 .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6
0000 7 *
0000 8 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 9 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 10 * ALL RIGHTS RESERVED.
0000 11 *
0000 12 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 13 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 14 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 15 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 16 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 17 * TRANSFERRED.
0000 18 *
0000 19 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 20 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 21 * CORPORATION.
0000 22 *
0000 23 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 24 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 25 *
0000 26 *****
0000 27
0000 28 AUTHOR: Peter George 01-MAR-83
0000 29
0000 30 ABSTRACT: These routines are called to evaluate the disallow expression
0000 31 for a command that has just been successfully parsed.
0000 32
0000 33 MODIFICATIONS:
0000 34
0000 35 V03-006 HWS0098 Harold Schultz 02-Aug-1984
0000 36 Treat all qualifiers as global for purposes of the
0000 37 disallow expression. Disregard notion of a "local"
0000 38 qualifier for the present. (the question of local
0000 39 qualifiers will be addressed in a future version)
0000 40
0000 41 V03-005 HWS0089 Harold Schultz 22-Jul-1984
0000 42 Fix checking for presence of local qualifiers.
0000 43
0000 44 V03-004 PCG0004 Peter George 03-May-1984
0000 45 Allowing checking of keywords on negated qualifiers.
0000 46
0000 47 V03-003 PCG0003 Peter George 06-Dec-1983
0000 48 Add NEG operator.
0000 49
0000 50 V03-002 PCG0002 Peter George 11-May-1983
0000 51 Make path list search independent of token sorting.
0000 52
0000 53 V03-001 PCG0001 Peter George 30-Apr-1983
0000 54 Make the expression evaluation more robust.
0000 55 ---
0000 56
```


DISALLOW
V04-000

- Evaluate Disallow Expressions B 14

15-SEP-1984 23:44:21 VAX/VMS Macro V04-00
4-SEP-1984 23:40:19 [DCL.SRC]DISALLOW.MAR;1

Page 2
(2)

```
0000 58 :  
0000 59 : Macro library calls  
0000 60 :  
0000 61 : $$CLITABDEF ; Define table structures  
0000 62 : PRCDEF ; Define process work area  
0000 63 : WRKDEF ; Define command work area  
0000 64 : PTRDEF ; Define result parse descriptor  
0000 65 :  
00000000 66 .PSECT DCL$ZCODE,BYTE,RD,NOWRT
```

```
0000 68 .SBTTL Evaluate Disallow Expression
0000 69 :---
0000 70 :DCL$EVAL_DISALLOW - Evaluate disallow expression
0000 71 :
0000 72 :This routine is called after the command verb has been parsed, to detect
0000 73 :qualifier and parameter conflicts.
0000 74 :
0000 75 :INPUTS:
0000 76 :
0000 77 :R10 = Address of command work area (WRK)
0000 78 :
0000 79 :OUTPUTS:
0000 80 :
0000 81 :R0 = Value of disallow expression - true or false
0000 82 :
0000 83 :REGISTERS USED GLOBALLY IN EXPRESSION EVALUATION:
0000 84 :
0000 85 :R1 = Address of expression block
0000 86 :R7 = Address of ENT block associated with last
0000 87 :conflicting entity
0000 88 :R8 = Address of PTR block associated with last
0000 89 :conflicting entity
0000 90 :R9 = The number of the parameter providing the qualifier context
0000 91 :
0000 92 :+++
0000 93 :
0000 94 :DCL$EVAL_DISALLOW::
0000 95 :
0000 96 :
0000 97 :Check for conflicts on a per-parameter basis.
0000 98 :
0000 99 :CLRL R9 : Assume there are no parameters
0000 100 :TSTB WRK_B_PARM_SUM(R10) : Are there?
0000 101 :BEQL 10$ : No, then skip
0000 102 :INCL R9 : Yes, then start with first
59 CF AA 90 0000 103 :MOVB WRK_B_PARM_SUM(R10),R9 : Get total # of parameters
51 E6 AA D0 0004 104 10$: :MOVL WRK_L_DISALLOW(R10),R1 : Is there a disallow expression?
0000 105 :BEQL 95$ : No, then skip
0000 106 :BSBW DISPATCH : Yes, evaluate it
0000 107 :BLBS R0,80$ : If success, then signal error
0000 108 :INCL R9 : Set for next parameter
0000 109 :CMPB R9,WRK_B_PARM_SUM(R10) : Are there more parameters?
0000 110 :BLEQU 10$ : Yes, then continue
0000 111 95$: :STATUS NORMAL : Set normal return status
0000 112 :RSB : Return
0000 113 :
0000 114 :
0000 115 :Set up the conflict error message. Use actually command string if available.
0000 116 :Otherwise, use "NO" + the entity name.
0000 117 :
0000 118 80$: :TSTL R8 : Was entity explicitly present?
0000 119 :BEQL 85$ : No, then use entity block
0000 120 :EXTZV #PTR_V_VALUE,#PTR_S_VALUE,- : Get length of name
0000 121 :PTR_C_DESCR(R8),R6 :
0000 122 :EXTZV #PTR_V_OFFSET,#PTR_S_OFFSET,- : Get address of name
0000 123 :PTR_C_DESCR(R8),R7 :
0000 124 :MOVAB WRK_G_BUFFER(R10)[R7],R7 :
```


		23	11	002C	125	BRB	90\$	:		
				002E	126					
	58	57	D0	002E	127	85\$:	MOVL	R7,R8	: Get address of ENT block	
50	16	A8	32	0031	128		CVTWL	ENT W NAME(R8),R0	: Get offset to ascic name	
57	68	40	9E	0035	129		MOVAB	(R8)[R0],R7	: Get address of ascic name	
	56	87	9A	0039	130		MOVZBL	(R7)+,R6	: Get name length/address	
F492	CA	4F4E	8F	B0	003C	131	MOVW	#^A/NO/,WRK_G_BUFFER(R10)	: Preface the name with 'NO'	
F494	CA	67	56	28	0043	132	MCVC3	R6,(R7),WRK_G_BUFFER+2(R10)	: Move string into the buffer	
	57	F492	CA	9E	0049	133	MOVAB	WRK_G_BUFFER(R10),R7	: Set address of string	
	56	02	C0	004E	134		ADDL	#2,R6	: Add in length of 'NO'	
				0051	135					
	F48A	CA	57	D0	0051	136	90\$:	MOVL	R7,WRK_L_MARKPTR(R10)	: Set start of string
F486	CA	57	56	C1	0056	137	ADDL3	R6,R7,WRK_L_EXPANDPTR(R10)	: Set end of string	
50	00000000	'8F	D0	005C	138		MOVL	#CLIS_CONFLICT,R0	: Set conflicting entity status	
			05	0063	139		RSB		: Return	

```
0064 141 .SBTTL Dispatch to Disallow Expression Evaluator
0064 142 :---
0064 143 :DISPATCH - Dispatch to disallow expression evaluator
0064 144 :
0064 145 :This routine is called to invoke the appropriate routines for evaluating
0064 146 :a disallow expression.
0064 147 :
0064 148 :INPUTS:
0064 149 :
0064 150 :R1 = Address of expression block
0064 151 :R10 = Address of command work area (WRK)
0064 152 :
0064 153 :OUTPUTS:
0064 154 :
0064 155 :R0 = Value of disallow expression - true or false
0064 156 :
0064 157 :+++
0064 158 :
0064 159 DISPATCH:
0064 160
0064 161 CASEB EXP_B SUBTYPE(R1),- : Dispatch to action routine
0064 162 #EXP_R PATH,#EXP_K_NEG : Action routines
0064 163 10$: .WORD PATH-10$ :
0064 164 .WORD NOT-10$ :
0064 165 .WORD ANY2-10$ :
0064 166 .WORD AND-10$ :
0064 167 .WORD OR-10$ :
0064 168 .WORD XOR-10$ :
0064 169 .WORD NEG-10$ :
0064 170 CLRL R0 : Clear value if error
0064 171 RSB : Return
007A 172
```

03 A1 8F 0064 161
07 01 0067 162
0011' 0069 163
0105' 006B 164
0111' 006D 165
0139' 006F 166
015A' 0071 167
017A' 0073 168
01A5' 0075 169
50 D4 0077 170
05 0079 171
007A 172


```
007A 174 .SBTTL Evaluate Path Expression
007A 175 :---
007A 176 : PATH - Evaluate path expression
007A 177 :
007A 178 : This routine is called to evaluate a path expression.
007A 179 :
007A 180 : INPUTS:
007A 181 :
007A 182 :     R1 = Address of expression block
007A 183 :     R9 = Number of parameter to conduct local qualifier search at
007A 184 :     R10 = Address of command work area (WRK)
007A 185 :
007A 186 : OUTPUTS:
007A 187 :
007A 188 :     R0 = Value of expression - true or false
007A 189 :     R5,R6 are trashed
007A 190 :     R7 = Address of ENT block associated with last
007A 191 :           conflicting entity
007A 192 :     R8 = Address of PTR block associated with last
007A 193 :           conflicting entity
007A 194 :     R9 = The number of the parameter providing the qualifier context
007A 195 :
007A 196 :+++
007A 197 :
007A 198 :
007A 199 : Init state for the search.
007A 200 :
007A 201 PATH: MOVQ R2,-(SP) ; Save R2,R3
007D 202 : MOVL #1,R2 ; Init value level
0080 203 : MOVZWL EXP_W_TRO_COUNT(R1),R3 ; Get operand count
0084 204 : MOVAB WRK_G_RESULT-PTR_K_LENGTH(R10),R8 ; Get addr of PTR blocks
0089 205 : MOVAB EXP_L_OPERAND_LIST-4(R1),R5 ; Get addr of entity block l
008D 206 :
008D 207 :
008D 208 : If the path specifies a parameter, find the first parameter value that
008D 209 : we could possibly be a value of the specified parameter.
008D 210 :
008D 211 : ADDL3 WRK_L_TAB_VEC(R10),(R5)[R2],R7 ; Get entity block address
0093 212 : CMPB #ENT_R_QUALIFIER,ENT_B_SUBTYPE(R7) ; Is it a qualifier?
0097 213 : BEQL 7$ ; Yes, then process it
0099 214 : MOVZBL ENT_B_NUMBER(R7),R6 ; Get parameter number
009D 215 :
009D 216 5$: ADDL #PTR_K_LENGTH,R8 ; Get next PTR block
00A0 217 : CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; End of line?
00A4 218 : #PTR_K_ENDLINE
00A5 219 : BEQL 85$ ; Yes, return false
00A7 220 : CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; Parameter?
00AB 221 : #PTR_K_PARAMETR
00AC 222 : BNEQ 5$ ; No, skip it
00AE 223 : CMPB PTR_B_PARMCNT(R8),R6 ; Is is the one we want?
00B2 224 : BNEQ 5$ ; No, get next PTR
00B4 225 :
00B4 226 : We are processing and have found a parameter specification. If we are
00B4 227 : not looking for a specific keyword, return success now. If we are, then
00B4 228 : set up the index and ptr block starting address for the search.
00B4 229 :
00B4 230 : CMPL #1,R3 ; Only one entity address?
```

```

      45 13 00B7 231      BEQL 80$
      53 53 D7 00B9 232      DECL R3
55 04 C0 00BB 233      ADDL #4,R5
58 0C C2 00BE 234      SUBL #PTR_K_LENGTH,R8
      OF 11 00C1 235      BRB 10$
      00C3 236
      00C3 237
      00C3 238 :: Process qualifier.
      00C3 239
      4C 10 00C3 240 7$: BSBB QUAL
3F 50 E9 00C5 241      BLBC R0,95$
53 01 D1 00C8 242      CMPL #1,R3
      3E 13 00CB 243      BEQL 97$
      52 D6 00CD 244      INCL R2
56 08 D0 00CF 245      MOVL #8,R6
      00D2 246
      00D2 247
      00D2 248 :: Find the specified keyword entities.
      00D2 249
57 6542 DE AA C1 00D2 250 10$: ADDL3 WRK_L_TAB_VEC(R10),(R5)[R2],R7
      58 0C C0 00D8 251 20$: ADDL #PTR_R_LENGTH,R8
68 04 1C ED 00DB 252      CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),-
      04 00DF 253      #PTR_K_ENDLINE
      21 13 00E0 254      BEQL 85$
56 06 A8 91 00E2 255      CMPB PTR_B_PARMCNT(R8),R6
      1B 14 00E6 256      BGTR 85$
52 04 A8 91 00E8 257      CMPB PTR_B_LEVEL(R8),R2
      15 1F 00EC 258      BLSSU 85$
      E8 12 00EE 259      BNEQ 20$
57 08 A8 D1 00FC 260      CMPL PTR_L_ENTITY(R8),R7
      E2 12 00F4 261      BNEQ 20$
D8 52 53 F3 00F6 262      AOBLEQ R3,R2,10$
07 68 14 E0 00FA 263      BBS #PTR_V_NEGATE,(R8),90$
      00FE 264
      00FE 265
      00FE 266 :: Clean up and return
      00FE 267
      50 01 D0 00FE 268 80$: MOVL #1,R0
      04 11 0101 269      BRB 95$
      0103 270
      58 D4 0103 271 85$: CLRL R8
      50 D4 0105 272 90$: CLRL R0
      52 8E 7D 0107 273 95$: MOVQ (SP)+,R2
      05 010A 274      RSB
      010B 275
EF 68 14 E1 010B 276 97$: BBC #PTR_V_NEGATE,(R8),80$
      F2 11 010F 277      BRB 85$
      :: If not negated, return true
      :: Else, return false
      : Yes, return true
      : Put in synch with level
      : Shift the table address
      : Start with this token
      : Process keywords
      : Process qualifier
      : Return false if not found
      : Only one entity address?
      : Yes, return
      : Incr the value level
      : Set parameter count
      : Get entity block address
      : Get next PTR block
      : End of line?
      : Yes, return false
      : Right parameter count?
      : No, return false
      : Value depth too shallow?
      : Yes, return false
      : Skip if not exact
      : Is is the one we want?
      : No, get next PTR
      : Yes, get next entity
      : If negated, return false
      : Return true
      : Mark entity not found
      : Return false
      : Restore R2,R3
      : Return
      : If not negated, return true
      : Else, return false
```



```
0111 279 :++
0111 280 : QUAL - Find qualifier on command line.
0111 281 :
0111 282 : Outputs:
0111 283 :
0111 284 :     R0 = true/false status
0111 285 :     R8 = PTR block address
0111 286 : --
0111 287 :
0111 288 QUAL:
56 D4 0111 289 : CLRL R6 ; Assume qual not found
0113 290 : BBC #ENT_V_PARM,ENT_W_FLAGS(R7),50$ ; Skip local search if unnec
0113 291 :
0113 292 :
0113 293 : Find local parameter context.
0113 294 : (We are temporarily treating all qualifiers as global. Therefore, the following
0113 295 : checks all of the local qualifiers on the command line for presence)
0113 296 :
0113 297 :     TSTL R9 ; Are there any parameters?
0113 298 :     BEQL 50$ ; No, then skip
0113 299 :     MOVAB WRK_G_RESULT-PTR_K_LENGTH(R10),R8 ; Get addr of PTR blocks
0113 300 :     CLRL R0 ; Init parameter count
0113 301 : 40$: ADDL #PTR_K_LENGTH,R8 ; Get next PTR block
0113 302 :     CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; Parameter value?
0113 303 :     ; #PTR_K_PARAMETR
0113 304 :     BNEQ 40$ ; No, get next PTR
0113 305 :     INCL R0 ; Incr param count
0113 306 :     CMPL R0,R9 ; Are we there yet?
0113 307 :     BNEQ 40$ ; No, get next PTR
0113 308 :
0113 309 :
0113 310 : Check for local qualifiers.
0113 311 :
0113 312 : 45$: ADDL #PTR_K_LENGTH,R8 ; Get next PTR block
0113 313 :     CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; End of line or next
0113 314 :     ; #PTR_K_PARAMETR ; parameter value?
0113 315 :     BGEQU 50$ ; Yes, then done
0113 316 :     CMPL PTR_L_ENTITY(R8),R7 ; Is is the one we want?
0113 317 :     BNEQ 45$ ; No, get next PTR
0113 318 :     MOVL R8,R6 ; Yes, save PTR
0113 319 :     BRB 45$ ; Get next PTR
0113 320 :
0113 321 :
0113 322 :
0113 323 :
0113 324 : Check for global qualifiers.
0113 325 :
0113 326 : 50$: BBC #ENT_V_VERB,ENT_W_FLAGS(R7),40$ ; Skip global search if unne
0118 327 :     TSTL R6 ; Was qual found
0118 328 :     BNEQ 40$ ; Yes, then skip
0118 329 :     MOVAB WRK_G_RESULT-PTR_K_LENGTH(R10),R8 ; Get addr of PTR blocks
011D 330 : 55$: ADDL #PTR_K_LENGTH,R8 ; Get next PTR block
0120 331 :     CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; End of line?
0124 332 :     ; #PTR_K_ENDLINE
0125 333 :     BEQL 40$ ; Yes, then done
0127 334 :     CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; Command qualifier?
012B 335 :     ; #PTR_K_COMDQUAL
```

```
57 08 EF 12 012C 336 BNEQ 55$ ; No, get next PTR
    A8 D1 012E 337 CMPL PTR_L_ENTITY(R8),R7 ; Is is the one we want?
    E9 12 0132 338 BNEQ 55$ ; No, get next PTR
56 58 D0 0134 339 MOVL R8,R6 ; Yes, save PTR
    E4 11 0137 340 BRB 55$ ; Get next PTR
    0139 341 ;
    0139 342 ; Look at all the local qualifiers on the command line.
    0139 343 ; (Checked after global qualifiers to establish a defined order)
    0139 344 ;
25 04 A7 09 E1 0139 345 40$: BBC #ENT_V_PARM,ENT_W_FLAGS(R7),60$ ; Skip local search if unnec
    59 D5 013E 346 TSTL R9 ; Are there any parameters?
    21 13 0140 347 BEQL 60$ ; No, then skip
58 F9AA CA 9E 0142 348 MOVAB WRK_G_RESULT-PTR_K_LENGTH(R10),R8 ; Get addr of PTR blocks
    58 0C C0 0147 349 45$: ADDL #PTR_K_LENGTH,R8 ; Get next PTR block
68 04 1C ED 014A 350 CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; End of line?
    04 014E 351 #PTR_K_ENDLINE ;
    12 13 014F 352 BEQL 60$ ; Yes, then done
68 04 1C ED 0151 353 CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; Parameter qualifier?
    01 0155 354 #PTR_K_PARMQUAL ;
    EF 12 0156 355 BNEQ 45$ ; No, get next PTR
57 08 A8 D1 0158 356 CMPL PTR_L_ENTITY(R8),R7 ; Is is the one we want?
    E9 12 015C 357 BNEQ 45$ ; No, get next PTR
56 58 D0 015E 358 MOVL R8,R6 ; Yes, save PTR
    E4 11 0161 359 BRB 45$ ; Get next PTR
    0163 360 ;
    0163 361 ;
    0163 362 ; Set up registers and return.
    0163 363 ;
    50 D4 0163 364 60$: CLRL R0 ; Assume not found
58 56 D0 0165 365 MOVL R6,R8 ; Save found PTR address
    03 13 0168 366 BEQL 90$ ; Branch if none found
50 01 D0 016A 367 MOVL #1,R0 ; Set present
    05 016D 368 90$: RSB ; Return
```



```
016E 370      .SBTTL Evaluate Not Expression
016E 371      :---
016E 372      : NOT - Evaluate not expression
016E 373      :
016E 374      : This routine is called to evaluate a not expression.
016E 375      :
016E 376      : INPUTS:
016E 377      :
016E 378      :     R1 = Address of expression block
016E 379      :     R10 = Address of command work area (WRK)
016E 380      :
016E 381      : OUTPUTS:
016E 382      :
016E 383      :     R0 = Value of expression - true or false
016E 384      :
016E 385      : +++
016E 386      :
016E 387      : NOT:
51  DE AA C1 016E 388      ADDL3 WRK_L_TAB_VEC(R10),-      : Get subexpression address
    08 A1 016E 389      EXP_L_OPERAND_LIST(R1),R1      :
    FEED 30 0171 389      :
    50 D6 0174 390      BSBW DISPATCH      : Evaluate the subexpression
    05 0177 391      INCL R0      : Toggle the result
    0179 392      RSB      : Return
    017A 393
```

```
017A 395 .SBTTL Evaluate Any2 Expression
017A 396 :---
017A 397 : ANY2 - Evaluate any2 expression
017A 398 :
017A 399 : This routine is called to evaluate a any2 expression.
017A 400 :
017A 401 : INPUTS:
017A 402 :
017A 403 : R1 = Address of expression block
017A 404 : R10 = Address of command work area (WRK)
017A 405 :
017A 406 : OUTPUTS:
017A 407 :
017A 408 : R0 = Value of expression - true or false
017A 409 :
017A 410 :+++
017A 411 :
017A 412 ANY2:
017A 413 MOVQ R2,-(SP)
017A 414 CLRL -(SP)
017A 415 MOVL R1,R3
017A 416 MOVZWL EXP_W_TRO_COUNT(R3),R2
017A 417 10$: ADDL3 WRK_L_TAB_VEC(R10),-
017A 418 EXP_L_OPERAND_LIST-4(R3)[R2],R1
017A 419 BSBW DISPATCH
017A 420 BLBC R0,20$
017A 421 BBSS #0,(SP),30$
017A 422 20$: SOBGTR R2,10$
017A 423 CLRL R0
017A 424 30$: TSTL (SP)+
017A 425 MOVQ (SP)+,R2
017A 426 RSB
017A 427

7E 52 7D 017A 413 MOVQ R2,-(SP)
53 7E D4 017D 414 CLRL -(SP)
52 53 51 D0 017F 415 MOVL R1,R3
06 A3 3C 0182 416 MOVZWL EXP_W_TRO_COUNT(R3),R2
DE AA C1 0186 417 10$: ADDL3 WRK_L_TAB_VEC(R10),-
51 04 A342 0189 418 EXP_L_OPERAND_LIST-4(R3)[R2],R1
FED4 30 018D 419 BSBW DISPATCH
04 50 E9 0190 420 BLBC R0,20$
05 6E 00 E2 0193 421 BBSS #0,(SP),30$
EC 52 F5 0197 422 20$: SOBGTR R2,10$
50 D4 019A 423 CLRL R0
8E D5 019C 424 30$: TSTL (SP)+
52 8E 7D 019E 425 MOVQ (SP)+,R2
05 01A1 426 RSB
01A2 427

: Save R2,R3
: Assume false
: Copy expression block addr
: Get operand count
: Get subexpression address
:
: Evaluate the subexpression
: Branch if false
: Branch if second true
: Loop until done
: Return false
: Pop state
: Restore R2,R3
: Return
```



```
01A2 429 .SBTTL Evaluate And Expression
01A2 430 :---
01A2 431 :AND - Evaluate and expression
01A2 432 :
01A2 433 :This routine is called to evaluate a and expression.
01A2 434 :
01A2 435 :INPUTS:
01A2 436 :
01A2 437 :R1 = Address of expression block
01A2 438 :R10 = Address of command work area (WRK)
01A2 439 :
01A2 440 :OUTPUTS:
01A2 441 :
01A2 442 :R0 = Value of expression - true or false
01A2 443 :
01A2 444 :+++
01A2 445 :
01A2 446 AND:
01A2 447 MOVQ R2,-(SP) ; Save R2,R3
01A5 448 MOVL R1,R3 ; Copy expression block addr
01A8 449 MOVZWL EXP_W_TRO_COUNT(R3),R2 ; Get operand count
01AC 450 10$: ADDL3 WRK_L_TAB_VEC(R10),- ; Get subexpression address
01AF 451 EXP_L_OPERAND_LIST-4(R3)[R2],R1 ;
01B3 452 BSBW DISPATCH ; Evaluate the subexpression
01B6 453 BLBC R0,30$ ; Branch if false
01B9 454 SOBGTR R2,10$ ; Loop until done
01BC 455 MOVL #1,R0 ; Return true
01BF 456 30$: MOVQ (SP)+,R2 ; Restore R2,R3
01C2 457 RSB ; Return
01C3 458
```

7E 52 7D 01A2 447
53 51 D0 01A5 448
52 06 A3 3C 01A8 449
DE AA C1 01AC 450 10\$:
51 04 A342 01AF 451
FEAE 30 01B3 452
06 50 E9 01B6 453
F0 52 F5 01B9 454
50 01 D0 01BC 455
52 8E 7D 01BF 456 30\$:
05 01C2 457
01C3 458

```
01C3 460 .SBTTL Evaluate Or Expression
01C3 461 :---
01C3 462 :OR - Evaluate or expression
01C3 463 :
01C3 464 :This routine is called to evaluate a or expression.
01C3 465 :
01C3 466 :INPUTS:
01C3 467 :
01C3 468 :R1 = Address of expression block
01C3 469 :R10 = Address of command work area (WRK)
01C3 470 :
01C3 471 :OUTPUTS:
01C3 472 :
01C3 473 :R0 = Value of expression - true or false
01C3 474 :
01C3 475 :+++
01C3 476 :
01C3 477 OR:
01C3 478 MOVQ R2,-(SP) ; Save R2,R3
01C6 479 MOVL R1,R3 ; Copy expression block addr
01C9 480 MOVZWL EXP_W_TRO_COUNT(R1),R2 ; Get operand count
01CD 481 10$: ADDL3 WRK_L_TAB_VEC(R10),- ; Get subexpression address
01D0 482 EXP_L_OPERAND_LIST-4(R3)[R2],R1 ;
01D4 483 BSBW DISPATCH ; Evaluate the subexpression
01D7 484 BLBS R0,30$ ; Branch if false
01DA 485 SOBGTR R2,10$ ; Loop until done
01DD 486 CLRL R0 ; Return false
01DF 487 30$: MOVQ (SP)+,R2 ; Restore R2,R3
01E2 488 RSB ; Return
01E3 489
```

7E 52 7D
53 51 D0
52 06 A1 3C
DE AA C1
51 04 A342 30
FE8D E8
05 50 F5
F0 52 01DA
50 D4
52 8E 7D
7D 01DF
05 01E2
01E3


```
01E3 491 .SBTTL Evaluate Xor Expression
01E3 492 ---
01E3 493 XOR - Evaluate xor expression
01E3 494
01E3 495 This routine is called to evaluate a xor expression.
01E3 496
01E3 497 INPUTS:
01E3 498
01E3 499 R1 = Address of expression block
01E3 500 R10 = Address of command work area (WRK)
01E3 501
01E3 502 OUTPUTS:
01E3 503
01E3 504 R0 = Value of expression - true or false
01E3 505
01E3 506 :+++
01E3 507
01E3 508 XOR:
01E3 509 MOVQ R2,-(SP) ; Save R2,R3
01E3 510 CLRL -(SP) ; Assume false
01E3 511 MOVL R1,R3 ; Copy expression block addr
01E3 512 MOVZWL EXP_W_TRO_COUNT(R3),R2 ; Get operand count
01E3 513 10$: ADDL3 WRK-L-TAB-VEC(R10),- ; Get subexpression address
01E3 514 EXP-L-OPERAND_LIST-4(R3)[R2],R1
01E3 515 BSBW DISPATCH ; Evaluate the subexpression
01E3 516 BLBC R0,20$ ; Branch if false
01E3 517 CLRL R0 ; Set false
01E3 518 BBSS #0,(SP),30$ ; Branch if second true
01E3 519 20$: SOBGTR R2,10$ ; Loop until done
01E3 520 MOVL (SP),R0 ; Return value
01E3 521 30$: TSTL (SP)+ ; Pop state
01E3 522 MOVQ (SP)+,R2 ; Restore R2,R3
01E3 523 RSB ; Return
01E3 524
```

7E	52	7D	01E3	509
	7E	D4	01E6	510
53	51	D0	01E8	511
52	06 A3	3C	01EB	512
	DE AA	C1	01EF	513
51	04 A342		01F2	514
	FE6B	30	01F6	515
	06 50	E9	01F9	516
	50	D4	01FC	517
06	6E 00	E2	01FE	518
	EA 52	F5	0202	519
	50 6E	D0	0205	520
	8E	D5	0208	521
52	8E	7D	020A	522
		05	020D	523
			020E	524

```
020E 526 .SBTTL Evaluate Neg Path Expression
020E 527 :---
020E 528 : NEG - Evaluate neg path expression
020E 529 :
020E 530 : This routine is called to evaluate a neg path expression.
020E 531 :
020E 532 : INPUTS:
020E 533 :
020E 534 : R1 = Address of expression block
020E 535 : R9 = Number of parameter to conduct local qualifier search at
020E 536 : R10 = Address of command work area (WRK)
020E 537 :
020E 538 : OUTPUTS:
020E 539 :
020E 540 : R0 = Value of expression - true or false
020E 541 : R5,R6 are trashed
020E 542 : R7 = Address of ENT block associated with last
020E 543 : conflicting entity
020E 544 : R8 = Address of PTR block associated with last
020E 545 : conflicting entity
020E 546 : R9 = The number of the parameter providing the qualifier context
020E 547 :
020E 548 :+++
020E 549 :
020E 550 :
020E 551 : Init state for the search.
020E 552 :
020E 553 NEG: ADDL3 WRK_L_TAB_VEC(R10),- ; Get subexpression address
0211 554 EXP_L_OPERAND_LIST(R1),R1 ;
0214 555 MOVQ R2,=(SP) ; Save R2,R3
0217 556 MOVL #1,R2 ; Init value level
021A 557 MOVZWL EXP_W_TRO_COUNT(R1),R3 ; Get operand count
021E 558 MOVAB WRK_G_RESULT-PTR_K_LENGTH(R10),R8 ; Get addr of PTR blocks
0223 559 MOVAB EXP_L_OPERAND_LIST-4(R1),R5 ; Get addr of entity block l
0227 560 :
0227 561 :
0227 562 : If the path specifies a parameter, find the first parameter value that
0227 563 : could possibly be a value of the specified parameter.
0227 564 :
0227 565 ADDL3 WRK_L_TAB_VEC(R10),(R5)[R2],R7 ; Get entity block address
022D 566 CMPB #ENT_R_QUALIFIER,ENT_B_SUBTYPE(R7) ; Is it a qualifier?
0231 567 BEQL 7$ ; Yes, then process it
0233 568 MOVZBL ENT_B_NUMBER(R7),R6 ; Get parameter number
0237 569 :
0237 570 5$: ADDL #PTR_K_LENGTH,R8 ; Get next PTR block
023A 571 CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; End of line?
023E 572 #PTR_K_ENDLINE ;
023F 573 BEQL 85$ ; Yes, return false
0241 574 CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; Parameter?
0245 575 #PTR_K_PARAMETER ;
0246 576 BNEQ 5$ ; No, skip it
0248 577 CMPB PTR_B_PARMCNT(R8),R6 ; Is is the one we want?
024C 578 BNEQ 5$ ; No, get next PTR
024E 579 :
024E 580 :
024E 581 : We are processing and have found a parameter specification. If we are
024E 582 : not looking for a specific keyword, return failure now. If we are, then
```



```
024E 583 ; set up the index and ptr block starting address for the search.
024E 584 ;
53 01 D1 024E 585 CMPL #1,R3 ; Only one entity address?
4B 13 0251 586 BEQL 85$ ; Yes, return false
53 D7 0253 587 DECL R3 ; Put in synch with level
55 04 C0 0255 588 ADDL #4,R5 ; Shift the table address
58 0C C2 0258 589 SUBL #PTR_K_LENGTH,R8 ; Start with this token
10 11 025B 590 BRB 10$ ; Process keywords
025D 591
025D 592
025D 593 ; Process qualifier.
025D 594
025D 595 7$: BSBW QUAL ; Process qualifier
3F 50 E9 0260 596 BLBC R0,95$ ; Return false if not found
53 01 D1 0263 597 CMPL #1,R3 ; Only one entity address?
3E 13 0266 598 BEQL 97$ ; Yes, return
52 D6 0268 599 INCL R2 ; Incr the value level
56 08 D0 026A 600 MOVL #8,R6 ; Set parameter count
026D 601
026D 602
026D 603 ; Find the specified keyword entities.
026D 604
57 6542 DE AA C1 026D 605 10$: ADDL3 WRK_L_TAB_VEC(R10),(R5)[R2],R7 ; Get entity block address
58 0C C0 0273 606 20$: ADDL #PTR_K_LENGTH,R8 ; Get next PTR block
68 04 1C ED 0276 607 CMPZV #PTR_V_TYPE,#PTR_S_TYPE,(R8),- ; End of line?
04 027A 608 #PTR_K_ENDLINE
21 13 027B 609 BEQL 85$ ; Yes, return false
56 06 A8 91 027D 610 CMPB PTR_B_PARMCNT(R8),R6 ; Right parameter count?
1B 14 0281 611 BGTR 85$ ; No, return false
52 04 A8 91 0283 612 CMPB PTR_B_LEVEL(R8),R2 ; Value depth too shallow?
15 1F 0287 613 BLSSU 85$ ; Yes, return false
E8 12 0289 614 BNEQ 20$ ; Skip if not exact
57 08 A8 D1 028B 615 CMPL PTR_L_ENTITY(R8),R7 ; Is is the one we want?
E2 12 028F 616 BNEQ 20$ ; No, get next PTR
D8 52 53 F3 0291 617 AOBLEQ R3,R2,10$ ; Yes, get next entity
07 68 14 E1 0295 618 BBC #PTR_V_NEGATE,(R8),90$ ; If not negated, return fal
0299 619
0299 620 ; Clean up and return
0299 621
0299 622
50 01 D0 0299 623 80$: MOVL #1,R0 ; Return true
04 11 029C 624 BRB 95$
029E 625
58 D4 029E 626 85$: CLRL R8 ; Mark entity not found
50 D4 02A0 627 90$: CLRL R0 ; Return false
52 8E 7D 02A2 628 95$: MOVQ (SP)+,R2 ; Restore R2,R3
05 02A5 629 RSB ; Return
02A6 630
EF 68 14 E0 02A6 631 97$: BBS #PTR_V_NEGATE,(R8),80$ ; If negated, return true
F2 11 02AA 632 BRB 85$ ; Else, return false
02AC 633
02AC 634 .END
```


DISALLOW
Symbol table

- Evaluate Disallow Expressions

D 15

15-SEP-1984 23:44:21 VAX/VMS Macro V04-00
4-SEP-1984 23:40:19 [DCL.SRC]DISALLOW.MAR;1

Page 17
(12)

AND	000001A2	R	02	PRC_L_PPFLIST	00000070
ANY2	0000017A	R	02	PRC_L_RECALLPTR	0000012F
CLIS_CONFLICT	*****	X	02	PRC_L_RESTART	00000058
CLIS_NORMAL	*****	X	02	PRC_L_SAVAP	00000000
DCLSEVAL_DISALLOW	00000000	RG	02	PRC_L_SAVFP	00000004
DISPATCH	00000064	R	02	PRC_L_SEVERITY	00000050
ENT_B_NUMBER	= 00000014			PRC_L_SPWN	000000C0
ENT_B_SUBTYPE	= 00000003			PRC_L_STACKLM	000000A4
ENT_K_QUALIFIER	= 00000002			PRC_L_STACKPT	000000A0
ENT_V_PARM	= 00000009			PRC_L_STATUS	00000054
ENT_V_VERB	= 00000008			PRC_L_STS	00000084
ENT_W_FLAGS	= 00000004			PRC_L_STV	00000088
ENT_W_NAME	= 00000016			PRC_L_SYMBOL	00000060
EXP_B_SUBTYPE	= 00000003			PRC_L_TMBX	00000074
EXP_K_NEG	= 00000007			PRC_L_TRLIST	00000010
EXP_K_PATH	= 00000001			PRC_Q_ALLOCREG	00000020
EXP_L_OPERAND_LIST	= 00000008			PRC_Q_COMMAND	000000E0
EXP_W_TRO_COUNT	= 00000006			PRC_Q_FLUSHTIME	000000D0
NEG	0000020E	R	02	PRC_Q_GLOBAL	00000028
NOT	0000016E	R	02	PRC_Q_IMAGENAME	000000D8
OR	000001C3	R	02	PRC_Q_KEYPAD	00000040
PATH	0000007A	R	02	PRC_Q_LABEL	00000030
PRC_B_CONTINUE	000000F3			PRC_Q_LOCAL	00000038
PRC_B_DEFRADIX	000000AE			PRC_Q_SAVEPRIV	000000E8
PRC_B_EXMDEPMOD	000000AD			PRC_T_OUTDVI	0000011C
PRC_B_EXMDEPWID	000000AC			PRC_W_ASTIOSB	000000C6
PRC_B_EXONLYL	0000012D			PRC_W_ASTRETN	000000C8
PRC_B_FLAGS2	000000AF			PRC_W_ASTSTATUS	000000C4
PRC_B_IMGFLAG	00000078			PRC_W_ATTMBX	0000007A
PRC_B_OUTFLAGS	0000012C			PRC_W_FLAGS	00000068
PRC_B_PROMPTLEN	000000F0			PRC_W_INPCHAN	00000064
PRC_C_LENGTH	00000534			PRC_W_ONLEVEL	0000006A
PRC_G_COMMANDS	00000133			PRC_W_OUTIFI	00000114
PRC_G_PROMPT	000000F4			PRC_W_OUTISI	00000116
PRC_K_LENGTH	00000534			PRC_W_OUTMBXCHN	000000CA
PRC_L_CURRKEY	00000048			PRC_W_OUTMBXREF	000000CE
PRC_L_EXMDEPADR	000000A8			PRC_W_OUTMBXSIZ	000000CC
PRC_L_EXTARG	00000094			PRC_W_PMPCTCTL	000000F1
PRC_L_EXTBLK	0000008C			PRC_W_WAITIOSB	00000066
PRC_L_EXTCOD	0000009C			PTR_B_LEVEL	00000004
PRC_L_EXTHND	00000090			PTR_B_NUMBER	00000005
PRC_L_EXTPRM	00000098			PTR_B_PARMCNT	00000006
PRC_L_IDFLNK	000000BC			PTR_B_VALUE	00000000
PRC_L_IMGACTSTS	00000080			PTR_C_LENGTH	0000000C
PRC_L_INDCLOCK	0000007C			PTR_K_CMDQUAL	= 00000000
PRC_L_INDEPTH	0000005C			PTR_K_ENDLINE	= 00000004
PRC_L_INDFAB	0000001C			PTR_K_LENGTH	0000000C
PRC_L_INDINPRAB	00000014			PTR_K_PARAMETR	= 00000003
PRC_L_INDOUTRAB	00000018			PTR_K_PARMQUAL	= 00000001
PRC_L_INPRAB	00000008			PTR_L_DESCR	00000000
PRC_L_LASTKEY	0000004C			PTR_L_ENTITY	00000008
PRC_L_LSTSTATUS	000000B0			PTR_S_OFFSET	= 0000000C
PRC_L_ONCTLY	000000B8			PTR_S_TYPE	= 00000004
PRC_L_ONERROR	0000006C			PTR_S_VALUE	= 00000008
PRC_L_OUTOFBAND	000000B4			PTR_V_NEGATE	= 00000014
PRC_L_OUTRAB	0000000C			PTR_V_OFFSET	= 00000008
PRC_L_OUTRABCTX	00000118			PTR_V_TYPE	= 0000001C

DISALLOW
Symbol table

- Evaluate Disallow Expressions

E 15

15-SEP-1984 23:44:21
4-SEP-1984 23:40:19

VAX/VMS Macro V04-00
[DCL.SRC]DISALLOW.MAR;1

Page 18
(12)

```

PTR_V_VALUE      = 00000000
QUAC             = 00000111 R    02
WRK_B_CMDOPT     = FFFFFFFC3
WRK_B_MAXPARM    = FFFFFFFD0
WRK_B_MINPARM    = FFFFFFFD1
WRK_B_PARMCNT    = FFFFFFFCE
WRK_B_PARMSUM    = FFFFFFFCF
WRK_B_RECALLCNT  = FFFFFFFC5
WRK_B_VALLEV     = FFFFFFFC4
WRK_B_VERBTYP    = FFFFFFFC2
WRK_C_LENGTH     = FFFFF486
WRK_G_BUFFER     = FFFFF492
WRK_G_INPBUF     = FFFFF896
WRK_G_RESULT     = FFFFF9B6
WRK_K_LENGTH     = FFFFF486
WRK_L_CHARPTR    = FFFFF48E
WRK_L_DISALLOW   = FFFFFFFE6
WRK_L_ERRORRTN   = FFFFF9AE
WRK_L_EXPANDPTR  = FFFFF486
WRK_L_IMAGE      = FFFFFFFE2
WRK_L_MARKPTR    = FFFFF48A
WRK_L_PAROUT     = FFFFFFFD2
WRK_L_PMPTADDR   = FFFFF9A2
WRK_L_PROMPTRTN  = FFFFF9A6
WRK_L_PROPTR     = FFFFFFFC6
WRK_L_QUABLK     = FFFFFFFCA
WRK_L_READRTN    = FFFFF9AA
WRK_L_RECALLPTR  = FFFFFFFEA
WRK_L_RSLND      = FFFFFFFB6
WRK_L_RSLNXT     = FFFFFFFBA
WRK_L_SAVAP      = FFFFFFFF8
WRK_L_SAVFP      = FFFFFFFFC
WRK_L_SAVSP      = FFFFFFFF4
WRK_L_SIGNALRTN  = FFFFFFFD6
WRK_L_SPECRTN    = FFFFF9B2
WRK_L_TAB_VEC    = FFFFFFFDE
WRK_L_VERB       = FFFFFFFBE
WRK_W_FLAGS      = FFFFFFFF0
WRK_W_FLAGS2     = FFFFFFFF2
WRK_W_IMGCHAN    = FFFFFFFEE
WRK_W_PMPTLEN    = FFFFF99E
XOR              = 000001E3 R    02
  
```

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

PSECT name	Allocation	PSECT No.	Attributes
.ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	FFFFFFFC (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
DCL\$ZCODE	000002AC (684.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

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

Phase	Page faults	CPU Time	Elapsed Time
-----	-----	-----	-----
Initialization	16	00:00:00.04	00:00:01.86
Command processing	122	00:00:00.72	00:00:08.80
Pass 1	194	00:00:06.26	00:00:21.95
Symbol table sort	0	00:00:00.69	00:00:01.57
Pass 2	111	00:00:01.69	00:00:04.81
Symbol table output	19	00:00:00.13	00:00:00.53
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	463	00:00:09.55	00:00:39.54

The working set limit was 1050 pages.
32656 bytes (64 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 456 non-local and 40 local symbols.
634 source lines were read in Pass 1, producing 14 object records in Pass 2.
29 pages of virtual memory were used to define 15 macros.

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

Macro library name	Macros defined
-----	-----
-\$255\$DUA28:[SYSLIB]SYSBLDMLB.MLB;1	0
-\$255\$DUA28:[DCL.OBJ]DCL.MLB;1	6
-\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	9

558 GETS were required to define 9 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:DISALLOW/OBJ=OBJ\$:DISALLOW MSRC\$:DISALLOW/UPDATE=(ENH\$:DISALLOW)+EXECML\$/LIB+LIB\$:DCL/LIB+SYSS\$LIBRARY:SYSBLDMLB/LIB

0069 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

INTDEF
SOL

COMMAND
LIS

DESCRVAL
LIS

CLIMAC
MAR

CLIMSG
LIS

CONNECT
LIS

DCLPARSE
LIS

CHARMANIP
LIS

EXAMDEP
LIS

EXIT
LIS

INTIMAGES
MAR

DCXSTART
LIS

CLIGBL
LIS

DISALLOW
LIS

CLINT
LIS

CANCEL
LIS