```
RRRRRRRRRRR      UUU          UUU  NNN          NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN          NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN          NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRR          RRR UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNNNNN       NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNNNNN       NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNNNNN       NNN  000         000  FFF              FFF
RRRRRRRRRRRR     UUU          UUU  NNN  NNN     NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN  NNN     NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN  NNN     NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRR    RRR       UUU          UUU  NNN       NNNNNN  000         000  FFF              FFF
RRR    RRR       UUU          UUU  NNN       NNNNNN  000         000  FFF              FFF
RRR    RRR       UUU          UUU  NNN       NNNNNN  000         000  FFF              FFF
RRR       RRR    UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR       RRR    UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR       RRR    UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
```

```
_$2

Sym
---
NDX
NDX
NUM
NUM
OPE
OUT
PAC
PAC
PAC
PAC
PAC
PAC
PAC
PAD
PAG
PAG
PAG
PAG
PAG
PAG
PAG
PER
PUT
RCO
RIN

RLI
RNO
RNO
RTY
SAV
STR
STR
STR
STR
STR
STR
STR

STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
```

FILEID**CNTVMS

```
  CCCCCCCC  NN      NN  TTTTTTTTTT  VV      VV  MM      MM   SSSSSSSS
  CCCCCCCC  NN      NN  TTTTTTTTTT  VV      VV  MM      MM   SSSSSSSS
CC          NN      NN      TT      VV      VV  MMMM  MMMM  SS
CC          NN      NN      TT      VV      VV  MMMM  MMMM  SS
CC          NNNN    NN      TT      VV      VV  MM  MM  MM  SS
CC          NNNN    NN      TT      VV      VV  MM  MM  MM  SS
CC          NN  NN  NN      TT      VV      VV  MM      MM    SSSSSS
CC          NN  NN  NN      TT      VV      VV  MM      MM    SSSSSS
CC          NN    NNNN      TT      VV      VV  MM      MM          SS
CC          NN    NNNN      TT      VV      VV  MM      MM          SS
CC          NN      NN      TT        VV  VV    MM      MM          SS      ....
CC          NN      NN      TT        VV  VV    MM      MM          SS      ....
  CCCCCCCC  NN      NN      TT          VV      MM      MM  SSSSSSSS        ....
  CCCCCCCC  NN      NN      TT          VV      MM      MM  SSSSSSSS        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;    1          0001  0 %TITLE 'CNTVMS - CONTENTS VMS command Line Interface'
;    2          0002  0 MODULE cntvms   (IDENT = 'V04-000',
;    3          0003  0                 ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE)
;    4          0004  0                 ) =
;    5          0005  1 BEGIN
;    6          0006  1
;    7          0007  1 !
;    8          0008  1 !*****************************************************************************
;    9          0009  1 !*                                                                           *
;   10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;   11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;   12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;   13          0013  1 !*                                                                           *
;   14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;   15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;   16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;   17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;   18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;   19          0019  1 !*  TRANSFERRED.                                                              *
;   20          0020  1 !*                                                                           *
;   21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;   22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;   23          0023  1 !*  CORPORATION.                                                              *
;   24          0024  1 !*                                                                           *
;   25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;   26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;   27          0027  1 !*                                                                           *
;   28          0028  1 !*                                                                           *
;   29          0029  1 !*****************************************************************************
;   30          0030  1 !
;   31          0031  1
;   32          0032  1 !++
;   33          0033  1 ! FACILITY:
;   34          0034  1 !   DSR (Digital Standard RUNOFF) /DSRPLUS DSRTOC/CONTENTS Utility
;   35          0035  1 !
;   36          0036  1 ! ABSTRACT:
;   37          0037  1 !   This module contains the command line interface for CONTENTS.
;   38          0038  1 !
;   39          0039  1 ! ENVIRONMENT:   VAX/VMS User Mode
```

```
;    41          0040  1 !
;    42          0041  1 ! AUTHOR:        JPK
;    43          0042  1 !
;    44          0043  1 ! CREATION DATE: April 1982
;    45          0044  1 !
;    46          0045  1 ! MODIFIED BY:
;    47          0046  1 !
;    48          0047  1 !       013     REM00013        Ray Marshall    13-Feb-1984
;    49          0048  1 !               Enabled /REQUIRE for DSRTOC.
;    50          0049  1 !
;    51          0050  1 !       012     KFA00012        29-Aug-1983
;    52          0051  1 !               Modified tms ==> TMS11.
;    53          0052  1 !
;    54          0053  1 !       011     JPK00012        27-May-1983
;    55          0054  1 !               Modified source modules to include REQ: in REQUIRE statements.
;    56          0055  1 !
;    57          0056  1 !       010     JPK00008        09-Mar-1983
;    58          0057  1 !               Modified CONTENTS and CAPTION to support new BRN formats,
;    59          0058  1 !               support SEND CONTENTS, /DOUBLE_SPACE, page numbered chapters,
;    60          0059  1 !               guarantee space after section number and to write new
;    61          0060  1 !               prologue and epilog for RUNOFF output.
;    62          0061  1 !               Modified FORMAT to quote only the RUNOFF flags used by CONTENTS.
;    63          0062  1 !               Modified CNTVMS to fix default for /DOUBLE_SPACE and do more
;    64          0063  1 !               value checking.
;    65          0064  1 !
;    66          0065  1 !       009     JPK00007        14-Feb-1983
;    67          0066  1 !               Global edit of all sources for CONTENTS/DSRTOC:
;    68          0067  1 !               - module names are now consistant with file names
;    69          0068  1 !               - copyright dates have been updated
;    70          0069  1 !               - facility names have been updated
;    71          0070  1 !               - revision history was updated to be consistant with DSR/DSRPLUS
;    72          0071  1 !
;    73          0072  1 !       008     JPK00006        14-Feb-1983
;    74          0073  1 !               Modified CNTVMS, CONTENTS, FORMAT and CNTVMSMSG to generate
;    75          0074  1 !               error messages for DSRTOC or CONTENTS depending on the
;    76          0075  1 !               compiletime variant for DSRPLUS (/VARIANT:8192)
;    77          0076  1 !
;    78          0077  1 !       007     JPK00004        11-Feb-1983
;    79          0078  1 !               Changed the global variable name INDENT to LINE_INDENT in
;    80          0079  1 !               modules CONTENTS, CAPTION, FORMAT and GBLDCL.
;    81          0080  1 !               Removed declarations of PDENTS in modules CNTVMS, CONTENTS,
;    82          0081  1 !               and CAPTION and replaced with a module wide BIND using the
;    83          0082  1 !               new name INDENTS.
;    84          0083  1 !               Changed handling of INDENTS [1]. It no longer represents the
;    85          0084  1 !               sum of the chapter and title indents.
;    86          0085  1 !
;    87          0086  1 !       006     JPK00003        11-Feb-1983
;    88          0087  1 !               Added condition handler to CNTVMS to set program exit status.
;    89          0088  1 !
;    90          0089  1 !       005     JPK00002        10-Feb-1983
;    91          0090  1 !               Merged in change KFA00002 to modules CNTVMS, CNTCLIDMP and
;    92          0091  1 !               CNTCLI.REQ. This change was done without reserving and
;    93          0092  1 !               replacing the files in the CMS Library. This work involved
;    94          0093  1 !               replacing CONTENTS$G_LM_INDENT, CONTENTS$G_HL1_INDENT through
;    95          0094  1 !               CONTENTS$G_HL6_INDENT and CONTENTS$G_HLN_INDENT with
;    96          0095  1 !               CONTENTS$AG_HL_INDENT.
;    97          0096  1 !               CONTENTS$v_TEX was removed - it was no longer needed.
```

```
;    98          0097  1 !                Changed /TMS to /FORMAT=(DSR ! TMS).
;    99          0098  1 !                Added code for /DOUBLE_SPACE.
;   100          0099  1 !
;   101          0100  1 !        004     RER00002        20-Jan-1983
;   102          0101  1 !                Modified CNTVMS and CNTCLI.REQ to add new fields for new and
;   103          0102  1 !                expanded qualifiers:
;   104          0103  1 !                CONTENTS$V_TEX - for tex output,
;   105          0104  1 !                CONTENTS$G_DOUBLE_SPACE - for /DOUBLE_SPACE,
;   106          0105  1 !                CONTENTS$G_HL2_INDENT through CONTENTS$G_HL6_INDENT for /INDENT.
;   107          0106  1 !
;   108          0107  1 !        003     RER00001        17-Dec-1982
;   109          0108  1 !                Added code to CNTVMS to treat keyword NORUNNING in same way as
;   110          0109  1 !                keyword STANDARD.
;   111          0110  1 !                Changed header level default value from 99 to 6.
;   112          0111  1 !                Deleted foreign-command code; CONTENTS is now called
;   113          0112  1 !                as a subcommand of DSR.
;   114          0113  1 !                Conditionalized code to compile for DSRPLUS if BLISS
;   115          0114  1 !                /VARIANT = 8192 is used; otherwise, to compile for DSR.
;   116          0115  1 !
;   117          0116  1 !        002     KFA00002        14-Oct-1982
;   118          0117  1 !                Modified CNTVMS, CONTENTS, CAPTION and CNTCLI.REQ to handle
;   119          0118  1 !                new syntax for /INDENT qualifier. All indents are now
;   120          0119  1 !                stored in the vector CMDBLK [CONTENTS$AG_HL_INDENT []],
;   121          0120  1 !                where 0 = chapter indent, 1 = header level one indent, etc.
;   122          0121  1 !
;   123          0122  1 !--
```

```
;  125     0123  1 !
;  126     0124  1 ! INCLUDE FILES:
;  127     0125  1 !
;  128     0126  1
;  129     0127  1 LIBRARY 'SYS$LIBRARY:STARLET';
;  130     0128  1
;  131     0129  1 LIBRARY 'SYS$LIBRARY:TPAMAC';    ! TPARSE macros
;  132     0130  1
;  133     0131  1 LIBRARY 'SYS$LIBRARY:XPORT';
;  134     0132  1
;  135     0133  1 SWITCHES LIST (REQUIRE);
;  136     0134  1
;  137     0135  1 REQUIRE 'REQ:CNTCLI';           ! Command line information block definition
```

```
;  R0136  1        !
;  R0137  1        ! Version:      'V04-000'
;  R0138  1        !
;  R0139  1        !****************************************************************************
;  R0140  1        !*                                                                          *
;  R0141  1        !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;  R0142  1        !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
;  R0143  1        !*  ALL RIGHTS RESERVED.                                                    *
;  R0144  1        !*                                                                          *
;  R0145  1        !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;  R0146  1        !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;  R0147  1        !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;  R0148  1        !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;  R0149  1        !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;  R0150  1        !*  TRANSFERRED.                                                            *
;  R0151  1        !*                                                                          *
;  R0152  1        !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;  R0153  1        !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;  R0154  1        !*  CORPORATION.                                                            *
;  R0155  1        !*                                                                          *
;  R0156  1        !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;  R0157  1        !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;  R0158  1        !*                                                                          *
;  R0159  1        !*                                                                          *
;  R0160  1        !****************************************************************************
;  R0161  1        !
;  R0162  1
;  R0163  1        !++
;  R0164  1        ! FACILITY:
;  R0165  1        !   DSR (Digital Standard RUNOFF) /DSRPLUS DSRTOC/CONTENTS Utility
;  R0166  1        !
;  R0167  1        ! ABSTRACT:
;  R0168  1        !   CONTENTS Command Line Data Structure Definitions
;  R0169  1        !
;  R0170  1        ! ENVIRONMENT:   Transportable
;  R0171  1        !
;  R0172  1        ! AUTHOR:        JPK
;  R0173  1        !
;  R0174  1        ! CREATION DATE: April 1982
;  R0175  1        !
;  R0176  1        ! MODIFIED BY:
;  R0177  1        !
;  R0178  1        !       005     JPK00007        14-Feb-1983
;  R0179  1        !               Global edit of all sources for CONTENTS/DSRTOC:
;  R0180  1        !               - module names are now consistant with file names
;  R0181  1        !               - copyright dates have been updated
;  R0182  1        !               - facility names have been updated
;  R0183  1        !               - revision history was updated to be consistant with DSR/DSRPLUS
;  R0184  1        !
;  R0185  1        !       004     JPK00002        10-Feb-1983
;  R0186  1        !               Merged in change KFA00002 to modules CNTVMS, CNTCLIDMP and
;  R0187  1        !               CNTCLI.REQ. This change was done without reserving and
;  R0188  1        !               replacing the files in the CMS Library. This work involved
;  R0189  1        !               replacing CONTENTS$G_LM_INDENT, CONTENTS$G_HL1_INDENT through
;  R0190  1        !               CONTENTS$G_HL6_INDENT and CONTENTS$G_HLN_INDENT with
;  R0191  1        !               CONTENTS$AG_HL_INDENT.
;  R0192  1        !               CONTENTS$V_TEX was removed - it was no longer needed.
```

```
;  R0193  1        !               Changed /TMS to /FORMAT={DSR ! TMS}.
;  R0194  1        !               Added code for /DOUBLE_SPACE.
;  R0195  1        !
;  R0196  1        !       003     RER00002        20-Jan-1983
;  R0197  1        !               Modified CNTVMS and CNTCLI.REQ to add new fields for new and
;  R0198  1        !               expanded qualifiers:
;  R0199  1        !               CONTENTS$V_TEX - for tex output,
;  R0200  1        !               CONTENTS$G_DOUBLE_SPACE - for /DOUBLE_SPACE,
;  R0201  1        !               CONTENTS$G_HL2_INDENT through CONTENTS$G_HL6_INDENT for /INDENT.
;  R0202  1        !
;  R0203  1        !       002     KFA00002        14-Oct-1982
;  R0204  1        !               Modified CNTVMS, CONTENTS, CAPTION and CNTCLI.REQ to handle
;  R0205  1        !               new syntax for /INDENT qualifier. All indents are now
;  R0206  1        !               stored in the vector CMDBLK [CONTENTS$AG_HL_INDENT []],
;  R0207  1        !               where 0 = chapter indent, 1 = header level one indent, etc.
;  R0208  1        !
;  R0209  1        !--
;  R0210  1
;  R0211  1        !
;  R0212  1        !  Contents command fields:
;  R0213  1        !
;  R0214  1        $field contents_cmd_fields =
;  R0215  1            SET
;  R0216  1
;  R0217  1            contents$v_options          = [$short_integer],     ! Command option indicators:
;  R0218  1
;  R0219  1                $overlay (contents$v_options)
;  R0220  1
;  R0221  1                contents$v_output               = [$bit],       ! Generate output file
;  R0222  1                contents$v_tms11                = [$bit],       ! Generate TMS11 output
;  R0223  1                contents$v_require              = [$bit],       ! Require file specified
;  R0224  1                contents$v_standard_page        = [$bit],       ! Use section type page numbers
;  R0225  1                contents$v_include_sections     = [$bit],       ! Include section numbers in entry
;  R0226  1                contents$v_log                  = [$bit],       ! Generate /LOG message
;  R0227  1
;  R0228  1                $continue
;  R0229  1
;  R0230  1            contents$h_leader           = [$short_integer],     ! Align to next fullword
;  R0231  1
;  R0232  1                $overlay (contents$h_leader)
;  R0233  1
;  R0234  1                contents$c_leader_char  = [$string(1)],         ! Leader dot character
;  R0235  1
;  R0236  1                $continue
;  R0237  1
;  R0238  1            contents$g_headers          = [$integer],           ! Deepest header level to include
;  R0239  1            contents$g_page_level       = [$integer],           ! Deepest level to include page references
;  R0240  1            contents$g_underline        = [$integer],           ! Deepest level to include underlining
;  R0241  1            contents$g_bold             = [$integer],           ! Deepest level to include bolding
;  R0242  1            contents$g_sections         = [$integer],           ! Deepest level to include section numbers
;  R0243  1            contents$g_page_width       = [$integer],           ! Page width (for RUNOFF only)
;  R0244  1            contents$g_double_space     = [$integer],           ! Double space value
;  R0245  1            contents$ag_hl_indent       = [$sub_block (7)],     ! Left margin and HL 1-6 indents
;  R0246  1            contents$t_input_file       = [$descriptor(dynamic)], ! Input file name descriptor
;  R0247  1            contents$t_output_file      = [$descriptor(dynamic)], ! Output file name descriptor
;  R0248  1            contents$t_require_file     = [$descriptor(dynamic)], ! Require file name descriptor
;  R0249  1            contents$t_command_line     = [$descriptor(dynamic)]  ! Copy of entire command line
```

```
;  R0250  1
;  R0251  1          TES;
;  R0252  1
;  R0253  1      !  End of contents_cmd_fields
;  R0254  1
;  R0255  1      LITERAL
;  R0256  1          contents_cmd$k_length = $field_set_size;
;  R0257  1
;  R0258  1      MACRO
;  R0259  1          $contents_cmd = BLOCK [contents_cmd$k_length] FIELD (contents_cmd_fields) %;
;  R0260  1
;  R0261  1      !
;  R0262  1      !--     End of CNTCLI.REQ
```

```
;  138          0263  1
;  139          0264  1 REQUIRE 'REQ:CNTVMSREQ';        ! VMS error messages
```

```
;  R0265  1        !
;  R0266  1        ! Version:      'V04-000'
;  R0267  1        !
;  R0268  1        !****************************************************************************
;  R0269  1        !*                                                                          *
;  R0270  1        !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;  R0271  1        !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
;  R0272  1        !*  ALL RIGHTS RESERVED.                                                    *
;  R0273  1        !*                                                                          *
;  R0274  1        !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;  R0275  1        !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
;  R0276  1        !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;  R0277  1        !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;  R0278  1        !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;  R0279  1        !*  TRANSFERRED.                                                            *
;  R0280  1        !*                                                                          *
;  R0281  1        !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;  R0282  1        !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;  R0283  1        !*  CORPORATION.                                                            *
;  R0284  1        !*                                                                          *
;  R0285  1        !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;  R0286  1        !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;  R0287  1        !*                                                                          *
;  R0288  1        !*                                                                          *
;  R0289  1        !****************************************************************************
;  R0290  1        !
;  R0291  1
;  R0292  1        !++
;  R0293  1        ! FACILITY:
;  R0294  1        !   DSR (Digital Standard RUNOFF) /DSRPLUS DSRTOC/CONTENTS Utility
;  R0295  1        !
;  R0296  1        ! ABSTRACT:
;  R0297  1        !       This file contains external references to the error message numbers
;  R0298  1        !       for DSRTOC/CONTENTS.
;  R0299  1        !
;  R0300  1        !       New messages must be defined in CNTVMSMSG.MSG and referenced here:
;  R0301  1        !       both in the MACRO section (for DSRTOC) and the EXTERNAL LITERAL
;  R0302  1        !       section (for CONTENTS)
;  R0303  1        !
;  R0304  1        ! ENVIRONMENT:   VAX/VMS User Mode
;  R0305  1        !
;  R0306  1        ! AUTHOR:        JPK
;  R0307  1        !
;  R0308  1        ! CREATION DATE: February 1983
;  R0309  1        !
;  R0310  1        ! MODIFIED BY:
;  R0311  1        !
;  R0312  1        !       002     JPK00009        24-Mar-1983
;  R0313  1        !               Modified CNTT20 to support new command line syntax.
;  R0314  1        !               Modified CONTENTS adding routines PROCESS_PAGE,
;  R0315  1        !               PROCESS_ENTITY_INFO and PROCESS_ENTITY_TXT to remove
;  R0316  1        !               code from routine TOC so that CONTENTS will compile on TOPS-20
;  R0317  1        !               Modified CNTVMSREQ to remove conditional require of RNODEF.
;  R0318  1        !
;  R0319  1        !--
;  R0320  1
;  R0321  1        REQUIRE 'REQ:RNODEF';
```

```
;  R0322  1        !
;  R0323  1        ! Version:      'V04-000'
;  R0324  1        !
;  R0325  1        !****************************************************************************
;  R0326  1        !*                                                                          *
;  R0327  1        !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;  R0328  1        !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
;  R0329  1        !*  ALL RIGHTS RESERVED.                                                    *
;  R0330  1        !*                                                                          *
;  R0331  1        !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;  R0332  1        !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;  R0333  1        !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;  R0334  1        !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;  R0335  1        !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;  R0336  1        !*  TRANSFERRED.                                                            *
;  R0337  1        !*                                                                          *
;  R0338  1        !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;  R0339  1        !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;  R0340  1        !*  CORPORATION.                                                            *
;  R0341  1        !*                                                                          *
;  R0342  1        !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;  R0343  1        !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;  R0344  1        !*                                                                          *
;  R0345  1        !*                                                                          *
;  R0346  1        !****************************************************************************
;  R0347  1        !
;  R0348  1
;  R0349  1        !++
;  R0350  1        ! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
;  R0351  1        !
;  R0352  1        ! ABSTRACT:
;  R0353  1        !       Converts BLISS/VARIANT values into useful names.
;  R0354  1        !
;  R0355  1        ! ENVIRONMENT:   Transportable BLISS
;  R0356  1        !
;  R0357  1        ! AUTHOR:        Rich Friday
;  R0358  1        !
;  R0359  1        ! CREATION DATE: 1978
;  R0360  1        !
;  R0361  1        ! MODIFIED BY:
;  R0362  1        !
;  R0363  1        !       016     KAD00016        Ray Marshall    19-Mar-1984
;  R0364  1        !               Added GERMAN, FRENCH, & ITALIAN.
;  R0365  1        !
;  R0366  1        !       015     KAD00015        Keith Dawson    18-Apr-1983
;  R0367  1        !               Made the LN01 conditional the default for vanilla DSR --
;  R0368  1        !                 its value is 0 (no variant supplied).
;  R0369  1        !
;  R0370  1        !       014     KAD00014        Keith Dawson    22-Mar-1983
;  R0371  1        !               Asserted the LN01 conditional when DSRPLUS is asserted.
;  R0372  1        !
;  R0373  1        !       013     KAD00013        Keith Dawson    20-Mar-1983
;  R0374  1        !               Removed all references to .BIX and .BTC files.
;  R0375  1        !
;  R0376  1        !       012     KAD00012        Keith Dawson    07-Mar-1983
;  R0377  1        !               Global edit of all modules. Updated module names, idents,
;  R0378  1        !               copyright dates. Changed require files to BLISS library.
```

```
;  R0379  1        !
;  R0380  1        !--
;  R0381  1
;  R0382  1        !++
;  R0383  1        !       D E F I N I T I O N   O F  /VARIANT   B I T S
;  R0384  1        !
;  R0385  1        !       The bit assignments are as follows:
;  R0386  1        !
;  R0387  1        !       Bit  Weight   Meaning
;  R0388  1        !-------------------------------------------------------------------------
;  R0389  1        !       --      0     If no /VARIANT is supplied (as for vanilla DSR),
;  R0390  1        !                     compile with LN01 support. LN01 support is also
;  R0391  1        !                     implied by the DSRPLUS variant.
;  R0392  1        !
;  R0393  1        !        0      1     CLEAR =   Unassigned
;  R0394  1        !                     SET   =   Unassigned
;  R0395  1        !
;  R0396  1        !        1      2     CLEAR =   Normal compile
;  R0397  1        !                     SET   =   Compile for DSRPLUS
;  R0398  1        !
;  R0399  1        !       4-6    16     CLEAR =   English (American) version
;  R0400  1        !                     SET   =   16 = German (Austrian)
;  R0401  1        !                               32 = French
;  R0402  1        !                               48 = Italian
;  R0403  1        !--
;  R0404  1
;  R0405  1        !-------------------------------------------------------------------------
;  R0406  1        !   This variable (LN01) controls whether or not to compile an LN01-flavored
;  R0407  1        !   DSR. It is asserted by default, and also whenever DSRPLUS is asserted.
;  R0408  1        !
;  R0409  1        !   Modules utilizing LN01 are:
;  R0410  1        !
;  R0411  1        !       DOOPTS  NOUT
;  R0412  1
;  R0413  1        COMPILETIME
;  R0414  1            ln01 =
;  R0415  2                ( (%VARIANT EQL 0) OR  %VARIANT/2 )
;  R0416  1            ;
;  R0417  1
;  R0418  1        !-------------------------------------------------------------------------
;  R0419  1        !   This variable (DSRPLUS) controls compilation for the DSRPLUS program.
;  R0420  1        !
;  R0421  1        !   All modules utilize DSRPLUS.
;  R0422  1
;  R0423  1        COMPILETIME
;  R0424  1            dsrplus =
;  R0425  2                ( %VARIANT/2 )
;  R0426  1            ;
;  R0427  1
;  R0428  1        !-------------------------------------------------------------------------
;  R0429  1        !   This variable (FLIP) controls compilation of FLIP features of DSRPLUS.
;  R0430  1        !   It assures that FLIP features are compiled only on VMS systems.
;  R0431  1        !
;  R0432  1        !   Modules utilizing FLIP are many and various.
;  R0433  1
;  R0434  1        COMPILETIME
;  R0435  1            flip =
```

```
;  R0436  2              ( %VARIANT/2 AND %BLISS(BLISS32) )
;  R0437  1          ;
;  R0438  1
;  R0439  1      !-----------------------------------------------------------------------
;  R0440  1      !       4-6    16     CLEAR =   English (American) version
;  R0441  1      !                     SET   =   16 = German (Austrian)
;  R0442  1      !                               32 = French
;  R0443  1      !                               48 = Italian
;  R0444  1      COMPILETIME
;  R0445  1          German  = ( %VARIANT/16 AND NOT %VARIANT/32 AND NOT %VARIANT/64 ) ;
;  R0446  1      COMPILETIME
;  R0447  1          French  = ( NOT %VARIANT/16 AND %VARIANT/32 AND NOT %VARIANT/64 ) ;
;  R0448  1      COMPILETIME
;  R0449  1          Italian = ( %VARIANT/16 AND %VARIANT/32 AND NOT %VARIANT/64 ) ;
;  R0450  1      !-----------------------------------------------------------------------
;  R0451  1      !                       End of RNODEF.REQ
```

```
;  R0452  1
; LR0453  1      %IF NOT DSRPLUS
;  R0454  1      %THEN
;  R0455  1
;  R0456  1      MACRO
;  R0457  1          CONTENTS$_BADVALUE  = DSRTOC$_BADVALUE  %,
;  R0458  1          CONTENTS$_OPENIN    = DSRTOC$_OPENIN    %,
;  R0459  1          CONTENTS$_OPENOUT   = DSRTOC$_OPENOUT   %,
;  R0460  1          CONTENTS$_VALERR    = DSRTOC$_VALERR    %,
;  R0461  1          CONTENTS$_CAPTIONS  = DSRTOC$_CAPTIONS  %,
;  R0462  1          CONTENTS$_CLOSEQUOT = DSRTOC$_CLOSEQUOT %,
;  R0463  1          CONTENTS$_CONFQUAL  = DSRTOC$_CONFQUAL  %,
;  R0464  1          CONTENTS$_CTRLCHAR  = DSRTOC$_CTRLCHAR  %,
;  R0465  1          CONTENTS$_EMPTYIN   = DSRTOC$_EMPTYIN   %,
;  R0466  1          CONTENTS$_IGNORED   = DSRTOC$_IGNORED   %,
;  R0467  1          CONTENTS$_INVINPUT  = DSRTOC$_INVINPUT  %,
;  R0468  1          CONTENTS$_INVRECORD = DSRTOC$_INVRECORD %,
;  R0469  1          CONTENTS$_OVERSTRK  = DSRTOC$_OVERSTRK  %,
;  R0470  1          CONTENTS$_COMPLETE  = DSRTOC$_COMPLETE  %,
;  R0471  1          CONTENTS$_CREATED   = DSRTOC$_CREATED   %,
;  R0472  1          CONTENTS$_IGNORENEW = DSRTOC$_IGNORENEW %,
;  R0473  1          CONTENTS$_IGNOREOLD = DSRTOC$_IGNOREOLD %,
;  R0474  1          CONTENTS$_PROCFILE  = DSRTOC$_PROCFILE  %,
;  R0475  1          CONTENTS$_TEXTD     = DSRTOC$_TEXTD     %,
;  R0476  1          CONTENTS$_TMS11     = DSRTOC$_TMS11     %,
;  R0477  1          CONTENTS$_IDENT     = DSRTOC$_IDENT     %;
;  R0478  1
;  R0479  1      %FI
;  R0480  1
;  R0481  1      EXTERNAL LITERAL
;  R0482  1          CONTENTS$_BADVALUE,     ! <'!AS' is an invalid keyword value>
;  R0483  1          CONTENTS$_OPENIN,       ! <error opening '!AS' for input>
;  R0484  1          CONTENTS$_OPENOUT,      ! <error opening '!AS' for output>
;  R0485  1          CONTENTS$_VALERR,       ! <specified value is out of legal range>
;  R0486  1          CONTENTS$_CAPTIONS,     ! <both ".HEADER x" and ".SEND TOC n" captions encountered>
;  R0487  1          CONTENTS$_CLOSEQUOT,    ! <the following line is missing a close quote>
;  R0488  1          CONTENTS$_CONFQUAL,     ! <conflicting qualifiers>
;  R0489  1          CONTENTS$_CTRLCHAR,     ! <the following line contains control characters - ignored>
;  R0490  1          CONTENTS$_EMPTYIN,      ! <empty input file '!AS'>
;  R0491  1          CONTENTS$_IGNORED,      ! <'!AS' ignored>
;  R0492  1          CONTENTS$_INVINPUT,     ! <invalid input file format in file '!AS'>
;  R0493  1          CONTENTS$_INVRECORD,    ! <invalid record type in file '!AS'>
;  R0494  1          CONTENTS$_OVERSTRK,     ! <the following line contains an overstrike sequence>
;  R0495  1          CONTENTS$_COMPLETE,     ! <processing complete '!AS'>
;  R0496  1          CONTENTS$_CREATED,      ! <'!AS' created>
;  R0497  1          CONTENTS$_IGNORENEW,    ! <".HEADER x" captions will be ignored>
;  R0498  1          CONTENTS$_IGNOREOLD,    ! <".SEND TOC n" captions will be ignored>
;  R0499  1          CONTENTS$_PROCFILE,     ! <processing file '!AS'>
;  R0500  1          CONTENTS$_TEXTD,        ! <!AD>
;  R0501  1          CONTENTS$_TMS11,        ! <output file full - continuing with file '!AS'>
;  R0502  1          CONTENTS$_IDENT;        ! <CONTENTS version !AD>
;  R0503  1
```

```
;  140          0504  1
;  141          0505  1 SWITCHES LIST (NOREQUIRE);
;  142          0506  1
;  143          0507  1 !
;  144          0508  1 ! TABLE OF CONTENTS:
;  145          0509  1 !
;  146          0510  1
;  147          0511  1 FORWARD ROUTINE
;  148          0512  1     cntcli,                 ! Command line interface routine
;  149          0513  1     call_tparse,            ! Procedure to invoke TPARSE
;  150          0514  1     enter_page,             ! Action routine - enter page number display type
;  151          0515  1     condition_handler,      ! Condition handler - sets termination status
;  152          0516  1     open_error;             ! File open error handler
;  153          0517  1
;  154        L 0518  1 %IF DSRPLUS
;  155        U 0519  1 %THEN
;  156        U 0520  1
;  157        U 0521  1 FORWARD ROUTINE
;  158        U 0522  1     enter_format,           ! Action routine - enter output format type
;  159        U 0523  1     enter_sect,             ! Action routine - enter section number display level
;  160        U 0524  1     enter_dot,              ! Action routine - enter leader dot character
;  161        U 0525  1     enter_hl_indent;        ! Action routine - enter HL indent value
;  162        U 0526  1
;  163          0527  1 %FI
;  164          0528  1
;  165          0529  1 !
;  166          0530  1 ! EQUATED SYMBOLS:
;  167          0531  1 !
;  168          0532  1
;  169          0533  1 LITERAL
;  170          0534  1     true  = 1,
;  171          0535  1     false = 0;
;  172          0536  1
;  173          0537  1 !
;  174          0538  1 ! OWN STORAGE:
;  175          0539  1 !
;  176          0540  1
;  177          0541  1 OWN
;  178          0542  1     qualifier_value,
;  179          0543  1     tmp_str : $str_descriptor (class = dynamic, string = (0, 0)),
;  180          0544  1     termination_status : INITIAL (sts$k_success);
;  181          0545  1
;  182          0546  1 !
;  183          0547  1 ! EXTERNAL REFERENCES:
;  184          0548  1 !
;  185          0549  1
;  186          0550  1 EXTERNAL LITERAL                ! Status codes returned by cli$present ()
;  187          0551  1     cli$_concat,
;  188          0552  1     cli$_present,
;  189          0553  1     cli$_defaulted,
;  190          0554  1     cli$_negated,
;  191          0555  1     cli$_absent;
;  192          0556  1
;  193          0557  1 EXTERNAL
;  194          0558  1     cmdblk : $contents_cmd,     ! Command line information block
;  195          0559  1     cntvrl,                     ! Length of version number string
;  196          0560  1     cntvrp;                     ! CH$PTR to version number string
```

```
;  197          0561  1
;  198          0562  1 BIND
;  199          0563  1     indents = cmdblk [contents$ag_hl_indent] : VECTOR;
;  200          0564  1
;  201          0565  1 EXTERNAL ROUTINE
;  202          0566  1     cli$present     : ADDRESSING_MODE (GENERAL),        ! Qualifier present?
;  203          0567  1     cli$get_value   : ADDRESSING_MODE (GENERAL),        ! Get parameter value
;  204          0568  1     lib$tparse      : ADDRESSING_MODE (GENERAL),        ! TPARSE parser
;  205          0569  1     toc,                                                ! Entry point
;  206          0570  1     tocfin;                                             ! Cleanup routine
;  207          0571  1
;  208          0572  1 !+
;  209          0573  1 !   TPARSE state tables
;  210          0574  1 !-
;  211          0575  1
;  212          0576  1 !
;  213          0577  1 ! Tables to parse a decimal number
;  214          0578  1 !
;  215          0579  1 $init_state (numb_state, numb_key);
;  216        P 0580  1 $state (,
;  217        P 0581  1     (tpa$_decimal,    ,           ,                   , qualifier_value),
;  218        P 0582  1     (tpa$_eos,        tpa$_exit)
;  219          0583  1     );
;  220        P 0584  1 $state (,
;  221        P 0585  1     (tpa$_eos,        tpa$_exit)
;  222          0586  1     );
;  223          0587  1
;  224          0588  1 !
;  225          0589  1 ! Tables to parse /PAGE_NUMBERS values
;  226          0590  1 !
;  227          0591  1 $init_state (page_state, page_key);
;  228        P 0592  1 $state (,
;  229        P 0593  1     ('RUNNING',       page_end,  enter_page,      , ,                 false),
;  230        P 0594  1     ('NORUNNING',     page_end,  enter_page,      , ,                 true),
;  231        P 0595  1
;  232        L 0596  1 %IF DSRPLUS
;  233        U 0597  1 %THEN
;  234        U 0598  1
;  235        U 0599  1     ('STANDARD',      page_end,  enter_page,      , ,                 true),
;  236        U 0600  1
;  237        P 0601  1 %FI
;  238        P 0602  1
;  239        P 0603  1     ('LEVEL')
;  240          0604  1     );
;  241        P 0605  1 $state (,
;  242        P 0606  1     ('='),
;  243        P 0607  1     (':')
;  244          0608  1     );
;  245        P 0609  1 $state (,
;  246        P 0610  1     (tpa$_decimal,    page_end,  ,                   , qualifier_value)
;  247          0611  1     );
;  248        P 0612  1 $state (page_end,
;  249        P 0613  1     (tpa$_eos,        tpa$_exit)
;  250          0614  1     );
;  251          0615  1
;  252        L 0616  1 %IF DSRPLUS
;  253        U 0617  1 %THEN
```

```
;  254          U 0618  1
;  255          U 0619  1 !
;  256          U 0620  1 ! Tables to parse /FORMAT values
;  257          U 0621  1 !
;  258          U 0622  1 $init_state (format_state, format_key);
;  259          U 0623  1 $state (,
;  260          U 0624  1     ('DSR',           ,          enter_format,    , ,          false),
;  261          U 0625  1     ('TMS11',         ,          enter_format,    , ,          true)
;  262          U 0626  1     );
;  263          U 0627  1 $state (,
;  264          U 0628  1     (tpa$_eos,        tpa$_exit)
;  265          U 0629  1     );
;  266          U 0630  1
;  267          U 0631  1 !
;  268          U 0632  1 ! Tables to parse /INDENT values
;  269          U 0633  1 !
;  270          U 0634  1 $init_state (ind_state, ind_key);
;  271          U 0635  1 $state (,
;  272          U 0636  1     ('CHAPTER',       ,          ,                , ,          0),
;  273          U 0637  1     ('TITLES',        ,          ,                , ,          1),
;  274          U 0638  1     ('PROGRESSIVE',   ,          ,                , ,          -1),
;  275          U 0639  1     ('HL1',           ,          ,                , ,          1),
;  276          U 0640  1     ('HL2',           ,          ,                , ,          2),
;  277          U 0641  1     ('HL3',           ,          ,                , ,          3),
;  278          U 0642  1     ('HL4',           ,          ,                , ,          4),
;  279          U 0643  1     ('HL5',           ,          ,                , ,          5),
;  280          U 0644  1     ('HL6',           ,          ,                , ,          6)
;  281          U 0645  1     );
;  282          U 0646  1 $state (,
;  283          U 0647  1     ('='),
;  284          U 0648  1     (':')
;  285          U 0649  1     );
;  286          U 0650  1 $state (,
;  287          U 0651  1     (tpa$_decimal,    ,          enter_hl_indent)
;  288          U 0652  1     );
;  289          U 0653  1 $state (,
;  290          U 0654  1     (tpa$_eos,        tpa$_exit)
;  291          U 0655  1     );
;  292          U 0656  1
;  293          U 0657  1 !
;  294          U 0658  1 ! Tables to parse /SECTION_NUMBERS values
;  295          U 0659  1 !
;  296          U 0660  1 $init_state (sect_state, sect_key);
;  297          U 0661  1 $state (,
;  298          U 0662  1     ('AS_INPUT',      sect_end,  enter_sect,      , ,          -1),
;  299          U 0663  1     ('ALL',           sect_end,  enter_sect,      , ,          6),
;  300          U 0664  1     ('NONE',          sect_end,  enter_sect,      , ,          -2),
;  301          U 0665  1     ('LEVEL')
;  302          U 0666  1     );
;  303          U 0667  1 $state (,
;  304          U 0668  1     ('='),
;  305          U 0669  1     (':')
;  306          U 0670  1     );
;  307          U 0671  1 $state (,
;  308          U 0672  1     (tpa$_decimal,    sect_end,  enter_sect,      , ,          0)
;  309          U 0673  1     );
;  310          U 0674  1 $state(sect_end,
```

```
;   311        U 0675  1     (tpa$_eos,       tpa$_exit)
;   312        U 0676  1     );
;   313        U 0677  1
;   314        U 0678  1 !
;   315        U 0679  1 ! Tables to parse /LEADER_DOTS values
;   316        U 0680  1 !
;   317        U 0681  1 $init_state (dot_state, dot_key);
;   318        U 0682  1 $state (,
;   319        U 0683  1     (tpa$_any,                ,          enter_dot),
;   320        U 0684  1     (tpa$_eos,       tpa$_exit)
;   321        U 0685  1     );
;   322        U 0686  1 $state (,
;   323        U 0687  1     (tpa$_eos,       tpa$_exit)
;   324        U 0688  1     );
;   325        U 0689  1
;   326          0690  1 %FI
```

```
;  328     0691  1 %SBTTL 'CNTCLI - CONTENTS VMS Command line interface'
;  329     0692  1 GLOBAL ROUTINE cntcli =
;  330     0693  1 !++
;  331     0694  1 !
;  332     0695  1 ! FUNCTIONAL DESCRIPTION:
;  333     0696  1 !
;  334     0697  1 !       This routine uses the VMS DCL CLE to obtain command
;  335     0698  1 !       line information that is, in turn, passed to the
;  336     0699  1 !       Table of Contents application in a transportable fashion.
;  337     0700  1 !
;  338     0701  1 ! FORMAL PARAMETERS:
;  339     0702  1 !
;  340     0703  1 !       None
;  341     0704  1 !
;  342     0705  1 ! IMPLICIT INPUTS:
;  343     0706  1 !
;  344     0707  1 !       The VMS command line
;  345     0708  1 !
;  346     0709  1 ! IMPLICIT OUTPUTS:
;  347     0710  1 !
;  348     0711  1 !       cmdblk - The command line information block is initialized
;  349     0712  1 !
;  350     0713  1 ! ROUTINE VALUE:
;  351     0714  1 ! COMPLETION CODES:
;  352     0715  1 !
;  353     0716  1 !       termination_status      - Set by CONDITION_HANDLER ()
;  354     0717  1 !
;  355     0718  1 ! SIDE EFFECTS:
;  356     0719  1 !
;  357     0720  1 !       None
;  358     0721  1 !
;  359     0722  1 !--
;  360     0723  1
;  361     0724  2     BEGIN
;  362     0725  2
;  363     0726  2     ENABLE
;  364     0727  2         condition_handler;
;  365     0728  2
;  366     0729  2     LOCAL
;  367     0730  2         status;
;  368     0731  2
;  369     0732  2     $str_desc_init (descriptor = cmdblk [contents$t_input_file], class = dynamic);
;  370     0733  2     $str_desc_init (descriptor = cmdblk [contents$t_output_file], class = dynamic);
;  371     0734  2     $str_desc_init (descriptor = cmdblk [contents$t_require_file], class = dynamic);
;  372     0735  2     $str_desc_init (descriptor = cmdblk [contents$t_command_line], class = dynamic);
;  373     0736  2
;  374     0737  2     !
;  375     0738  2     ! Get a copy of the whole command line.
;  376     0739  2     !
;  377     0740  2     cli$get_value (%ASCID'$LINE', cmdblk [contents$t_command_line]);
;  378     0741  2
;  379     0742  2     !
;  380     0743  2     ! /FORMAT = { DSR ! TMS11 }
;  381     0744  2     !
;  382     0745  2     !           * W A R N I N G *
;  383     0746  2     !
;  384     0747  2     !   This qualifier must be processed before any other qualifier.
```

```
;  385          0748  2      !   Other qualifiers depend on the value of this qualifier.
;  386          0749  2      !
;  387          0750  2      !           * W A R N I N G *
;  388          0751  2      !
;  389          0752  2      cmdblk [contents$v_tms11] = false;
;  390          0753  2
;  391        L 0754  2  %IF DSRPLUS
;  392        U 0755  2  %THEN
;  393        U 0756  2
;  394        U 0757  2      IF  cli$present (%ASCID'FORMAT')
;  395        U 0758  2      THEN
;  396        U 0759  2          BEGIN
;  397        U 0760  2          cli$get_value (%ASCID'FORMAT', tmp_str);
;  398        U 0761  2
;  399        U 0762  2          IF NOT call_tparse (tmp_str, format_state, format_key, false)
;  400        U 0763  2          THEN
;  401        U 0764  2              SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;  402        U 0765  2
;  403        U 0766  2          END;
;  404        U 0767  2
;  405          0768  2  %FI
;  406          0769  2
;  407          0770  2      !
;  408          0771  2      ! /LINE_WIDTH = n
;  409          0772  2      !
;  410          0773  2      !           * W A R N I N G *
;  411          0774  2      !
;  412          0775  2      !   This qualifier must be processed after /FORMAT before any other
;  413          0776  2      !   qualifier. It depends on the value of /FORMAT and other qualifiers
;  414          0777  2      !   depend on the value of this qualifier.
;  415          0778  2      !
;  416          0779  2      !           * W A R N I N G *
;  417          0780  2      !
;  418          0781  2      qualifier_value = 70;
;  419          0782  2
;  420        L 0783  2  %IF DSRPLUS
;  421        U 0784  2  %THEN
;  422        U 0785  2
;  423        U 0786  2      IF  .cmdblk [contents$v_tms11]
;  424        U 0787  2      THEN
;  425        U 0788  2          BEGIN
;  426        U 0789  2          !
;  427        U 0790  2          ! Generating TMS11 output.
;  428        U 0791  2          !
;  429        U 0792  2          IF  cli$present (%ASCID'LINE_WIDTH')
;  430        U 0793  2          THEN
;  431        U 0794  2              SIGNAL (contents$_ignored, 1, %ASCID'LINE_WIDTH', contents$_confqual);
;  432        U 0795  2
;  433        U 0796  2          qualifier_value = -1;
;  434        U 0797  2          END
;  435        U 0798  2      ELSE
;  436        U 0799  2          BEGIN
;  437        U 0800  2
;  438        U 0801  2          IF  cli$present (%ASCID'LINE_WIDTH')
;  439        U 0802  2          THEN
;  440        U 0803  2              BEGIN
;  441        U 0804  2              cli$get_value (%ASCID'LINE_WIDTH', tmp_str);
```

```
;   442          U 0805  2
;   443          U 0806  2              IF  NOT call_tparse (tmp_str, numb_state, numb_key, false)
;   444          U 0807  2              THEN
;   445          U 0808  2                  SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;   446          U 0809  2
;   447          U 0810  2              IF  (.qualifier_value LSS 20) OR (.qualifier_value GTR 120)
;   448          U 0811  2              THEN
;   449          U 0812  2                  SIGNAL_STOP (contents$_badvalue, 1, tmp_str, contents$_valerr);
;   450          U 0813  2
;   451          U 0814  2              END;
;   452          U 0815  2
;   453          U 0816  2          END;
;   454          U 0817  2
;   455            0818  2 %FI
;   456            0819  2
;   457            0820  2      cmdblk [contents$g_page_width] = .qualifier_value;
;   458            0821  2
;   459            0822  2      !
;   460            0823  2      ! /DEEPEST_HEADER = n
;   461            0824  2      !
;   462            0825  2      !           * W A R N I N G *
;   463            0826  2      !
;   464            0827  2      !   This qualifier must be processed before /INDENT.
;   465            0828  2      !   The value of indent depends on the value of this qualifier.
;   466            0829  2      !
;   467            0830  2      !           * W A R N I N G *
;   468            0831  2      !
;   469            0832  2      qualifier_value = 6;
;   470            0833  2
;   471            0834  2      IF  cli$present (%ASCID'DEEPEST_HEADER')
;   472            0835  2      THEN
;   473            0836  3          BEGIN
;   474            0837  3          cli$get_value (%ASCID'DEEPEST_HEADER', tmp_str);
;   475            0838  3
;   476            0839  3          IF  NOT call_tparse (tmp_str, numb_state, numb_key, false)
;   477            0840  3          THEN
;   478            0841  3              SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;   479            0842  3
;   480            0843  3          IF  .qualifier_value GTR 6
;   481            0844  3          THEN
;   482            0845  3              SIGNAL_STOP (contents$_badvalue, 1, tmp_str, contents$_valerr);
;   483            0846  3
;   484            0847  2          END;
;   485            0848  2
;   486            0849  2      cmdblk [contents$g_headers] = .qualifier_value;
;   487            0850  2
;   488            0851  2      !
;   489            0852  2      ! /INDENT = ([CHAPTER = n], [TITLES = m], [PROGRESSIVE = p],
;   490            0853  2      !           [HL1 = m], [HL2 = q], [HL3 = r], [HL4 = s], [HL5 = t], [HL6 = u])
;   491            0854  2      !
;   492            0855  2      !           * W A R N I N G *
;   493            0856  2      !
;   494            0857  2      !   This qualifier must be processed after /FORMAT, /LINE_WIDTH and
;   495            0858  2      !   /DEEPEST_HEADER. It depends on their values.
;   496            0859  2      !
;   497            0860  2      !           * W A R N I N G *
;   498            0861  2      !
```

```
;  499          0862  2      indents [0] = 8;
;  500          0863  2      indents [1] = 8;
;  501          0864  2      indents [2] = 2;
;  502          0865  2
;  503        L 0866  2  %IF DSRPLUS
;  504        U 0867  2  %THEN
;  505        U 0868  2
;  506        U 0869  2      indents [3] = 2;
;  507        U 0870  2      indents [4] = 2;
;  508        U 0871  2      indents [5] = 2;
;  509        U 0872  2      indents [6] = 2;
;  510        U 0873  2
;  511          0874  2  %ELSE
;  512          0875  2
;  513          0876  2      indents [3] = 0;
;  514          0877  2      indents [4] = 0;
;  515          0878  2      indents [5] = 0;
;  516          0879  2      indents [6] = 0;
;  517          0880  2
;  518          0881  2  %FI
;  519          0882  2
;  520          0883  2      IF cli$present (%ASCID'INDENT')
;  521          0884  2      THEN
;  522          0885  3          BEGIN
;  523          0886  3
;  524        L 0887  3  %IF DSRPLUS
;  525        U 0888  3  %THEN
;  526        U 0889  3
;  527        U 0890  3          IF  .cmdblk [contents$v_tms11]
;  528        U 0891  3          THEN
;  529        U 0892  3              BEGIN
;  530        U 0893  3              !
;  531        U 0894  3              ! Generating TMS11 output.
;  532        U 0895  3              !
;  533        U 0896  3              SIGNAL (contents$_ignored, 1, %ASCID'INDENT', contents$_confqual);
;  534        U 0897  3
;  535        U 0898  3              INCR i FROM 0 TO 6 DO indents [.i] = 0;
;  536        U 0899  3              END
;  537        U 0900  3          ELSE
;  538        U 0901  3              BEGIN
;  539        U 0902  3
;  540        U 0903  3              LOCAL
;  541        U 0904  3                  total_indent;
;  542        U 0905  3
;  543        U 0906  3              WHILE cli$get_value (%ASCID'INDENT', tmp_str) DO
;  544        U 0907  3                  IF  NOT call_tparse (tmp_str, ind_state, ind_key, false)
;  545        U 0908  3                  THEN
;  546        U 0909  3                      SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;  547        U 0910  3
;  548        U 0911  3              !
;  549        U 0912  3              ! Validate left margin indent.
;  550        U 0913  3              !
;  551        U 0914  3              total_indent = .indents [0];
;  552        U 0915  3
;  553        U 0916  3              IF  .total_indent GEQ .cmdblk [contents$g_page_width]
;  554        U 0917  3              THEN
;  555        U 0918  3                  BEGIN
```

```
;  556        U 0919  3                $str_copy (target = tmp_str,
;  557        U 0920  3                    string = $str_concat ('CHAPTER=', $str_ascii (.indents [0])));
;  558        U 0921  3
;  559        U 0922  3                SIGNAL_STOP (contents$_badvalue, 1, tmp_str, contents$_valerr);
;  560        U 0923  3                END;
;  561        U 0924  3
;  562        U 0925  3            !
;  563        U 0926  3            ! Validate HL1 indent.
;  564        U 0927  3            !
;  565        U 0928  3            total_indent = .total_indent + .indents [1];
;  566        U 0929  3
;  567        U 0930  3            IF  .total_indent GEQ .cmdblk [contents$g_page_width]
;  568        U 0931  3            THEN
;  569        U 0932  3                BEGIN
;  570        U 0933  3                $str_copy (target = tmp_str,
;  571        U 0934  3                    string = $str_concat ('HL1=', $str_ascii (.indents [1])));
;  572        U 0935  3
;  573        U 0936  3                SIGNAL_STOP (contents$_badvalue, 1, tmp_str, contents$_valerr);
;  574        U 0937  3                END;
;  575        U 0938  3
;  576        U 0939  3            !
;  577        U 0940  3            ! Validate progressive indents
;  578        U 0941  3            !
;  579        U 0942  3            INCR i FROM 2 TO .cmdblk [contents$g_headers] DO
;  580        U 0943  3                BEGIN
;  581        U 0944  3                total_indent = .total_indent + .indents [.i];
;  582        U 0945  3
;  583        U 0946  3                IF .total_indent GEQ .cmdblk [contents$g_page_width]
;  584        U 0947  3                THEN
;  585        U 0948  3                    BEGIN
;  586        U 0949  3                    $str_copy (target = tmp_str,
;  587        U 0950  3                        string = $str_concat ('HL', $str_ascii (.i), '=',
;  588        U 0951  3                            $str_ascii (.indents [.i])));
;  589        U 0952  3
;  590        U 0953  3                    SIGNAL_STOP (contents$_badvalue, 1, tmp_str, contents$_valerr);
;  591        U 0954  3                    END;
;  592        U 0955  3
;  593        U 0956  3                END;
;  594        U 0957  3
;  595        U 0958  3            END;
;  596        U 0959  3
;  597          0960  2        %ELSE
;  598          0961  2
;  599          0962  2            indents [3] = 2;
;  600          0963  2            indents [4] = 2;
;  601          0964  2            indents [5] = 2;
;  602          0965  2            indents [6] = 2;
;  603          0966  2
;  604          0967  2        %FI
;  605          0968  2
;  606          0969  2            END;
;  607          0970  2
;  608          0971  2        !
;  609          0972  2        ! /[NO]LEADER_DOTS = k
;  610          0973  2        !
;  611          0974  2        !           * W A R N I N G *
;  612          0975  2        !
```

```
;  613          0976  2      !   This qualifier must be processed after /FORMAT.
;  614          0977  2      !   It depends on the value of /FORMAT.
;  615          0978  2      !
;  616          0979  2      !           * W A R N I N G *
;  617          0980  2      !
;  618          0981  2      CH$WCHAR (%C'.', CH$PTR (cmdblk [contents$c_leader_char]));
;  619          0982  2
;  620        L 0983  2  %IF DSRPLUS
;  621        U 0984  2  %THEN
;  622        U 0985  2
;  623        U 0986  2      IF  .cmdblk [contents$v_tms11]
;  624        U 0987  2      THEN
;  625        U 0988  2          BEGIN
;  626        U 0989  2          !
;  627        U 0990  2          ! Generating TMS11 output.
;  628        U 0991  2          !
;  629        U 0992  2          SELECTONE cli$present (%ASCID'LEADER_DOTS') OF
;  630        U 0993  2              SET
;  631        U 0994  2
;  632        U 0995  2              [cli$_present, cli$_negated]:
;  633        U 0996  2                  !
;  634        U 0997  2                  ! Value explicitly given or negated - ignored for TMS.
;  635        U 0998  2                  !
;  636        U 0999  2                  SIGNAL (contents$_ignored, 1, %ASCID'LEADER_DOTS', contents$_confqual);
;  637        U 1000  2
;  638        U 1001  2              [OTHERWISE]:
;  639        U 1002  2                  !
;  640        U 1003  2                  ! Value defaulted
;  641        U 1004  2                  ! or value absent (not possible, value is present by default).
;  642        U 1005  2                  ! In any event, do nothing.
;  643        U 1006  2                  !
;  644        U 1007  2                  ;
;  645        U 1008  2
;  646        U 1009  2              TES;
;  647        U 1010  2
;  648        U 1011  2          CH$WCHAR (%C' ', CH$PTR (cmdblk [contents$c_leader_char]));
;  649        U 1012  2          END
;  650        U 1013  2      ELSE
;  651        U 1014  2          BEGIN
;  652        U 1015  2
;  653        U 1016  2          SELECTONE cli$present (%ASCID'LEADER_DOTS') OF
;  654        U 1017  2              SET
;  655        U 1018  2
;  656        U 1019  2              [cli$_present]:
;  657        U 1020  2                  BEGIN
;  658        U 1021  2                  !
;  659        U 1022  2                  ! Value explicitly given.
;  660        U 1023  2                  !
;  661        U 1024  2                  cli$get_value (%ASCID'LEADER_DOTS', tmp_str);
;  662        U 1025  2
;  663        U 1026  2                  IF  NOT call_tparse (tmp_str, dot_state, dot_key, true)
;  664        U 1027  2                  THEN
;  665        U 1028  2                      SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;  666        U 1029  2
;  667        U 1030  2                  END;
;  668        U 1031  2
;  669        U 1032  2              [cli$_negated]:
```

```
;  670          U 1033  2                    !
;  671          U 1034  2                    ! Value explicitly negated.
;  672          U 1035  2                    !
;  673          U 1036  2                    CH$WCHAR (%C' ', CH$PTR (cmdblk [contents$c_leader_char]));
;  674          U 1037  2
;  675          U 1038  2                [OTHERWISE]:
;  676          U 1039  2                    !
;  677          U 1040  2                    ! Value defaulted
;  678          U 1041  2                    ! or qualifier absent (not possible - present by default).
;  679          U 1042  2                    ! In any event, do nothing.
;  680          U 1043  2                    !
;  681          U 1044  2                    ;
;  682          U 1045  2
;  683          U 1046  2                TES;
;  684          U 1047  2
;  685          U 1048  2            END;
;  686          U 1049  2
;  687            1050  2 %FI
;  688            1051  2
;  689            1052  2        !
;  690            1053  2        ! /PAGE_NUMBERS = ([[NO]RUNNING], [STANDARD], [LEVEL = n])
;  691            1054  2        !
;  692            1055  2        ! NORUNNING is the same as STANDARD.
;  693            1056  2        !
;  694            1057  2        !           * W A R N I N G *
;  695            1058  2        !
;  696            1059  2        !   This qualifier must be processed after /FORMAT.
;  697            1060  2        !   It depends on the value of /FORMAT.
;  698            1061  2        !
;  699            1062  2        !           * W A R N I N G *
;  700            1063  2        !
;  701            1064  2        cmdblk [contents$v_standard_page] = true;
;  702            1065  2        qualifier_value = 6;
;  703            1066  2
;  704            1067  2        IF  cli$present (%ASCID'PAGE_NUMBERS')
;  705            1068  2        THEN
;  706            1069  3            BEGIN
;  707            1070  3
;  708            1071  3            WHILE cli$get_value (%ASCID'PAGE_NUMBERS', tmp_str) DO
;  709            1072  3                IF  NOT call_tparse (tmp_str, page_state, page_key, false)
;  710            1073  3                THEN
;  711            1074  4                    SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;  712            1075  3
;  713            1076  4                IF  .cmdblk [contents$v_tms11] and (.qualifier_value NEQ 6)
;  714            1077  3                THEN
;  715            1078  4                    BEGIN
;  716            1079  4                    !
;  717            1080  4                    ! LEVEL = n not allowed for TMS11 output.
;  718            1081  4                    !
;  719            1082  4                    SIGNAL (contents$_ignored, 1, %ASCID'LEVEL', contents$_confqual);
;  720            1083  4
;  721            1084  4                    qualifier_value = 6;
;  722            1085  3                    END;
;  723            1086  3
;  724            1087  2            END;
;  725            1088  2
;  726            1089  2        cmdblk [ccntents$g_page_level] = .qualifier_value;
```

```
;  727          1090  2
;  728          1091  2      !
;  729          1092  2      ! /[NO]BOLD = n
;  730          1093  2      !
;  731          1094  2      qualifier_value = -1;
;  732          1095  2
;  733          1096  2      IF  cli$present (%ASCID'BOLD')
;  734          1097  2      THEN
;  735          1098  2
;  736        L 1099  2  %IF DSRPLUS
;  737        U 1100  2  %THEN
;  738        U 1101  2
;  739        U 1102  2          BEGIN
;  740        U 1103  2          qualifier_value = 6;
;  741        U 1104  2          cli$get_value (%ASCID'BOLD', tmp_str);
;  742        U 1105  2
;  743        U 1106  2          IF  NOT call_tparse (tmp_str, numb_state, numb_key, false)
;  744        U 1107  2          THEN
;  745        U 1108  2              SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;  746        U 1109  2
;  747        U 1110  2          END;
;  748        U 1111  2
;  749          1112  2  %ELSE
;  750          1113  2
;  751          1114  2          qualifier_value = 6;
;  752          1115  2
;  753          1116  2  %FI
;  754          1117  2
;  755          1118  2      cmdblk [contents$g_bold] = .qualifier_value;
;  756          1119  2
;  757          1120  2      !
;  758          1121  2      ! /[NO]DOUBLE_SPACE[=n]
;  759          1122  2      !
;  760          1123  2      qualifier_value = -1;
;  761          1124  2
;  762        L 1125  2  %IF DSRPLUS
;  763        U 1126  2  %THEN
;  764        U 1127  2
;  765        U 1128  2      IF cli$present (%ASCID'DOUBLE_SPACE')
;  766        U 1129  2      THEN
;  767        U 1130  2          BEGIN
;  768        U 1131  2          qualifier_value = 1;
;  769        U 1132  2          cli$get_value (%ASCID'DOUBLE_SPACE', tmp_str);
;  770        U 1133  2
;  771        U 1134  2          IF NOT call_tparse (tmp_str, numb_state, numb_key, false)
;  772        U 1135  2          THEN
;  773        U 1136  2              SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;  774        U 1137  2
;  775        U 1138  2          IF .qualifier_value LSS 1
;  776        U 1139  2          THEN
;  777        U 1140  2              SIGNAL_STOP (contents$_badvalue, 1, tmp_str, contents$_valerr);
;  778        U 1141  2
;  779        U 1142  2          END;
;  780        U 1143  2
;  781          1144  2  %FI
;  782          1145  2
;  783          1146  2      cmdblk [contents$g_double_space] = .qualifier_value;
```

```
;  784          1147  2
;  785          1148  2      !
;  786          1149  2      ! /IDENTIFICATION
;  787          1150  2      !
;  788          1151  2      IF  cli$present (%ASCID'IDENTIFICATION')
;  789          1152  2      THEN
;  790          1153  2          SIGNAL (contents$_ident, 2, .cntvrl, .cntvrp);
;  791          1154  2
;  792          1155  2      !
;  793          1156  2      ! /[NO]LOG
;  794          1157  2      !
;  795          1158  2      IF  cli$present (%ASCID'LOG')
;  796          1159  2      THEN
;  797          1160  2          cmdblk [contents$v_log] = true
;  798          1161  2      ELSE
;  799          1162  2          cmdblk [contents$v_log] = false;
;  800          1163  2
;  801          1164  2      !
;  802          1165  2      ! /[NO]OUTPUT = filespec
;  803          1166  2      !
;  804          1167  2      IF  cli$present (%ASCID'OUTPUT')
;  805          1168  2      THEN
;  806          1169  3          BEGIN
;  807          1170  3          cmdblk [contents$v_output] = true;
;  808          1171  3          cli$get_value (%ASCID'OUTPUT', cmdblk [contents$t_output_file]);
;  809          1172  3          END
;  810          1173  2      ELSE
;  811          1174  2          cmdblk [contents$v_output] = false;
;  812          1175  2
;  813          1176  2      !
;  814          1177  2      ! /REQUIRE = filespec
;  815          1178  2      !
;  816          1179  2
;  817          1180  2      IF  cli$present (%ASCID'REQUIRE')
;  818          1181  2      THEN
;  819          1182  3          BEGIN
;  820          1183  3          cmdblk [contents$v_require] = true;
;  821          1184  3          cli$get_value (%ASCID'REQUIRE', cmdblk [contents$t_require_file]);
;  822          1185  3          END
;  823          1186  2      ELSE
;  824          1187  2          cmdblk [contents$v_require] = false;
;  825          1188  2
;  826          1189  2      !
;  827          1190  2      ! /SECTION_NUMBERS = ( AS_INPUT ! ALL ! NONE ! LEVEL = n )
;  828          1191  2      !
;  829          1192  2      cmdblk [contents$v_include_sections] = true;
;  830          1193  2
;  831        L 1194  2  %IF DSRPLUS
;  832        U 1195  2  %THEN
;  833        U 1196  2
;  834        U 1197  2      cmdblk [contents$g_sections] = -1;          ! Display section numbers AS_INPUT
;  835        U 1198  2
;  836        U 1199  2      IF  cli$present (%ASCID'SECTION_NUMBERS')
;  837        U 1200  2      THEN
;  838        U 1201  2          BEGIN
;  839        U 1202  2          cli$get_value (%ASCID'SECTION_NUMBERS', tmp_str);
;  840        U 1203  2
```

```
;  841          U 1204  2          IF  NOT call_tparse (tmp_str, sect_state, sect_key, false)
;  842          U 1205  2          THEN
;  843          U 1206  2              SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;  844          U 1207  2
;  845          U 1208  2          END;
;  846          U 1209  2
;  847            1210  2 %ELSE
;  848            1211  2
;  849            1212  2      cmdblk [contents$g_sections] = 6;           ! Display ALL section numbers
;  850            1213  2
;  851            1214  2      IF cli$present (%ASCID'SECTION_NUMBERS') EQL cli$_negated
;  852            1215  2      THEN
;  853            1216  3          BEGIN
;  854            1217  3          cmdblk [contents$v_include_sections] = false;
;  855            1218  3          cmdblk [contents$g_sections] = 0;
;  856            1219  2          END;
;  857            1220  2
;  858            1221  2 %FI
;  859            1222  2
;  860            1223  2      !
;  861            1224  2      ! /[NO]UNDERLINE = n
;  862            1225  2      !
;  863            1226  2      qualifier_value = -1;
;  864            1227  2
;  865            1228  2      IF  cli$present (%ASCID'UNDERLINE')
;  866            1229  2      THEN
;  867            1230  2
;  868          L 1231  2 %IF DSRPLUS
;  869          U 1232  2 %THEN
;  870          U 1233  2
;  871          U 1234  2          BEGIN
;  872          U 1235  2          qualifier_value = 6;
;  873          U 1236  2          cli$get_value (%ASCID'UNDERLINE', tmp_str);
;  874          U 1237  2
;  875          U 1238  2          IF  NOT call_tparse (tmp_str, numb_state, numb_key, false)
;  876          U 1239  2          THEN
;  877          U 1240  2              SIGNAL_STOP (contents$_badvalue, 1, tmp_str);
;  878          U 1241  2
;  879          U 1242  2          IF  .qualifier_value GTR 6
;  880          U 1243  2          THEN
;  881          U 1244  2              SIGNAL_STOP (contents$_badvalue, 1, tmp_str, contents$_valerr);
;  882          U 1245  2
;  883          U 1246  2          END;
;  884          U 1247  2
;  885            1248  2 %ELSE
;  886            1249  2
;  887            1250  2          qualifier_value = 6;
;  888            1251  2
;  889            1252  2 %FI
;  890            1253  2
;  891            1254  2      cmdblk [contents$g_underline] = .qualifier_value;
;  892            1255  2
;  893            1256  2      !
;  894            1257  2      ! Process input file(s) and positional qualifiers.
;  895            1258  2      !
;  896            1259  2      WHILE (status = cli$get_value (%ASCID'INPUT', cmdblk [contents$t_input_file])) DO
;  897            1260  3          BEGIN
```

```
;  898          1261  3          toc ();                                  ! Process input file.
;  899          1262  3
;  900          1263  3          !
;  901          1264  3          ! Generate Figures, Tables, and Examples and finish up if
;  902          1265  3          ! the current input file is not concatenated to the next.
;  903          1266  3          !
;  904          1267  3          IF .status NEQ cli$_concat THEN tocfin ();
;  905          1268  2          END;
;  906          1269  2
;  907          1270  2      RETURN (.termination_status OR sts$m_inhib_msg);
;  908          1271  1      END;


                                                        .TITLE  CNTVMS CNTVMS - CONTENTS VMS command Line Inter
;                                                                       face
                                                        .IDENT  \V04-000\

                                                        .PSECT  _LIB$KEY1$,NOWRT,  SHR,  PIC,1

                                                00000 ;TPA$KEYSTO
                                                      U.9:      .BLKB   0
                                 47  4E  49  4E  4E  55  52  00000 ;TPA$KEYST
                                                      U.11:     .ASCII  \RUNNING\                               ;
                                                FF    00007             .BYTE   -1                              ;
                                                      00008 ;TPA$KEYSTO
                                                      U.18:     .BLKB   0
                             47  4E  49  4E  4E  55  52  4F  4E  00008 ;TPA$KEYST
                                                      U.20:     .ASCII  \NORUNNING\                             ;
                                                FF    00011             .BYTE   -1                              ;
                                                      00012 ;TPA$KEYSTO
                                                      U.26:     .BLKB   0
                                     4C  45  56  45  4C  00012 ;TPA$KEYST
                                                      U.28:     .ASCII  \LEVEL\                                 ;
                                                FF    00017             .BYTE   -1                              ;
                                                FF    00018 ;TPA$KEYFILL
                                                      U.30:     .BYTE   -1                                      ;

                                                                .PSECT  _LIB$STATE$,NOWRT,  SHR,  PIC,1

                                                00000 NUMB_STATE::
                                                                .BLKB   0
                                           41F3 00000 ;TPA$TYPE
                                                      U.2:      .WORD   16883                                   ;
                                      00000000* 00002 ;TPA$ADDR
                                                      U.3:      .LONG   <<QUALIFIER_VALUE-U.3>-4>               ;
                                           15F7 00006 ;TPA$TYPE
                                                      U.4:      .WORD   5623                                    ;
                                           FFFF 00008 ;TPA$TARGET
                                                      U.5:      .WORD   -1                                      ;
                                           15F7 0000A ;TPA$TYPE
                                                      U.6:      .WORD   5623                                    ;
                                           FFFF 0000C ;TPA$TARGET
                                                      U.7:      .WORD   -1                                      ;
                                                0000E             .BLKB   2
                                                00010 PAGE_STATE::
                                                                .BLKB   0
                                           9300 00010 ;TPA$TYPE
```

```
                                U.12:   .WORD   -27904                          ;
      01  00012                ;TPA$FLAGS2
                                U.13:   .BYTE   1                               ;
00000000  00013                ;TPA$PARAM
                                U.14:   .LONG   0                               ;
00000000V 00017                ;TPA$ACTION
                                U.15:   .LONG   <<ENTER_PAGE-U.15>-4>           ;
    0000* 0001B                ;TPA$TARGET
                                U.17:   .WORD   <<U.16-U.17>-2>                 ;
    9301  0001D                ;TPA$TYPE
                                U.21:   .WORD   -27903                          ;
      01  0001F                ;TPA$FLAGS2
                                U.22:   .BYTE   1                               ;
00000001  00020                ;TPA$PARAM
                                U.23:   .LONG   1                               ;
00000000V 00024                ;TPA$ACTION
                                U.24:   .LONG   <<ENTER_PAGE-U.24>-4>           ;
    0000* 00028                ;TPA$TARGET
                                U.25:   .WORD   <<U.16-U.25>-2>                 ;
    0502  0002A                ;TPA$TYPE
                                U.29:   .WORD   1282                            ;
    003D  0002C                ;TPA$TYPE
                                U.31:   .WORD   61                              ;
    043A  0002E                ;TPA$TYPE
                                U.32:   .WORD   1082                            ;
    55F3  00030                ;TPA$TYPE
                                U.33:   .WORD   22003                           ;
00000000* 00032                ;TPA$ADDR
                                U.34:   .LONG   <<QUALIFIER_VALUE-U.34>-4>      ;
    0000* 00036                ;TPA$TARGET
                                U.35:   .WORD   <<U.16-U.35>-2>                 ;
          00038                ;PAGE_END
                                U.16:   .BLKB   0
    15F7  00038                ;TPA$TYPE
                                U.36:   .WORD   5623                            ;
    FFFF  0003A                ;TPA$TARGET
                                U.37:   .WORD   -1                              ;

                                        .PSECT  _LIB$KEY0$,NOWRT,  SHR,  PIC,1

          00000 NUMB_KEY::
                                        .BLKB   0
          00000                 ;TPA$KEY0
                                U.1:    .BLKB   0
          00000 PAGE_KEY::
                                        .BLKB   0
          00000                 ;TPA$KEY0
                                U.8:    .BLKB   0
    0000* 00000                 ;TPA$KEY
                                U.10:   .WORD   <U.9-U.8>                       ;
    0000* 00002                 ;TPA$KEY
                                U.19:   .WORD   <U.18-U.8>                      ;
    0000* 00004                 ;TPA$KEY
                                U.27:   .WORD   <U.26-U.8>                      ;

                                        .PSECT  $PLIT$,NOWRT,NOEXE,2
```

```
                                00  00  00  45  4E  49  4C  24  00000 P.AAB:   .ASCII  \$LINE\<0><0><0>
                                                    010E0005  00008 P.AAA:   .LONG   17694725
                                                    00000000' 0000C         .ADDRESS P.AAB
00  52  45  44  41  45  48  5F  54  53  45  50  45  45  44  00010 P.AAD:   .ASCII  \DEEPEST_HEADER\<0><0>
                                                        00  0001F
                                                    010E000E  00020 P.AAC:   .LONG   17694734
                                                    00000000' 00024         .ADDRESS P.AAD
00  52  45  44  41  45  48  5F  54  53  45  50  45  45  44  00028 P.AAF:   .ASCII  \DEEPEST_HEADER\<0><0>
                                                        00  00037
                                                    010E000E  00038 P.AAE:   .LONG   17694734
                                                    00000000' 0003C         .ADDRESS P.AAF
                                00  00  54  4E  45  44  4E  49  00040 P.AAH:   .ASCII  \INDENT\<0><0>
                                                    010E0006  00048 P.AAG:   .LONG   17694726
                                                    00000000' 0004C         .ADDRESS P.AAH
            53  52  45  42  4D  55  4E  5F  45  47  41  50  00050 P.AAJ:   .ASCII  \PAGE_NUMBERS\
                                                    010E000C  0005C P.AAI:   .LONG   17694732
                                                    00000000' 00060         .ADDRESS P.AAJ
            53  52  45  42  4D  55  4E  5F  45  47  41  50  00064 P.AAL:   .ASCII  \PAGE_NUMBERS\
                                                    010E000C  00070 P.AAK:   .LONG   17694732
                                                    00000000' 00074         .ADDRESS P.AAL
                                00  00  00  4C  45  56  45  4C  00078 P.AAN:   .ASCII  \LEVEL\<0><0><0>
                                                    010E0005  00080 P.AAM:   .LONG   17694725
                                                    00000000' 00084         .ADDRESS P.AAN
                                                44  4C  4F  42  00088 P.AAP:   .ASCII  \BOLD\
                                                    010E0004  0008C P.AAO:   .LONG   17694724
                                                    00000000' 00090         .ADDRESS P.AAP
00  4E  4F  49  54  41  43  49  46  49  54  4E  45  44  49  00094 P.AAR:   .ASCII  \IDENTIFICATION\<0><0>
                                                        00  000A3
                                                    010E000E  000A4 P.AAQ:   .LONG   17694734
                                                    00000000' 000A8         .ADDRESS P.AAR
                                            00  47  4F  4C  000AC P.AAT:   .ASCII  \LOG\<0>
                                                    010E0003  000B0 P.AAS:   .LONG   17694723
                                                    00000000' 000B4         .ADDRESS P.AAT
                                00  00  54  55  50  54  55  4F  000B8 P.AAV:   .ASCII  \OUTPUT\<0><0>
                                                    010E0006  000C0 P.AAU:   .LONG   17694726
                                                    00000000' 000C4         .ADDRESS P.AAV
                                00  00  54  55  50  54  55  4F  000C8 P.AAX:   .ASCII  \OUTPUT\<0><0>
                                                    010E0006  000D0 P.AAW:   .LONG   17694726
                                                    00000000' 000D4         .ADDRESS P.AAX
                                00  45  52  49  55  51  45  52  000D8 P.AAZ:   .ASCII  \REQUIRE\<0>
                                                    010E0007  000E0 P.AAY:   .LONG   17694727
                                                    00000000' 000E4         .ADDRESS P.AAZ
                                00  45  52  49  55  51  45  52  000E8 P.ABB:   .ASCII  \REQUIRE\<0>
                                                    010E0007  000F0 P.ABA:   .LONG   17694727
                                                    00000000' 000F4         .ADDRESS P.ABB
53  52  45  42  4D  55  4E  5F  4E  4F  49  54  43  45  53  000F8 P.ABD:   .ASCII  \SECTION_NUMBERS\<0>
                                                        00  00107
                                                    010E000F  00108 P.ABC:   .LONG   17694735
                                                    00000000' 0010C         .ADDRESS P.ABD
            00  00  00  45  4E  49  4C  52  45  44  4E  55  00110 P.ABF:   .ASCII  \UNDERLINE\<0><0><0>
                                                    010E0009  0011C P.ABE:   .LONG   17694729
                                                    00000000' 00120         .ADDRESS P.ABF
                                00  00  00  54  55  50  4E  49  00124 P.ABH:   .ASCII  \INPUT\<0><0><0>
                                                    010E0005  0012C P.ABG:   .LONG   17694725
                                                    00000000' 00130         .ADDRESS P.ABH

                                                                            .PSECT  $OWN$,NOEXE,2
```

```
                                00000 QUALIFIER_VALUE:
                                      .BLKB   4
                           0000 00004 TMP_STR:.WORD   0                                                    ;
                          02   0E 00006        .BYTE   14, 2                                               ;
                    00000000 00008        .LONG   0                                                       ;
                    00000001 0000C TERMINATION_STATUS:
                                      .LONG   1                                                           ;

                                      .EXTRN  DSRTOC$_BADVALUE
                                      .EXTRN  DSRTOC$_OPENIN, DSRTOC$_OPENOUT
                                      .EXTRN  DSRTOC$_VALERR, DSRTOC$_CAPTIONS
                                      .EXTRN  DSRTOC$_CLOSEQUOT
                                      .EXTRN  DSRTOC$_CONFQUAL
                                      .EXTRN  DSRTOC$_CTRLCHAR
                                      .EXTRN  DSRTOC$_EMPTYIN
                                      .EXTRN  DSRTOC$_IGNORED
                                      .EXTRN  DSRTOC$_INVINPUT
                                      .EXTRN  DSRTOC$_INVRECORD
                                      .EXTRN  DSRTOC$_OVERSTRK
                                      .EXTRN  DSRTOC$_COMPLETE
                                      .EXTRN  DSRTOC$_CREATED
                                      .EXTRN  DSRTOC$_IGNORENEW
                                      .EXTRN  DSRTOC$_IGNOREOLD
                                      .EXTRN  DSRTOC$_PROCFILE
                                      .EXTRN  DSRTOC$_TEXTD, DSRTOC$_TMS11
                                      .EXTRN  DSRTOC$_IDENT, CLI$_CONCAT
                                      .EXTRN  CLI$_PRESENT, CLI$_DEFAULTED
                                      .EXTRN  CLI$_NEGATED, CLI$_ABSENT
                                      .EXTRN  CMDBLK, CNTVRL, CNTVRP
                                      .EXTRN  CLI$PRESENT, CLI$GET_VALUE
                                      .EXTRN  LIB$TPARSE, TOC
                                      .EXTRN  TOCFIN

                                      .PSECT  $CODE$,NOWRT,2

                                0FFC 00000        .ENTRY  CNTCLI, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,-      ; 0692
                                                          R11
    5B 00000000V   EF   9E 00002        MOVAB   CALL_TPARSE, R11                                          ;
    5A 00000000'   EF   9E 00009        MOVAB   NUMB_KEY, R10                                             ;
    59 00000000G   00   9E 00010        MOVAB   LIB$STOP, R9                                              ;
    58 00000000G   8F   D0 00017        MOVL    #DSRTOC$_BADVALUE, R8                                     ;
    57 00000000G   00   9E 0001E        MOVAB   CLI$GET_VALUE, R7                                         ;
    56 00000000G   00   9E 00025        MOVAB   CLI$PRESENT, R6                                           ;
    55 00000000'   EF   9E 0002C        MOVAB   P.AAA, R5                                                 ;
    54 00000000'   EF   9E 00033        MOVAB   QUALIFIER_VALUE, R4                                       ;
    53 00000000G   EF   9E 0003A        MOVAB   CMDBLK, R3                                                ;
    6D      020A   CF   DE 00041        MOVAL   19$, (FP)                                                 ; 0724
3C  A3 020E0000    8F   D0 00046        MOVL    #34471936, $STR$DESC                                      ; 0732
              40   A3   D4 0004E        CLRL    $STR$DESC+4
44  A3 020E0000    8F   D0 00051        MOVL    #34471936, $STR$DESC                                      ; 0733
              48   A3   D4 00059        CLRL    $STR$DESC+4
4C  A3 020E0000    8F   D0 0005C        MOVL    #34471936, $STR$DESC                                      ; 0734
              50   A3   D4 00064        CLRL    $STR$DESC+4
54  A3 020E0000    8F   D0 00067        MOVL    #34471936, $STR$DESC                                      ; 0735
              58   A3   D4 0006F        CLRL    $STR$DESC+4
              54   A3   9F 00072        PUSHAB  CMDBLK+84                                                 ; 0740
```

```
                                55  DD 00075          PUSHL   R5
                             67 02  FB 00077          CALLS   #2, CLI$GET_VALUE
                             63 02  8A 0007A          BICB2   #2, CMDBLK                              0752
                             64     8F  9A 0007D      MOVZBL  #70, QUALIFIER_VALUE                    0781
                          18 A3 64  D0 00081          MOVL    QUALIFIER_VALUE, CMDBLK+24              0820
                             64 06  D0 00085          MOVL    #6, QUALIFIER_VALUE                     0832
                                18  A5 9F 00088      PUSHAB  P.AAC                                   0834
                             66 01  FB 0008B          CALLS   #1, CLI$PRESENT
                             38 50  E9 0008E          BLBC    R0, 2$
                                04  A4 9F 00091      PUSHAB  TMP_STR                                 0837
                                30  A5 9F 00094      PUSHAB  P.AAE
                             67 02  FB 00097          CALLS   #2, CLI$GET_VALUE
                                7E  D4 0009A          CLRL    -(SP)                                   0839
                                5A  DD 0009C          PUSHL   R10
                     00000000'  EF  9F 0009E          PUSHAB  NUMB_STATE
                            04  A4 9F 000A4          PUSHAB  TMP_STR
                             6B 04  FB 000A7          CALLS   #4, CALL_TPARSE
                             0A 50  E8 000AA          BLBS    R0, 1$
                                04  A4 9F 000AD      PUSHAB  TMP_STR                                 0841
                                01  DD 000B0          PUSHL   #1
                                58  DD 000B2          PUSHL   R8
                             69 03  FB 000B4          CALLS   #3, LIB$STOP
                             06 64  D1 000B7 1$:      CMPL    QUALIFIER_VALUE, #6                     0843
                                10  15 000BA          BLEQ    2$
                     00000000G  8F  DD 000BC          PUSHL   #DSRTOC$_VALERR                         0845
                            04  A4 9F 000C2          PUSHAB  TMP_STR
                                01  DD 000C5          PUSHL   #1
                                58  DD 000C7          PUSHL   R8
                             69 04  FB 000C9          CALLS   #4, LIB$STOP
                          04 A3 64  D0 000CC 2$:      MOVL    QUALIFIER_VALUE, CMDBLK+4               0849
                          20 A3 08  D0 000D0          MOVL    #8, INDENTS                             0862
                          24 A3 08  D0 000D4          MOVL    #8, INDENTS+4                           0863
                          28 A3 02  7D 000D8          MOVQ    #2, INDENTS+8                           0864
                                30  A3 7C 000DC      CLRQ    INDENTS+16                              0877
                                38  A3 D4 000DF      CLRL    INDENTS+24                              0879
                                40  A5 9F 000E2      PUSHAB  P.AAG                                   0883
                             66 01  FB 000E5          CALLS   #1, CLI$PRESENT
                             10 50  E9 000E8          BLBC    R0, 3$
                          2C A3 02  D0 000EB          MOVL    #2, INDENTS+12                          0962
                          30 A3 02  D0 000EF          MOVL    #2, INDENTS+16                          0963
                          34 A3 02  D0 000F3          MOVL    #2, INDENTS+20                          0964
                          38 A3 02  D0 000F7          MOVL    #2, INDENTS+24                          0965
                          02 A3 2E  90 000FB 3$:      MOVB    #46, CMDBLK+2                           0981
                             63 08  88 000FF          BISB2   #8, CMDBLK                              1064
                             64 06  D0 00102          MOVL    #6, QUALIFIER_VALUE                     1065
                                54  A5 9F 00105      PUSHAB  P.AAI                                   1067
                             66 01  FB 00108          CALLS   #1, CLI$PRESENT
                             4F 50  E9 0010B          BLBC    R0, 6$
                                04  A4 9F 0010E 4$:  PUSHAB  TMP_STR                                 1071
                                68  A5 9F 00111      PUSHAB  P.AAK
                             67 02  FB 00114          CALLS   #2, CLI$GET_VALUE
                             1F 50  E9 00117          BLBC    R0, 5$
                                7E  D4 0011A          CLRL    -(SP)                                   1072
                                5A  DD 0011C          PUSHL   R10
                     00000000'  EF  9F 0011E          PUSHAB  PAGE_STATE
                            04  A4 9F 00124          PUSHAB  TMP_STR
                             6B 04  FB 00127          CALLS   #4, CALL_TPARSE
```

```
                      E1     50 E8 0012A        BLBS    R0, 4$                          ;
                  04  A4     9F 0012D           PUSHAB  TMP_STR                         ; 1074
                      01     DD 00130           PUSHL   #1                              ;
                      58     DD 00132           PUSHL   R8                              ;
                      69     03 FB 00134        CALLS   #3, LIB$STOP                    ;
                      D5     11 00137           BRB     4$                              ; 1072
         20           63     01 E1 00139 5$:    BBC     #1, CMDBLK, 6$                  ; 1076
                      06     64 D1 0013D        CMPL    QUALIFIER_VALUE, #6             ;
                      1B     13 00140           BEQL    6$                              ;
           00000000G  8F     DD 00142           PUSHL   #DSRTOC$_CONFQUAL               ; 1082
                  78  A5     9F 00148           PUSHAB  P.AAM                           ;
                      01     DD 0014B           PUSHL   #1                              ;
           00000000G  8F     DD 0014D           PUSHL   #DSRTOC$_IGNORED                ;
  00000000G  00              04 FB 00153        CALLS   #4, LIB$SIGNAL                  ;
                      64     06 D0 0015A        MOVL    #6, QUALIFIER_VALUE             ; 1084
         08           A3     64 D0 0015D 6$:    MOVL    QUALIFIER_VALUE, CMDBLK+8       ; 1089
                      64     01 CE 00161        MNEGL   #1, QUALIFIER_VALUE             ; 1094
                0084  C5     9F 00164           PUSHAB  P.AAO                           ; 1096
                      66     01 FB 00168        CALLS   #1, CLI$PRESENT                 ;
                      03     50 E9 0016B        BLBC    R0, 7$                          ;
                      64     06 D0 0016E        MOVL    #6, QUALIFIER_VALUE             ; 1114
         10           A3     64 D0 00171 7$:    MOVL    QUALIFIER_VALUE, CMDBLK+16      ; 1118
                      64     01 CE 00175        MNEGL   #1, QUALIFIER_VALUE             ; 1123
         1C           A3     64 D0 00178        MOVL    QUALIFIER_VALUE, CMDBLK+28      ; 1146
                009C  C5     9F 0017C           PUSHAB  P.AAQ                           ; 1151
                      66     01 FB 00180        CALLS   #1, CLI$PRESENT                 ;
                      1B     50 E9 00183        BLBC    R0, 8$                          ;
           00000000G  EF     DD 00186           PUSHL   CNTVRP                          ; 1153
           00000000G  EF     DD 0018C           PUSHL   CNTVRL                          ;
                      02     DD 00192           PUSHL   #2                              ;
           00000000G  8F     DD 00194           PUSHL   #DSRTOC$_IDENT                  ;
  00000000G  00              04 FB 0019A        CALLS   #4, LIB$SIGNAL                  ;
                00A8  C5     9F 001A1 8$:       PUSHAB  P.AAS                           ; 1158
                      66     01 FB 001A5        CALLS   #1, CLI$PRESENT                 ;
                      05     50 E9 001A8        BLBC    R0, 9$                          ;
                      63     20 88 001AB        BISB2   #32, CMDBLK                     ; 1160
                      03     11 001AE           BRB     10$                             ;
                      63     20 8A 001B0 9$:    BICB2   #32, CMDBLK                     ; 1162
                00B8  C5     9F 001B3 10$:      PUSHAB  P.AAU                           ; 1167
                      66     01 FB 001B7        CALLS   #1, CLI$PRESENT                 ;
                      0F     50 E9 001BA        BLBC    R0, 11$                         ;
                      63     01 88 001BD        BISB2   #1, CMDBLK                      ; 1170
                  44  A3     9F 001C0           PUSHAB  CMDBLK+68                       ; 1171
                00C8  C5     9F 001C3           PUSHAB  P.AAW                           ;
                      67     02 FB 001C7        CALLS   #2, CLI$GET_VALUE               ;
                      03     11 001CA           BRB     12$                             ; 1167
                      63     01 8A 001CC 11$:   BICB2   #1, CMDBLK                      ; 1174
                00D8  C5     9F 001CF 12$:      PUSHAB  P.AAY                           ; 1180
                      66     01 FB 001D3        CALLS   #1, CLI$PRESENT                 ;
                      0F     50 E9 001D6        BLBC    R0, 13$                         ;
                      63     04 88 001D9        BISB2   #4, CMDBLK                      ; 1183
                  4C  A3     9F 001DC           PUSHAB  CMDBLK+76                       ; 1184
                00E8  C5     9F 001DF           PUSHAB  P.ABA                           ;
                      67     02 FB 001E3        CALLS   #2, CLI$GET_VALUE               ;
                      03     11 001E6           BRB     14$                             ; 1180
                      63     04 8A 001E8 13$:   BICB2   #4, CMDBLK                      ; 1187
                      63     10 88 001EB 14$:   BISB2   #16, CMDBLK                     ; 1192
```

```
                            14   A3                  06  D0 001EE          MOVL    #6, CMDBLK+20                           ; 1212
                                           0100     C5  9F 001F2          PUSHAB  P.ABC                                   ; 1214
                                 66                 01  FB 001F6          CALLS   #1, CLI$PRESENT                         ;
                     00000000G   8F                 50  D1 001F9          CMPL    R0, #CLI$_NEGATED                       ;
                                                    06  12 00200          BNEQ    15$                                     ;
                                 63                 10  8A 00202          BICB2   #16, CMDBLK                             ; 1217
                                                14  A3  D4 00205          CLRL    CMDBLK+20                               ; 1218
                                 64                 01  CE 00208 15$:      MNEGL   #1, QUALIFIER_VALUE                     ; 1226
                                           0114     C5  9F 0020B          PUSHAB  P.ABE                                   ; 1228
                                 66                 01  FB 0020F          CALLS   #1, CLI$PRESENT                         ;
                                 03                 50  E9 00212          BLBC    R0, 16$                                 ;
                                 64                 06  D0 00215          MOVL    #6, QUALIFIER_VALUE                     ; 1250
                            0C   A3                 64  D0 00218 16$:      MOVL    QUALIFIER_VALUE, CMDBLK+12              ; 1254
                                                3C  A3  9F 0021C 17$:      PUSHAB  CMDBLK+60                               ; 1259
                                           0124     C5  9F 0021F          PUSHAB  P.ABG                                   ;
                                 67                 02  FB 00223          CALLS   #2, CLI$GET_VALUE                       ;
                                 52                 50  D0 00226          MOVL    R0, STATUS                              ;
                                 19                 52  E9 00229          BLBC    STATUS, 18$                             ;
                     00000000G   EF                 00  FB 0022C          CALLS   #0, TOC                                 ; 1261
                     00000000G   8F                 52  D1 00233          CMPL    STATUS, #CLI$_CONCAT                    ; 1267
                                                    E0  13 0023A          BEQL    17$                                     ;
                     00000000G   EF                 00  FB 0023C          CALLS   #0, TOCFIN                              ;
                                                    D7  11 00243          BRB     17$                                     ; 1259
                 50      0C   A4 10000000   8F     C9  00245 18$:      BISL3   #268435456, TERMINATION_STATUS, R0          ; 1270
                                                        04  0024E          RET                                             ; 1271
                                                  0000  0024F 19$:      .WORD   Save nothing                            ; 0724
                                                7E  D4  00251          CLRL    -(SP)                                   ;
                                                5E  DD  00253          PUSHL   SP                                      ;
                                 7E         04  AC  7D  00255          MOVQ    4(AP), -(SP)                            ;
                     00000000V   EF                 03  FB 00259          CALLS   #3, CONDITION_HANDLER                   ;
                                                        04  00260          RET                                             ;

; Routine Size:  609 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  910     1272  1 %SBTTL 'CALL_TPARSE -- Invoke TPARSE to process qualifier values'
;  911     1273  1 ROUTINE call_tparse (string : REF $str_descriptor (), state_tab, key_tab, blanks) =
;  912     1274  1 !++
;  913     1275  1 !
;  914     1276  1 ! FUNCTIONAL DESCRIPTION:
;  915     1277  1 !
;  916     1278  1 !       This routine calls TPARSE to parse the given string with
;  917     1279  1 !       the given state and key tables.
;  918     1280  1 !
;  919     1281  1 ! FORMAL PARAMETERS:
;  920     1282  1 !
;  921     1283  1 !       string    - Address of a string descriptor of string to parse
;  922     1284  1 !       state_tab - Address of TPARSE state tables
;  923     1285  1 !       key_tab   - Address of TPARSE key tables
;  924     1286  1 !       blanks    - true if blanks are to be processed explicitly
;  925     1287  1 !
;  926     1288  1 ! IMPLICIT INPUTS:
;  927     1289  1 !
;  928     1290  1 !       None
;  929     1291  1 !
;  930     1292  1 ! IMPLICIT OUTPUTS:
;  931     1293  1 !
;  932     1294  1 !       None
;  933     1295  1 !
;  934     1296  1 ! ROUTINE VALUE:
;  935     1297  1 ! COMPLETION CODES:
;  936     1298  1 !
;  937     1299  1 !       Returns completion code of lib$tparse.
;  938     1300  1 !
;  939     1301  1 ! SIDE EFFECTS:
;  940     1302  1 !
;  941     1303  1 !       None
;  942     1304  1 !
;  943     1305  1 !--
;  944     1306  1
;  945     1307  2     BEGIN
;  946     1308  2
;  947     1309  2     LOCAL
;  948     1310  2         tparse_block : BLOCK [tpa$k_length0, BYTE],
;  949     1311  2         status;
;  950     1312  2
;  951     1313  2     !
;  952     1314  2     ! Initialize the TPARSE parameter block.
;  953     1315  2     !
;  954     1316  2     tparse_block [tpa$l_count]     = tpa$k_count0;
;  955     1317  2     tparse_block [ipa$l_options]   = tpa$m_abbrev;
;  956     1318  2     tparse_block [tpa$v_blanks]    = .blanks;
;  957     1319  2     tparse_block [tpa$l_stringcnt] = .string [str$h_length];
;  958     1320  2     tparse_block [tpa$l_stringptr] = .string [str$a_pointer];
;  959     1321  2     tparse_block [tpa$l_tokencnt]  = 0;
;  960     1322  2     tparse_block [tpa$l_tokenptr]  = 0;
;  961     1323  2     tparse_block [tpa$l_number]    = 0;
;  962     1324  2     tparse_block [tpa$l_param]     = 0;
;  963     1325  2
;  964     1326  2     !
;  965     1327  2     ! Parse the string and return parse status.
;  966     1328  2     !
```

```
;  967          1329  2     RETURN lib$tparse (tparse_block, .state_tab, .key_tab);
;  968          1330  1     END;



                                                0000 00000 CALL_TPARSE:
                                                                .WORD   Save nothing                                    ; 1273
                                                     5E         20  C2 00002          SUBL2   #32, SP                         ;
                                                                08  DD 00005          PUSHL   #8                              ; 1316
                                                04   AE         02  D0 00007          MOVL    #2, TPARSE_BLOCK+4              ; 1317
    04   AE                 01                              00  10  AC F0 0000B      INSV    BLANKS, #0, #1, TPARSE_BLOCK+4  ; 1318
                                                                50  04  AC D0 00012      MOVL    STRING, R0                      ; 1319
                                                08   AE         60  3C 00016          MOVZWL  (R0), TPARSE_BLOCK+8            ;
                                                0C   AE         04  A0 D0 0001A       MOVL    4(R0), TPARSE_BLOCK+12          ; 1320
                                                                10  AE 7C 0001F       CLRQ    TPARSE_BLOCK+16                 ; 1321
                                                                1C  AE 7C 00022       CLRQ    TPARSE_BLOCK+28                 ; 1323
                                                     7E         08  AC 7D 00025       MOVQ    STATE_TAB, -(SP)                ; 1329
                                                                08  AE 9F 00029       PUSHAB  TPARSE_BLOCK                    ;
                                           00000000G 00                 03  FB 0002C  CALLS   #3, LIB$TPARSE                  ;
                                                                        04 00033      RET                                     ; 1330

; Routine Size:  52 bytes,    Routine Base:  $CODE$ + 0261
```

```
;  970          1331  1 %SBTTL 'ENTER_PAGE -- Action routine - enter page display type'
;  971          1332  1 ROUTINE enter_page =
;  972          1333  1 !++
;  973          1334  1 !
;  974          1335  1 ! FUNCTIONAL DESCRIPTION:
;  975          1336  1 !
;  976          1337  1 !       This routine is called as an action routine by TPARSE.
;  977          1338  1 !       It sets contents$v_standard page from tpa$l_param.
;  978          1339  1 !
;  979          1340  1 ! FORMAL PARAMETERS:
;  980          1341  1 !
;  981          1342  1 !       AP [tpa$l_param] - value to set contents$v_standard_page
;  982          1343  1 !
;  983          1344  1 ! IMPLICIT INPUTS:
;  984          1345  1 !
;  985          1346  1 !       None
;  986          1347  1 !
;  987          1348  1 ! IMPLICIT OUTPUTS:
;  988          1349  1 !
;  989          1350  1 !       cmdblk [contents$v_standard_page]
;  990          1351  1 !
;  991          1352  1 ! ROUTINE VALUE:
;  992          1353  1 ! COMPLETION CODES:
;  993          1354  1 !
;  994          1355  1 !       ss$_normal
;  995          1356  1 !
;  996          1357  1 ! SIDE EFFECTS:
;  997          1358  1 !
;  998          1359  1 !       None
;  999          1360  1 !
; 1000          1361  1 !--
; 1001          1362  1
; 1002          1363  2     BEGIN
; 1003          1364  2
; 1004          1365  2     BUILTIN
; 1005          1366  2         AP;
; 1006          1367  2
; 1007          1368  2     MAP
; 1008          1369  2         AP : REF BLOCK [, BYTE];
; 1009          1370  2
; 1010          1371  2     cmdblk [contents$v_standard_page] = .AP [tpa$l_param];
; 1011          1372  2
; 1012          1373  2     RETURN ss$_normal;
; 1013          1374  1     END;


                                                0000 00000 ENTER_PAGE:
                                                           .WORD   Save nothing                                    ; 1332
00000000G  EF           01          03      20  AC  F0 00002      INSV    32(AP), #3, #1, CMDBLK                  ; 1371
                                    50          01  D0 0000C      MOVL    #1, R0                                  ; 1373
                                                    04 0000F      RET                                             ; 1374

; Routine Size:  16 bytes,    Routine Base:  $CODE$ + 0295
```

```
;  1015          L 1375  1 %IF DSRPLUS
;  1016          U 1376  1 %THEN
;  1017          U 1377  1
;  1018          U 1378  1 %SBTTL 'ENTER_FORMAT -- Action routine - Enter output format'
;  1019          U 1379  1 ROUTINE enter_format =
;  1020          U 1380  1 !++
;  1021          U 1381  1 !
;  1022          U 1382  1 ! FUNCTIONAL DESCRIPTION:
;  1023          U 1383  1 !
;  1024          U 1384  1 !       This routine is called as an action routine by TPARSE.
;  1025          U 1385  1 !
;  1026          U 1386  1 !       It sets the value of contents$v_tms11 based on the value of
;  1027          U 1387  1 !       the parameter passed by TPARSE.
;  1028          U 1388  1 !
;  1029          U 1389  1 ! FORMAL PARAMETERS:
;  1030          U 1390  1 !
;  1031          U 1391  1 !       AP [tpa$l_param]        - true if generating TMS output, false otherwise
;  1032          U 1392  1 !
;  1033          U 1393  1 ! IMPLICIT INPUTS:
;  1034          U 1394  1 !
;  1035          U 1395  1 !       None
;  1036          U 1396  1 !
;  1037          U 1397  1 ! IMPLICIT OUTPUTS:
;  1038          U 1398  1 !
;  1039          U 1399  1 !       cmdblk [contents$v_tms11] - set to the value of AP [tpa$l_param]
;  1040          U 1400  1 !
;  1041          U 1401  1 ! ROUTINE VALUE:
;  1042          U 1402  1 ! COMPLETION CODES:
;  1043          U 1403  1 !
;  1044          U 1404  1 !       ss$_normal
;  1045          U 1405  1 !
;  1046          U 1406  1 ! SIDE EFFECTS:
;  1047          U 1407  1 !
;  1048          U 1408  1 !       None
;  1049          U 1409  1 !--
;  1050          U 1410  1     BEGIN
;  1051          U 1411  1     BUILTIN
;  1052          U 1412  1         AP;
;  1053          U 1413  1
;  1054          U 1414  1     MAP
;  1055          U 1415  1         AP : REF BLOCK [, BYTE];
;  1056          U 1416  1
;  1057          U 1417  1     cmdblk [contents$v_tms11] = .AP [tpa$l_param];
;  1058          U 1418  1     RETURN ss$_normal;
;  1059          U 1419  1     END;
;  1060          U 1420  1
;  1061            1421  1 %FI
```

```
;  1063        L 1422  1 %IF DSRPLUS
;  1064        U 1423  1 %THEN
;  1065        U 1424  1
;  1066        U 1425  1 %SBTTL 'ENTER_SECT -- Action routine - enter section number display level'
;  1067        U 1426  1 ROUTINE enter_sect =
;  1068        U 1427  1 !++
;  1069        U 1428  1 !
;  1070        U 1429  1 ! FUNCTIONAL DESCRIPTION:
;  1071        U 1430  1 !
;  1072        U 1431  1 !       This routine is called as an action routine by TPARSE.
;  1073        U 1432  1 !       It sets the level of section number to be displayed based on the
;  1074        U 1433  1 !       value passed by TPARSE.
;  1075        U 1434  1 !
;  1076        U 1435  1 ! FORMAL PARAMETERS:
;  1077        U 1436  1 !
;  1078        U 1437  1 !       AP [tpa$l_param]        = -1 - display section numbers AS_INPUT
;  1079        U 1438  1 !                                 =  6 - display ALL section numbers
;  1080        U 1439  1 !                                 = -2 - don't display section numbers
;  1081        U 1440  1 !                                 =  0 - user specified deepest level to display
;  1082        U 1441  1 !       AP [tpa$l_number]       =      user specified level
;  1083        U 1442  1 !
;  1084        U 1443  1 ! IMPLICIT INPUTS:
;  1085        U 1444  1 !
;  1086        U 1445  1 !       None
;  1087        U 1446  1 !
;  1088        U 1447  1 ! IMPLICIT OUTPUTS:
;  1089        U 1448  1 !
;  1090        U 1449  1 !       cmdblk [contents$v_include_sections]    = true if sections are to be
;  1091        U 1450  1 !                                                 displayed
;  1092        U 1451  1 !       cmdblk [contents$g_sections]            = deepest level to display
;  1093        U 1452  1 !
;  1094        U 1453  1 ! ROUTINE VALUE:
;  1095        U 1454  1 ! COMPLETION CODES:
;  1096        U 1455  1 !
;  1097        U 1456  1 !       ss$_normal
;  1098        U 1457  1 !
;  1099        U 1458  1 ! SIDE EFFECTS:
;  1100        U 1459  1 !
;  1101        U 1460  1 !       None
;  1102        U 1461  1 !
;  1103        U 1462  1 !--
;  1104        U 1463  1
;  1105        U 1464  1     BEGIN
;  1106        U 1465  1
;  1107        U 1466  1     BUILTIN
;  1108        U 1467  1         AP;
;  1109        U 1468  1
;  1110        U 1469  1     MAP
;  1111        U 1470  1         AP : REF BLOCK [, BYTE];
;  1112        U 1471  1
;  1113        U 1472  1     SELECTONE .AP [tpa$l_param] OF
;  1114        U 1473  1         SET
;  1115        U 1474  1
;  1116        U 1475  1         [-2]:
;  1117        U 1476  1             BEGIN
;  1118        U 1477  1             !
;  1119        U 1478  1             ! Don't display section numbers.
```

```
;  1120        U 1479  1              !
;  1121        U 1480  1              cmdblk [contents$v_include_sections] = false;
;  1122        U 1481  1              cmdblk [contents$g_sections] = 0;
;  1123        U 1482  1              END;
;  1124        U 1483  1
;  1125        U 1484  1          [-1]:
;  1126        U 1485  1              BEGIN
;  1127        U 1486  1              !
;  1128        U 1487  1              ! Display section numbers AS_INPUT.
;  1129        U 1488  1              !
;  1130        U 1489  1              cmdblk [contents$v_include_sections] = true;
;  1131        U 1490  1              cmdblk [contents$g_sections] = -1;
;  1132        U 1491  1              END;
;  1133        U 1492  1
;  1134        U 1493  1          [0]:
;  1135        U 1494  1              BEGIN
;  1136        U 1495  1              !
;  1137        U 1496  1              ! User specified deepest level to display.
;  1138        U 1497  1              !
;  1139        U 1498  1              cmdblk [contents$v_include_sections] = true;
;  1140        U 1499  1              cmdblk [contents$g_sections] = .AP [tpa$l_number];
;  1141        U 1500  1              END;
;  1142        U 1501  1
;  1143        U 1502  1          [6]:
;  1144        U 1503  1              BEGIN
;  1145        U 1504  1              !
;  1146        U 1505  1              ! Display ALL section numbers.
;  1147        U 1506  1              !
;  1148        U 1507  1              cmdblk [contents$v_include_sections] = true;
;  1149        U 1508  1              cmdblk [contents$g_sections] = 6;
;  1150        U 1509  1              END;
;  1151        U 1510  1
;  1152        U 1511  1          TES;
;  1153        U 1512  1
;  1154        U 1513  1      RETURN ss$_normal;
;  1155        U 1514  1      END;
;  1156        U 1515  1
;  1157          1516  1 %FI
```

```
;  1159          L 1517  1 %IF DSRPLUS
;  1160          U 1518  1 %THEN
;  1161          U 1519  1
;  1162          U 1520  1 %SBTTL 'ENTER_DOT -- Action routine - enter leader dot character'
;  1163          U 1521  1 ROUTINE enter_dot =
;  1164          U 1522  1 !++
;  1165          U 1523  1 !
;  1166          U 1524  1 ! FUNCTIONAL DESCRIPTION:
;  1167          U 1525  1 !
;  1168          U 1526  1 !       This routine is called as an action routine by TPARSE.
;  1169          U 1527  1 !       This routine saves the leader dot character specified by the user.
;  1170          U 1528  1 !
;  1171          U 1529  1 ! FORMAL PARAMETERS:
;  1172          U 1530  1 !
;  1173          U 1531  1 !       AP [tpa$b_char] - leader dot character
;  1174          U 1532  1 !
;  1175          U 1533  1 ! IMPLICIT INPUTS:
;  1176          U 1534  1 !
;  1177          U 1535  1 !       None
;  1178          U 1536  1 !
;  1179          U 1537  1 ! IMPLICIT OUTPUTS:
;  1180          U 1538  1 !
;  1181          U 1539  1 !       cmdblk [contents$c_leader_char] - character is saved here
;  1182          U 1540  1 !
;  1183          U 1541  1 ! ROUTINE VALUE:
;  1184          U 1542  1 ! COMPLETION CODES:
;  1185          U 1543  1 !
;  1186          U 1544  1 !       ss$_normal
;  1187          U 1545  1 !
;  1188          U 1546  1 ! SIDE EFFECTS:
;  1189          U 1547  1 !
;  1190          U 1548  1 !       None
;  1191          U 1549  1 !
;  1192          U 1550  1 !--
;  1193          U 1551  1
;  1194          U 1552  1     BEGIN
;  1195          U 1553  1
;  1196          U 1554  1     BUILTIN
;  1197          U 1555  1         AP;
;  1198          U 1556  1
;  1199          U 1557  1     MAP
;  1200          U 1558  1         AP : REF BLOCK [, BYTE];
;  1201          U 1559  1
;  1202          U 1560  1     CH$WCHAR (.AP [tpa$b_char], CH$PTR (cmdblk [contents$c_leader_char]));
;  1203          U 1561  1
;  1204          U 1562  1     RETURN ss$_normal;
;  1205          U 1563  1     END;
;  1206          U 1564  1
;  1207            1565  1 %FI
```

```
;  1209        L 1566  1 %IF DSRPLUS
;  1210        U 1567  1 %THEN
;  1211        U 1568  1
;  1212        U 1569  1 %SBTTL 'ENTER_HL_INDENT - Action routine - enter HL indent value'
;  1213        U 1570  1 ROUTINE enter_hl_indent =
;  1214        U 1571  1 !
;  1215        U 1572  1 ! FUNCTIONAL DESCRIPTION:
;  1216        U 1573  1 !
;  1217        U 1574  1 !       This routine is called as an action routine by TPARSE.
;  1218        U 1575  1 !
;  1219        U 1576  1 !       It assigns values for progressive indenting.
;  1220        U 1577  1 !
;  1221        U 1578  1 ! FORMAL PARAMETERS:
;  1222        U 1579  1 !
;  1223        U 1580  1 !       AP [tpa$l_number]       - value to set contents$ag_hl_indent
;  1224        U 1581  1 !       AP [tpa$l_param]        - HL number (0 = CHAPTER, -1 = PROGRESSIVE)
;  1225        U 1582  1 !
;  1226        U 1583  1 ! IMPLICIT INPUTS:
;  1227        U 1584  1 !
;  1228        U 1585  1 !       None
;  1229        U 1586  1 !
;  1230        U 1587  1 ! IMPLICIT OUTPUTS:
;  1231        U 1588  1 !
;  1232        U 1589  1 !       None
;  1233        U 1590  1 !
;  1234        U 1591  1 ! ROUTINE VALUE:
;  1235        U 1592  1 ! COMPLETION CODES:
;  1236        U 1593  1 !
;  1237        U 1594  1 !       ss$_normal
;  1238        U 1595  1 !
;  1239        U 1596  1 ! SIDE EFFECTS:
;  1240        U 1597  1 !
;  1241        U 1598  1 !       None
;  1242        U 1599  1 !
;  1243        U 1600  1 !--
;  1244        U 1601  1     BEGIN
;  1245        U 1602  1
;  1246        U 1603  1     BUILTIN
;  1247        U 1604  1         AP;
;  1248        U 1605  1
;  1249        U 1606  1     MAP
;  1250        U 1607  1         AP : REF BLOCK [, BYTE];
;  1251        U 1608  1
;  1252        U 1609  1     IF .AP [tpa$l_param] EQL -1
;  1253        U 1610  1     THEN
;  1254        U 1611  1         BEGIN
;  1255        U 1612  1         !
;  1256        U 1613  1         ! /INDENT=PROGRESSIVE=n.
;  1257        U 1614  1         ! Set HL2 through HL6
;  1258        U 1615  1         !
;  1259        U 1616  1         INCR i FROM 2 TO 6 DO
;  1260        U 1617  1             indents [.i] = .AP [tpa$l_number];
;  1261        U 1618  1
;  1262        U 1619  1         END
;  1263        U 1620  1     ELSE
;  1264        U 1621  1         BEGIN
;  1265        U 1622  1         !
```

```
;  1266          U 1623  1          ! CHAPTER and HL1 through HL6
;  1267          U 1624  1          !
;  1268          U 1625  1          indents [.AP [tpa$l_param]] = .AP [tpa$l_number];
;  1269          U 1626  1          END;
;  1270          U 1627  1
;  1271          U 1628  1      RETURN ss$_normal;
;  1272          U 1629  1      END;
;  1273          U 1630  1
;  1274            1631  1 %FI
```

```
;1276     1632  1 %SBTTL 'CONDITION_HANDLER - Main program condition handler - sets termination status'
;1277     1633  1 ROUTINE condition_handler (sig : REF BLOCK [, BYTE], mch : REF BLOCK [, BYTE]) =
;1278     1634  1 !++
;1279     1635  1 !
;1280     1636  1 ! FUNCTIONAL DESCRIPTION:
;1281     1637  1 !
;1282     1638  1 !       This routine is enabled by CNTCLI as a condition handler.
;1283     1639  1 !       Whenever a signal is generated, the signal severity is examined.
;1284     1640  1 !       If the condition is more severe than any previous condition,
;1285     1641  1 !       (success, warning, error, severe error) the severity is recorded
;1286     1642  1 !       in termination_status which is the condition severity. CNTCLI
;1287     1643  1 !       returns the value of TERMINATION_STATUS as the program status
;1288     1644  1 !       which will set the value of the DCL $STATUS variable.
;1289     1645  1 !
;1290     1646  1 ! FORMAL PARAMETERS:
;1291     1647  1 !
;1292     1648  1 !       sig                     - address of signal array
;1293     1649  1 !       mch                     - address of mechanism array
;1294     1650  1 !
;1295     1651  1 ! IMPLICIT INPUTS:
;1296     1652  1 !
;1297     1653  1 !       termination_status      - current termination severity
;1298     1654  1 !
;1299     1655  1 ! IMPLICIT OUTPUTS:
;1300     1656  1 !
;1301     1657  1 !       termination_status      - may be set to the severity level in the
;1302     1658  1 !                                 signalled condition if it is more severe
;1303     1659  1 !
;1304     1660  1 ! ROUTINE VALUE:
;1305     1661  1 ! COMPLETION CODES:
;1306     1662  1 !
;1307     1663  1 !       ss$_resignal
;1308     1664  1 !
;1309     1665  1 ! SIDE EFFECTS:
;1310     1666  1 !
;1311     1667  1 !       None
;1312     1668  1 !--
;1313     1669  2     BEGIN
;1314     1670  2
;1315     1671  2     BIND
;1316     1672  2         signalled_condition = sig [chf$l_sig_name] : BLOCK [, BYTE];
;1317     1673  2
;1318     1674  2     SELECTONE .signalled_condition [sts$v_severity] OF
;1319     1675  2         SET
;1320     1676  2
;1321     1677  2         [sts$k_warning]:
;1322     1678  2             IF .termination_status EQL sts$k_success
;1323     1679  2             THEN
;1324     1680  2                 !
;1325     1681  2                 ! A warning changes the termination status only if it was
;1326     1682  2                 ! 'success' previously.
;1327     1683  2                 !
;1328     1684  2                 termination_status = sts$k_warning;
;1329     1685  2
;1330     1686  2         [sts$k_error]:
;1331     1687  2             IF .termination_status LSS sts$k_error
;1332     1688  2             THEN
```

```
;  1333          1689  2                 !
;  1334          1690  2                 ! An error status changes the termination status only if it
;  1335          1691  2                 ! was 'success' or 'warning' previously.
;  1336          1692  2                 !
;  1337          1693  2                 termination_status = sts$k_error;
;  1338          1694  2
;  1339          1695  2             [sts$k_severe]:                         ! Severe error
;  1340          1696  2                 termination_status = sts$k_severe;  ! set the termination status
;  1341          1697  2
;  1342          1698  2             [OTHERWISE]:                            ! Success or Informational
;  1343          1699  2                 ;                                   ! Do nothing
;  1344          1700  2
;  1345          1701  2             TES;
;  1346          1702  2
;  1347          1703  2         RETURN ss$_resignal;                        ! Continue processing condition
;  1348          1704  1         END;


                                        0004 00000 CONDITION_HANDLER:
                                                   .WORD   Save R2                                                  ;  1633
                    52 00000000'    EF  9E 00002           MOVAB   TERMINATION_STATUS, R2
                                                                                                                    ;
          50        04              AC  C1 00009           ADDL3   #4, SIG, R0                                      ;  1672
                    07              60  93 0000E           BITB    (R0), #7                                         ;  1677
                                    09  12 00011           BNEQ    1$
                    01              62  D1 00013           CMPL    TERMINATION_STATUS, #1                           ;  1678
                                    1F  12 00016           BNEQ    3$
                                    62  D4 00018           CLRL    TERMINATION_STATUS                               ;  1684
                                    1B  11 0001A           BRB     3$                                               ;  1678
     02             60              03      00  ED 0001C 1$:  CMPZV   #0, #3, (R0), #2                             ;  1686
                                    0A  12 00021           BNEQ    2$
                    02              62  D1 00023           CMPL    TERMINATION_STATUS, #2                           ;  1687
                                    0F  18 00026           BGEQ    3$
                    62              02  D0 00028           MOVL    #2, TERMINATION_STATUS                           ;  1693
                                    0A  11 0002B           BRB     3$                                               ;  1687
     04             60              03      00  ED 0002D 2$:  CMPZV   #0, #3, (R0), #4                             ;  1695
                                    03  12 00032           BNEQ    3$
                    62              04  D0 00034           MOVL    #4, TERMINATION_STATUS                           ;  1696
                    50      0918    8F  3C 00037 3$:       MOVZWL  #2328, R0                                        ;  1703
                                        04 0003C           RET                                                      ;  1704

; Routine Size:  61 bytes,    Routine Base:  $CODE$ + 02A5
```

```
;  1350     1705  1 %SBTTL 'OPEN_ERROR - Handle File Open Errors'
;  1351     1706  1 GLOBAL ROUTINE open_error (function_code, primary_code, secondary_code, iob : REF $xpo_iob ()) =
;  1352     1707  1 !++
;  1353     1708  1 !
;  1354     1709  1 ! FUNCTIONAL DESCRIPTION:
;  1355     1710  1 !
;  1356     1711  1 !       This routine is called as an action routine to report
;  1357     1712  1 !       file open errors.
;  1358     1713  1 !
;  1359     1714  1 ! FORMAL PARAMETERS:
;  1360     1715  1 !
;  1361     1716  1 !       function_code   - XPORT failure action routine function code
;  1362     1717  1 !       primary_code    - primary failure completion code
;  1363     1718  1 !       secondary_code  - secondary failure completion code
;  1364     1719  1 !       iob             - Address of file IOB
;  1365     1720  1 !
;  1366     1721  1 ! IMPLICIT INPUTS:      None
;  1367     1722  1 !
;  1368     1723  1 ! IMPLICIT OUTPUTS:     None
;  1369     1724  1 !
;  1370     1725  1 ! ROUTINE VALUE:
;  1371     1726  1 ! COMPLETION CODES:
;  1372     1727  1 !
;  1373     1728  1 !       Returns the value of primary_code if success is indicated.
;  1374     1729  1 !
;  1375     1730  1 ! SIDE EFFECTS:
;  1376     1731  1 !
;  1377     1732  1 !       Signals a fatal error terminating program execution if failure
;  1378     1733  1 !       is indicated by primary_code.
;  1379     1734  1 !--
;  1380     1735  1
;  1381     1736  2     BEGIN
;  1382     1737  2
;  1383     1738  2     BIND
;  1384     1739  2         file_spec = .iob [iob$a_file_spec] : $str_descriptor (),
;  1385     1740  2         resultant =  iob [iob$t_resultant] : $str_descriptor ();
;  1386     1741  2
;  1387     1742  2     LOCAL
;  1388     1743  2         file_name : REF $str_descriptor ();
;  1389     1744  2
;  1390     1745  2     !
;  1391     1746  2     ! Point to best file name.
;  1392     1747  2     !
;  1393     1748  2     file_name = (IF    .resultant [str$h_length] NEQ 0 THEN resultant ELSE file_spec);
;  1394     1749  2
;  1395     1750  2     IF  NOT .primary_code
;  1396     1751  2     THEN
;  1397     1752  3         BEGIN
;  1398     1753  3         !
;  1399     1754  3         ! File was not opened.
;  1400     1755  3         !
;  1401     1756  3         IF  .iob [iob$v_input]
;  1402     1757  3         THEN
;  1403     1758  3             SIGNAL_STOP (contents$_openin, 1, .file_name,
;  1404     1759  3                 .iob [iob$g_comp_code], 1, .iob [iob$g_2nd_code])
;  1405     1760  3
;  1406     1761  3         ELSE
```

```
;  1407          1762  3              SIGNAL_STOP (contents$_openout, 1, .file_name,
;  1408          1763  3                  .iob [iob$g_comp_code], 1, .iob [iob$g_2nd_code]);
;  1409          1764  3
;  1410          1765  2          END;
;  1411          1766  2
;  1412          1767  2      RETURN .primary_code;
;  1413          1768  1      END;


                                        0000 00000          .ENTRY  OPEN_ERROR, Save nothing                        ; 1706
                                50        10  AC   D0 00002          MOVL    IOB, R0                                 ; 1739
                                          1C  A0   B5 00006          TSTW    28(R0)                                  ; 1748
                                              06   13 00009          BEQL    1$                                      ;
                                51        1C  A0   9E 0000B          MOVAB   28(R0), FILE_NAME                       ;
                                              04   11 0000F          BRB     2$                                      ;
                                51        04  A0   D0 00011 1$:      MOVL    4(R0), FILE_NAME                        ;
                                35        08  AC   E8 00015 2$:      BLBS    PRIMARY_CODE, 5$                        ; 1750
                                16        2E  A0   E9 00019          BLBC    46(R0), 3$                              ; 1756
                                        00DC  C0   DD 0001D          PUSHL   220(R0)                                 ; 1759
                                              01   DD 00021          PUSHL   #1                                      ; 1758
                                        00D8  C0   DD 00023          PUSHL   216(R0)                                 ; 1759
                                              51   DD 00027          PUSHL   FILE_NAME                               ; 1758
                                              01   DD 00029          PUSHL   #1                                      ;
                                   00000000G  8F   DD 0002B          PUSHL   #DSRTOC$_OPENIN                         ;
                                              14   11 00031          BRB     4$                                      ;
                                        00DC  C0   DD 00033 3$:      PUSHL   220(R0)                                 ; 1763
                                              01   DD 00037          PUSHL   #1                                      ; 1762
                                        00D8  C0   DD 00039          PUSHL   216(R0)                                 ; 1763
                                              51   DD 0003D          PUSHL   FILE_NAME                               ; 1762
                                              01   DD 0003F          PUSHL   #1                                      ;
                                   00000000G  8F   DD 00041          PUSHL   #DSRTOC$_OPENOUT                        ;
                        00000000G 00              06   FB 00047 4$:      CALLS   #6, LIB$STOP                            ;
                                50        08  AC   D0 0004E 5$:      MOVL    PRIMARY_CODE, R0                        ; 1767
                                              04   00052          RET                                             ; 1768

; Routine Size:  83 bytes,    Routine Base:  $CODE$ + 02E2

;  1414          1769  1
;  1415          1770  1 END                                   ! End of module
;  1416          1771  0 ELUDOM


                                                                .EXTRN  LIB$SIGNAL, LIB$STOP

;                                   PSECT SUMMARY
;
;       Name                            Bytes                           Attributes
;
;  $OWN$                                  16  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  _LIB$KEY0$                              6  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$STATE$                            60  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
```

```
;  _LIB$KEY1$                    25  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  $PLIT$                       308  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                       821  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                          Library Statistics
;
;                                    --------- Symbols ---------      Pages       Processing
;       File                         Total     Loaded    Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1   9776         22          0         581       00:01.1
;  _$255$DUA28:[SYSLIB]TPAMAC.L32;1      42         25         59          14       00:00.1
;  _$255$DUA28:[SYSLIB]XPORT.L32;1      590        130         22         252       00:00.6



;                                  COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:CNTVMS/OBJ=OBJ$:CNTVMS MSRC$:CNTVMS/UPDATE=(ENH$:CNTVMS)

; Size:          821 code + 415 data bytes
; Run Time:        00:39.3
; Elapsed Time:    01:34.5
; Lines/CPU Min:    2704
; Lexemes/CPU-Min: 52940
; Memory Used:  270 pages
; Compilation Complete
```

CNVDAT
LIS

CONVBB
LIS

CONVBR
LIS

CNTVMSMSG
LIS

CONTENTS
LIS

CNTVMS
LIS

CNTVRS
LIS

CONTRL
LIS

CONVBI
LIS