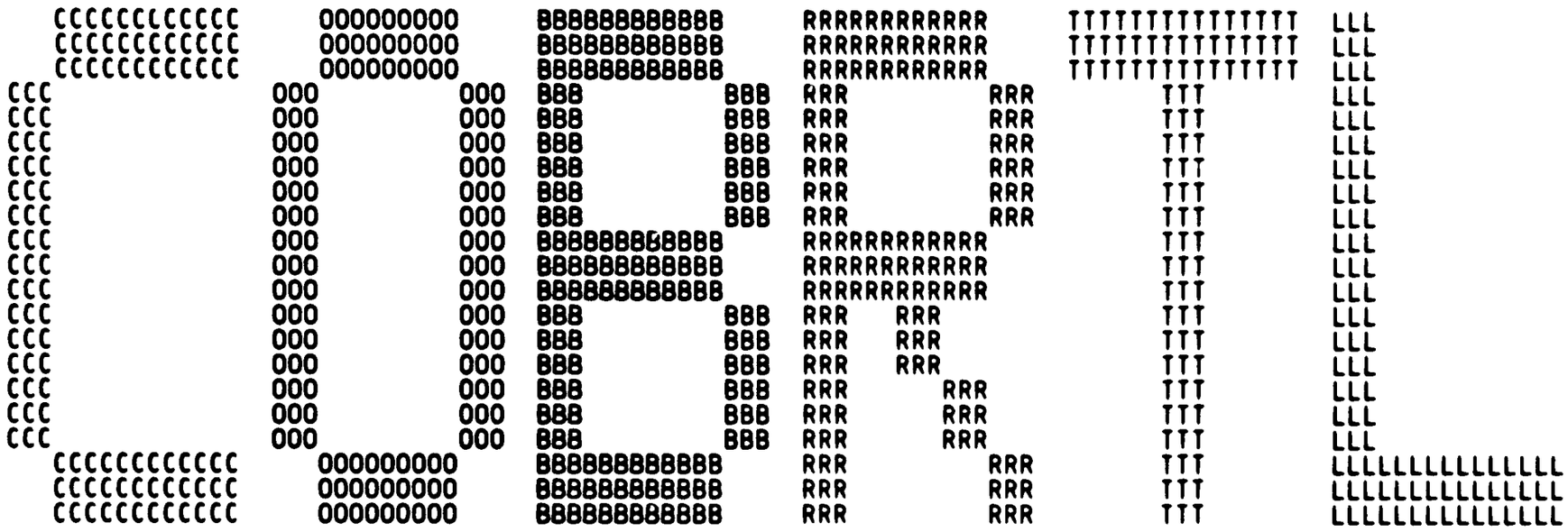
CCCCCCCCCCC 000000000 BBBBBBBBBBBB RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
CCCCCCCCCCC 000000000 BBBBBBBBBBBB RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
CCCCCCCCCCC 000000000 BBBBBBBBBBBB RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBBBBBBBBBB RRRRRRRRRRR TTT LLL
CCC 000 000 BBBBBBBBBBBB RRRRRRRRRRR TTT LLL
CCC 000 000 BBBBBBBBBBBB RRRRRRRRRRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCC 000 000 BBB BBB RRR RRR TTT LLL
CCCCCCCCCCC 000000000 BBBBBBBBBBBB RRR RRR TTT LLLLLLLLLLLLLLL
CCCCCCCCCCC 000000000 BBBBBBBBBBBB RRR RRR TTT LLLLLLLLLLLLLLL
CCCCCCCCCCC 000000000 BBBBBBBBBBBB RRR RRR TTT LLLLLLLLLLLLLLL

```
  CCCCCCCC    000000    BBBBBBBB    PPPPPPPP      000000      SSSSSSSS  EEEEEEEEEE  RRRRRRRR      AAAAAA
  CCCCCCCC    000000    BBBBBBBB    PPPPPPPP      000000      SSSSSSSS  EEEEEEEEEE  RRRRRRRR      AAAAAA
CC          00      00  BB      BB  PP      PP  00      00  SS          EE          RR      RR  AA      AA
CC          00      00  BB      BB  PP      PP  00      00  SS          EE          RR      RR  AA      AA
CC          00      00  BB      BB  PP      PP  00      00  SS          EE          RR      RR  AA      AA
CC          00      00  BB      BB  PP      PP  00      00  SS          EE          RR      RR  AA      AA
CC          00      00  BBBBBBBB    PPPPPPPP    00      00    SSSSSS    EEEEEEEE    RRRRRRRR    AA      AA
CC          00      00  BBBBBBBB    PPPPPPPP    00      00    SSSSSS    EEEEEEEE    RRRRRRRR    AA      AA
CC          00      00  BB      BB  PP          00      00          SS  EE          RR  RR      AAAAAAAAAA
CC          00      00  BB      BB  PP          00      00          SS  EE          RR  RR      AAAAAAAAAA
CC          00      00  BB      BB  PP          00      00          SS  EE          RR    RR    AA      AA
CC          00      00  BB      BB  PP          00      00          SS  EE          RR    RR    AA      AA
  CCCCCCCC    000000    BBBBBBBB    PP            000000    SSSSSSSS  EEEEEEEEEE  RR      RR  AA      AA
  CCCCCCCC    000000    BBBBBBBB    PP            000000    SSSSSSSS  EEEEEEEEEE  RR      RR  AA      AA

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 MODULE COB$POS_ERASE ( %TITLE 'Position and erase'
;     2          0002  0                  IDENT = '1-003'          ! File: COBPOSERA.B32 Edit: LGB1003
;     3          0003  0                  ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !*****************************************************************************
;     7          0007  1 !*                                                                           *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                     *
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    17          0017  1 !*  TRANSFERRED.                                                             *
;    18          0018  1 !*                                                                           *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    21          0021  1 !*  CORPORATION.                                                             *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*****************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:     VAX-11 COBOL Support
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       Used with ACCEPT and DISPLAY, this routine provides direct cursor
;    36          0036  1 !       positioning, relative cursor positioning, screen erasing, and
;    37          0037  1 !       line erasing.
;    38          0038  1 !
;    39          0039  1 ! ENVIRONMENT:  User mode - AST reentrant
;    40          0040  1 !
;    41          0041  1 ! AUTHOR: P. Levesque, CREATION DATE: 27-Jan-1983
;    42          0042  1 !
;    43          0043  1 ! MODIFIED BY:
;    44          0044  1 !
;    45          0045  1 ! 1-001 - Original.  PLL 27-Jan-1983
;    46          0046  1 ! 1-002 - Put out sequence to make sure terminal is in VT100 or VT52 mode.
;    47          0047  1 !         Put in kludge for erasing whole screen w/cursor positioning
;    48          0048  1 !         on a vt52.
;    49          0049  1 !         After a screen erase on a vt52, always call the cursor
;    50          0050  1 !         positioning routine again, whether direct or indirect.
;    51          0051  1 !         More code for COB$$AB_PREV to handle Version 3 ACCEPT statement
;    52          0052  1 !         with ADVANCING and with NO ADVANCING.
;    53          0053  1 !         Created two entry points, COB$POS_ACCEPT and COB$POS_DISPLAY.
;    54          0054  1 !         Created COB$ACC_TERM_TYPE for ACCEPT and left COB$TERM_TYPE for
;    55          0055  1 !         DISPLAY.
;    56          0056  1 !         Open both SYS$INPUT and SYS$OUTPUT for ACCEPT to be able to
;    57          0057  1 !         write to terminal.  SYS$INPUT is Read-only under VMS V4.
```

```
;   58          0058  1 !          Added second parameter for COB$$OPEN_IN and COB$$OPEN_OUT.
;   59          0059  1 !          Use COBPROLOG.REQ                                  LGB 16-JAN-1984
;   60          0060  1 ! 1-003 - Create placeholder routine COB$POS_ERASE.     LGB 27-FEB-1984
;   61          0061  1 !--
;   62          0062  1
```

```
;    64          0063  1 %SBTTL 'Declarations'
;    65          0064  1 !
;    66          0065  1 ! PROLOGUE FILE
;    67          0066  1 !
;    68          0067  1
;    69          0068  1 REQUIRE 'RTLIN:COBPROLOG' ;                    ! Switches, Psects, Include
;    70          1585  1                                               ! files
;    71          1586  1 !
;    72          1587  1 ! LINKAGES:
;    73          1588  1 !
;    74          1589  1 !       NONE
;    75          1590  1 !
;    76          1591  1 ! TABLE OF CONTENTS:
;    77          1592  1 !
;    78          1593  1
;    79          1594  1 FORWARD ROUTINE
;    80          1595  1     COB$POS_ACCEPT  :  NOVALUE,                ! Open for ACCEPT
;    81          1596  1     COB$POS_DISPLAY :  NOVALUE,                ! Open for DISPLAY
;    82          1597  1     COB$POS_ERASE   :  NOVALUE,                ! Place-holder routine
;    83          1598  1     COB$$POS_ERASE ;                           ! Position and erase
;    84          1599  1
;    85          1600  1 !
;    86          1601  1 ! INCLUDE FILES:
;    87          1602  1 !
;    88          1603  1
;    89          1604  1 REQUIRE 'RTLIN:COBLNK';                       ! Define linkages for Screen
;    90          1679  1                                               ! routines
;    91          1680  1 LITERAL
;    92          1681  1         DISP    = 0,                          ! Code for DISPLAY
;    93          1682  1         DNA     = 1,                          ! Code for DISPLAY No Advancing
;    94          1683  1         POS     = 2,                          ! Code for Positioning
;    95          1684  1         POS_DNA = 3,                          ! Code for Positioning No Adv
;    96          1685  1         ACC_ADV = 4,                          ! Code for Accept (V3)
;    97          1686  1         ACC_DNA = 5 ;                         ! Code for Accept No Adv (V3)
;    98          1687  1 !
;    99          1688  1 ! MACROS:
;   100          1689  1 !
;   101          1690  1     MACRO
;   102        M 1691  1         $OUTPUT_ESC_SEQ_BUF =
;   103        M 1692  1         !+
;   104        M 1693  1         !  Output sequences.
;   105        M 1694  1         !-
;   106        M 1695  1         BEGIN
;   107        M 1696  1         RAB [RAB$L_RBF] = ESC_SEQ_BUF [0];
;   108        M 1697  1         RAB [RAB$W_RSZ] = MIN (.COR_BUF_LEN, K_MAX_RMS_LEN);
;   109        M 1698  1         WHILE $PUT (RAB = .RAB) EQL RMS$_RSA DO $WAIT (RAB = .RAB);
;   110        M 1699  1         IF NOT .RAB [RAB$L_STS]
;   111        M 1700  1         THEN
;   112        M 1701  1             LIB$STOP (COB$_ERRDURPOS, 1, .RAB + RAB$C_BLN,
;   113        M 1702  1                       .RAB [RAB$L_STS], .RAB [RAB$L_STV]);
;   114        M 1703  1         CUR_BUF_LEN = 0;
;   115        M 1704  1         END;
;   116          1705  1     %;                                           ! end macro $OUTPUT_ESC_SEQ_BUF
;   117          1706  1
;   118          1707  1 !
;   119          1708  1 ! EQUATED SYMBOLS:
;   120          1709  1 !
```

```
;  121          1710  1 !       NONE
;  122          1711  1 !
;  123          1712  1 ! FIELDS:
;  124          1713  1 !
;               1714  1 !       NONE
;    26         1715  1 !
;  127          1716  1 !
;  128          1717  1 ! OWN STORAGE
;  129          1718  1 !
;  130          1719  1 GLOBAL
;  131          1720  1     COB$TERM_TYPE : INITIAL (0),                ! Terminal type for DISPLAY
;  132          1721  1     COB$ACC_TERM_TYPE : INITIAL (0) ;           ! Terminal type for ACCEPT
;  133          1722  1 OWN
;  134          1723  1     RAB  :  REF $RAB_DECL ;                     ! RAB for output/input device
;  135          1724  1 !
;  136          1725  1 ! EXTERNAL REFERENCES:
;  137          1726  1 !
;  138          1727  1
;  139          1728  1 EXTERNAL ROUTINE
;  140          1729  1     COB$$SETUP_TERM_TYPE,                       ! Setup terminal type
;  141          1730  1     COB$$ERASE_LINE_R2 : COB$$ESC_R2_LNK,       ! Get the seq to erase a line
;  142          1731  1     COB$$ERASE_PAGE_R2 : COB$$ESC_R2_LNK,       ! Get the seq to erase the screen
;  143          1732  1     COB$$SET_CURSOR_REL,                        ! Get seq for relative cursor
;  144          1733  1                                                 !  positioning
;  145          1734  1     COB$$ERASE_WHOLE_PAGE_R2 : COB$$ESC_R2_LNK, ! Get seq to erase screen w/out
;  146          1735  1                                                 !  changing cursor
;  147          1736  1     COB$$ERASE_WHOLE_LINE_R2 : COB$$ESC_R2_LNK, ! Get seq to erase line w/out
;  148          1737  1                                                 !  changing cursor
;  149          1738  1     COB$$OPEN_IN,                               ! Create unit for input
;  150          1739  1     COB$$OPEN_OUT,                              ! Open unit for output
;  151          1740  1     LIB$STOP;                                   ! Signal error
;  152          1741  1
;  153          1742  1 EXTERNAL LITERAL                                ! Condition value symbols
;  154          1743  1     COB$_INVARG,                                ! Invalid argument
;  155          1744  1     COB$_ERRDURDIS,                             ! Error during DISPLAY
;  156          1745  1     COB$_ERRDURACC,                             ! Error during ACCEPT
;  157          1746  1     COB$_ERRDURPOS ;                            ! Error during Positioning
;  158          1747  1
;  159          1748  1 EXTERNAL
;  160          1749  1     COB$$AL_WRITE_RAB : VECTOR,                 ! Address of RAB
;  161          1750  1     COB$$AB_USPCODE : VECTOR [,BYTE],           ! Prefix and Post Upspacing
;  162          1751  1     COB$$AB_PREV : VECTOR [,BYTE] ;             ! History of Previous call
```

```
;  164          1752  1 %SBTTL 'COB$POS_ACCEPT - Position and erase for ACCEPT'
;  165          1753  1 GLOBAL ROUTINE COB$POS_ACCEPT (
;  166          1754  1                                 UNIT : VECTOR [2,BYTE], ! unit for inout
;  167          1755  1                                 ERASE_FLAG,     ! type of erasing
;  168          1756  1                                 LINE,           ! line number
;  169          1757  1                                 COLUMN,         ! column number
;  170          1758  1                                 LINE_PLUS,      ! number of lines to advance
;  171          1759  1                                 COLUMN_PLUS     ! number of cols to advance
;  172          1760  1     ) : NOVALUE =
;  173          1761  1
;  174          1762  1 !++
;  175          1763  1 ! FUNCTIONAL DESCRIPTION:
;  176          1764  1 !
;  177          1765  1 !       This routine is used with ACCEPT and DISPLAY.  Its purpose is to
;  178          1766  1 !       open UNIT for input if necessary, then call COB$$POS_ERASE to
;  179          1767  1 !       perform positioning and erasing.
;  180          1768  1 !
;  181          1769  1 ! CALLING SEQUENCE:
;  182          1770  1 !
;  183          1771  1 !       ret_status.wlc.v = COB$POS_ERASE (UNIT.rbu.va,
;  184          1772  1 !                                         ERASE_FLAG.rl.v,
;  185          1773  1 !                                         LINE.rlu.v,
;  186          1774  1 !                                         COLUMN.rlu.v,
;  187          1775  1 !                                         LINE_PLUS.rlu.v,
;  188          1776  1 !                                         COLUMN_PLUS.rlu.v)
;  189          1777  1 !
;  190          1778  1 ! FORMAL PARAMETERS:
;  191          1779  1 !
;  192          1780  1 !       UNIT.rbu.va             Byte integer unit number designating unit
;  193          1781  1 !                               from which string is to be read, followed
;  194          1782  1 !                               by byte flag indicating whether a routine
;  195          1783  1 !                               should abort or return status on RMS$_EOF.
;  196          1784  1 !                               (This second byte is not used here)
;  197          1785  1 !       ERASE_FLAG.rl.v         type of erasing to be done
;  198          1786  1 !                                       0 = no erasing
;  199          1787  1 !                                       1 = erase whole screen
;  200          1788  1 !                                       2 = erase whole line
;  201          1789  1 !                                       3 = erase to end of screen
;  202          1790  1 !                                       4 = erase to end of line
;  203          1791  1 !       LINE.rlu.v              line number to which cursor should be moved
;  204          1792  1 !                               if  ero, the current line is retained
;  205          1793  1 !       COLUMN.rlu.v            col mn number to which cursor should be moved
;  206          1794  1 !                               if  ro, the current column is retained
;  207          1795  1 !       LINE_PLUS.rlu.v         numb r of lines to advance with line-feeds
;  208          1796  1 !       COLUMN_PLUS.rlu.v       numb r of columns to advance
;  209          1797  1 !
;  210          1798  1 ! IMPLICIT INPUTS:
;  211          1799  1 !
;  212          1800  1 !       NONE
;  213          1801  1 !
;  214          1802  1 ! IMPLICIT OUTPUTS·
;  215          1803  1 !
;  216          1804  1 !       NON:
;  217          1805  1 !
;  218          1806  1 ! COMPLETION STATUS:
;  219          1807  1 !
;  220          1808  1 !       SS$_NORMAL      Normal successful completion
```

```
;  221          1809  1 !       COB$_INVARG     Invalid argument
;  222          1810  1 !       COB$_ERRDURACC  Error during ACCEPT
;  223          1811  1 !
;  224          1812  1 ! SIDE EFFECTS:
;  225          1813  1 !
;  226          1814  1 !       NONE
;  227          1815  1 !
;  228          1816  1 !--
;  229          1817  1
;  230          1818  2     BEGIN
;  231          1819  2         LOCAL
;  232          1820  2             TERM_TYPE,                          ! Parameter to COB$POS_ERASE
;  233          1821  2             STATUS ;
;  234          1822  2
;  235          1823  2 !+
;  236          1824  2 ! Check to see if the unit has been opened.  The unit will be associated
;  237          1825  2 ! with either a COB$xxxx name or a SYS$xxxx name.  If not already open,
;  238          1826  2 ! call a routine to set up a RAB.
;  239          1827  2 !-
;  240          1828  2
;  241          1829  2     IF .UNIT[0] GTRU COB$K_UNIT_MAX
;  242          1830  2     THEN
;  243          1831  2         LIB$STOP (COB$_INVARG);
;  244          1832  2
;  245          1833  2 !+
;  246          1834  2 !  Open SYS$INPUT for call to COB$$SETUP_TERM_TYPE to find out if we are
;  247          1835  2 !  dealing with a file or not.
;  248          1836  2 !-
;  249          1837  2
;  250          1838  2     IF .COB$$AL_WRITE_RAB [.UNIT[0]] EQL 0
;  251          1839  2     THEN
;  252          1840  2         !+
;  253          1841  2         !  Second parameter signifies COB$$OPEN_IN called on behalf of VAX COBOL
;  254          1842  2         !-
;  255          1843  2         COB$$OPEN_IN (.UNIT[0], 0) ;
;  256          1844  2
;  257          1845  2     RAB = .COB$$AL_WRITE_RAB [.UNIT[0]] ;
;  258          1846  2
;  259          1847  2 !+
;  260          1848  2 ! First determine the terminal type and save it for later use.
;  261          1849  2 ! Notice that COB$TERM_TYPE is set only once, on the first call.
;  262          1850  2 ! A descriptor for the resultant name and the name itself are stored
;  263          1851  2 ! after the RAB.
;  264          1852  2 !-
;  265          1853  2
;  266          1854  3     BEGIN                                       ! find and store terminal type
;  267          1855  3         LOCAL
;  268          1856  3             NAM_DSC : REF BLOCK [,BYTE];        ! name dsc
;  269          1857  3
;  270          1858  3     NAM_DSC = .RAB + RAB$C_BLN;                 ! point past RAB to name dsc
;  271          1859  3
;  272          1860  3     IF .COB$ACC_TERM_TYPE EQL 0
;  273          1861  3     THEN
;  274          1862  4         IF NOT ( COB$$SETUP_TERM_TYPE (.NAM_DSC [DSC$A_POINTER],
;  275          1863  4                                        .NAM_DSC [DSC$W_LENGTH],
;  276          1864  4                                        COB$ACC_TERM_TYPE ) )
;  277          1865  3         THEN LIB$STOP (COB$_ERRDURACC);
```

```
;  278          1866  3
;  279          1867  3      !+
;  280          1868  3      !  IF COB$ACC_TERM_TYPE is UNKNOWN, we can assume we are dealing with
;  281          1869  3      !  a file, therefore no need to go thro ' the positioning routine.
;  282          1870  3      !  If not dealing with a file call COB$$POS_ERASE.
;  283          1871  3      !-
;  284          1872  3
;  285          1873  3      IF .COB$ACC_TERM_TYPE NEQ UNKNOWN
;  286          1874  3      THEN
;  287          1875  4          BEGIN
;  288          1876  4          TERM_TYPE = .COB$ACC_TERM_TYPE ;
;  289          1877  4          IF .COB$$AL_WRITE_RAB [1] EQL 0
;  290          1878  4          THEN
;  291          1879  4          !+
;  292          1880  4          !  SYS$INPUT is read-only in VMS V3B.
;  293          1881  4          !  There is not need to open  SYS$INPUT (.UNIT[0] = 0) at this point,
;  294          1882  4          !  leave this for routine COB$ACC_SRC.
;  295          1883  4          !  For VMS V3B you have to $CREATE i.e. $OPEN SYS$OUTPUT in order to
;  296          1884  4          !  write the Escape Sequences to the terminal.  Use COB$$OPEN_OUT
;  297          1885  4          !  instead of COB$$OPEN_IN because we need the RHB (= COB$$AB_USPCODE)
;  298          1886  4          !  field of $RAB-INIT.  (SYS$OUTPUT is the second element in the
;  299          1887  4          !  SYS_TABLE table, hence COB$$OPEN_OUT (1) ).
;  300          1888  4          !  Make sure that the RAB is pointing to SYS$OUTPUT for the macro
;  301          1889  4          !  $OUTPUT_ESC_SEQ_BUF.
;  302          1890  4          !  Second parameter signifies COB$$OPEN_OUT called on behalf of VAX COBOL
;  303          1891  4          !-
;  304          1892  4          COB$$OPEN_OUT (1, 0) ;
;  305          1893  4          RAB = .COB$$AL_WRITE_RAB [1] ;
;  306          1894  4
;  307          1895  4          COB$$POS_ERASE ( .TERM_TYPE, .ERASE_FLAG, .LINE, .COLUMN,
;  308          1896  4                                                    .LINE_PLUS, .COLUMN_PLUS ) ;
;  309          1897  3          END ;
;  310          1898  2      END ;
;  311          1899  2
;  312          1900  1      END ;                                  ! End of COB$POS_ACCEPT


                                                                    .TITLE  COB$POS_ERASE Position and erase
                                                                    .IDENT  \1-003\

                                                                    .PSECT  _COB$DATA,NOEXE,  PIC,2

                                                    00000000  00000 COB$TERM_TYPE::
                                                                    .LONG   0                                   ;
                                                    00000000  00004 COB$ACC_TERM_TYPE::
                                                                    .LONG   0                                   ;
                                                              00008 RAB:    .BLKB   4

                                                                    .EXTRN  COB$$SETUP_TERM_TYPE
                                                                    .EXTRN  COB$$ERASE_LINE_R2
                                                                    .EXTRN  COB$$ERASE_PAGE_R2
                                                                    .EXTRN  COB$$SET_CURSOR_REL
                                                                    .EXTRN  COB$$ERASE_WHOLE_PAGE_R2
                                                                    .EXTRN  COB$$ERASE_WHOLE_LINE_R2
                                                                    .EXTRN  COB$$OPEN_IN, COB$$OPEN_OUT
                                                                    .EXTRN  LIB$STOP, COB$_INVARG
                                                                    .EXTRN  COB$_ERRDURDIS, COB$_ERRDURACC
```

```
                                        .EXTRN  COB$_ERRDURPOS, COB$$AL_WRITE_RAB
                                        .EXTRN  COB$$AB_USPCODE
                                        .EXTRN  COB$$AB_PREV

                                        .PSECT  _COB$CODE,NOWRT,  SHR,  PIC,2

                                007C 00000      .ENTRY  COB$POS_ACCEPT, Save R2,R3,R4,R5,R6     ; 1753
                    56 00000000G  00  9E 00002  MOVAB   LIB$STOP, R6                            ;
                    55 00000000G  00  9E 00009  MOVAB   COB$$AL_WRITE_RAB+4, R5                 ;
                    54 00000000'  EF  9E 00010  MOVAB   RAB, R4                                 ;
                    52        04  AC  9A 00017  MOVZBL  UNIT, R2                               ; 1829
                    06            52  91 0001B  CMPB    R2, #6                                ;
                                  09  1B 0001E  BLEQU   1$                                    ;
                       00000000G  8F  DD 00020  PUSHL   #COB$_INVARG                          ; 1831
                    66            01  FB 00026  CALLS   #1, LIB$STOP                         ;
                    53        FC A542  DE 00029  1$:     MOVAL   COB$$AL_WRITE_RAB[R2], R3     ; 1838
                                  63  D5 0002E  TSTL    (R3)                                  ;
                                  0B  12 00030  BNEQ    2$                                    ;
                                  7E  D4 00032  CLRL    -(SP)                                 ; 1843
                                  52  DD 00034  PUSHL   R2                                    ;
                 00000000G  00        02  FB 00036  CALLS   #2, COB$$OPEN_IN                  ;
                    64            63  D0 0003D  2$:     MOVL    (R3), RAB                     ; 1845
                50  64 00000044  8F  C1 00040  ADDL3   #68, RAB, NAM_DSC                     ; 1858
                           FC     A4  D5 00048  TSTL    COB$ACC_TERM_TYPE                     ; 1860
                                  1C  12 0004B  BNEQ    3$                                    ;
                           FC     A4  9F 0004D  PUSHAB  COB$ACC_TERM_TYPE                     ; 1862
                    7E            60  3C 00050  MOVZWL  (NAM_DSC), -(SP)                    ; 1863
                           04     A0  DD 00053  PUSHL   4(NAM_DSC)                          ; 1862
                 00000000G  00        03  FB 00056  CALLS   #3, COB$$SETUP_TERM_TYPE          ;
                    09            50  E8 0005D  BLBS    R0, 3$                              ;
                       00000000G  8F  DD 00060  PUSHL   #COB$_ERRDURACC                       ; 1865
                    66            01  FB 00066  CALLS   #1, LIB$STOP                         ;
                    50        FC  A4  D0 00069  3$:     MOVL    COB$ACC_TERM_TYPE, R0         ; 1873
                                  26  13 0006D  BEQL    5$                                    ;
                    52            50  D0 0006F  MOVL    R0, TERM_TYPE                         ; 1876
                                  65  D5 00072  TSTL    COB$$AL_WRITE_RAB+4                   ; 1877
                                  0A  12 00074  BNEQ    4$                                    ;
                    7E            01  7D 00076  MOVQ    #1, -(SP)                             ; 1892
                 00000000G  00        02  FB 00079  CALLS   #2, COB$$OPEN_OUT                ;
                    64            65  D0 00080  4$:     MOVL    COB$$AL_WRITE_RAB+4, RAB      ; 1893
                    7E        14  AC  7D 00083  MOVQ    LINE_PLUS, -(SP)                      ; 1896
                    7E        0C  AC  7D 00087  MOVQ    LINE, -(SP)                           ; 1895
                              08  AC  DD 0008B  PUSHL   ERASE_FLAG                           ;
                                  52  DD 0008E  PUSHL   TERM_TYPE                            ;
                      0000V   CF        06  FB 00090  CALLS   #6, COB$$POS_ERASE            ;
                                        04  00095  5$:     RET                                 ; 1900

; Routine Size:  150 bytes,    Routine Base:  _COB$CODE + 0000
```

```
;  314          1901  1 %SBTTL 'COB$POS_DISPLAY - Pos and erase for DISPLAY'
;  315          1902  1 GLOBAL ROUTINE COB$POS_DISPLAY (
;  316          1903  1                                 UNIT,           ! unit for output
;  317          1904  1                                 ERASE_FLAG,     ! type of erasing
;  318          1905  1                                 LINE,           ! line number
;  319          1906  1                                 COLUMN,         ! column number
;  320          1907  1                                 LINE_PLUS,      ! number of lines to advance
;  321          1908  1                                 COLUMN_PLUS     ! number of cols to advance
;  322          1909  1     ) : NOVALUE =
;  323          1910  1
;  324          1911  1 !++
;  325          1912  1 ! FUNCTIONAL DESCRIPTION:
;  326          1913  1 !
;  327          1914  1 !       This routine is used with ACCEPT and DISPLAY.  Its purpose is to
;  328          1915  1 !       open UNIT for input if necessary, then call COB$$POS_ERASE to
;  329          1916  1 !       perform positioning and erasing.
;  330          1917  1 !
;  331          1918  1 ! CALLING SEQUENCE:
;  332          1919  1 !
;  333          1920  1 !       ret_status.wlc.v = COB$POS_ERASE (UNIT.rlu.v,
;  334          1921  1 !                                         ERASE_FLAG.rl.v,
;  335          1922  1 !                                         LINE.rlu.v,
;  336          1923  1 !                                         COLUMN.rlu.v,
;  337          1924  1 !                                         LINE_PLUS.rlu.v,
;  338          1925  1 !                                         COLUMN_PLUS.rlu.v)
;  339          1926  1 !
;  340          1927  1 ! FORMAL PARAMETERS:
;  341          1928  1 !
;  342          1929  1 !       UNIT.rlu.v              unit number of output device
;  343          1930  1 !       ERASE_FLAG.rl.v         type of erasing to be done
;  344          1931  1 !                                       0 = no erasing
;  345          1932  1 !                                       1 = erase whole screen
;  346          1933  1 !                                       2 = erase whole line
;  347          1934  1 !                                       3 = erase to end of screen
;  348          1935  1 !                                       4 = erase to end of line
;  349          1936  1 !       LINE.rlu.v              line number to which cursor should be moved
;  350          1937  1 !                               if zero, the current line is retained
;  351          1938  1 !       COLUMN.rlu.v            column number to which cursor should be moved
;  352          1939  1 !                               if zero, the current column is retained
;  353          1940  1 !       LINE_PLUS.rlu.v         number of lines to advance with line-feeds
;  354          1941  1 !       COLUMN_PLUS.rlu.v       number of columns to advance
;  355          1942  1 !
;  356          1943  1 ! IMPLICIT INPUTS:
;  357          1944  1 !
;  358          1945  1 !       NONE
;  359          1946  1 !
;  360          1947  1 ! IMPLICIT OUTPUTS:
;  361          1948  1 !
;  362          1949  1 !       NONE
;  363          1950  1 !
;  364          1951  1 ! COMPLETION STATUS:
;  365          1952  1 !
;  366          1953  1 !       SS$_NORMAL      Normal successful completion
;  367          1954  1 !       COB$_INVARG     Invalid argument
;  368          1955  1 !       COB$_ERRDURDIS  Error during DISPLAY
;  369          1956  1 !
;  370          1957  1 ! SIDE EFFECTS:
```

```
;   371          1958  1 !
;   372          1959  1 !       NONE
;   373          1960  1 !
;   374          1961  1 !--
;   375          1962  1
;   376          1963  2     BEGIN
;   377          1964  2         LOCAL
;   378          1965  2             TERM_TYPE,                          ! Parameter to COB$POS_ERASE
;   379          1966  2             STATUS ;
;   380          1967  2
;   381          1968  2 !+
;   382          1969  2 ! Check to see if the unit has been opened.  The unit will be associated
;   383          1970  2 ! with either a COB$xxxx name or a SYS$xxxx name.  If not already open,
;   384          1971  2 ! call a routine to set up a RAB.
;   385          1972  2 !-
;   386          1973  2
;   387          1974  2     IF .UNIT GTRU COB$K_UNIT_MAX
;   388          1975  2     THEN
;   389          1976  2         LIB$STOP (COB$_INVARG);
;   390          1977  2
;   391          1978  2     IF .COB$$AL_WRITE_RAB [.UNIT] EQL 0
;   392          1979  2     THEN
;   393          1980  2         !+
;   394          1981  2         ! Second parameter signifies COB$$OPEN_OUT called on behalf of VAX COBOL
;   395          1982  2         !-
;   396          1983  2         COB$$OPEN_OUT (.UNIT, 0);
;   397          1984  2
;   398          1985  2     RAB = .COB$$AL_WRITE_RAB [.UNIT];
;   399          1986  2
;   400          1987  2 !+
;   401          1988  2 ! First determine the terminal type and save it for later use.
;   402          1989  2 ! Notice that COB$TERM_TYPE is set only once, on the first call.
;   403          1990  2 ! A descriptor for the resultant name and the name itself are stored
;   404          1991  2 ! after the RAB.
;   405          1992  2 !-
;   406          1993  2
;   407          1994  3     BEGIN                                       ! find and store terminal type
;   408          1995  3         LOCAL
;   409          1996  3             NAM_DSC : REF BLOCK [,BYTE];        ! name dsc
;   410          1997  3
;   411          1998  3     NAM_DSC = .RAB + RAB$C_BLN  ;               ! point past RAB to name dsc
;   412          1999  3
;   413          2000  3     IF .COB$TERM_TYPE EQL 0
;   414          2001  3     THEN
;   415          2002  4         IF NOT ( COB$$SETUP_TERM_TYPE (.NAM_DSC [DSC$A_POINTER],
;   416          2003  4                                        .NAM_DSC [DSC$W_LENGTH],
;   417          2004  4                                        COB$TERM_TYPE ) )
;   418          2005  3         THEN LIB$STOP (COB$_ERRDURDIS);
;   419          2006  3
;   420          2007  3     !+
;   421          2008  3     !  IF COB$ACC_TERM_TYPE is UNKNOWN, we can assume we are dealing with
;   422          2009  3     !  a file, therefore no need to go through the positioning routine.
;   423          2010  3     !  If not dealing with a file call COB$$POS_ERASE.
;   424          2011  3     !-
;   425          2012  3
;   426          2013  3     IF .COB$TERM_TYPE NEQ UNKNOWN
;   427          2014  3     THEN
```

```
;  428          2015  4          BEGIN
;  429          2016  4          TERM_TYPE = .COB$TERM_TYPE ;
;  430          2017  4          COB$$POS_ERASE ( .TERM_TYPE, .ERASE_FLAG, .LINE, .COLUMN,
;  431          2018  4                                                    .LINE_PLUS, .COLUMN_PLUS ) ;
;  432          2019  3          END ;
;  433          2020  2      END ;
;  434          2021  1      END ;                                        ! End of COB$POS_DISPLAY



                                                003C 00000          .ENTRY  COB$POS_DISPLAY, Save R2,R3,R4,R5       ; 1902
                                    55 00000000G    00   9E 00002      MOVAB   LIB$STOP, R5                            ;
                                    54 00000000'    EF   9E 00009      MOVAB   COB$TERM_TYPE, R4                       ;
                                    52          04  AC   D0 00010      MOVL    UNIT, R2                                ; 1974
                                    06              52   D1 00014      CMPL    R2, #6                                  ;
                                                    09   1B 00017      BLEQU   1$                                      ;
                                       00000000G    8F   DD 00019      PUSHL   #COB$_INVARG                            ; 1976
                                    65              01   FB 0001F      CALLS   #1, LIB$STOP                            ;
                                    53 00000000G0042     DE 00022 1$:   MOVAL   COB$$AL_WRITE_RAB[R2], R3               ; 1978
                                                    63   D5 0002A      TSTL    (R3)                                    ;
                                                    0B   12 0002C      BNEQ    2$                                      ;
                                                    7E   D4 0002E      CLRL    -(SP)                                   ; 1983
                                                    52   DD 00030      PUSHL   R2                                      ;
                                 00000000G          00   FB 00032      CALLS   #2, COB$$OPEN_OUT                       ;
                                         08         A4   63   D0 00039 2$:   MOVL    (R3), RAB                       ; 1985
                                   50    08         A4 00000044  8F   C1 0003D      ADDL3   #68, RAB, NAM_DSC               ; 1998
                                                       64   D5 00046      TSTL    COB$TERM_TYPE                           ; 2000
                                                       1B   12 00048      BNEQ    3$                                      ;
                                                       54   DD 0004A      PUSHL   R4                                      ; 2002
                                             7E            60   3C 0004C      MOVZWL  (NAM_DSC), -(SP)                        ; 2003
                                                      04   A0   DD 0004F      PUSHL   4(NAM_DSC)                              ; 2002
                                 00000000G          00   03   FB 00052      CALLS   #3, COB$$SETUP_TERM_TYPE                ;
                                                    09   50   E8 00059      BLBS    R0, 3$                                  ;
                                          00000000G      8F   DD 0005C      PUSHL   #COB$_ERRDURDIS                         ; 2005
                                                    65   01   FB 00062      CALLS   #1, LIB$STOP                            ;
                                                    50   64   D0 00065 3$:   MOVL    COB$TERM_TYPE, R0                       ; 2013
                                                         12   13 00068      BEQL    4$                                      ;
                                                    7E        14  AC   7D 0006A      MOVQ    LINE_PLUS, -(SP)                ; 2018
                                                    7E        0C  AC   7D 0006E      MOVQ    LINE, -(SP)                     ; 2017
                                                              08  AC   DD 00072      PUSHL   ERASE_FLAG                      ;
                                                                   50   DD 00075      PUSHL   TERM_TYPE                       ;
                                               0000V        CF        06   FB 00077      CALLS   #6, COB$$POS_ERASE              ;
                                                                        04   0007C 4$:   RET                                 ; 2021

; Routine Size:  125 bytes,    Routine Base:  _COB$CODE + 0096
```

```
;  436          2022  1 %SBTTL 'COB$POS_ERASE - Position and erase'
;  437          2023  1 GLOBAL ROUTINE COB$POS_ERASE : NOVALUE =
;  438          2024  1
;  439          2025  1 !++
;  440          2026  1 ! FUNCTIONAL DESCRIPTION:
;  441          2027  1 !
;  442          2028  1 !       This routine is a place-holder routine needed by COBVECTOR.MAR.
;  443          2029  1 !
;  444          2030  1 ! CALLING SEQUENCE:
;  445          2031  1 !
;  446          2032  1 !       NONE
;  447          2033  1 !
;  448          2034  1 ! FORMAL PARAMETERS:
;  449          2035  1 !
;  450          2036  1 !       NONE
;  451          2037  1 !
;  452          2038  1 ! IMPLICIT INPUTS:
;  453          2039  1 !
;  454          2040  1 !       NONE
;  455          2041  1 !
;  456          2042  1 ! IMPLICIT OUTPUTS:
;  457          2043  1 !
;  458          2044  1 !       NONE
;  459          2045  1 !
;  460          2046  1 ! SIDE EFFECTS:
;  461          2047  1 !
;  462          2048  1 !       NONE
;  463          2049  1 !
;  464          2050  1 !--
;  465          2051  1
;  466          2052  2     BEGIN
;  467          2053  2     0;
;  468          2054  1     END ;


                                        0000 00000          .ENTRY  COB$POS_ERASE, Save nothing                 ; 2023
                                          04 00002          RET                                                 ; 2054

; Routine Size:  3 bytes,    Routine Base:  _COB$CODE + 0113
```

```
;  470          2055  1 %SBTTL 'COB$$POS_ERASE'
;  471          2056  1 ROUTINE COB$$POS_ERASE (
;  472          2057  1                                  TERM_TYPE,      ! Terminal type
;  473          2058  1                                  ERASE_FLAG,     ! type of erasing
;  474          2059  1                                  LINE,           ! line number
;  475          2060  1                                  COLUMN,         ! column number
;  476          2061  1                                  LINE_PLUS,      ! number of lines to advance
;  477          2062  1                                  COLUMN_PLUS     ! number of cols to advance
;  478          2063  1     ) =
;  479          2064  1
;  480          2065  1 !++
;  481          2066  1 ! FUNCTIONAL DESCRIPTION:
;  482          2067  1 !
;  483          2068  1 !       This routine is used with ACCEPT and DISPLAY.  Its purpose is to
;  484          2069  1 !       set the cursor to the specified line and column, and to erase
;  485          2070  1 !       a line or the whole screen if specified.
;  486          2071  1 !
;  487          2072  1 ! CALLING SEQUENCE:
;  488          2073  1 !
;  489          2074  1 !       ret_status.wlc.v = COB$$POS_ERASE (TERM_TYPE.rlu.v,
;  490          2075  1 !                                          ERASE_FLAG.rl.v,
;  491          2076  1 !                                          LINE.rlu.v,
;  492          2077  1 !                                          COLUMN.rlu.v,
;  493          2078  1 !                                          LINE_PLUS.rlu.v,
;  494          2079  1 !                                          COLUMN_PLUS.rlu.v)
;  495          2080  1 !
;  496          2081  1 ! FORMAL PARAMETERS:
;  497          2082  1 !
;  498          2083  1 !       TERM_TYPE.rlu.v         input or output terminal type
;  499          2084  1 !       ERASE_FLAG.rl.v         type of erasing to be done
;  500          2085  1 !                                       0 = no erasing
;  501          2086  1 !                                       1 = erase whole screen
;  502          2087  1 !                                       2 = erase whole line
;  503          2088  1 !                                       3 = erase to end of screen
;  504          2089  1 !                                       4 = erase to end of line
;  505          2090  1 !       LINE.rlu.v              line number to which cursor should be moved
;  506          2091  1 !                               if zero, the current line is retained
;  507          2092  1 !       COLUMN.rlu.v            column number to which cursor should be moved
;  508          2093  1 !                               if zero, the current column is retained
;  509          2094  1 !       LINE_PLUS.rlu.v         number of lines to advance with line-feeds
;  510          2095  1 !       COLUMN_PLUS.rlu.v       number of columns to advance
;  511          2096  1 !
;  512          2097  1 ! IMPLICIT INPUTS:
;  513          2098  1 !
;  514          2099  1 !       NONE
;  515          2100  1 !
;  516          2101  1 ! IMPLICIT OUTPUTS:
;  517          2102  1 !
;  518          2103  1 !       NONE
;  519          2104  1 !
;  520          2105  1 ! COMPLETION STATUS:
;  521          2106  1 !
;  522          2107  1 !       SS$_NORMAL      Normal successful completion
;  523          2108  1 !       COB$_INVARG     Invalid argument
;  524          2109  1 !       COB$_ERRDURPOS  Error during Positioning
;  525          2110  1 !
;  526          2111  1 ! SIDE EFFECTS:
```

```
;   527          2112  1 !
;   528          2113  1 !       NONE
;   529          2114  1 !
;   530          2115  1 !--
;   531          2116  1
;   532          2117  2     BEGIN
;   533          2118  2
;   534          2119  2     LOCAL
;   535          2120  2         ESC_SEQ_BUF : VECTOR [255, BYTE],       ! Buffer to store escape
;   536          2121  2                                                 ! sequences
;   537          2122  2         CUR_BUF_LEN : INITIAL (0),              ! Current length of ESC_SEQ_BUF
;   538          2123  2         ABS_CURSOR_FLAG : INITIAL (0),          ! Flag used w/vt52 erase
;   539          2124  2         TEMP_LINE_PLUS,                         ! Local for param LINE_PLUS
;   540          2125  2         STATUS;                                 ! Return status from calls
;   541          2126  2
;   542          2127  2     LITERAL
;   543          2128  2         K_MAX_RMS_LEN = 255,                    ! Max RMS output buffer size
;   544          2129  2         INIT_VALUE    = 9 ;                     ! Initial COB$$AB_PREV value
;   545          2130  2
;   546          2131  2 !+
;   547          2132  2 !   If COB$$AB_PREV says a LINEFEED is necessary, perform the lf now,
;   548          2133  2 !   then set COB$$AB_PREV to POS_DNA to signify that no further lf need be performed
;   549          2134  2 !   in either COB$DISPLAY or COB$ACCEPT.
;   550          2135  2 !   Move parameter LINE_PLUS to local TEMP_LINE_PLUS in case it will have to
;   551          2136  2 !   be incremented.
;   552          2137  2 !-
;   553          2138  2
;   554          2139  2     TEMP_LINE_PLUS = .LINE_PLUS ;
;   555          2140  2     IF .COB$$AB_PREV [0] EQL DISP  OR  .COB$$AB_PREV [0] EQL ACC_ADV
;   556          2141  2     THEN
;   557          2142  2     !+
;   558          2143  2     !   Echo linefeed for advancing via TEMP_LINE_PLUS.
;   559          2144  2     !-
;   560          2145  2         IF .LINE EQL 0
;   561          2146  2         THEN
;   562          2147  2             IF .TEMP_LINE_PLUS NEQ 0
;   563          2148  2             THEN
;   564          2149  2                 TEMP_LINE_PLUS = .TEMP_LINE_PLUS + 1
;   565          2150  2             ELSE
;   566          2151  2                 TEMP_LINE_PLUS = 1 ;
;   567          2152  2
;   568          2153  2     !+
;   569          2154  2     !   If COB$$AB_PREV is in its initial state do no advancing and leave
;   570          2155  2     !   it in the initial state if no positioning was requested.
;   571          2156  2     !-
;   572          2157  3     IF (.COB$$AB_PREV [0] EQL INIT_VALUE)
;   573          2158  3                         AND ( .LINE           EQL 0  AND  .COLUMN      EQL 0
;   574          2159  3                         AND   .TEMP_LINE_PLUS EQL 0  AND  .COLUMN_PLUS EQL 0 )
;   575          2160  2     THEN
;   576          2161  2         COB$$AB_PREV [0] = INIT_VALUE
;   577          2162  2     ELSE
;   578          2163  2         COB$$AB_PREV [0] = POS_DNA ;
;   579          2164  2
;   580          2165  2     COB$$AB_USPCODE [0] = 0;            ! turn off prefix carriage control
;   581          2166  2     COB$$AB_USPCODE [1] = 0;            ! turn off postfix carriage control
;   582          2167  2
;   583          2168  2 !+
```

```
;  584          2169  2 ! Put cursor positioning sequence in buffer.  The relative routine
;  585          2170  2 ! will do direct positioning if possible.
;  586          2171  2 !-
;  587          2172  2
;  588          2173  2     STATUS = COB$$SET_CURSOR_REL (.TERM_TYPE, ABS (.LINE),
;  589          2174  2                                   ABS (.COLUMN), ABS (.TEMP_LINE_PLUS),
;  590          2175  2                                   ABS (.COLUMN_PLUS), ESC_SEQ_BUF [0],
;  591          2176  2                                   CUR_BUF_LEN);
;  592          2177  2     IF (NOT .STATUS) THEN LIB$STOP (COB$_ERRDURPOS);
;  593          2178  2
;  594          2179  2 !+
;  595          2180  2 ! The relative cursor positioning routine may have been able to directly
;  596          2181  2 ! position the cursor.  This is significant only on a vt52 with whole
;  597          2182  2 ! screen erasing.
;  598          2183  2 !-
;  599          2184  2
;  600          2185  2     IF .LINE NEQ 0 AND
;  601          2186  2        .COLUMN NEQ 0
;  602          2187  2     THEN
;  603          2188  2         ABS_CURSOR_FLAG = 1;
;  604          2189  2
;  605          2190  2 !+
;  606          2191  2 ! It's possible for the buffer to be full already, particularly if
;  607          2192  2 ! relative cursor positioning was requested.  Check for that now,
;  608          2193  2 ! to make sure that an erase sequence would fit.  (This also avoids
;  609          2194  2 ! splitting an escape sequence between two $PUTs.)
;  610          2195  2 !-
;  611          2196  2
;  612          2197  3     IF .CUR_BUF_LEN GEQ (K_MAX_RMS_LEN - 10)
;  613          2198  2     THEN
;  614          2199  2         $OUTPUT_ESC_SEQ_BUF;
;  615          2200  2
;  616          2201  2 !+
;  617          2202  2 ! The set cursor sequence, if needed, is now in ESC_SEQ_BUF.  Append
;  618          2203  2 ! in the erase sequence, if one is requested.
;  619          2204  2 !-
;  620          2205  2
;  621          2206  2     IF .ERASE_FLAG NEQ 0
;  622          2207  2     THEN
;  623          2208  3         BEGIN
;  624          2209  3         CASE .ERASE_FLAG FROM 1 TO 4 OF
;  625          2210  3         SET
;  626          2211  3             [1]:                                    ! Erase whole screen
;  627          2212  4                 BEGIN
;  628          2213  4                 !+
;  629          2214  4                 ! Kludge for VT52.  If we erase the whole page, we will
;  630          2215  4                 ! lose our cursor position.  So put out the cursor sequence
;  631          2216  4                 ! again to restore it.
;  632          2217  4                 !-
;  633          2218  4                 IF .TERM_TYPE EQL VT52
;  634          2219  4                 THEN
;  635          2220  5                     BEGIN
;  636          2221  5                     IF .ABS_CURSOR_FLAG NEQ 0
;  637          2222  5                     THEN
;  638          2223  6                         BEGIN                       ! abs cursor, erase whole screen
;  639          2224  6                         STATUS = COB$$ERASE_WHOLE_PAGE_R2 (.TERM_TYPE,
;  640          2225  6                                                            ESC_SEQ_BUF [0],
```

```
;  641          2226  6                                                    CUR_BUF_LEN);
;  642          2227  6                          COB$$SET_CURSOR_REL (.TERM_TYPE, ABS (.LINE),
;  643          2228  6                                               ABS (.COLUMN), ABS (.TEMP_LINE_PLUS),
;  644          2229  6                                               ABS (.COLUMN_PLUS), ESC_SEQ_BUF [0],
;  645          2230  6                                               CUR_BUF_LEN);
;  646          2231  6                          END
;  647          2232  5                      ELSE
;  648          2233  6                          BEGIN                        ! rel cursor, erase to end
;  649          2234  6                          STATUS = COB$$ERASE_PAGE_R2 (.TERM_TYPE, ESC_SEQ_BUF [0],
;  650          2235  6                                                       CUR_BUF_LEN);
;  651          2236  5                          END;
;  652          2237  5                      END
;  653          2238  4                  ELSE
;  654          2239  4                      STATUS = COB$$ERASE_WHOLE_PAGE_R2 (.TERM_TYPE, ESC_SEQ_BUF [0],
;  655          2240  4                                                         CUR_BUF_LEN);
;  656          2241  3                  END;
;  657          2242  3              [2]:                                     ! Erase whole line
;  658          2243  3                  STATUS = COB$$ERASE_WHOLE_LINE_R2 (.TERM_TYPE, ESC_SEQ_BUF [0],
;  659          2244  3                                                     CUR_BUF_LEN);
;  660          2245  3
;  661          2246  3              [3]:                                     ! Erase from cursor to end of screen
;  662          2247  3                  STATUS = COB$$ERASE_PAGE_R2 (.TERM_TYPE, ESC_SEQ_BUF [0],
;  663          2248  3                                              CUR_BUF_LEN);
;  664          2249  3
;  665          2250  3              [4]:                                     ! Erase from cursor to end of line
;  666          2251  3                  STATUS = COB$$ERASE_LINE_R2 (.TERM_TYPE, ESC_SEQ_BUF [0],
;  667          2252  3                                              CUR_BUF_LEN);
;  668          2253  3             TES;
;  669          2254  3
;  670          2255  3          IF NOT .STATUS
;  671          2256  3          THEN LIB$STOP (COB$_ERRDURPOS);
;  672          2257  3
;  673          2258  2          END;
;  674          2259  2
;  675          2260  2 !+
;  676          2261  2 ! ESC_SEQ_BUF now contains everything.  Put it out to the terminal.
;  677          2262  2 !-
;  678          2263  2
;  679          2264  2      IF .CUR_BUF_LEN EQL 0                            ! nothing to output
;  680          2265  2      THEN
;  681          2266  2          RETURN (SS$_NORMAL);
;  682          2267  2
;  683          2268  2      $OUTPUT_ESC_SEQ_BUF;                             ! do RMS $PUT
;  684          2269  2
;  685          2270  2      RETURN (SS$_NORMAL);
;  686          2271  2
;  687          2272  1      END;                                             ! End of routine COB$$POS_ERASE


                                                    .EXTRN  SYS$PUT, SYS$WAIT

                                    0FFC 00000 COB$$POS_ERASE:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 2056
                5B 00000000G   00    9E 00002          MOVAB   LIB$STOP, R11                           ;
                5A 00000000G   8F    D0 00009          MOVL    #COB$_ERRDURPOS, R10                    ;
                59 00000000'   EF    9E 00010          MOVAB   RAB, R9                                 ;
```

```
                                 5E      FF00    CE  9E 00017          MOVAB   -256(SP), SP                                    ;
                                                 7E  D4 0001C          CLRL    CUR_BUF_LEN                                     ; 2117
                                                 52  D4 0001E          CLRL    ABS_CURSOR_FLAG                                 ;
                                 50        14    AC  D0 00020          MOVL    LINE_PLUS, TEMP_LINE_PLUS                       ; 2139
                                 51 00000000G    00  9A 00024          MOVZBL  COB$$AB_PREV, R1                                ; 2140
                                                 05  13 0002B          BEQL    1$                                              ;
                                 04              51  91 0002D          CMPB    R1, #4                                          ;
                                                 10  12 00030          BNEQ    3$                                              ;
                                         0C      AC  D5 00032 1$:      TSTL    LINE                                            ; 2145
                                                 0B  12 00035          BNEQ    3$                                              ;
                                                 50  D5 00037          TSTL    TEMP_LINE_PLUS                                  ; 2147
                                                 04  13 00039          BEQL    2$                                              ;
                                                 50  D6 0003B          INCL    TEMP_LINE_PLUS                                  ; 2149
                                                 03  11 0003D          BRB     3$                                              ;
                                 50              01  D0 0003F 2$:      MOVL    #1, TEMP_LINE_PLUS                              ; 2151
                                 09              51  91 00042 3$:      CMPB    R1, #9                                          ; 2157
                                                 1C  12 00045          BNEQ    4$                                              ;
                                         0C      AC  D5 00047          TSTL    LINE                                            ; 2158
                                                 17  12 0004A          BNEQ    4$                                              ;
                                         10      AC  D5 0004C          TSTL    COLUMN                                          ;
                                                 12  12 0004F          BNEQ    4$                                              ;
                                                 50  D5 00051          TSTL    TEMP_LINE_PLUS                                  ; 2159
                                                 0E  12 00053          BNEQ    4$                                              ;
                                         18      AC  D5 00055          TSTL    COLUMN_PLUS                                     ;
                                                 09  12 00058          BNEQ    4$                                              ;
                 00000000G       00              09  90 0005A          MOVB    #9, COB$$AB_PREV                                ; 2161
                                                 07  11 00061          BRB     5$                                              ;
                 00000000G       00              03  90 00063 4$:      MOVB    #3, COB$$AB_PREV                                ; 2163
                             00000000G           00  B4 0006A 5$:      CLRW    COB$$AB_USPCODE                                 ; 2165
                                                 5E  DD 00070          PUSHL   SP                                              ; 2175
                                         08      AE  9F 00072          PUSHAB  ESC_SEQ_BUF                                     ;
                                 57      18      AC  D0 00075          MOVL    COLUMN_PLUS, R7                                 ;
                                                 03  18 00079          BGEQ    6$                                              ;
                                 57              57  CE 0007B          MNEGL   R7, R7                                          ;
                                                 57  DD 0007E 6$:      PUSHL   R7                                              ;
                                 56              50  D0 00080          MOVL    TEMP_LINE_PLUS, R6                              ; 2174
                                                 03  18 00083          BGEQ    7$                                              ;
                                 56              56  CE 00085          MNEGL   R6, R6                                          ;
                                                 56  DD 00088 7$:      PUSHL   R6                                              ; 2175
                                 55      10      AC  D0 0008A          MOVL    COLUMN, R5                                      ; 2174
                                                 03  18 0008E          BGEQ    8$                                              ;
                                 55              55  CE 00090          MNEGL   R5, R5                                          ;
                                                 55  DD 00093 8$:      PUSHL   R5                                              ; 2175
                                 54      0C      AC  D0 00095          MOVL    LINE, R4                                        ; 2173
                                                 03  18 00099          BGEQ    9$                                              ;
                                 54              54  CE 0009B          MNEGL   R4, R4                                          ;
                                                 54  DD 0009E 9$:      PUSHL   R4                                              ; 2175
                                 53      04      AC  D0 000A0          MOVL    TERM_TYPE, R3                                   ; 2173
                                                 53  DD 000A4          PUSHL   R3                                              ; 2175
                 00000000G       00              07  FB 000A6          CALLS   #7, COB$$SET_CURSOR_REL                         ;
                                 58              50  D0 000AD          MOVL    R0, STATUS                                      ;
                                 05              58  E8 000B0          BLBS    STATUS, 10$                                     ; 2177
                                                 5A  DD 000B3          PUSHL   R10                                             ;
                                 6B              01  FB 000B5          CALLS   #1, LIB$STOP                                    ;
                                         0C      AC  D5 000B8 10$:     TSTL    LINE                                            ; 2185
                                                 08  13 000BB          BEQL    11$                                             ;
                                         10      AC  D5 000BD          TSTL    COLUMN                                          ; 2186
```

```
                                         03  13 000C0        BEQL    11$                                ;
                              52         01  D0 000C2        MOVL    #1, ABS_CURSOR_FLAG                ; 2188
                   000000F5   8F         6E  D1 000C5 11$:   CMPL    CUR_BUF_LEN, #245                  ; 2197
                                         50  19 000CC        BLSS    16$                                ;
                              50         69  D0 000CE        MOVL    RAB, R0                            ; 2198
                        28    A0    04   AE  9E 000D1        MOVAB   ESC_SEQ_BUF, 40(R0)                ;
                              51         6E  D0 000D6        MOVL    CUR_BUF_LEN, R1                    ;
                   000000FF   8F         51  D1 000D9        CMPL    R1, #255                           ;
                                         04  15 000E0        BLEQ    12$                                ;
                              51    FF   8F  9A 000E2        MOVZBL  #255, R1                           ;
                        22    A0         51  B0 000E6 12$:   MOVW    R1, 34(R0)                         ;
                                         69  DD 000EA 13$:   PUSHL   RAB                                ;
                   00000000G  00         01  FB 000EC        CALLS   #1, SYS$PUT                        ;
                   000182DA   8F         50  D1 000F3        CMPL    R0, #99034                         ;
                                         0B  12 000FA        BNEQ    14$                                ;
                                         69  DD 000FC        PUSHL   RAB                                ;
                   00000000G  00         01  FB 000FE        CALLS   #1, SYS$WAIT                       ;
                                         E3  11 00105        BRB     13$                                ;
                              50         69  D0 00107 14$:   MOVL    RAB, R0                            ;
                              0E    08   A0  E8 0010A        BLBS    8(R0), 15$                         ;
                              7E    08   A0  7D 0010E        MOVQ    8(R0), -(SP)                       ;
                                    44   A0  9F 00112        PUSHAB  68(R0)                             ;
                                         01  DD 00115        PUSHL   #1                                 ;
                                         5A  DD 00117        PUSHL   R10                                ;
                              6B         05  FB 00119        CALLS   #5, LIB$STOP                       ;
                                         6E  D4 0011C 15$:   CLRL    CUR_BUF_LEN                        ;
                                    08   AC  D5 0011E 16$:   TSTL    ERASE_FLAG                         ; 2206
                                         03  12 00121        BNEQ    17$                                ;
                                       008C  31 00123        BRW     26$                                ;
                 03           01    08   AC  CF 00126 17$:   CASEL   ERASE_FLAG, #1, #3                 ; 2209
006C             0036       005A       0008     0012B 18$:   .WORD   19$-18$,-                          ;
                                                                     22$-18$,-                          ;
                                                                     20$-18$,-                          ;
                                                                     23$-18$                            ;
                              02         53  D1 00133 19$:   CMPL    R3, #2                             ; 2218
                                         3B  12 00136        BNEQ    21$                                ;
                                         52  D5 00138        TSTL    ABS_CURSOR_FLAG                    ; 2221
                                         25  13 0013A        BEQL    20$                                ;
                              52         6E  9E 0013C        MOVAB   CUR_BUF_LEN, R2                    ; 2225
                              51    04   AE  9E 0013F        MOV'    ESC_SEQ_BUF, R1                    ;
                              50         53  D0 00143        MOV     R3, R0                             ;
                                 00000000G  00  16 00146     JSc     COB$$ERASE_WHOLE_PAGE_R2           ;
                              58         50  D0 0014C        MOVL    R0, STATUS                         ;
                                         5E  DD 0014F        PUSHL   SP                                 ; 2229
                                    08   AE  9F 00151        PUSHAB  ESC_SEQ_BUF                        ;
                                  00F8   8F  BB 00154        PUSHR   #^M<R3,R4,R5,R6,R7>                ;
                   00000000G  00         07  FB 00158        CALLS   #7, COB$$SET_CURSOR_REL            ;
                                         49  11 0015F        BRB     25$                                ; 2221
                              52         6E  9E 00161 20$:   MOVAB   CUR_BUF_LEN, R2                    ; 2234
                              51    04   AE  9E 00164        MOVAB   ESC_SEQ_BUF, R1                    ;
                              50         53  D0 00168        MOVL    R3, R0                             ;
                                 00000000G  00  16 0016B     JSB     COB$$ERASE_PAGE_R2                 ;
                                         34  11 00171        BRB     24$                                ;
                              52         6E  9E 00173 21$:   MOVAB   CUR_BUF_LEN, R2                    ; 2239
                              51    04   AE  9E 00176        MOVAB   ESC_SEQ_BUF, R1                    ;
                              50         53  D0 0017A        MOVL    R3, R0                             ;
                                 00000000G  00  16 0017D     JSB     COB$$ERASE_WHOLE_PAGE_R2           ;
```

```
                                 22  11 00183                BRB     24$                                     ;
                              52 6E  9E 00185 22$:           MOVAB   CUR_BUF_LEN, R2                         ; 2243
                              51 04  AE 9E 00188             MOVAB   ESC_SEQ_BUF, R1                         ;
                              50     53 D0 0018C             MOVL    R3, R0                                  ;
                    00000000G       00 16 0018F             JSB     COB$$ERASE_WHOLE_LINE_R2                ;
                                    10 11 00195             BRB     24$                                     ;
                              52 6E  9E 00197 23$:           MOVAB   CUR_BUF_LEN, R2                         ; 2251
                              51 04  AE 9E 0019A             MOVAB   ESC_SEQ_BUF, R1                         ;
                              50     53 D0 0019E             MOVL    R3, R0                                  ;
                    00000000G       00 16 001A1             JSB     COB$$ERASE_LINE_R2                      ;
                              58     50 D0 001A7 24$:        MOVL    R0, STATUS                              ;
                              05     58 E8 001AA 25$:        BLBS    STATUS, 26$                             ; 2255
                                    5A DD 001AD             PUSHL   R10                                     ; 2256
                              6B    01 FB 001AF             CALLS   #1, LIB$STOP                            ;
                                    6E D5 001B2 26$:        TSTL    CUR_BUF_LEN                             ; 2264
                                    50 13 001B4             BEQL    31$                                     ;
                              50    69 D0 001B6             MOVL    RAB, R0                                 ; 2266
                        28    A0 04  AE 9E 001B9             MOVAB   ESC_SEQ_BUF, 40(R0)                     ;
                              51    6E D0 001BE             MOVL    CUR_BUF_LEN, R1                         ;
                 000000FF     8F    51 D1 001C1             CMPL    R1, #255                                ;
                                    04 15 001C8             BLEQ    27$                                     ;
                              51 FF 8F 9A 001CA             MOVZBL  #255, R1                                ;
                        22    A0    51 B0 001CE 27$:        MOVW    R1, 34(R0)                              ;
                                    69 DD 001D2 28$:        PUSHL   RAB                                     ;
                 00000000G    00    01 FB 001D4             CALLS   #1, SYS$PUT                             ;
                 000182DA     8F    50 D1 001DB             CMPL    R0, #99034                              ;
                                    0B 12 001E2             BNEQ    29$                                     ;
                                    69 DD 001E4             PUSHL   RAB                                     ;
                 00000000G    00    01 FB 001E6             CALLS   #1, SYS$WAIT                            ;
                                    E3 11 001ED             BRB     28$                                     ;
                              50    69 D0 001EF 29$:        MOVL    RAB, R0                                 ;
                              0E 08 A0 E8 001F2             BLBS    8(R0), 30$                              ;
                              7E 08 A0 7D 001F6             MOVQ    8(R0), -(SP)                            ;
                                 44 A0 9F 001FA             PUSHAB  68(R0)                                  ;
                                    01 DD 001FD             PUSHL   #1                                      ;
                                    5A DD 001FF             PUSHL   R10                                     ;
                              6B    05 FB 00201             CALLS   #5, LIB$STOP                            ;
                                    6E D4 00204 30$:        CLRL    CUR_BUF_LEN                             ;
                              50    01 D0 00206 31$:        MOVL    #1, R0                                  ; 2270
                                       04 00209             RET                                             ; 2272

; Routine Size:  522 bytes,    Routine Base:  _COB$CODE + 0116

;  688          2273  1 END                                          ! End of module COB$POS_ERASE
;  689          2274  0 ELUDOM


;                                    PSECT SUMMARY
;
;      Name                      Bytes                          Attributes
;
;  _COB$DATA                       12  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
;  _COB$CODE                          800  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                                Library Statistics
;
;                                          -------- Symbols --------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1         9776        20         0         581        00:00.7
;  _$255$DUA28:[COBRTL.OBJ]SMGLIB.L32;1       469         2         0          38        00:00.3




;                                      COMMAND QUALIFIERS

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

; Size:         800 code + 12 data bytes
; Run Time:       00:13.5
; Elapsed Time:   00:51.0
; Lines/CPU Min:  10084
; Lexemes/CPU-Min: 27104
; Memory Used:  187 pages
; Compilation Complete
```

COBPAUSE
LIS

COBMSG
LIS

COBHANDLE
LIS

COBINTAR
LIS

COBINTER
LIS

COBMULQ
LIS

COBPOSERA
LIS

COBIOEXCE
LIS

COBLINAGE
LIS

COBKEY
LIS

COBINVUSE
LIS

COBRMSBLO
LIS

COBSPANC
LIS

COBSWITCH
LIS

COBRESTVA
LIS

CONVSHR
MAP

CONVCOMIO
LIS

COBSETSWI
LIS

CONVDEF
MDL

CONVLINK
REQ

COBSPANC2
LIS

CONV

CONVERT
MAP

CONVCALL
LIS

COBVECTOR
LIS

CONVERT
REQ

RECLAIM
MAP

ADDKEY
LIS