

```

MMM          MMM          SSSSSSSSSSSS          GGGGGGGGGGGG          FFFFFFFFFFFFFFFF          IIIIIIIII          LLL
MMM          MMM          SSSSSSSSSSSS          GGGGGGGGGGGG          FFFFFFFFFFFFFFFF          IIIIIIIII          LLL
MMM          MMM          SSSSSSSSSSSS          GGGGGGGGGGGG          FFFFFFFFFFFFFFFF          IIIIIIIII          LLL
MMMMMM      MMMMM      SSS          GGG          FFF          III          LLL
MMMMMM      MMMMM      SSS          GGG          FFF          III          LLL
MMMMMM      MMMMM      SSS          GGG          FFF          III          LLL
MMM      MMM      MMM      SSS          GGG          FFF          III          LLL
MMM      MMM      MMM      SSS          GGG          FFF          III          LLL
MMM      MMM      MMM      SSS          GGG          FFF          III          LLL
MMM          MMM          SSSSSSSSSS          GGG          FFFFFFFFFFFFFFFF          III          LLL
MMM          MMM          SSSSSSSSSS          GGG          FFFFFFFFFFFFFFFF          III          LLL
MMM          MMM          SSSSSSSSSS          GGG          FFFFFFFFFFFFFFFF          III          LLL
MMM          MMM          SSS          GGG          GGGGGGGGGG          FFF          III          LLL
MMM          MMM          SSS          GGG          GGGGGGGGGG          FFF          III          LLL
MMM          MMM          SSS          GGG          GGGGGGGGGG          FFF          III          LLL
MMM          MMM          SSS          GGG          GGG          GGG          FFF          III          LLL
MMM          MMM          SSS          GGG          GGG          GGG          FFF          III          LLL
MMM          MMM          SSS          GGG          GGG          GGG          FFF          III          LLL
MMM          MMM          SSS          GGG          GGG          GGG          FFF          III          LLL
MMM          MMM          SSSSSSSSSSSS          GGGGGGGGGG          FFF          IIIIIIIII          LLLLLLLLLLLLLLLLLL
MMM          MMM          SSSSSSSSSSSS          GGGGGGGGGG          FFF          IIIIIIIII          LLLLLLLLLLLLLLLLLL
MMM          MMM          SSSSSSSSSSSS          GGGGGGGGGG          FFF          IIIIIIIII          LLLLLLLLLLLLLLLLLL

```

```

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
LLLLLLLLLLLLLL  IIIIII               SSSSSSSS
LLLLLLLLLLLLLL  IIIIII               SSSSSSSS

```

.....

```
1 0001 0 MODULE mdlgen ( IDENT = 'V04-000' ) =
2 0002 0
3 0003 1 BEGIN
4 0004 1
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: Message compiler
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This compiler translates message definition language
36 0036 1 into object modules. This module is called by the parser
37 0037 1 to generate an MDL file.
38 0038 1
39 0039 1 ENVIRONMENT:
40 0040 1
41 0041 1 VAX/VMS operating system, unprivileged user mode.
42 0042 1
43 0043 1 AUTHOR: Peter George, December 1980
44 0044 1
45 0045 1 Modified by:
46 0046 1
47 0047 1 V03-001 JWT0048 Jim Teague 12-Aug-1982
48 0048 1 Don't keep concatenating comments on conflicting literal
49 0049 1 definitions.
50 0050 1
51 0051 1 V02-003 PCG0003 Peter George 04-Mar-1981
52 0052 1 Alter CONCAT and DESC macro calls so that multiple
53 0053 1 UPLIT's are no longer generated.
54 0054 1
55 0055 1 V02-002 PCG0002 Peter George 02-Feb-1981
56 0056 1 Generate FACILITY constant definition to
57 0057 1 specify facility numbers. Also, change
```

MDLGEN
V04-000

D 3
16-Sep-1984 02:10:16
14-Sep-1984 12:46:14

VAX-11 Bliss-32 V4.0-742
[MSGF:L.SRC]MDLGEN.B32;1

Page 2
(1)

```

: 58      0058 1 | format of hexadecimal output.
: 59      0059 1 |
: 60      0060 1 | V02-001 PCG0001 Peter George 15-Jan-1981
: 61      0061 1 | Add processing for the /MACRO facility
: 62      0062 1 | qualifier.
: 63      0063 1 |
: 64      0064 1 | --
: 65      0065 1 |
: 66      0066 1 |
: 67      0067 1 |
: 68      0068 1 | Include files
: 69      0069 1 |
: 70      0070 1 |
: 71      0071 1 | LIBRARY 'SYSS$LIBRARY:STARLET';      ! VMS common definitions
: 72      0072 1 |
: 73      0073 1 | REQUIRE 'SRC$:MSG.REQ';      ! Message common definitions
: 74      0311 1 |
```

```
76 0312 1
77 0313 1
78 0314 1 Table of contents
79 0315 1
80 0316 1
81 0317 1 FORWARD ROUTINE
82 0318 1     mdl_start_struct,      ! Output $STRUCT command
83 0319 1     mdl_define_constant, ! Define message or literal constant
84 0320 1     mdl_end_struct,      ! Output E command
85 0321 1     mdl_comment,        ! Output a comment
86 0322 1     mdl_put_record;     ! Output a line to mdl file
87 0323 1
88 0324 1 Macros
89 0325 1
90 0326 1
91 0327 1 MACRO
92 0328 1
93 0329 1 ! Create a descriptor for a static string
94 0330 1
95 M 0331 1     DESC (string) =
96 0332 1         %CHARCOUNT (string), UPLIT BYTE (string)%,
97 0333 1
98 0334 1 ! Concatenate two or more strings
99 0335 1
100 M 0336 1     CONCAT (result, source) [] =
101 M 0337 1         CH$MOVE (.source, .(source+4), .result[0] + .result[1]);
102 M 0338 1         result[0] = .result[0] + .source;
103 0339 1         concat (result, %REMAINING)%;
104 0340 1
105 0341 1
106 0342 1 Literals
107 0343 1
108 0344 1
109 0345 1 LITERAL
110 0346 1     system_bit = 1,          ! /SYSTEM facility qualifier indicator
111 0347 1     prefix_bit = 2,         ! /PREFIX facility qualifier indicator
112 0348 1     macro_bit = 3,          ! /MACRO facility qualifier indicator
113 0349 1     dectohex_switch = %X'FFFF', ! Value to switch from decimal to hex at
114 0350 1     record_offset = 8,      ! Column where record starts
115 0351 1     comment_offset = 5,      ! Number of spaces to insert when appending a comment
116 0352 1     symbol_size = obj$c_symsiz, ! Maximum symbol name size
117 0353 1     line_size = 132;        ! Length of output record buffer
118 0354 1
119 0355 1
120 0356 1 Storage definitions
121 0357 1
122 0358 1
123 0359 1 OWN
124 0360 1     value_offset,
125 0361 1     struct_init: INITIAL (false), ! Column where values starts
126 0362 1     prefix: VECTOR [line_size, BYTE], ! Structure initialized flag
127 0363 1     prefix_desc: VECTOR [2] INITIAL (0, prefix), ! Prefix buffer
128 0364 1     output_buffer: VECTOR [line_size, BYTE], ! Symbol name prefix descriptor
129 0365 1     output_desc: VECTOR [2] INITIAL (0, output_buffer); ! Output buffer
130 0366 1 ! Descriptor of output buffer
131 0367 1 OWN
132 0368 1     bracket: VECTOR [2] ! '>' descriptor
```

```
133 0369 1 INITIAL (DESC ('>')),
134 0370 1 cbracket: VECTOR [2] ! 'C <' descriptor
135 0371 1 INITIAL (DESC ('C <')),
136 0372 1 comma: VECTOR [2] ! ',' descriptor
137 0373 1 INITIAL (DESC (',')),
138 0374 1 e: VECTOR [2] ! 'E' descriptor
139 0375 1 INITIAL (DESC ('E')),
140 0376 1 facility: VECTOR [2] ! 'FACILITY,' descriptor
141 0377 1 INITIAL (DESC ('FACILITY,')),
142 0378 1 semicolon: VECTOR [2] ! ';' descriptor
143 0379 1 INITIAL (DESC (';')),
144 0380 1 struct: VECTOR [2] ! '$STRUCT ' descriptor
145 0381 1 INITIAL (DESC ('$STRUCT ')),
146 0382 1 ul: VECTOR [2] ! '!UL' descriptor
147 0383 1 INITIAL (DESC ('!UL')),
148 0384 1 xl: VECTOR [2] ! '<^X!XL>' descriptor
149 0385 1 INITIAL (DESC (%STRING ('<^X', '!XL', '>')));
150 0386 1
151 0387 1 !
152 0388 1 ! External storage
153 0389 1 !
154 0390 1
155 0391 1 EXTERNAL
156 0392 1 md1_rab: BBLOCK, ! Output file RAB
157 0393 1 md1_fab: BBLOCK; ! Output file FAB
158 0394 1
159 0395 1 !
160 0396 1 ! External routines
161 0397 1 !
162 0398 1
163 0399 1 EXTERNAL ROUTINE
164 0400 1 syntax_error, ! Signal syntax error
165 0401 1 rms_error; ! Signal RMS-type error
166 0402 1
```

```
168 0403 1 GLOBAL ROUTINE mdl_start_struc (facility_desc, facility_number, macro_desc, facility_flags) =
169 0404 1
170 0405 1 --
171 0406 1
172 0407 1 This routine outputs the following two lines:
173 0408 1
174 0409 1 $STRUCT facility_name,macro_suffix
175 0410 1 C <constant_prefix,tag_string,
176 0411 1 FACILITY,facility_number
177 0412 1
178 0413 1 Where, 1) facility_name is defined by either /MACRO, /PREFIX,
179 0414 1 or the default facility name, depending on which is present;
180 0415 1 2) macro_suffix is defined only when /MACRO is present and is
181 0416 1 then the last three characters of the macro name; 3) constant_prefix
182 0417 1 is defined by the /PREFIX qualifier or the default facility
183 0418 1 name if the qualifier is not present; 4) tag_string is either
184 0419 1 '$' or '.' depending on whether the /SYSTEM qualifier is
185 0420 1 present or not; and 5) facility_number is the facility number
186 0421 1 specified.
187 0422 1
188 0423 1 Inputs:
189 0424 1
190 0425 1 facility_desc = Address of descriptor of facility name.
191 0426 1 /PREFIX qualifier is used if present.
192 0427 1
193 0428 1 facility_number = Facility number passed by value.
194 0429 1
195 0430 1 macro_desc = Address of descriptor of string specified by the
196 0431 1 /MACRO qualifier.
197 0432 1
198 0433 1 facility_flags = Bitvector indicating which facility
199 0434 1 qualifiers are present.
200 0435 1
201 0436 1 Outputs:
202 0437 1
203 0438 1 None
204 0439 1
205 0440 1 --
206 0441 1
207 0442 2 BEGIN
208 0443 2
209 0444 2 MAP
210 0445 2 facility_desc: REF VECTOR,
211 0446 2 macro_desc: REF VECTOR,
212 0447 2 facility_flags: REF BITVECTOR;
213 0448 2
214 0449 2 LOCAL
215 0450 2 buffer: VECTOR [line_size, BYTE],
216 0451 2 buffer_desc: VECTOR [2],
217 0452 2 constant_desc: VECTOR [2],
218 0453 2 faoout_buf: VECTOR [line_size, BYTE],
219 0454 2 faoout_desc: VECTOR [2],
220 0455 2 struc_desc: VECTOR [2],
221 0456 2 suffix_desc: VECTOR [2],
222 0457 2 tag_desc: VECTOR [2],
223 0458 2 tag: VECTOR [line_size, BYTE];
224 0459 2
```

Line to be output
Buffer descriptor
Facility name descriptor for constant line
FAO output buffer
Descriptor to accept output from FAO
Structure name descriptor
Macro name suffix descriptor
Tag field descriptor
Tag built when /PREFIX is specified

```
225 0460 2 struc_init = true; ! Set structure active flag
226 0461 2 constant_desc[0] = .facility_desc[0]; ! Initialize structure name
227 0462 2 constant_desc[1] = .facility_desc[1]; ! descriptor from parameter
228 0463 2
229 0464 2 IF .facility_flags [prefix_bit] ! /PREFIX specified
230 0465 2 THEN BEGIN
231 0466 2
232 0467 2 prefix_desc[0] = .constant_desc[0]; ! Pickup prefix as literally specified
233 0468 2 prefix_desc[1] = .constant_desc[1];
234 0469 2
235 0470 2 tag_desc[0] = 2; ! Remove last character or two of
236 0471 2 tag_desc[1] = tag; ! prefix as tag
237 0472 2
238 0473 2 constant_desc[0] = .constant_desc[0] - 2; ! Remove tag from structure name
239 0474 2
240 0475 2 CH$MOVE (2, .constant_desc[0] + .constant_desc[1], .tag_desc[1]);
241 0476 2
242 0477 4 IF ( NOT .facility_flags [system_bit] ) ! /SYSTEM not specified
243 0478 4 THEN BEGIN
244 0479 4 tag_desc[0] = .tag_desc[0] - 1; ! Remove '$' from tag
245 0480 4 tag_desc[1] = .tag_desc[1] + 1;
246 0481 4 constant_desc[0] = .constant_desc[0] + 1;
247 0482 4 END;
248 0483 2
249 0484 2 END
250 0485 2
251 0486 2 ELSE BEGIN ! /PREFIX not specified
252 0487 2
253 0488 2 tag_desc[0] = 2; ! Initialize tag field
254 0489 2 tag_desc[1] = UPLIT ('$_'); ! Assuming /SYSTEM specified
255 0490 2
256 0491 4 IF ( NOT .facility_flags [system_bit] ) ! /SYSTEM not specified
257 0492 4 THEN BEGIN
258 0493 4 tag_desc[0] = .tag_desc[0] - 1; ! Remove '$' from tag
259 0494 4 tag_desc[1] = .tag_desc[1] + 1;
260 0495 4 END;
261 0496 2
262 0497 2 prefix_desc[0] = 0; ! Clear prefix length
263 0498 2 prefix_desc[1] = prefix; ! Set up prefix buffer
264 0499 2
265 P 0500 2 CONCAT (prefix_desc, ! Put together symbol name prefix
266 P 0501 2 constant_desc,
267 0502 2 tag_desc);
268 0503 2
269 0504 2 END;
270 0505 2
271 0506 2 value_offset = record_offset + symbol_size - .prefix_desc[0]; ! Initialize symbol value offset
272 0507 2
273 0508 2 mdl_put_record ( UPLIT (DESC (' ')) ); ! Output a blank line
274 0509 2
275 0510 2 buffer_desc[0] = record_offset; ! Initialize buffer descriptor
276 0511 2 buffer_desc[1] = buffer;
277 0512 2 CH$FILC (' ', record_offset, buffer); ! Clear beginning of line
278 0513 2
279 0514 2 IF .facility_flags [macro_bit] ! /MACRO qualifier present
280 0515 2 THEN BEGIN
281 0516 2
```

```
282      struc_desc[0] = .macro_desc[0] - 4;          ! Remove $ and suffix
283      struc_desc[1] = .macro_desc[1] + 1;
284
285      suffix_desc[0] = 3;          ! Initialize suffix
286      suffix_desc[1] = .struc_desc[1] + .struc_desc[0];
287
288      P 0523      CONCAT (buffer_desc,          ! Put together $STRUCT line
289      P 0524      struct,
290      P 0525      struc_desc,
291      P 0526      comma,
292      P 0527      suffix_desc);
293
294      0528      END
295
296      0531      ELSE BEGIN
297
298      0533      struc_desc[0] = .constant_desc[0];    ! Use default structure name
299      0534      struc_desc[1] = .constant_desc[1];
300
301      P 0536      CONCAT (buffer_desc,          ! Put together $STRUCT line
302      P 0537      struct,
303      P 0538      struc_desc);
304
305      0539      END;
306
307      0542      mdl_put_record (buffer_desc);          ! Output $STRUCT line
308      0543      mdl_put_record ( UPLIT (DESC (' ')) ); ! Output blank line
309
310      0545      buffer_desc[0] = record_offset;        ! Reset buffer size
311
312      P 0547      CONCAT (buffer_desc,          ! Put together constant header
313      P 0548      cbracket,
314      P 0549      constant_desc,
315      P 0550      comma,
316      P 0551      tag_desc,
317      P 0552      comma);
318
319      0554      mdl_put_record (buffer_desc);          ! Output constant header
320
321      0556      buffer_desc[0] = record_offset;        ! Initialize buffer descriptor
322      0557      CH$FIL (' ', .value_offset, buffer);   ! Clear beginning of line
323
324      P 0559      CONCAT (buffer_desc,          ! Put together FACILITY constant definition
325      P 0560      facility);
326
327      0562      buffer_desc[0] = .value_offset;        ! Move buffer ptr to column for values
328
329      P 0564      CONCAT (buffer_desc,          ! Use decimal notation
330      P 0565      ul);
331
332      0567      faoout_desc[0] = line_size;            ! Set maximum buffer size
333      0568      faoout_desc[1] = faoout_buf;          ! Set FAO output buffer
334
335      P 0570      $FAO (buffer_desc, faoout_desc[0],   ! Convert numeric value to string descriptor
336      P 0571      faoout_desc, .facility_number);
337
338      0573      mdl_put_record (faoout_desc);          ! Output constant statement
```

```

: 339          0574 2
: 340          0575 2 RETURN true;
: 341          0576 2
: 342          0577 1 END;

```

										TITLE	MDLGEN	
										IDENT	\V04-000\	
										PSECT	\$SPLIT\$,NOWRT,NOEXE,2	
							3E	00000	P.AAA:	.ASCII	\>\	
					3C	20	43	00001	P.AAB:	.ASCII	\C <\	
							2C	00004	P.AAC:	.ASCII	\,\	
							45	00005	P.AAD:	.ASCII	\E\	
2C	59	54	49	4C	49	43	41	46	00006	P.AAE:	.ASCII	\FACILITY,\
								3B	0000F	P.AAF:	.ASCII	\;\
	20	54	43	55	52	54	53	24	00010	P.AAG:	.ASCII	\\$STRUCT \
						4C	55	21	00018	P.AAH:	.ASCII	\!UL\
		3E	4C	58	21	58	5E	3C	0001B	P.AAI:	.ASCII	\<^X!XL>\
									00022		.BLKB	2
					00	00	5F	24	00024	P.AAJ:	.ASCII	\\$, \<0><0>
								20	00028	P.AAL:	.ASCII	\
									00029		.BLKB	3
									0002C	P.AAK:	.LONG	1
									00030		.ADDRESS	P.AAL
								20	00034	P.AAN:	.ASCII	\ \
									00035		.BLKB	3
									00038	P.AAM:	.LONG	1
									0003C		.ADDRESS	P.AAN
										PSECT	\$OWN\$,NOEXE,2	
									00000	VALUE_OFFSET:		
										.BLKB	4	
									00004	STRUC_INIT:		
										.LONG	0	
									00008	PREFIX:	.BLKB	132
									00008C	PREFIX_DESC:		
										.LONG	0	
									000090		.ADDRESS	PREFIX
									00094	OUTPUT_BUFFER:		
										.BLKB	132	
									00118	OUTPUT_DESC:		
										.LONG	0	
									0011C		.ADDRESS	OUTPUT_BUFFER
									00120	BRACKET:	.LONG	1
									00124		.ADDRESS	P.AAA
									00128	CBRACKET:		
										.LONG	3	
									0012C		.ADDRESS	P.AAB
									00130	COMMA:	.LONG	1
									00134		.ADDRESS	P.AAC
									00138	E:	.LONG	1
									0013C		.ADDRESS	P.AAD
									00140	FACILITY:		
										.LONG	9	

Si
RU
EL
Li
Le
Me
Co

```
00000000' 00144 .ADDRESS P.AAE
00000001' 00148 SEMICOLON:
00000000' 0014C .LONG 1
00000008' 00150 STRUCT: .ADDRESS P.AAF
00000000' 00154 .LONG 8
00000003' 00154 .ADDRESS P.AAG
00000003' 00158 UL: .LONG 3
00000000' 0015C .ADDRESS P.AAH
00000007' 00160 XL: .LONG 7
00000000' 00164 .ADDRESS P.AAI
```

```
.EXTRN MDL_RAB, MDL_FAB
.EXTRN SYNTAX_ERROR, RMS_ERROR
.EXTRN SYS$FAD
```

```
.PSECT $CODE$,NOWRT,2
```

```
03FC 00000
```

```
.ENTRY MDL_START_STRUC, Save R2,R3,R4,R5,R6,R7,R8,-, 0403
R9
```

	59	0000V	CF	9E	00002	MOVAB	MDL_PUT_RECORD, R9		
	58	0000'	CF	9E	00007	MOVAB	PREFIX_DESC, R8		
	5E	FE44	CE	9E	0000C	MOVAB	-444(SP), SP		
	FF78	C8	01	D0	00011	MOVL	#1, STRUC_INIT	0460	
		50	04	AC	D0	00016	MOVL	FACILITY_DESC, R0	0461
	FF6C	CD	60	7D	0001A	MOVQ	(R0), CONSTANT_DESC		
		56	FF6C	CD	D0	0001F	MOVL	CONSTANT_DESC, R6	0467
	0084	CE	02	D0	00024	MOVL	#2, TAG_DESC	0470	
33	10	BC	02	E1	00029	BBC	#2, @FACILITY_FLAGS, 1\$	0464	
		68	56	D0	0002E	MOVL	R6, PREFIX_DESC	0467	
	04	A8	FF70	CD	D0	00031	MOVL	CONSTANT_DESC+4, PREFIX_DESC+4	0468
	0088	CE	6E	9E	00037	MOVAB	TAG, TAG_DESC+4	0471	
	FF6C	CD	02	C2	0003C	SUBL2	#2, CONSTANT_DESC	0473	
50	FF6C	CD	FF70	CD	C1	00041	ADDL3	CONSTANT_DESC+4, CONSTANT_DESC, R0	0475
	0088	DE	60	B0	00049	MOVW	(R0), @TAG_DESC+4		
4A	10	BC	01	E0	0004E	BBS	#1, @FACILITY_FLAGS, 3\$	0477	
		0084	CE	D7	00053	DECL	TAG_DESC	0479	
		0088	CE	D6	00057	INCL	TAG_DESC+4	0480	
		FF6C	CD	D6	0005B	INCL	CONSTANT_DESC	0481	
			3C	11	0005F	BRB	3\$	0464	
	0088	CE	0000'	CF	9E	00061	MOVAB	P.AAJ, TAG_DESC+4	0489
08	10	BC	01	E0	00068	BBS	#1, @FACILITY_FLAGS, 2\$	0491	
		0084	CE	D7	0006D	DECL	TAG_DESC	0493	
		0088	CE	D6	00071	INCL	TAG_DESC+4	0494	
			68	D4	00075	CLRL	PREFIX_DESC	0497	
	04	A8	FF7C	C8	9E	00077	MOVAB	PREFIX, PREFIX_DESC+4	0498
50		68	04	A8	C1	0007D	ADDL3	PREFIX_DESC+4, -PREFIX_DESC, R0	0502
60	FF70	DD		56	28	00082	MOVC3	R6, @CONSTANT_DESC+4, -(R0)	
		68		56	C0	00088	ADDL2	R6, PREFIX_DESC	
50		68	04	A8	C1	0008B	ADDL3	PREFIX_DESC+4, PREFIX_DESC, R0	
60	0088	DE	0084	CE	28	00090	MOVC3	TAG_DESC, @TAG_DESC+4, (R0)	
		68	0084	CE	C0	00098	ADDL2	TAG_DESC, PREFIX_DESC	
FF74	C8	27		68	C3	0009D	SUBL3	PREFIX_DESC, #39, VALUE_OFFSET	0506
			0000'	CF	9F	000A3	PUSHAB	P.AAK	0508
		69		01	FB	000A7	CALLS	#1, MDL_PUT_RECORD	
	FF74	CD		08	D0	000AA	MOVL	#8, BUFFER_DESC	0510
	FF78	CD	FF7C	CD	9E	000AF	MOVAB	BUFFER, BUFFER_DESC+4	0511
08	20	6E		00	2C	000B6	MOVC5	#0, (SP), #32, #8, BUFFER	0512

		57	FF7C	CD	DO	000BB	MOVL	STRUCT, R7	0527
		00C4	FF78	C8	C1	000BE	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R1	0514
	51	FF74		BC	E1	000CB	BBC	#3, @FACILITY_FLAGS, Z\$	0517
	71	10		50	OC	AC	MOVL	MACRO_DESC, R0	
0094	CE			60		04	SUBL3	#4, (R0), STRUC_DESC	0518
0098	CE	04		A0	01	C1	ADDL3	#1, 4(R0), STRUC_DESC+4	0520
		008C		CE	03	D0	MOVL	#3, SUFFIX_DESC	0521
			0094	56	CE	D0	MOVL	STRUC_DESC, R6	
		0090	0098	DE	46	9E	MOVAB	@STRUC_DESC+4[R6], SUFFIX_DESC+4	0527
61		00C8		D8	57	28	MOVC3	R7, @STRUCT+4, (R1)	
		FF74		CD	57	C0	ADDL2	R7, BUFFER_DESC	
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	
60		0098		DE	56	28	MOVC3	R6, @STRUC_DESC+4, (R0)	
		FF74		CD	56	C0	ADDL2	R6, BUFFER_DESC	
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	
60		00A8		D8	00A4	C8	MOVC3	COMMA, @COMMA+4, (R0)	
		FF74		CD	00A4	C8	ADDL2	COMMA, BUFFER_DESC	
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	
60		0090		DE	008C	CE	MOVC3	SUFFIX_DESC, @SUFFIX_DESC+4, (R0)	
		FF74		CD	008C	CE	ADDL2	SUFFIX_DESC, BUFFER_DESC	
					29	11	BRB	5\$	0514
		0094		CE	FF6C	CD	MOVQ	CONSTANT_DESC, STRUC_DESC	0533
61		00C8		D8	57	28	MOVC3	R7, @STRUCT+4, (R1)	0538
		FF74		CD	57	C0	ADDL2	R7, BUFFER_DESC	
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	
60		0098		DE	0094	CE	MOVC3	STRUC_DESC, @STRUC_DESC+4, (R0)	
		FF74		CD	0094	CE	ADDL2	STRUC_DESC, BUFFER_DESC	
					FF74	CD	PUSHAB	BUFFER_DESC	0542
				69	01	FB	CALLS	#1, MDC_PUT_RECORD	
					CF	9F	PUSHAB	P.AAM	0543
				69	01	FB	CALLS	#1, MDL_PUT_RECORD	
		FF74		CD	08	D0	MOVL	#8, BUFFER_DESC	0545
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	0552
60		00A0		D8	009C	C8	MOVC3	CBRACKET, @CBRACKET+4, (R0)	
		FF74		CD	009C	C8	ADDL2	CBRACKET, BUFFER_DESC	
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	
60		FF70		DD	FF6C	CD	MOVC3	CONSTANT_DESC, @CONSTANT_DESC+4, (R0)	
		FF74		CD	FF6C	CD	ADDL2	CONSTANT_DESC, BUFFER_DESC	
				56	00A4	C8	MOVL	COMMA, R6	
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	
60		00A8		D8	56	28	MOVC3	R6, @COMMA+4, (R0)	
		FF74		CD	56	C0	ADDL2	R6, BUFFER_DESC	
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	
60		0088		DE	0084	CE	MOVC3	TAG_DESC, @TAG_DESC+4, (R0)	
		FF74		CD	0084	CE	ADDL2	TAG_DESC, BUFFER_DESC	
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	
60		00A8		D8	56	28	MOVC3	R6, @COMMA+4, (R0)	
		FF74		CD	56	C0	ADDL2	R6, BUFFER_DESC	
					FF74	CD	PUSHAB	BUFFER_DESC	0554
				69	01	FB	CALLS	#1, MDC_PUT_RECORD	
		FF74		CD	08	D0	MOVL	#8, BUFFER_DESC	0556
20				6E	00	2C	MOVC5	#0, (SP), #32, VALUE_OFFSET, BUFFER	0557
					FF7C	CD			
50		FF74		CD	FF78	C1	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	0560
60		00B8		D8	00B4	C8	MOVC3	FACILITY, @FACILITY+4, (R0)	
		FF74		CD	00B4	C8	ADDL2	FACILITY, BUFFER_DESC	

MDLGEN
V04-000

M 3
16-Sep-1984 02:10:16
14-Sep-1984 12:46:14

VAX-11 Bliss-32 V4.0-742
[MSGFIL.SRC]MDLGEN.B32;1

Page 11
(3)

MESS

50	FF74	CD	FF74	C8	D0	0021A	MOVL	VALUE OFFSET, BUFFER_DESC	:	0562
60	FF74	CD	FF78	CD	C1	00221	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	:	0565
	00D0	D8	00CC	C8	28	00229	MOVCL	UL, @UC+4, (R0)	:	
	FF74	CD	00CC	C8	C0	00231	ADDL2	UL, BUFFER_DESC	:	
	009C	CE	84	8F	9A	00238	MOVZBL	#132, FAOOUT_DESC	:	0567
	00A0	CE	00A4	CE	9E	0023E	MOVAB	FAOOUT_BUF, FAOOUT_DESC+4	:	0568
			08	AC	DD	00245	PUSHL	FACILITY_NUMBER	:	0571
			00A0	CE	9F	00248	PUSHAB	FAOOUT_DESC	:	
			00A4	CE	9F	0024C	PUSHAB	FAOOUT_DESC	:	
			FF74	CD	9F	00250	PUSHAB	BUFFER_DESC	:	
00000000G	00			04	FB	00254	CALLS	#4, SYSSFAO	:	
	69		009C	CE	9F	0025B	PUSHAB	FAOOUT_DESC	:	0573
	50			01	FB	0025F	CALLS	#1, MDC_PUT_RECORD	:	
				01	D0	00262	MOVL	#1, R0	:	0575
				04	00	00265	RET		:	0577

; Routine Size: 614 bytes, Routine Base: \$CODE\$ + 0000

Ther
MESS

```
344 0578 1 GLOBAL ROUTINE mdl_define_constant (name_desc, value, msg_flag, tparse) =
345 0579 1
346 0580 1 --
347 0581 1
348 0582 1 This routine outputs constant lines in the following format:
349 0583 1
350 0584 1     symbol_name, symbol_value
351 0585 1
352 0586 1 It can be called either to output message symbols or literals.
353 0587 1 In the latter case, it checks to be sure that the literal name
354 0588 1 is consistent with the $STRUCT statement. If it is not, a warning
355 0589 1 is issued and the literal is omitted.
356 0590 1
357 0591 1 Inputs:
358 0592 1
359 0593 1     name_desc =     Address of descriptor of symbol name.
360 0594 1
361 0595 1     value =         Symbol value.
362 0596 1
363 0597 1     msg_flag =      Set if a message symbol and not a literal.
364 0598 1
365 0599 1     tparse =        Address of tparse block, when present.
366 0600 1
367 0601 1 Outputs:
368 0602 1
369 0603 1     None
370 0604 1
371 0605 1 --
372 0606 1
373 0607 2 BEGIN
374 0608 2
375 0609 2 MAP
376 0610 2     name_desc:      REF VECTOR;
377 0611 2
378 0612 2 LOCAL
379 0613 2     buffer:          VECTOR [line_size, BYTE],
380 0614 2     buffer_desc:     VECTOR [2],
381 0615 2     faout_desc:      VECTOR [2],
382 0616 2     faout_buf:       VECTOR [line_size, BYTE],
383 0617 2     symbol_desc:     VECTOR [2];
384 0618 2
385 0619 3 IF ( NOT .struc_init )
386 0620 3 THEN BEGIN
387 0621 3     syntax_error (.tparse, emsg (nofacil));
388 0622 3     RETURN true;
389 0623 3 END;
390 0624 2
391 0625 2 symbol_desc[0] = .name_desc[0];
392 0626 2 symbol_desc[1] = .name_desc[1];
393 0627 2
394 0628 3 IF ( NOT .msg_flag )
395 0629 3 THEN BEGIN
396 0630 3
397 0631 3     IF CH$NEQ (.prefix_desc[0], .prefix_desc[1],
398 0632 3         .prefix_desc[0], .symbol_desc[1])
399 0633 4 THEN BEGIN
400 0634 4     syntax_error (.tparse, emsg (litconf));
```

```
! Line to be output
! Buffer descriptor
! Descriptor to accept output from FAO
! FAO output buffer
! Symbol name descriptor
! Structure not initialized
! Signal warning
! Exit
! Initialize symbol name descriptor
! Symbol is a literal
! Be sure that literal prefix is
! consistent with $STRUCT statement
! Signal warning
```

```

401      0635      4      RETURN false;      ! Do not output
402      0636      END;
403      0637      symbol_desc[0] = .symbol_desc[0] - .prefix_desc[0];      ! Remove prefix
404      0638      symbol_desc[1] = .symbol_desc[1] + .prefix_desc[0];
405      0639      END;
406      0640      buffer_desc[0] = record_offset;      ! Initialize buffer descriptor
407      0641      buffer_desc[1] = buffer;
408      0642      CH$FILL (' ', .value_offset, buffer);      ! Clear beginning of line
409      0643      CONCAT (buffer_desc,      ! Put together constant statement
410      0644      symbol_desc,      ! Symbol name comes first
411      0645      comma);      ! Then a comma
412      0646      buffer_desc[0] = .value_offset;      ! Move buffer ptr along
413      0647      IF ..value LEQU dectohex_switch
414      P 0648      THEN (CONCAT (buffer_desc,      ! Use decimal notation
415      P 0649      ul))
416      0650      ELSE (CONCAT (buffer_desc,      ! Use hexadecimal notation
417      0651      xl));
418      0652      faoout_desc[0] = line_size;      ! Set maximum buffer size
419      0653      faoout_desc[1] = faoout_buf;      ! Set FAO output buffer
420      0654      $FAO (buffer_desc, faoout_desc[0],      ! Convert numeric value to string descriptor
421      P 0655      faoout_desc, ..value);
422      0656      mdl_put_record (faoout_desc);      ! Output constant statement
423      P 0657
424      0658      RETURN true;
425      0659
426      0660
427      0661
428      0662
429      P 0663
430      0664
431      0665
432      0666
433      0667
434      0668
435      0669
436      0670      1 END;
```

```

                                007C 00000
56      0000' CF 9E 00002
5E      FEE0 CE 9E 00007
11      FF78 C6 E8 0000C
                                00000000G 8F DD 00011
                                10 AC DD 00017
0000G CF 02 FB 0001A
                                00CE 31 0001F
50      04 AC D0 00022 1$:
6E      60 7D 00026
20      0C AC E8 00029
04 BE 04 B6 66 29 0002D
                                11 13 00033
                                00000000G 8F DD 00035
                                10 AC DD 0003B
0000G CF 02 FB 0003E
```

```

.EXTRN MSG$_NOFACIL, MSG$_LITCONF
.ENTRY MDL DEFINE CONSTANT, Save R2,R3,R4,R5,R6 ; 0578
MOVAB PREFIX_DESC, R6
MOVAB -288(SP), SP
BLBS STRUC_INIT, 1$ ; 0619
PUSHL #MSG$_NOFACIL ; 0621
PUSHL TPARSE
CALLS #2, SYNTAX_ERROR
BRW 6$ ; 0622
MOVL NAME_DESC, R0 ; 0625
MOVQ (R0), SYMBOL_DESC
BLBS MSG_FLAG, 3$ ; 0628
CMPC3 PREFIX_DESC, @PREFIX_DESC+4, @SYMBOL_DESC+4 ; 0631
BEQL 2$ ; 0634
PUSHL #MSG$_LITCONF
PUSHL TPARSE
CALLS #2, SYNTAX_ERROR
```

FF74	C6	20	04	6E	00AE	31	00043	BRW	7\$	:	0635	
			FF74	AE	66	C2	00046	SUBL2	PREFIX_DESC, SYMBOL_DESC	:	0638	
			FF78	CD	66	C0	00049	ADDL2	PREFIX_DESC, SYMBOL_DESC+4	:	0639	
				CD	08	D0	0004D	MOVL	#8, BUFFER_DESC	:	0643	
				CD	9E	00052		MOVAB	BUFFER, BUFFER_DESC+4	:	0644	
				6E	00	2C	00059	MOVCS	#0, (SP), #32, VALUE_OFFSET, BUFFER	:	0646	
					CD		00060			:		
50			FF74	CD	FF7C	CD	C1	00063	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	:	0650
60			04	BE	FF78	CD	6E	28	MOVCS	SYMBOL_DESC, @SYMBOL_DESC+4, (R0)	:	
			FF74	CD		6E	C0	00070	ADDL2	SYMBOL_DESC, BUFFER_DESC	:	
50			FF74	CD	FF78	CD	C1	00075	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	:	
60			00A8	D6	00A4	C6	28	0007D	MOVCS	COMMA, @COMMA+4, (R0)	:	
			FF74	CD	00A4	C6	C0	00085	ADDL2	COMMA, BUFFER_DESC	:	
			FF74	CD	FF74	C6	D0	0008C	MOVL	VALUE_OFFSET, BUFFER_DESC	:	0652
50			FF74	CD	FF78	CD	C1	00093	ADDL3	BUFFER_DESC+4, BUFFER_DESC, R0	:	0656
			0000FFFF	8F	08	BC	D1	0009B	CML	@VALUE, #65535	:	0654
						11	1A	000A3	BGTRU	4\$	:	
60			00D0	D6	00CC	C6	28	000A5	MOVCS	UL, @UL+4, (R0)	:	0656
			FF74	CD	00CC	C6	C0	000AD	ADDL2	UL, BUFFER_DESC	:	
						0F	11	000B4	JRB	5\$	:	0654
60			00D8	D6	00D4	C6	28	000B6	MOVCS	XL, @XL+4, (R0)	:	0658
			FF74	CD	00D4	C6	C0	000BE	ADDL2	XL, BUFFER_DESC	:	
			008C	CE	84	8F	9A	000C5	MOVZBL	#132, FAOOUT_DESC	:	0660
			FF70	CD	08	AE	9E	000CB	MOVAB	FAOOUT_BUF, FAOOUT_DESC+4	:	0661
					08	BC	DD	000D1	PUSHL	@VALUE	:	0664
					0090	CE	9F	000D4	PUSHAB	FAOOUT_DESC	:	
					FF6C	CD	9F	000D8	PUSHAB	FAOOUT_DESC	:	
					FF74	CD	9F	000DC	PUSHAB	BUFFER_DESC	:	
			00000000G	00		04	FB	000E0	CALLS	#4, SYSSFAO	:	
					008C	CE	9F	000E7	PUSHAB	FAOOUT_DESC	:	0666
			0000V	CF	01	FB	000EB	CALLS	#1, MDC_PUT_RECORD	:	0668	
				50	01	D0	000F0	MOVL	#1, R0	:		
						04	000F3	RET		:		
					50	D4	000F4	CLRL	R0	:	0670	
						04	000F6	RET		:		

; Routine Size: 247 bytes, Routine Base: \$CODE\$ + 0266

; 437 0671 1

```
439 0672 1 GLOBAL ROUTINE mdl_end_struct =
440 0673 1
441 0674 1 --
442 0675 1
443 0676 1 This routine outputs an end statement and cleans up
444 0677 1 the previous series of constant commands.
445 0678 1
446 0679 1 >
447 0680 1 E
448 0681 1
449 0682 1 Inputs:
450 0683 1
451 0684 1 None
452 0685 1
453 0686 1 Outputs:
454 0687 1
455 0688 1 None
456 0689 1
457 0690 1 --
458 0691 1
459 0692 2 BEGIN
460 0693 2
461 0694 2 LOCAL
462 0695 2     buffer:      VECTOR [line_size, BYTE],      ! Line to be output
463 0696 2     buffer_desc: VECTOR [2];                    ! Buffer descriptor
464 0697 2
465 0698 3 IF ( NOT .struct_init)                          ! Structure not initialized
466 0699 2 THEN RETURN true;                               ! Exit
467 0700 2
468 0701 2 struct_init = false;                             ! Clear structure active flag
469 0702 2 buffer_desc[0] = record_offset;                  ! Initialize buffer descriptor
470 0703 2 buffer_desc[1] = buffer;
471 0704 2 CH$FIL[ (' ', record_offset, buffer);           ! Clear beginning of line
472 0705 2
473 P 0706 2 CONCAT (buffer_desc,                          ! Put together '>' statement
474 0707 2     bracket);
475 0708 2 mdl_put_record (buffer_desc);                    ! Output that line
476 0709 2
477 0710 2 buffer_desc[0] = record_offset;                  ! Reset buffer size
478 0711 2
479 P 0712 2 CONCAT (buffer_desc,                          ! Put together 'E' statement
480 0713 2     e);
481 0714 2 mdl_put_record (buffer_desc);                    ! Output that line
482 0715 2
483 0716 2 mdl_put_record ( UPLIT (DESC (' ')) );          ! Output blank line
484 0717 2
485 0718 2 RETURN true;
486 0719 2
487 0720 1 END;
```

```
.PSECT $SPLIT$,NOWRT,NOEXE,2
20 00040 P.AAP: .ASCII \ \
00041 .BLKB 3
00000001 00044 P.AAO: .LONG 1
```

: 0672

MOVIE
RET

: 0720

; Routine Size: 96 bytes, Routine Base: \$CODE\$ + 035D

```

489 0721 1 GLOBAL ROUTINE mdl_comment (comment_desc, new_line_flag) =
490 0722 1
491 0723 1  |--
492 0724 1
493 0725 1      This routine outputs a comment either appended directly
494 0726 1      to the end of the previous record, i.e. on the same line
495 0727 1      as the previous record, or on a new line, depending on
496 0728 1      the value of the new_line_flag.
497 0729 1
498 0730 1  Inputs:
499 0731 1
500 0732 1      comment_desc = Descriptor of comment.
501 0733 1
502 0734 1      new_line_flag = Set if comment should be placed on a
503 0735 1      new line.
504 0736 1
505 0737 1  Outputs:
506 0738 1
507 0739 1      None
508 0740 1
509 0741 1  |--
510 0742 1
511 0743 2 BEGIN
512 0744 2
513 0745 2 MAP
514 0746 2     comment_desc : REF VECTOR;
515 0747 2
516 0748 2 LOCAL
517 0749 2     buffer:      VECTOR [line_size, BYTE],      ! Line to be output
518 0750 2     buffer_desc:  VECTOR [2];                    ! Buffer descriptor
519 0751 2
520 0752 2     buffer_desc[0] = 0;                            ! Initialize buffer descriptor
521 0753 2     buffer_desc[1] = buffer;
522 0754 2
523 0755 3 IF ( NOT .new_line_flag )                        ! Append to previous line
524 0756 3 THEN BEGIN
525 0757 3
526 0758 3     buffer_desc[0] = comment_offset;                ! Skip a few spaces
527 0759 3     CH$FILC (' ', comment_offset, buffer);
528 0760 3
529 0761 3 END;
530 0762 2
531 P 0763 2 CONCAT (buffer_desc,                            ! Put together comment record
532 P 0764 2     semicolon,
533 0765 2     .comment_desc);
534 0766 2
535 0767 2 mdl_put_record (buffer_desc,                        ! Output record
536 0768 2     IF NOT .new_line_flag THEN true ELSE false);
537 0769 2
538 0770 2 RETURN true;
539 0771 1 END;
```

007C 00000

.ENTRY MDL_COMMENT, Save R2,R3,R4,R5,R6

; 0721

; Routine Size: 85 bytes, Routine Base: \$CODE\$ + 03BD

```

541 0772 1 GLOBAL ROUTINE mdl_put_record (record_desc, append_to_buffer_flag) =
542 0773 1
543 0774 1 --
544 0775 1
545 0776 1 This routine puts a record into the mdl file.
546 0777 1
547 0778 1 Inputs:
548 0779 1
549 0780 1 record_desc = Descriptor of record to be buffered.
550 0781 1
551 0782 1 append_to_buffer_flag = Set if buffer should be expanded.
552 0783 1 [Default is clear.]
553 0784 1
554 0785 1 Outputs:
555 0786 1
556 0787 1 None
557 0788 1
558 0789 1 --
559 0790 1
560 0791 2 BEGIN
561 0792 2
562 0793 2 MAP
563 0794 2 record_desc: REF VECTOR;
564 0795 2
565 0796 2 BUILTIN
566 0797 2 NULLPARAMETER; ! True if not specified
567 0798 2
568 0799 2 LOCAL
569 0800 2 status; ! Status code
570 0801 2
571 0802 2 IF .mdl_fab [fab$w_ifi] EQL 0 ! If file not opened
572 0803 2 THEN RETURN true; ! Exit
573 0804 2
574 0805 2 IF NULLPARAMETER(2) ! New_line_flag not input or null
575 0806 2
576 0807 2 THEN BEGIN ! Output and then reset buffer
577 0808 2 mdl_rab [rab$w_rsz] = .output_desc[0]; ! Initialize output record
578 0809 2 mdl_rab [rab$l_rbf] = .output_desc[1];
579 0810 2
580 0811 2 status = $PUT (RAB = mdl_rab); ! Output line
581 0812 2
582 0813 2 IF ( NOT .status ) ! If error
583 0814 2 THEN rms_error (emsg (writeerr), mdl_fab, mdl_rab); ! Then report it
584 0815 2
585 0816 2 output_desc[0] = .record_desc[0]; ! Reset buffer size
586 0817 2 CH$MOVE (.record_desc[0], .record_desc[1], .output_desc[1]); ! Reset buffer
587 0818 2
588 0819 2 RETURN .status;
589 0820 2 END
590 0821 2
591 0822 2 ELSE BEGIN ! Simply modify old buffer
592 0823 2 CONCAT (output_desc, .record_desc); ! Append record to buffer
593 0824 2 RETURN true;
594 0825 2 END;
595 0826 2
596 0827 1 END;
```

				.EXTRN SYSS\$PUT			
				01FC	00000		
	58	0000G	CF	9E	00002	.ENTRY MDL_PUT_RECORD, Save R2,R3,R4,R5,R6,R7,R8 : 0772	
	57	0000	CF	9E	00007	MOVAB MDL_RAB, R8 : 0772	
		0000G	CF	B5	0000C	MOVAB OUTPUT_DESC, R7 : 0772	
			55	13	00010	TSTW MDL_FAB+2 : 0802	
	52	04	AC	D0	00012	BEQL 4\$: 0802	
	02		6C	91	00016	MOVL RECORD_DESC, R2 : 0817	
			05	1F	00019	CMPB (AP), #2 : 0805	
		08	AC	D5	0001B	BLSSU 1\$: 0805	
			38	12	0001E	TSTL 8(AP) : 0805	
	2	A8	67	B0	00020	BNEQ 3\$: 0805	
	28	A8	04	A7	00024	MOVW OUTPUT_DESC, MDL_RAB+34 : 0808	
			58	DD	00029	MOVL OUTPUT_DESC+4, MDL_RAB+40 : 0809	
	00000000G	00	01	FB	0002B	PUSHL R8 : 0811	
		56	50	D0	00032	CALLS #1, SYSS\$PUT : 0811	
		11	56	E8	00035	MOVL R0, STATUS : 0813	
			58	DD	00038	BLBS STATUS, 2\$: 0814	
			CF	9F	0003A	PUSHL R8 : 0814	
		0000G	8F	DD	0003E	PUSHAB MDL_FAB : 0814	
		009710D2	03	FB	00044	PUSHL #9900242 : 0814	
			04	BC	00049	CALLS #3, RMS_ERROR : 0814	
04	B7	04	04	BC	28	0004D	MOVL @RECORD_DESC, OUTPUT_DESC : 0816
			56	D0	00054	MOVC3 @RECORD_DESC, @4(R2), @OUTPUT_DESC+4 : 0817	
			04		00057	MOVL STATUS, R0 : 0822	
	50		04	A7	C1	00058	RET : 0823
	60	04	04	BC	28	0005D	ADDL3 OUTPUT_DESC+4, OUTPUT_DESC, R0 : 0823
			04	BC	C0	00063	MOVC3 @RECORD_DESC, @4(R2), (R0) : 0823
			50	01	D0	00067	ADDL2 @RECORD_DESC, OUTPUT_DESC : 0824
				04	0006A	MOVL #1, R0 : 0824	
						RET : 0827	

; Routine Size: 107 bytes, Routine Base: \$CODE\$ + 0412

: 597 0828 1
: 598 0829 1 END
: 599 0830 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	360	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$PLITS	76	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	1149	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics					
File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	22	0	581	00:01.1

;
; COMMAND QUALIFIERS
;
; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:MDLGEN/OBJ=OBJ\$:MDLGEN MSRC\$:MDLGEN/UPDATE=(ENH\$:MDLGEN)
;
; Size: 1149 code + 436 data bytes
; Run Time: 00:22.4
; Elapsed Time: 01:06.5
; Lines/CPU Min: 2226
; Lexemes/CPU-Min: 29777
; Memory Used: 181 pages
; Compilation Complete

0252

AH-BT13A-SE
VAX/VMS V4.

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY