```
EEEEEEEEEEEEEEE  DDDDDDDDDDD     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFFFFFFFFFFFFFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEEEEEEEEEEE     DDD         DDD  FFFFFFFFFFFF
EEEEEEEEEEEE     DDD         DDD  FFFFFFFFFFFF
EEEEEEEEEEEE     DDD         DDD  FFFFFFFFFFFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEE              DDD         DDD  FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDD     FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFF
```

```
Va
--
00
00
00
00
00
00
00
00
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
```

```
EEEEEEEEEE  DDDDDDDD    FFFFFFFFFF    GGGGGGGG  RRRRRRRR    FFFFFFFFFF
EEEEEEEEEE  DDDDDDDD    FFFFFFFFFF    GGGGGGGG  RRRRRRRR    FFFFFFFFFF
EE          DD      DD  FF          GG          RR      RR  FF
EE          DD      DD  FF          GG          RR      RR  FF
EE          DD      DD  FF          GG          RR      RR  FF
EE          DD      DD  FF          GG          RR      RR  FF
EEEEEEEE    DD      DD  FFFFFFFF    GG          RRRRRRRR    FFFFFFFF
EEEEEEEE    DD      DD  FFFFFFFF    GG          RRRRRRRR    FFFFFFFF
EE          DD      DD  FF          GG  GGGGGG  RR  RR      FF
EE          DD      DD  FF          GG  GGGGGG  RR  RR      FF
EE          DD      DD  FF          GG      GG  RR    RR    FF              ....
EE          DD      DD  FF          GG      GG  RR    RR    FF              ....
EEEEEEEEEE  DDDDDDDD    FF            GGGGGG    RR      RR  FF              ....
EEEEEEEEEE  DDDDDDDD    FF            GGGGGG    RR      RR  FF              ....

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 'EDF$GRAPH plotting module'
;     2          0002  0 MODULE EDF$GRAPH (
;     3          0003  0                   IDENT = 'V04-000',
;     4          0004  0                   ADDRESSING_MODE ( EXTERNAL = GENERAL ),
;     5          0005  0                   ADDRESSING_MODE ( NONEXTERNAL = GENERAL ),
;     6          0006  0                   OPTLEVEL=3
;     7          0007  0                    )  =
;     8          0008  1 BEGIN
;     9          0009  1
;    10          0010  1 !++
;    11          0011  1 !*****************************************************************************
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    14          0014  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    15          0015  1 !*  ALL RIGHTS RESERVED.                                                      *
;    16          0016  1 !*                                                                           *
;    17          0017  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    18          0018  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    19          0019  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    20          0020  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    21          0021  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    22          0022  1 !*  TRANSFERRED.                                                              *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    25          0025  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    26          0026  1 !*  CORPORATION.                                                              *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    29          0029  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    30          0030  1 !*                                                                           *
;    31          0031  1 !*                                                                           *
;    32          0032  1 !*****************************************************************************
;    33          0033  1 !
;    34          0034  1 !
;    35          0035  1 ! FACILITY:              VAX/VMS EDF (EDIT/FDL) UTILITY
;    36          0036  1 !
;    37          0037  1 ! ABSTRACT:              This facility is used to create, modify, and optimize
;    38          0038  1 !                        FDL specification files.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:           NATIVE/USER MODE
;    41          0041  1 !
;    42          0042  1 ! AUTHOR:                Tamar Krichevsky
;    43          0043  1 !
;    44          0044  1 ! CREATION DATE:         31-Jul-1981
;    45          0045  1 !
;    46          0046  1 ! MODIFIED BY:
;    47          0047  1 !
;    48          0048  1 !                        V03-002 KFH0002         Ken Henderson   20 Jan 1983
;    49          0049  1 !                                Fixed definition of xxx_INTENSITY
;    50          0050  1 !                                constants.
;    51          0051  1 !
;    52          0052  1 !                        V03-001 KFH0001         Ken Henderson   22 Nov 1982
;    53          0053  1 !                                Commented out references to
;    54          0054  1 !                                EDF$_INTSWERR; added OPTLEVEL=3
;    55          0055  1 !                                Add require of lib$:edfstruct
;    56          0056  1 !
;    57          0057  1 ! --
```

```
;    59          0058  1 !
;    60          0059  1 ! TABLE OF CONTNETS
;    61          0060  1 !
;    62          0061  1
;    63          0062  1 FORWARD ROUTINE
;    64          0063  1     EDF$GRAPH           : NOVALUE,      !Driver for the plot routines
;    65          0064  1     GET_GRAPH_INFO      : NOVALUE,      !Determines plot characteristics
;    66          0065  1     PLOT_LINE_GRAPH     : NOVALUE,      !Plots a simple graph (line graph)
;    67          0066  1     MOVE_LINE_GRAPH     : NOVALUE,      !Changes the shape of a simple graph
;    68          0067  1     PLOT_SURFACE_GRAPH: NOVALUE,        !Plots a tabular graph (surface plot)
;    69          0068  1     ERASE_PAGE          : NOVALUE,      !Erases a screen
;    70          0069  1     DRAW_Y_AXIS         : NOVALUE,      !Draws the y-axis line
;    71          0070  1     LABEL_Y_AXIS        : NOVALUE,      !Puts the y-axis descriptive labels out
;    72          0071  1     PUT_ROW_SEGMENT     : NOVALUE,      !Put a segment of a graph's row to the terminal
;    73          0072  1     MOVE_CURSOR_REGIS : NOVALUE,        !Position the cursor (REGIS terminals)
;    74          0073  1     PUT_REGIS_TEXT      : NOVALUE,      !Build and put a REGIS text command
;    75          0074  1     SHADE_ROW_REGIS     : NOVALUE,      !Shade a segment of the graph -- VT125
;    76          0075  1     DRAW_BARS_REGIS     : NOVALUE,      !Draw the bars for the VT125 line graph
;    77          0076  1     DRAW_X_AXIS         : NOVALUE       !Format x-axis lablels
;    78          0077  1     ;
;    79          0078  1
;    80          0079  1
;    81          0080  1 !
;    82          0081  1 ! INCLUDE FILES
;    83          0082  1 !
;    84          0083  1
;    85          0084  1
;    86          0085  1 LIBRARY
;    87          0086  1     'SYS$LIBRARY:STARLET';
;    88          0087  1 REQUIRE
;    89          0088  1     'LIB$:EDFSTRUCT';
;    90          0382  1
;    91          0383  1 !
;    92          0384  1 ! MACROS
;    93          0385  1 !
;    94          0386  1
;    95          0387  1 MACRO
;    96          0388  1     DSC$L_U1 = 32, 0, 32, 0 %,  !Field in the array descriptor for the first
;    97          0389  1                                 !dimension's upper bound
;    98          0390  1     DSC$L_U2 = 40, 0, 32, 0 %,  !Field in the array descriptor for the second
;    99          0391  1                                 !dimension's upper bound
;   100          0392  1
;   101          0393  1
;   102          0394  1     !Translate the given value into ASCII.
;   103        M 0395  1     TRANSLATE_VALUE ( CTRSTR_ADR, RESULT_LEN, DESC_ADR, PARAM_LIST) =
;   104        M 0396  1         BEGIN
;   105        M 0397  1         LOCAL  RTN_STATUS;      !Return status from $FAOL system service
;   106        M 0398  1         !Translate, and if not successful, then signal the error.
;   107        M 0399  1         IF NOT ( RTN_STATUS = $FAOL( CTRSTR = CTRSTR_ADR,
;   108        M 0400  1                                      OUTLEN = RESULT_LEN,
;   109        M 0401  1                                      OUTBUF = DESC_ADR,
;   110        M 0402  1                                      PRMLST = PARAM_LIST ))
;   111        M 0403  1         THEN
;   112        M 0404  1             SIGNAL( .RTN_STATUS );
;   113        M 0405  1         END
;   114          0406  1     %,
;   115          0407  1
```

```
;  116          0408  1
;  117          0409  1      !Put the given text to regested location on the screen.
;  118        M 0410  1      PUT_TEXT( DESC_ADR, ROW, COLUMN ) =
;  119        M 0411  1          BEGIN
;  120        M 0412  1          LOCAL RTN_STATUS;       !Return status from RTL call
;  121        M 0413  1          IF NOT ( RTN_STATUS = LIB$PUT_SCREEN( DESC_ADR,
;  122        M 0414  1                                                ROW,
;  123        M 0415  1                                                COLUMN
;  124        M 0416  1                                               ))
;  125        M 0417  1          THEN
;  126        M 0418  1              SIGNAL( .RTN_STATUS );
;  127        M 0419  1          END
;  128          0420  1      %,
;  129          0421  1
;  130          0422  1
;  131          0423  1      !Put the given text to the terminal; let REGIS position it.
;  132        M 0424  1      PUT_REGIS( DESC_ADR ) =
;  133        M 0425  1          BEGIN
;  134        M 0426  1          LOCAL RTN_STATUS;       !Return status from RTL call
;  135        M 0427  1          IF NOT ( RTN_STATUS = LIB$PUT_OUTPUT( DESC_ADR ))
;  136        M 0428  1          THEN
;  137        M 0429  1              SIGNAL( .RTN_STATUS );
;  138        M 0430  1          END
;  139          0431  1      %
;  140          0432  1
;  141          0433  1 ;
;  142          0434  1
```

```
;  144     0435  1 !
;  145     0436  1 ! EQUATED SYMBOLS
;  146     0437  1 !
;  147     0438  1
;  148     0439  1 LITERAL
;  149     0440  1     TRUE                = 1,
;  150     0441  1     FALSE               = 0,
;  151     0442  1     SUCCESS             = 1,
;  152     0443  1     FAILURE             = 0,
;  153     0444  1     ALL_BITS_SET        = 255,  !Unsigned -1, used for comparisons
;  154     0445  1     FULLWORD            = 4,    !Number of bytes in a BLISS-32 fullword
;  155     0446  1     QUAD                = 8,    !Size of a quad word -- for declarations
;  156     0447  1     NULL                =       !Longword of null characters
;  157     0448  1                           %CHAR(0,0,0,0)
;  158     0449  1     ;
;  159     0450  1
;  160     0451  1 LITERAL
;  161     0452  1 !    INVALID_GRAPH_CODE  = 100, !Signal arguement - graph code is off the wall
;  162     0453  1 !    INVALID_LABEL       = 101, !Signal arguement - y-axis label is too long
;  163     0454  1     FIRST_ROW           = 1,    !Position on screen of first row
;  164     0455  1     FIRST_COLUMN        = 1,    !Position on screen of first column
;  165     0456  1
;  166     0457  1     NO_LAST_INDEX       = -1,   !LAST_INDEX value -- signals that graph is to
;  167     0458  1                                 !be drawn with axes (as opposed to moved)
;  168     0459  1     NO_VALUE            = -1,   !Signals that or y of the pixel address is to
;  169     0460  1                                 !be left out (i.e. - [,y] or [x,] )
;  170     0461  1
;  171     0462  1     MAX_PAGE_WIDTH      = 132,  !Maximum number of columns on a page
;  172     0463  1     SCREEN_BUFFER_SIZE  = 512,  !Size of buffer for formatting graphs
;  173     0464  1     MAX_BUCKET_SIZE     = 32,   !Maximum number of block a bucket can contain
;  174     0465  1     SURFACE_GRAPH_LEN   = 19,   !Maximum number of rows in a tabular graph
;  175     0466  1     LINE_GRAPH_LEN      = 10,   !Maximum number of rows in a line graph
;  176     0467  1     SEPARATOR_WIDTH     = 2,    !Number columns a plotted value occupies
;  177     0468  1     HEADER_HIEGHT       = 3,    !Number of lines in the x-axis header
;  178     0469  1
;  179     0470  1 !    GRF$_LINE           =  0,   !Code for line  graph
;  180     0471  1 !    GRF$_SRF_INCREASING =  1,   !Code for surface graphs with increasing values
;  181     0472  1 !    GRF$_SRF_DECREASING =  2,   !Code for surface graphs with decreasing values
;  182     0473  1     GRF$_LINE           =  EDF$C_LINE,  !Code for line  graph
;  183     0474  1     GRF$_SRF_INCREASING =  EDF$C_SRF_INCREASING,   !Code for surface graphs with increasing values
;  184     0475  1     GRF$_SRF_DECREASING =  EDF$C_SRF_DECREASING,        !Code for surface graphs with decreasing values
;  185     0476  1
;  186     0477  1     GRF$_REGIS_POS      =  0,   !Code for position command
;  187     0478  1     GRF$_REGIS_VCTR     =  1,   !Code for vector command
;  188     0479  1     GRF$_REGIS_SHADE    =  2,   !Code for shaded vector command
;  189     0480  1
;  190     0481  1     CHAR_WIDTH          =  9,   !Width, in pixels,  of a text-character cell
;  191     0482  1     CHAR_HEIGHT         = 15,   !Height, in pixels, of a text_character cell
;  192     0483  1     UNIT_WIDTH          = 18,   !Width, in pixels, of a column
;  193     0484  1     UNIT_HEIGHT         = 18,   !Height, in pixels, of a row
;  194     0485  1     RIGHT_SIDE_LOC      = 767,  !Right most pixel address
;  195     0486  1     BOTTOM_LOC          = 479,  !Bottom most pixel address
;  196     0487  1     AXIS_SHIFT          = 5,    !Number of pixels from origin axes are located
;  197     0488  1
;  198     0489  1 !    BACKGROUND_INTENSITY= '0', !Code for background intensity
;  199     0490  1 !    LIGHT_INTENSITY     = '3', !Code for light color (green on color monitor)
;  200     0491  1 !    MEDIUM_INTENSITY    = '2', !Code for medium color (yellow)
```

```
;  201          0492  1 !     DARK_INTENSITY      = '1',  !code for dark color (red)
;  202          0493  1      BACKGROUND_INTENSITY= '0' + EDF$C_BACKGROUND_COLOR, !Code for background intensity
;  203          0494  1      LIGHT_INTENSITY     = '0' + EDF$C_LIGHT_GREEN,      !Code for light color (green on color monitor)
;  204          0495  1      MEDIUM_INTENSITY    = '0' + EDF$C_MEDIUM_YELLOW,    !Code for medium color (yellow)
;  205          0496  1      DARK_INTENSITY      = '0' + EDF$C_DARK_RED,         !code for dark color (red)
;  206          0497  1
;  207          0498  1                                  !Location of graph's x-origin on a VT125
;  208          0499  2      X_ORIGIN            = RIGHT_SIDE_LOC - (UNIT_WIDTH * MAX_BUCKET_SIZE)
;  209          0500  1      ;
;  210          0501  1
;  211          0502  1 BIND
;  212        L 0503  1      REGIS_SET_UP = %ASCID %STRING(
;  213        L 0504  1                                  !Set terminal into REGIS mode
;  214        L 0505  1                               %CHAR(27),'Pp',
;  215        L 0506  1                               !Set default screen characteristics
;  216        L 0507  1                               'S[0,0] (A[0,0] [767,479], I(d), S1)',
;  217        L 0508  1                               !Set default writing characteristics
;  218        L 0509  1                               'W(V, I(w), F3, M1, N0, P1, P(M2), S0)'
;  219          0510  1                             ),
;  220          0511  1
;  221          0512  1                                  !Change output color map to RED, YELLOW
;  222          0513  1                                  !and GREEN colors to signify good, fair
;  223          0514  1                                  !poor regions of the surface graph.OP
;  224        L 0515  1      COLOR_SET_RYG = %ASCID %STRING(
;  225        L 0516  1                               'S( M0 (l0)   (a      l0)',      ! background, black
;  226        L 0517  1                               '   M1 (l35)  (a h120 l30 s30)', ! dark RED
;  227        L 0518  1                               '   M2 (l65)  (a h150 l70 s50)', ! medium YELLOW
;  228        L 0519  1                               '   M3 (l100) (a h240 l35 s30))' ! light GREEN
;  229          0520  1                             ),
;  230          0521  1
;  231          0522  1                                  !Change output color mapping to a
;  232          0523  1                                  !more pleasing set of colors
;  233        L 0524  1      COLOR_SET_BLUE = %ASCID %STRING(
;  234        L 0525  1                               'S( M0 (l0)   (a      l0)',      ! background, black
;  235        L 0526  1                               '   M1 (l35)  (a h5   l30 s20)', ! dark blue
;  236        L 0527  1                               '   M2 (l65)  (a h325 l60 s60)', ! medium blue
;  237        L 0528  1                               '   M3 (l100) (a h245 l80 s10))' ! light blue
;  238          0529  1                             ),
;  239          0530  1
;  240          0531  1
;  241          0532  1      REGIS_ON    =                  !Set terminal into REGIS mode
;  242          0533  1                   %ASCID %STRING(  %CHAR(27),'Pp' ),
;  243          0534  1
;  244          0535  1      REGIS_OFF   =                  !Turn off regis mode
;  245          0536  1                   %ASCID %STRING( %CHAR(27), '\' )
;  246          0537  1      ;
```

```
;  248     0538  1 !
;  249     0539  1 ! Structure declarations used for system defined structures to
;  250     0540  1 ! save typing.  These structures are byte sized.
;  251     0541  1 !
;  252     0542  1
;  253     0543  1 STRUCTURE
;  254     0544  1     BBLOCK [O, P, S, E; N] =
;  255     0545  1         [N]
;  256     0546  1         (BBLOCK+O)<P,S,E>,
;  257     0547  1
;  258     0548  1     BBLOCKVECTOR [I, O, P, S, E; N, BS] =
;  259     0549  1         [N*BS]
;  260     0550  1         ((BBLOCKVECTOR+I*BS)+O)<P,S,E>,
;  261     0551  1
;  262     0552  1     TWO_DIM_ARRAY[ ROW, COL, DESC_ADR; S, E ] =
;  263     0553  1         [ 0 ]
;  264     0554  2             BEGIN
;  265     0555  2             LOCAL DESC_ADR_LCL : REF BBLOCK;
;  266     0556  2
;  267     0557  2             DESC_ADR_LCL = DESC_ADR;
;  268     0558  3             (.DESC_ADR_LCL[ DSC$A_POINTER ] +
;  269     0559  2                 ( ROW * .DESC_ADR_LCL[ DSC$L_M2 ] + COL ) * %UPVAL)<0, S, E>
;  270     0560  2             END
;  271     0561  1         ;
```

```
;  273     0562  1 !
;  274     0563  1 ! OWN STORAGE
;  275     0564  1 !
;  276     0565  1
;  277     0566  1 OWN
;  278     0567  1     !The following information is needed each time EDF$GRAPH is entered.
;  279     0568  1     !Therefore, it is maintained across calls.
;  280     0569  1     DEVICE_TYPE  : BYTE INITIAL( ALL_BITS_SET ),!Type of terminal for this session
;  281     0570  1     DEVICE_FLAGS : BLOCK[ 1 ],                  !DEC supported terminal
;  282     0571  1     Y_AXIS_LINE                                 !Column on which y-axis falls
;  283     0572  1     ;
;  284     0573  1
;  285     0574  1 !
;  286     0575  1 ! EXTERNAL REFERENCES
;  287     0576  1 !
;  288     0577  1 !EXTERNAL LITERAL
;  289     0578  1 !    EDF$_INTSWERR                          !Message code for internal software
;  290     0579  1 !    ;                                              !error.  This value is used when
;  291     0580  1                                            !signalling a wrong graph code or
;  292     0581  1                                            !peculiar y-axis label.
;  293     0582  1
;  294     0583  1
;  295     0584  1 EXTERNAL ROUTINE
;  296     0585  1     LIB$SCREEN_INFO,                       !Retrieve terminal characteristics.
;  297     0586  1     LIB$SET_BUFFER,                        !Set buffer mode on.
;  298     0587  1     LIB$ERASE_PAGE,                        !Erase the screen.
;  299     0588  1     LIB$PUT_SCREEN,                        !Write output to screen.
;  300     0589  1     LIB$PUT_BUFFER,                        !Set buffer mode off.
;  301     0590  1     LIB$PUT_OUTPUT                         !Write a line of output to SYS$OUTPUT
;  302     0591  1     ;
```

```
;  304          0592  1 %SBTTL 'EDF$GRAPH main routine'
;  305          0593  1 GLOBAL ROUTINE EDF$GRAPH( GRAPH_TYPE, XY_ARRAY_DESC, CURRENT_INDEX, LAST_INDEX,
;  306          0594  1                           Y_HIGH,     Y_LOW,          Y_INCR,
;  307          0595  1                           Y_LABEL_DESC,               SHADE_ARRAY_DESC
;  308          0596  1                         ) : NOVALUE =
;  309          0597  1
;  310          0598  1 !++
;  311          0599  1 !
;  312          0600  1 ! FUNCTIONAL DESCRIPTION:
;  313          0601  1 !
;  314          0602  1 !       Determine the user's terminal type and characteristics.  Set up a
;  315          0603  1 !       buffer to hold the graph as it is being created.  Plot the requested
;  316          0604  1 !       graph.  (Line, Surface)  Put the graph to the user's terminal.
;  317          0605  1 !
;  318          0606  1 !       If a line graph is requested and the last index (LAST_INDEX
;  319          0607  1 !       parameter) equals -1, then the plot will be drawn with the axes and
;  320          0608  1 !       labels; if the previous index (LAST_INDEX) is 0 or greater, then
;  321          0609  1 !       the graph will be "moved" -- only the inside of the graph will be
;  322          0610  1 !       redrawn.  Should the user's terminal be something other than a
;  323          0611  1 !       DIGITAL supported video terminal, hard copy or foreign terminal,
;  324          0612  1 !       the graph will always be completely replotted.
;  325          0613  1 !
;  326          0614  1 !
;  327          0615  1 ! FORMAL PARAMETERS:
;  328          0616  1 !
;  329          0617  1 !       GRAPH_TYPE      : one byte code which determines which type of graph
;  330          0618  1 !                         will be plotted on this code.
;  331          0619  1 !       XY_ARRAY_DESC   : address of the array descriptor which points to the
;  332          0620  1 !                         array containing the data to be plotted.
;  333          0621  1 !       CURRENT_INDEX   : the current index on the y axsis of the array which
;  334          0622  1 !                         contains the data for the requested line  graph.
;  335          0623  1 !       Y_LOW, Y_HIGH, Y_INCR : the low, high and increment values for the y
;  336          0624  1 !                         axsis.
;  337          0625  1 !       XY_ARRAY_DESC   : address of the array descriptor which points to the
;  338          0626  1 !                         array containing the shading information for VT125
;  339          0627  1 !                         surface plots.
;  340          0628  1 !
;  341          0629  1 !
;  342          0630  1 ! IMPLICIT INPUTS:
;  343          0631  1 !
;  344          0632  1 !       DEVICE_FLAGS :  a longword of flags which are set or cleared by
;  345          0633  1 !                       LIB$SCREEN_INFO depending on the terminal
;  346          0634  1 !                       characteristics.
;  347          0635  1 !       DEVICE_TYPE  :  also returned by LIB$SCREEN_INFO.  If all bits are set,
;  348          0636  1 !                       then this is the first time EDF$GRAPH is being called.
;  349          0637  1 !
;  350          0638  1 ! IMPLICIT OUTPUTS:
;  351          0639  1 !
;  352          0640  1 !       None
;  353          0641  1 !
;  354          0642  1 ! ROUTINE VALUE:
;  355          0643  1 !
;  356          0644  1 !       None, unless an nonrecoverable condition is encountered.  EDF$GRAPH
;  357          0645  1 !       will signal the condition for the caller to handle.
;  358          0646  1 !
;  359          0647  1 ! COMPLETION CODES:
;  360          0648  1 !
```

```
;   361         0649  1 !       None
;   362         0650  1 !
;   363         0651  1 ! SIDE EFFECTS:
;   364         0652  1 !
;   365         0653  1 !       None
;   366         0654  1 !
;   367         0655  1 !--
;   368         0656  1
;   369         0657  2 BEGIN
;   370         0658  2
;   371         0659  2     MAP
;   372         0660  2         XY_ARRAY_DESC    : REF BBLOCK,  !Descriptor for the data array
;   373         0661  2         SHADE_ARRAY_DESC : REF BBLOCK   !Descriptor for the shading array
;   374         0662  2         ;
;   375         0663  2
;   376         0664  2     LOCAL
;   377         0665  2         OLD_BUFFER,                    !Address of previous screen buffer,
;   378         0666  2                                        !if any existed -- otherwise, zero.
;   379         0667  2         RTN_STATUS,                    !Status returned from external calls
;   380         0668  2         NEW_SURFACE_GRAPH : BYTE,      !Set if graph will be completely rewritten
;   381         0669  2         XY_ARRAY         :             !Array to hold the data to be plotted
;   382         0670  2             REF TWO_DIM_ARRAY[ LONG, UNSIGNED ],
;   383         0671  2         SHADE_ARRAY      :             !Array of shading information
;   384         0672  2             REF TWO_DIM_ARRAY[ LONG, UNSIGNED ],
;   385         0673  2
;   386         0674  2
;   387         0675  2         !This vector and its descriptor will be used to format the different
;   388         0676  2         !graphs, before they are written to the user's terminal.
;   389         0677  2         !
;   390         0678  2         SCREEN_BUFFER    :             !Buffer for the output.
;   391         0679  2             VECTOR[ SCREEN_BUFFER_SIZE, BYTE ],
;   392         0680  2         SCREEN_BUFFER_DESC :           !Descriptor for the buffer
;   393         0681  2             BBLOCK[ DSC$K_Z_BLN ]
;   394         0682  2         ;
;   395         0683  2
```

```
;  397     0684  2      !Do we know what kind of terminal are we dealing with?
;  398     0685  2      !
;  399     0686  2      IF .DEVICE_TYPE EQLU ALL_BITS_SET
;  400     0687  2      THEN
;  401     0688  2
;  402     0689  2          !This is the first time EDF$GRAPH has been called.  Determine the
;  403     0690  2          !terminal type and check to make sure it is possible to write a
;  404     0691  2          !plot to the terminal given its charateristics.
;  405     0692  2          !
;  406     0693  2          GET_GRAPH_INFO();
;  407     0694  2
;  408     0695  2
;  409     0696  2      !Initialize the screen buffer's descriptor and then set buffer mode on
;  410     0697  2      !
;  411     0698  2      SCREEN_BUFFER_DESC[ DSC$B_CLASS ]   = DSC$K_CLASS_Z;
;  412     0699  2      SCREEN_BUFFER_DESC[ DSC$B_DTYPE ]   = DSC$K_DTYPE_Z;
;  413     0700  2      SCREEN_BUFFER_DESC[ DSC$W_LENGTH ]  = SCREEN_BUFFER_SIZE;
;  414     0701  2      SCREEN_BUFFER_DESC[ DSC$A_POINTER ] = SCREEN_BUFFER;
;  415     0702  2
;  416     0703  2
;  417     0704  2      !Prepare for putting graphs to the user's terminal.
;  418     0705  2      !
;  419     0706  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  420     0707  2      THEN
;  421     0708  2
;  422     0709  2          !Turn REGIS mode on and set up default screen characteristics.
;  423     0710  2          !
;  424     0711  3          PUT_REGIS( REGIS_SET_UP )
;  425     0712  3
;  426     0713  2      ELSE
;  427     0714  2
;  428     0715  2          !Set up bufferring for hard copy and ANSI terminals
;  429     0716  2          !
;  430     0717  3          IF NOT ( RTN_STATUS = LIB$SET_BUFFER( SCREEN_BUFFER_DESC, OLD_BUFFER ))
;  431     0718  2          THEN
;  432     0719  2
;  433     0720  2              !An error has occured while setting buffer mode.
;  434     0721  2              !
;  435     0722  2              SIGNAL( .RTN_STATUS );
;  436     0723  2
;  437     0724  2
;  438     0725  2      !Which kind of graph was requested?
;  439     0726  2      !
;  440     0727  2      SELECTONE ..GRAPH_TYPE OF
;  441     0728  2          SET
;  442     0729  2
;  443     0730  2          !Line graph
;  444     0731  2          !
;  445     0732  2          [ GRF$_LINE ] :
;  446     0733  3              BEGIN
;  447     0734  3              IF NOT .DEVICE_FLAGS[ SCR$V_REGIS ]
;  448     0735  3                              AND
;  449     0736  3                ..LAST_INDEX NEQU NO_LAST_INDEX
;  450     0737  3                              AND
;  451     0738  3                 .DEVICE_FLAGS[ SCR$V_SCREEN ]
;  452     0739  3              THEN
;  453     0740  3
```

```
;  454     0741  3                !A line graph was plotted during the last call and the
;  455     0742  3                !terminal is a video;  just the shape of the "curve" needs to
;  456     0743  3                !be changed.
;  457     0744  3                !
;  458     0745  3                MOVE_LINE_GRAPH( .XY_ARRAY_DESC, ..CURRENT_INDEX, ..LAST_INDEX)
;  459     0746  3
;  460     0747  3            ELSE
;  461     0748  3
;  462     0749  3                !Either this is the first time for the graph or the user's
;  463     0750  3                !terminal is hard copy.  The whole graph must be redrawn.
;  464     0751  3                !
;  465     0752  3                PLOT_LINE_GRAPH( .XY_ARRAY_DESC, ..CURRENT_INDEX,
;  466     0753  3                                 .Y_LABEL_DESC, .SHADE_ARRAY_DESC );
;  467     0754  2            END;
;  468     0755  2
;  469     0756  2        !Surface graph
;  470     0757  2        !
;  471     0758  2        [ GRF$_SRF_INCREASING, GRF$_SRF_DECREASING ] :
;  472     0759  3            BEGIN
;  473     0760  3            IF ..LAST_INDEX NEQU NO_LAST_INDEX AND .DEVICE_FLAGS[ SCR$V_SCREEN ]
;  474     0761  3            THEN
;  475     0762  3
;  476     0763  3                !The last graph which was plotted was the same type of surface
;  477     0764  3                !graph.  Rewrite the contents of the surface graph and leave
;  478     0765  3                !the axes alone.
;  479     0766  3                !
;  480     0767  3                NEW_SURFACE_GRAPH = FALSE
;  481     0768  3
;  482     0769  3            ELSE
;  483     0770  3
;  484     0771  3                !Write a whole new surface graph, icluding the axes.
;  485     0772  3                !
;  486     0773  3                NEW_SURFACE_GRAPH = TRUE;
;  487     0774  3
;  488     0775  3            PLOT_SURFACE_GRAPH( ..GRAPH_TYPE, .XY_ARRAY_DESC,   ..Y_HIGH,
;  489     0776  3                                ..Y_LOW,      ..Y_INCR,        .Y_LABEL_DESC,
;  490     0777  3                                .NEW_SURFACE_GRAPH,    .SHADE_ARRAY_DESC
;  491     0778  3                              );
;  492     0779  2            END;
;  493     0780  2
;  494     0781  2
;  495     0782  2            [ OTHERWISE ] :
;  496     0783  2 !              SIGNAL( EDF$_INTSWERR, 1, INVALID_GRAPH_CODE );
;  497     0784  2                0;
;  498     0785  2
;  499     0786  2            TES
;  500     0787  2        ;
;  501     0788  2
;  502     0789  2
;  503     0790  2        !Clean up before exitting.
;  504     0791  2        !
;  505     0792  3        IF (NOT .DEVICE_FLAGS[ SCR$V_REGIS ])
;  506     0793  2        THEN
;  507     0794  3            BEGIN
;  508     0795  3
;  509     0796  3            !Put graph to the terminal.
;  510     0797  3            !
```

```
;  511          0798  4          IF NOT (RTN_STATUS = LIB$PUT_BUFFER( OLD_BUFFER ))
;  512          0799  3          THEN
;  513          0800  3              SIGNAL( .RTN_STATUS);
;  514          0801  3          END
;  515          0802  2      ELSE
;  516          0803  2
;  517          0804  2          !Turn REGIS mode off
;  518          0805  2          !
;  519          0806  2          PUT_REGIS( REGIS_OFF );
;  520          0807  2
;  521          0808  2      RETURN;
;  522          0809  1 END;                                    ! End of routine EDF$GRAPH


                                                                .TITLE  EDF$GRAPH EDF$GRAPH plotting module
                                                                .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

2C 30 5B 41 28 20 5D 30 2C 30 5B 53 70 50 1B  00000 P.AAB:  .ASCII  <27>\PpS[0,0] (A[0,0] [767,479], I(d), \
49 20 2C 5D 39 37 34 2C 37 36 37 5B 20 5D 30  0000F
                              20 2C 29 64 28  0001E
46 20 2C 29 77 28 49 20 2C 56 28 57 29 31 53  00023          .ASCII  \S1)W(V, I(w), F3, M1, N0, P1, P(M2), S0)\
20 2C 31 50 20 2C 30 4E 20 2C 31 4D 20 2C 33  00032
                  29 30 53 20 2C 29 32 4D 28  00041
                                          00    0004B          .ASCII  <0>
                                 010E004B  C004C P.AAA:  .LONG   17694795
                                 00000000' 00050         .ADDRESS P.AAB
61 28 20 20 20 29 30 6C 28 20 30 4D 20 28 53  00054 P.AAD:  .ASCII  \S( M0 (l0)   (a      l0)   M1 (l35)  (a \
20 31 4D 20 20 20 29 30 6C 20 20 20 20 20 20  00063
                  20 61 28 20 20 29 35 33 6C  00072
20 20 29 30 33 73 20 30 33 6C 20 30 32 31 68  0007C          .ASCII  \h120 l30 s30)   M2 (l65)  (a h150 l70 s5\
68 20 61 28 20 20 29 35 36 6C 28 20 32 4D 20  0008B
                  35 73 20 30 37 6C 20 30 35  0009A
20 29 30 30 31 6C 28 20 33 4D 20 20 20 29 30  000A4          .ASCII  \0)   M3 (l100) (a h240 l35 s30))\
30 33 73 20 35 33 6C 20 30 34 32 68 20 61 28  000B3
                                    29 29     000C2
                                 010E0070  000C4 P.AAC:  .LONG   17694832
                                 00000000' 000C8         .ADDRESS P.AAD
61 28 20 20 20 29 30 6C 28 20 30 4D 20 28 53  000CC P.AAF:  .ASCII  \S( M0 (l0)   (a      l0)   M1 (l35)  (a \
20 31 4D 20 20 20 29 30 6C 20 20 20 20 20 20  000DB
                  20 61 28 20 20 29 35 33 6C  000EA
20 20 29 30 32 73 20 30 33 6C 20 20 20 35 68  000F4          .ASCII  \h5   l30 s20)   M2 (l65)  (a h325 l60 s6\
68 20 61 28 20 20 29 35 36 6C 28 20 32 4D 20  00103
                  36 73 20 30 36 6C 20 35 32  00112
20 29 30 30 31 6C 28 20 33 4D 20 20 20 29 30  0011C          .ASCII  \0)   M3 (l100) (a h245 l80 s10))\
30 31 73 20 30 38 6C 20 35 34 32 68 20 61 28  0012B
                                    29 29     0013A
                                 010E0070  0013C P.AAE:  .LONG   17694832
                                 00000000' 00140         .ADDRESS P.AAF
                              00 70 50 1B  00144 P.AAH:  .ASCII  <27>\Pp\<0>
                                 010E0003  00148 P.AAG:  .LONG   17694723
                                 00000000' 0014C         .ADDRESS P.AAH
                              00 00 5C 1B  00150 P.AAJ:  .ASCII  <27><92><0><0>
                                 010E0002  00154 P.AAI:  .LONG   17694722
                                 00000000' 00158         .ADDRESS P.AAJ
```

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

                                                FF  00000 DEVICE_TYPE:
                                                          .BYTE   -1                                    ;
                                                    00001 DEVICE_FLAGS:
                                                          .BLKB   3
                                                    00004 Y_AXIS_LINE:
                                                          .BLKB   4
                                                    00008 Y_AXIS_LINE:
                                                          .BLKB   4

                                                          REGIS_SET_UP=     P.AAA
                                                          COLOR_SET_RYG=    P.AAC
                                                          COLOR_SET_BLUE=   P.AAE
                                                          REGIS_ON=         P.AAG
                                                          REGIS_OFF=        P.AAI
                                                          .EXTRN  LIB$SCREEN_INFO
                                                          .EXTRN  LIB$SET_BUFFER, LIB$ERASE_PAGE
                                                          .EXTRN  LIB$PUT_SCREEN, LIB$PUT_BUFFER
                                                          .EXTRN  LIB$PUT_OUTPUT

                                                          .PSECT  $CODE$,NOWRT,2

                                                  007C 00000      .ENTRY  EDF$GRAPH, Save R2,R3,R4,R5,R6        ; 0593
                                    56 00000000G  00   9E 00002   MOVAB   LIB$SIGNAL, R6                       ;
                                    55 00000000G  00   9E 00009   MOVAB   LIB$PUT_OUTPUT, R5                   ;
                                    54 00000000'  00   9E 00010   MOVAB   DEVICE_FLAGS, R4                     ;
                                    5E      FDF4  CE   9E 00017   MOVAB   -524(SP), SP                        ;
                                FF  8F        FC  A4   91 0001C   CMPB    DEVICE_TYPE, #255                  ; 0686
                                                  07   12 00021   BNEQ    1$                                 ;
                            00000000V  00         00   FB 00023   CALLS   #0, GET_GRAPH_INFO                 ; 0693
                                    04  AE      0200  8F   3C 0002A 1$:    MOVZWL  #512, SCREEN_BUFFER_DESC   ; 0700
                                    08  AE        0C  AE   9E 00030   MOVAB   SCREEN_BUFFER, SCREEN_BUFFER_DESC+4 ; 0701
                                10          64        02   E1 00035   BBC     #2, DEVICE_FLAGS, 2$           ; 0706
                                                  00000000' 00   9F 00039   PUSHAB  REGIS_SET_UP                ; 0711
                                            65            01   FB 0003F   CALLS   #1, LIB$PUT_OUTPUT          ;
                                            1B            50   E8 00042   BLBS    RTN_STATUS, 4$             ;
                                                          50   DD 00045   PUSHL   RTN_STATUS                 ;
                                                          14   11 00047   BRB     3$                         ;
                                                          5E   DD 00049 2$:   PUSHL   SP                     ; 0717
                                                08        AE   9F 0004B   PUSHAB  SCREEN_BUFFER_DESC         ;
                                    00000000G   00        02   FB 0004E   CALLS   #2, LIB$SET_BUFFER         ;
                                                53        50   D0 00055   MOVL    R0, RTN_STATUS             ;
                                                05        53   E8 00058   BLBS    RTN_STATUS, 4$             ;
                                                          53   DD 0005B   PUSHL   RTN_STATUS                 ; 0722
                                                66        01   FB 0005D 3$:   CALLS   #1, LIB$SIGNAL         ;
                                                52        04   BC   D0 00060 4$:   MOVL    @GRAPH_TYPE, R2   ; 0727
                                                          36   12 00064   BNEQ    6$                         ; 0732
                                10          64        02   E0 00066   BBS     #2, DEVICE_FLAGS, 5$           ; 0734
                                    FFFFFFFF  8F        10   BC   D1 0006A   CMPL    @LAST_INDEX, #-1           ; 0736
                                                          15   13 00072   BEQL    5$                         ;
                                                12        64   E9 00074   BLBC    DEVICE_FLAGS, 5$           ; 0738
                                                          10   BC   DD 00077   PUSHL   @LAST_INDEX            ; 0745
                                                          0C   BC   DD 0007A   PUSHL   @CURRENT_INDEX        ;
                                                          08   AC   DD 0007D   PUSHL   XY_ARRAY_DESC         ;
                                    00000000V   00        03   FB 00080   CALLS   #3, MOVE_LINE_GRAPH       ;
                                                          4C   11 00087   BRB     9$                         ;
                                                7E        20   AC   7D 00089 5$:   MOVQ    Y_LABEL_DESC, -(SP) ; 0753
```

```
                                    0C   BC   DD 0008D          PUSHL   @CURRENT_INDEX                          ; 0752
                                    08   AC   DD 00090          PUSHL   XY_ARRAY_DESC
                          00000000V 00        04   FB 00093    CALLS   #4, PLOT_LINE_GRAPH
                                              39   11 0009A    BRB     9$                                    ; 0727
                                              37   15 0009C 6$: BLEQ    9$                                    ; 0758
                                    02        52   D1 0009E    CMPL    R2, #2
                                              32   14 000A1    BGTR    9$
                          FFFFFFFF  8F        10   BC   D1 000A3    CMPL    @LAST_INDEX, #-1                   ; 0760
                                              07   13 000AB    BEQL    7$
                                    04        64   E9 000AD    BLBC    DEVICE_FLAGS, 7$
                                              50   94 000B0    CLRB    NEW_SURFACE_GRAPH                     ; 0767
                                              03   11 000B2    BRB     8$
                                    50        01   90 000B4 7$: MOVB    #1, NEW_SURFACE_GRAPH                 ; 0773
                                    24        AC   DD 000B7 8$: PUSHL   SHADE_ARRAY_DESC                      ; 0777
                                    7E        50   9A 000BA    MOVZBL  NEW_SURFACE_GRAPH, -(SP)
                                    20        AC   DD 000BD    PUSHL   Y_LABEL_DESC                          ; 0776
                                    1C        BC   DD 000C0    PUSHL   @Y_INCR
                                    18        BC   DD 000C3    PUSHL   @Y_LOW
                                    14        BC   DD 000C6    PUSHL   @Y_HIGH                               ; 0775
                                    08        AC   DD 000C9    PUSHL   XY_ARRAY_DESC
                                    52        DD 000CC         PUSHL   R2
                          00000000V 00        08   FB 000CE    CALLS   #8, PLOT_SURFACE_GRAPH
                       13           64        02   E0 000D5 9$: BBS     #2, DEVICE_FLAGS, 10$                 ; 0792
                                              5E   DD 000D9    PUSHL   SP                                    ; 0798
                          00000000G 00        01   FB 000DB    CALLS   #1, LIB$PUT_BUFFER
                                    53        50   D0 000E2    MOVL    R0, RTN_STATUS
                                    15        53   E8 000E5    BLBS    RTN_STATUS, 12$
                                              53   DD 000E8    PUSHL   RTN_STATUS                            ; 0800
                                              0E   11 000EA    BRB     11$
                             00000000'    00   9F   000EC 10$: PUSHAB  REGIS_OFF                            ; 0806
                          65              01   FB 000F2    CALLS   #1, LIB$PUT_OUTPUT
                          05              50   E8 000F5    BLBS    RTN_STATUS, 12$
                                          50   DD 000F8    PUSHL   RTN_STATUS
                          66              01   FB 000FA 11$: CALLS   #1, LIB$SIGNAL
                                              04   000FD 12$: RET                                           ; 0809

; Routine Size:  254 bytes,    Routine Base:  $CODE$ + 0000
```

```
;   524     0810  1 %SBTTL 'Get the graph characteristics'
;   525     0811  1 ROUTINE GET_GRAPH_INFO  : NOVALUE =
;   526     0812  1
;   527     0813  1 !++
;   528     0814  1 !
;   529     0815  1 ! FUNCTIONAL DESCRIPTION:
;   530     0816  1 !
;   531     0817  1 !       Determine the terminal characteristics.  Determine the width of the
;   532     0818  1 !       graph.  Set up any screen characteristics, if this is a terminal
;   533     0819  1 !       which understands REGIS.
;   534     0820  1 !
;   535     0821  1 ! FORMAL PARAMETERS:
;   536     0822  1 !
;   537     0823  1 !       None
;   538     0824  1 !
;   539     0825  1 ! IMPLICIT INPUTS:
;   540     0826  1 !
;   541     0827  1 !       None
;   542     0828  1 !
;   543     0829  1 ! IMPLICIT OUTPUTS:
;   544     0830  1 !
;   545     0831  1 !       Y_AXIS_LINE   :  column which contains the y-axis line.  This value
;   546     0832  1 !                        is only output for non-REGIS terminals.
;   547     0833  1 !       DEVICE_FLAGS  :  a longword of flags which are set or cleared by
;   548     0834  1 !                        LIB$SCREEN_INFO depending on the terminal
;   549     0835  1 !                        characteristics.
;   550     0836  1 !       DEVICE_TYPE   :  also returned by LIB$SCREEN_INFO.  If all bits are set,
;   551     0837  1 !                        then this is the first time EDF$GRAPH is being called.
;   552     0838  1 !
;   553     0839  1 ! ROUTINE VALUE:
;   554     0840  1 !
;   555     0841  1 !       None
;   556     0842  1 !
;   557     0843  1 ! COMPLETION CODES:
;   558     0844  1 !
;   559     0845  1 !       None
;   560     0846  1 !
;   561     0847  1 ! SIDE EFFECTS:
;   562     0848  1 !
;   563     0849  1 !       If the user's terminla understands REGIS, the screen defaults may be
;   564     0850  1 !       altered.
;   565     0851  1 !
;   566     0852  1 !--
;   567     0853  1
;   568     0854  2 BEGIN
;   569     0855  2
;   570     0856  2 LITERAL
;   571     0857  2     SKINNY_LINE         = 80,
;   572     0858  2     FAT_LINE            = 132,
;   573     0859  2     WIDTH_BOUNDARY      = 100   !Minimum columns needed to format a wider graph
;   574     0860  2     ;
;   575     0861  2
;   576     0862  2 LOCAL
;   577     0863  2     RTN_STATUS,                 !Return status from external calls
;   578     0864  2     LINE_WIDTH      : WORD,     !Width of page for this session
;   579     0865  2     LINES_PER_PAGE  : WORD      !Number of lines per page on the terminal
;   580     0866  2     ;
```

```
;  581          0867  2
```

```
;  583     0868  2      !Determine the terminal type and characteristics.
;  584     0869  2      !
;  585     0870  3      IF NOT ( RTN_STATUS = LIB$SCREEN_INFO( DEVICE_FLAGS,  DEVICE_TYPE,
;  586     0871  3                                             LINE_WIDTH,    LINES_PER_PAGE
;  587     0872  3                                           ))
;  588     0873  2      THEN
;  589     0874  2
;  590     0875  2          !An error has occured while obtaining the device information.
;  591     0876  2          !
;  592     0877  2          SIGNAL( .RTN_STATUS );
;  593     0878  2
;  594     0879  2
;  595     0880  2      !If terminal understands REGIS, set up screen characteristics needed
;  596     0881  2      !for plotting the graphs.  Otherwise, position the y-axis line.
;  597     0882  2      !
;  598     0883  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  599     0884  2      THEN
;  600     0885  2          Y_AXIS_LINE  = X_ORIGIN - AXIS_SHIFT
;  601     0886  2
;  602     0887  2      ELSE
;  603     0888  2
;  604     0889  2          !Determine where the y-axis line should be positioned.
;  605     0890  2          !
;  606     0891  3          IF  (.LINE_WIDTH LEQ WIDTH_BOUNDARY
;  607     0892  3                  OR
;  608     0893  3              (NOT .DEVICE_FLAGS[ SCR$V_SCREEN ]))
;  609     0894  2          THEN
;  610     0895  2
;  611     0896  2              !Have the right most column of the graph fall on the right most
;  612     0897  2              !column of the page.  If the user's terminal is a hard copy (bit
;  613     0898  2              !zero clear in DEVICE_FLAGS), then have the y-axis line fall to
;  614     0899  2              !the left -- to reduce the number of characters being printed.
;  615     0900  2              !
;  616     0901  2              Y_AXIS_LINE = SKINNY_LINE -
;  617     0902  2                            ((MAX_BUCKET_SIZE + 1) * SEPARATOR_WIDTH) + 1
;  618     0903  2          ELSE
;  619     0904  2
;  620     0905  2              !If the line is wide enough, then center the graph across the line.
;  621     0906  2              !
;  622     0907  3              Y_AXIS_LINE = (FAT_LINE -
;  623     0908  2                            ((MAX_BUCKET_SIZE + 1) * SEPARATOR_WIDTH)) / 2;
;  624     0909  2
;  625     0910  2      RETURN;
;  626     0911  1  END;                                   ! End of routine GET_GRAPH_INFO


                                0004 00000 GET_GRAPH_INFO:
                                                .WORD   Save R2                                 ; 0811
                    52 00000000'  00  9E 00002          MOVAB   DEVICE_FLAGS, R2                ;
                    5E            08  C2 00009          SUBL2   #8, SP                          ;
                                  5E  DD 0000C          PUSHL   SP                              ; 0870
                            08    AE  9F 0000E          PUSHAB  LINE_WIDTH                      ;
                            FC    A2  9F 00011          PUSHAB  DEVICE_TYPE                     ;
                                  52  DD 00014          PUSHL   R2                              ;
```

```
                    00000000G  00              04  FB 00016          CALLS   #4, LIB$SCREEN_INFO                      ;
                               09              50  E8 0001D          BLBS    RTN_STATUS, 1$                          ;
                                               50  DD 00020          PUSHL   RTN_STATUS                              ; 0877
                    00000000G  00              01  FB 00022          CALLS   #1, LIB$SIGNAL                          ;
                          06                   02  E1 00029 1$:      BBC     #2, DEVICE_FLAGS, 2$                    ; 0883
                          04   A2         BA   8F  9A 0002D          MOVZBL  #186, Y_AXIS_LINE                       ; 0885
                                               04  00032            RET                                             ;
                        0064   8F         04   AE  B1 00033 2$:      CMPW    LINE_WIDTH, #100                        ; 0891
                                               03  1B 00039          BLEQU   3$                                      ;
                               05              62  E8 0003B          BLBS    DEVICE_FLAGS, 4$                        ; 0893
                          04   A2              0F  D0 0003E 3$:      MOVL    #15, Y_AXIS_LINE                        ; 0902
                                               04  00042            RET                                             ; 0901
                          04   A2              21  D0 00043 4$:      MOVL    #33, Y_AXIS_LINE                        ; 0908
                                               04  00047            RET                                             ; 0911

; Routine Size:  72 bytes,    Routine Base:  $CODE$ + 00FE
```

```
;  628     0912  1 %SBTTL 'Plot a line graph'
;  629     0913  1 ROUTINE PLOT_LINE_GRAPH( XY_ARRAY_DESC, CURRENT_INDEX, Y_LABEL_DESC,
;  630     0914  1                         SHADE_ARRAY_DESC ) : NOVALUE =
;  631     0915  1
;  632     0916  1 !++
;  633     0917  1 !
;  634     0918  1 ! FUNCTIONAL DESCRIPTION:
;  635     0919  1 !
;  636     0920  1 !       For each row in the graph:  write the y-axis label and mark
;  637     0921  1 !       all x-values in the x,y pairs with the y-values which belong
;  638     0922  1 !       in this row.  Draw and label the x-axis.
;  639     0923  1 !
;  640     0924  1 !
;  641     0925  1 ! FORMAL PARAMETERS:
;  642     0926  1 !
;  643     0927  1 !       XY_ARRAY_DESC   :  address of the descriptor for the data array
;  644     0928  1 !       CURRENT_INDEX   :  index of the "row" (first dimension) requested
;  64_     0929  1 !                          for graphing
;  646     0930  1 !
;  647     0931  1 ! IMPLICIT INPUTS:
;  648     0932  1 !
;  649     0933  1 !       None
;  650     0934  1 !
;  651     0935  1 ! IMPLICIT OUTPUTS:
;  652     0936  1 !
;  653     0937  1 !       A line graph
;  654     0938  1 !
;  655     0939  1 ! ROUTINE VALUE:
;  656     0940  1 !
;  657     0941  1 !       None
;  658     0942  1 !
;  659     0943  1 ! COMPLETION CODES:
;  660     0944  1 !
;  661     0945  1 !       None
;  662     0946  1 !
;  663     0947  1 ! SIDE EFFECTS:
;  664     0948  1 !
;  665     0949  1 !       Filling of SCREEN_BUFFER; adjusting the values in SCREEN_BUFFER_DESC.
;  666     0950  1 !
;  667     0951  1 !--
;  668     0952  1
;  669     0953  2 BEGIN
;  670     0954  2
;  671     0955  2 MAP
;  672     0956  2     XY_ARRAY_DESC : REF BBLOCK; !Descriptor for the data array
;  673     0957  2
;  674     0958  2
;  675     0959  2 LOCAL
;  676     0960  2     XY_ARRAY                :               !Array to hold the data to be plotted
;  677     0961  2             REF TWO_DIM_ARRAY[ LONG, UNSIGNED ],
;  678     0962  2     RTN_STATUS,                 !Status returned from external calls
;  679     0963  2     DIM2,                       !Index for the array's second dimension
;  680     0964  2     ROW_LABEL,                  !Label for y-axis value
;  681     0965  2     ROW_POS,                    !Current row on graph
;  682     0966  2     COLUMN_POS,                 !Column postion for output
;  683     0967  2     SEPARATOR_CHAR,             !Dummy arguement to PUT_ROW_SEGMENT
;  684     0968  2     CURRENT_VALUE,              !Current xy-value to be plotted
```

```
;  685          0969  2      'EPEAT_COUNT                .Number of times current xy-value is in row
;  686          0970  2
;  687          0971  2
```

```
;  689          0972  2     !Erase the screen to give the graph a "clean slate".
;  690          0973  2     !
;  691          0974  2     ERASE_PAGE();
;  692          0975  2
;  693          0976  2
;  694          0977  2     !Set up colors for the graph.
;  695          0978  2     !
;  696          0979  2     IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  697          0980  2     THEN
;  698          0981  2         PUT_REGIS( COLOR_SET_BLUE );
;  699          0982  2
;  700          0983  2
;  701          0984  2     !Draw and label the y-axis
;  702          0985  2     !
;  703          0986  2     DRAW_Y_AXIS( LINE_GRAPH_LEN, GRF$_LINE, NULL, NULL );
;  704          0987  2     LABEL_Y_AXIS( .Y_LABEL_DESC, LINE_GRAPH_LEN );
;  705          0988  2
;  706          0989  2
;  707          0990  2     !Create x-axis labels and put them to the screen
;  708          0991  2     !
;  709          0992  2     DRAW_X_AXIS( LINE_GRAPH_LEN );
;  710          0993  2
;  711          0994  2
;  712          0995  2     !Inititalize those variables needed to create the graph's "curve".
;  713          0996  2     !
;  714          0997  2     DIM2            = 0;
;  715          0998  2     SEPARATOR_CHAR = NULL;
;  716          0999  2
;  717          1000  2
;  718          1001  2     !For each row (y-value) in the graph:
;  719          1002  2     !
;  720          1003  3     INCR ROW_COUNT FROM 0 TO (LINE_GRAPH_LEN - 1)
;  721          1004  2     DO
;  722          1005  3         BEGIN
;  723          1006  3
;  724          1007  3         !Determine which row is being written and what it's label is.
;  725          1008  3         !
;  726          1009  3         ROW_POS   = .ROW_COUNT + 1;
;  727          1010  3         ROW_LABEL = LINE_GRAPH_LEN - .ROW_COUNT;
;  728          1011  3
;  729          1012  3
;  730          1013  3         !If there are more elements for this row of the xy-array and they have
;  731          1014  3         !the correct value, then print their value.
;  732          1015  3         !
;  733          1016  4         WHILE (.DIM2 LEQ .XY_ARRAY_DESC[ DSC$L_U2 ]
;  734          1017  4                         AND
;  735          1018  5               (.XY_ARRAY[ .CURRENT_INDEX,.DIM2,.XY_ARRAY_DESC ] GEQ .ROW_LABEL
;  736          1019  5                         OR
;  737          1020  4               .XY_ARRAY[ .CURRENT_INDEX,.DIM2,.XY_ARRAY_DESC ] LEQ 0))
;  738          1021  3         DO
;  739          1022  4             BEGIN
;  740          1023  4
;  741          1024  4             !Save the current xy-value for comparisons; determine the position
;  742          1025  4             !in the row.
;  743          1026  4             !
;  744          1027  4             CURRENT_VALUE = .XY_ARRAY[ .CURRENT_INDEX, .DIM2, .XY_ARRAY_DESC ];
;  745          1028  4             COLUMN_POS    = .DIM2;
```

```
;  746     1029  4
;  747     1030  4
;  748     1031  4              !Compare the current value to all subsequent values in the row
;  749     1032  4              !until a value which is not equal is encountered.  Keep a count
;  750     1033  4              !of the equal values, so the right number can be printed in the
;  751     1034  4              !graph.
;  752     1035  4              !
;  753     1036  4              DIM2 = .DIM2 + 1;
;  754     1037  4              WHILE .DIM2 LEQ .XY_ARRAY_DESC[ DSC$L_U2 ]
;  755     1038  4                    AND
;  756     1039  5                 (.CURRENT_VALUE EQL .XY_ARRAY[ .CURRENT_INDEX, .DIM2, .XY_ARRAY_DESC ]
;  757     1040  5                    OR
;  758     1041  5                 .XY_ARRAY[ .CURRENT_INDEX, .DIM2, .XY_ARRAY_DESC] LEQ 0)
;  759     1042  4              DO
;  760     1043  4                  DIM2 = .DIM2 + 1;
;  761     1044  4
;  762     1045  4
;  763     1046  4              !Calculate the number of equal values and write them out.
;  764     1047  4              !
;  765     1048  4              REPEAT_COUNT = .DIM2 - .COLUMN_POS;
;  766     1049  4
;  767     1050  4              IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  768     1051  4              THEN
;  769     1052  5                  BEGIN
;  770     1053  5                  IF .CURRENT_VALUE GTR 0
;  771     1054  5                  THEN
;  772     1055  5                      DRAW_BARS_REGIS( .ROW_POS, .COLUMN_POS, .REPEAT_COUNT );
;  773     1056  5                  END
;  774     1057  5
;  775     1058  4              ELSE
;  776     1059  4                  PUT_ROW_SEGMENT( .CURRENT_VALUE, .REPEAT_COUNT,   .COLUMN_POS,
;  777     1060  4                                   .ROW_POS, .SEPARATOR_CHAR, .CURRENT_INDEX,
;  778     1061  4                                   .SHADE_ARRAY_DESC);
;  779     1062  4
;  780     1063  3              END;                          !DIM2
;  781     1064  3
;  782     1065  2          END;                              !ROW_COUNT
;  783     1066  2
;  784     1067  1 END;                                       ! End of routine PLOT_LINE_GRAPH


                                    0FFC 00000 PLOT_LINE_GRAPH:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 0913
                          5B 00000000'  00  9E 00002        MOVAB   DEVICE_FLAGS, R11                 ;
    00000000V  00                       00  FB 00009        CALLS   #0, ERASE_PAGE                    ; 0974
            19 6B                       02  E1 00010        BBC     #2, DEVICE_FLAGS, 1$              ; 0979
                             00000000'  00  9F 00014        PUSHAB  COLOR_SET_BLUE                    ; 0981
    00000000G  00                       01  FB 0001A        CALLS   #1, LIB$PUT_OUTPUT                ;
               09                       50  E8 00021        BLBS    RTN_STATUS, 1$                    ;
                                        50  DD 00024        PUSHL   RTN_STATUS                        ;
    00000000G  00                       01  FB 00026        CALLS   #1, LIB$SIGNAL                    ;
                                        7E  7C 0002D 1$:     CLRQ    -(SP)                             ; 0986
               7E                       0A  7D 0002F        MOVQ    #10, -(SP)                        ;
    00000000V  00                       04  FB 00032        CALLS   #4, DRAW_Y_AXIS                   ;
```

```
                                      0A   DD 00039          PUSHL   #10                                          ; 0987
                                  0C  AC   DD 0003B          PUSHL   Y_LABEL_DESC                                ;
                         00000000V    00   02   FB 0003E          CALLS   #2, LABEL_Y_AXIS                            ;
                                      0A   DD 00045          PUSHL   #10                                         ; 0992
                         00000000V    00   01   FB 00047          CALLS   #1, DRAW_X_AXIS                             ;
                                      56   D4 0004E          CLRL    DIM2                                        ; 0997
                                      5A   D4 00050          CLRL    SEPARATOR_CHAR                              ; 0998
                                 55   04   AC   D0 00052          MOVL    XY_ARRAY_DESC, R5                           ; 1016
                                      52   D4 00056          CLRL    ROW_COUNT                                   ; 1027
                                 53   01   A2   9E 00058 2$:       MOVAB   1(R2), ROW_POS                              ; 1009
                            54        0A   52   C3 0005C          SUBL3   ROW_COUNT, #10, ROW_LABEL                   ; 1010
                                 28   A5   56   D1 00060 3$:       CMPL    DIM2, 40(R5)                                ; 1016
                                      23   14 00064          BGTR    4$                                          ;
                                 50        55   D0 00066          MOVL    R5, DESC_ADR_LCL                            ; 1018
                            51   08   AC   18   A0 00069          MULL3   24(DESC_ADR_LCL), CURRENT_INDEX, R1         ;
                                 51        56   C0 0006F          ADDL2   DIM2, R1                                    ;
                                 54   04 B041   D1 00072          CMPL    @4(DESC_ADR_LCL)[R1], ROW_LABEL             ;
                                      12   18 00077          BGEQ    5$                                          ;
                                 50        55   D0 00079          MOVL    R5, DESC_ADR_LCL                            ; 1020
                            51   08   AC   18   A0 0007C          MULL3   24(DESC_ADR_LCL), CURRENT_INDEX, R1         ;
                                 51        56   C0 00082          ADDL2   DIM2, R1                                    ;
                                      04 B041   D5 00085          TSTL    @4(DESC_ADR_LCL)[R1]                        ;
                                      74   14 00089 4$:       BGTR    10$                                         ;
                                 51        55   D0 0008B 5$:       MOVL    R5, DESC_ADR_LCL                            ; 1027
                            50   08   AC   18   A1 0008E          MULL3   24(DESC_ADR_LCL), CURRENT_INDEX, R0         ;
                                 50        56   C0 00094          ADDL2   DIM2, R0                                    ;
                                 57   04 B140   D0 00097          MOVL    @4(DESC_ADR_LCL)[R0], CURRENT_VALUE         ;
                                 58        56   D0 0009C          MOVL    DIM2, COLUMN_POS                            ; 1028
                                      56   D6 0009F 6$:       INCL    DIM2                                        ; 1036
                                 28   A5   56   D1 000A1          CMPL    DIM2, 40(R5)                                ; 1037
                                      25   14 000A5          BGTR    7$                                          ;
                                 50        55   D0 000A7          MOVL    R5, DESC_ADR_LCL                            ; 1039
                            51   08   AC   18   A0 000AA          MULL3   24(DESC_ADR_LCL), CURRENT_INDEX, R1         ;
                                 51        56   C0 000B0          ADDL2   DIM2, R1                                    ;
                                 04 B041   57   D1 000B3          CMPL    CURRENT_VALUE, @4(DESC_ADR_LCL)[R1]         ;
                                      E5   13 000B8          BEQL    6$                                          ;
                                 50        55   D0 000BA          MOVL    R5, DESC_ADR_LCL                            ; 1041
                            51   08   AC   18   A0 000BD          MULL3   24(DESC_ADR_LCL), CURRENT_INDEX, R1         ;
                                 51        56   C0 000C3          ADDL2   DIM2, R1                                    ;
                                      04 B041   D5 000C6          TSTL    @4(DESC_ADR_LCL)[R1]                        ;
                                      D3   15 000CA          BLEQ    6$                                          ;
                            59        56        58   C3 000CC 7$:       SUBL3   COLUMN_POS, DIM2, REPEAT_COUNT      ; 1048
                            11        6B        02   E1 000D0          BBC     #2, DEVICE_FLAGS, 8$                ; 1050
                                      57   D5 000D4          TSTL    CURRENT_VALUE                               ; 1053
                                      88   15 000D6          BLEQ    3$                                          ;
                                 0308   8F   BB 000D8          PUSHR   #^M<R3,R8,R9>                               ; 1055
                         00000000V    00   03   FB 000DC          CALLS   #3, DRAW_BARS_REGIS                         ;
                                      17   11 000E3          BRB     9$                                          ; 1050
                                   10   AC   DD 000E5 8$:       PUSHL   SHADE_ARRAY_DESC                            ; 1061
                                   08   AC   DD 000E8          PUSHL   CURRENT_INDEX                               ; 1060
                                 0408   8F   BB 000EB          PUSHR   #^M<R3,R10>                                 ;
                                      58   DD 000EF          PUSHL   COLUMN_POS                                  ; 1059
                                 0280   8F   BB 000F1          PUSHR   #^M<R7,R9>                                  ;
                         00000000V    00   07   FB 000F5          CALLS   #7, PUT_ROW_SEGMENT                         ;
                                    FF61   31 000FC 9$:       BRW     3$                                          ; 1016
                       FF53  52        01        09   F1 000FF 10$:      ACBL    #9, #1, ROW_COUNT, 2$               ; 1003
                                              04 00105          RET                                                 ; 1067
```

; Routine Size: 262 bytes,   Routine Base: $CODE$ + 0146

```
;   786          1068  1 %SBTTL 'Move the curve on a line graph'
;   787          1069  1 ROUTINE MOVE_LINE_GRAPH (XY_ARRAY_DESC, CURRENT_INDEX, LAST_INDEX ) :
;   788          1070  1 NOVALUE =
;   789          1071  1
;   790          1072  1 !++
;   791          1073  1 !
;   792          1074  1 ! FUNCTIONAL DESCRIPTION:
;   793          1075  1 !
;   794          1076  1 !       MOVE_LINE_GRAPH changes the existing curve of the graph.
;   795          1077  1 !       It checks each value in the current row of th xy-array against
;   796          1078  1 !       the appropriate value in the last row.  If there is a difference,
;   797          1079  1 !       the old value is erased and the new value is written out at the
;   798          1080  1 !       proper location.
;   799          1081  1 !
;   800          1082  1 ! FORMAL PARAMETERS:
;   801          1083  1 !
;   802          1084  1 !       XY_ARRAY_DESC   :  Address of xy-array descriptor
;   803          1085  1 !       CURRENT_INDEX   :  Index of current row to be plotted, in xy-array
;   804          1086  1 !       LAST_INDEX      :  Index of last row to be plotted
;   805          1087  1 !
;   806          1088  1 ! IMPLICIT INPUTS:
;   807          1089  1 !
;   808          1090  1 !       None
;   809          1091  1 !
;   810          1092  1 ! IMPLICIT OUTPUTS:
;   811          1093  1 !
;   812          1094  1 !       None
;   813          1095  1 !
;   814          1096  1 ! ROUTINE VALUE:
;   815          1097  1 !
;   816          1098  1 !       None
;   817          1099  1 !
;   818          1100  1 ! COMPLETION CODES:
;   819          1101  1 !
;   820          1102  1 !       None
;   821          1103  1 !
;   822          1104  1 ! SIDE EFFECTS:
;   823          1105  1 !
;   824          1106  1 !       None
;   825          1107  1 !
;   826          1108  1 !--
;   827          1109  1
;   828          1110  2 BEGIN
;   829          1111  2
;   830          1112  2 MAP
;   831          1113  2     XY_ARRAY_DESC : REF BBLOCK; !Descriptor for the data array
;   832          1114  2
;   833          1115  2 LITERAL
;   834          1116  2     CTRSTR_LEN = 4,             !Length of the control string
;   835          1117  2     LEN_FIELD  = 1              !Position of length field in the control string
;   836          1118  2     ;
;   837          1119  2
;   838          1120  2 LOCAL
;   839          1121  2     XY_ARRAY                :                  !Array to hold the data to be plotted
;   840          1122  2             REF TWO_DIM_ARRAY[ LONG, UNSIGNED ],
;   841          1123  2     ROW_POS,                    !Current row on graph
;   842          1124  2     COLUMN_POS,                 !Column postion for output
```

```
;  843          1125  2     BUFFER                  :       !Buffer for building the output
;  844          1126  2         VECTOR[ MAX_PAGE_WIDTH, BYTE ],
;  845          1127  2     DESC                    :       !Descriptor for BUFFER
;  846          1128  2         BBLOCK[ DSC$K_Z_BLN ]
;  847          1129  2     ;
;  848          1130  2
;  849          1131  2 BIND
;  850          1132  2     CTRSTR_DESC = %ASCID'!2UB'
;  851          1133  2     ;
```

```
;  853          1134  2      !Intialize variables
;  854          1135  2      !
;  855          1136  2      (BUFFER)                = '    ';
;  856          1137  2      DESC[ DSC$B_CLASS ]     = DSC$K_CLASS_Z;
;  857          1138  2      DESC[ DSC$B_DTYPE ]     = DSC$K_DTYPE_Z;
;  858          1139  2      DESC[ DSC$A_POINTER ] = BUFFER;
;  859          1140  2
;  860          1141  2
;  861          1142  2      !For each value:
;  862          1143  2      INCR DIM2 FROM 0 TO .XY_ARRAY_DESC[ DSC$L_U2 ]
;  863          1144  2      DO
;  864          1145  2
;  865          1146  2          !If the value at the current index is not equal to the value at the
;  866          1147  2          !old index, then the curve must be altered... erase the old value
;  867          1148  2          !and write the new one in the appropriate row.
;  868          1149  2          !
;  869          1150  2          IF .XY_ARRAY[ .CURRENT_INDEX, .DIM2, .XY_ARRAY_DESC ]
;  870          1151  2                NEQ
;  871          1152  2             .XY_ARRAY[ .LAST_INDEX, .DIM2, .XY_ARRAY_DESC ]
;  872          1153  2          THEN
;  873          1154  3              BEGIN
;  874          1155  3
;  875          1156  3              !Erase the old value.
;  876          1157  3              !
;  877          1158  3              INCR CHAR_POS FROM 0 TO ( SEPARATOR_WIDTH - 1 ) DO
;  878          1159  3                  BUFFER[ .CHAR_POS ] = ' ';
;  879          1160  3              DESC[ DSC$W_LENGTH ]    = SEPARATOR_WIDTH;
;  880          1161  3
;  881          1162  3
;  882          1163  3              !Determine which row the value to be erased is in.
;  883          1164  3              !
;  884          1165  3              IF .XY_ARRAY[ .LAST_INDEX, .DIM2, .XY_ARRAY_DESC ] GEQ LINE_GRAPH_LEN
;  885          1166  3                        OR
;  886          1167  3                 .XY_ARRAY[ .LAST_INDEX, .DIM2, .XY_ARRAY_DESC ] LEQ 0
;  887          1168  3              THEN
;  888          1169  3                  ROW_POS = FIRST_ROW
;  889          1170  3              ELSE
;  890          1171  3                  ROW_POS = LINE_GRAPH_LEN -
;  891          1172  3                            .XY_ARRAY[ .LAST_INDEX, .DIM2, .XY_ARRAY_DESC ] + 1;
;  892          1173  3
;  893          1174  3
;  894          1175  3              !Which column does it reside in?
;  895          1176  3              !
;  896          1177  3              COLUMN_POS  = (.DIM2 + 1) * SEPARATOR_WIDTH + .Y_AXIS_LINE;
;  897          1178  3
;  898          1179  3
;  899          1180  3              !Write over the old value with blanks.
;  900          1181  3              !
;  901          1182  3              PUT_TEXT( DESC, ROW_POS, COLUMN_POS );
;  902          1183  3
;  903          1184  3
;  904          1185  3              !Translate the new value to be placed on the curve.
;  905          1186  3              !
;  906          1187  3              IF .XY_ARRAY[ .CURRENT_INDEX, .DIM2, .XY_ARRAY_DESC ] LEQ 0
;  907          1188  3              THEN
;  908          1189  3
;  909          1190  4                  BEGIN
```

```
;  910          1191  4                (BUFFER[ 0 ])          = '  ';
;  911          1192  4                DESC[ DSC$W_LENGTH ] = SEPARATOR_WIDTH;
;  912          1193  4                END
;  913          1194  4
;  914          1195  3            ELSE
;  915        P 1196  3                TRANSLATE_VALUE( CTRSTR_DESC   DESC[ DSC$W_LENGTH ], DESC,
;  916        P 1197  3                                 XY_ARRAY[ .CURRENT_INDEX, .DIM2,
;  917          1198  3                                           .XY_ARRAY_DESC ]);
;  918          1199  3
;  919          1200  3
;  920          1201  3            !Determine which row the value to be written should be in.
;  921          1202  3            !
;  922          1203  3            IF .XY_ARRAY[ .CURRENT_INDEX, .DIM2, .XY_ARRAY_DESC ] GEQ LINE_GRAPH_LEN
;  923          1204  3                        OR
;  924          1205  3               .XY_ARRAY[ .CURRENT_INDEX, .DIM2, .XY_ARRAY_DESC ] LEQ 0
;  925          1206  3            THEN
;  926          1207  3                ROW_POS = FIRST_ROW
;  927          1208  3            ELSE
;  928          1209  3                ROW_POS = LINE_GRAPH_LEN -
;  929          1210  3                          .XY_ARRAY[ .CURRENT_INDEX,.DIM2,.XY_ARRAY_DESC ] + 1;
;  930          1211  3
;  931          1212  3
;  932          1213  3            !Write over the old value with blanks.
;  933          1214  3            !
;  934          1215  3            PUT_TEXT( DESC, ROW_POS, COLUMN_POS );
;  935          1216  2            END;
;  936          1217  1 END;                                    ! End of routine MOVE_LINE_GRAPH


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                42  55  32  21  0015C P.AAL:   .ASCII  \!2UB\                                    ;
                                  010E0004  00160 P.AAK:   .LONG   17694724                                      ;
                                  00000000' 00164          .ADDRESS P.AAL                                        ;

                                                    CTRSTR_DESC=        P.AAK
                                                        .EXTRN  SYS$FAOL

                                                        .PSECT  $CODE$,NOWRT,2

                                          01FC 00000 MOVE_LINE_GRAPH:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8                       ; 1069
                                    58 00000000G   00  9E 00002         MOVAB   LIB$PUT_SCREEN, R8              ;
                                    57 00000000G   00  9E 00009         MOVAB   LIB$SIGNAL, R7                  ;
                                    5E      FF6C   CE  9E 00010         MOVAB   -148(SP), SP                    ;
                                10  AE  20202020   8F  D0 00015         MOVL    #538976288, BUFFER              ; 1136
                                        0A  AE     B4 0001D             CLRW    DESC+2                          ; 1138
                                0C  AE      10  AE  9E 00020            MOVAB   BUFFER, DESC+4                  ; 1139
                                    55      04  AC  7D 00025            MOVQ    XY_ARRAY_DESC, R5                ; 1143
                                    54          01  CE 00029            MNEGL   #1, DIM2                        ; 1152
                                              0121  31 0002C 1$:        BRW     13$                             ;
                                    52          55  D0 0002F 2$:        MOVL    R5, DESC_ADR_LCL                ; 1150
                51                  56      18  A2  C5 00032            MULL3   24(DESC_ADR_LCL), R6, R1        ;
                                    51          54  C0 00037            ADDL2   DIM2, R1                        ;
                                    50          55  D0 0003A            MOVL    R5, DESC_ADR_LCL                ; 1152
                53          0C  AC      18  A0  C5 0003D                MULL3   24(DESC_ADR_LCL), LAST_INDEX, R3 ;
```

```
                53           54  C0 00043           ADDL2   DIM2, R3                                            ;
            04 B043       04 B241  D1 00046           CMPL    @4(DESC_ADR_LCL)[R1], @4(DESC_ADR_LCL)[R3]          ;
                             DD  13 0004D           BEQL    1$                                                  ;
                             50  D4 0004F           CLRL    CHAR_POS                                            ; 1158
            10 AE40          20  90 00051 3$:       MOVB    #32, BUFFER[CHAR_POS]                               ; 1159
      F7        50           01  F3 00056           AOBLEQ  #1, CHAR_POS, 3$                                    ;
            08  AE           02  B0 0005A           MOVW    #2, DESC                                            ; 1160
                50           55  D0 0005E           MOVL    R5, DESC_ADR_LCL                                    ; 1165
      51     0C AC        18  A0  C5 00061           MULL3   24(DESC_ADR_LCL), LAST_INDEX, R1                    ;
                51           54  C0 00067           ADDL2   DIM2, R1                                            ;
                0A        04 B041  D1 0006A           CMPL    @4(DESC_ADR_LCL)[R1], #10                           ;
                             12  18 0006F           BGEQ    4$                                                  ;
                50           55  D0 00071           MOVL    R5, DESC_ADR_LCL                                    ; 1167
      51     0C AC        18  A0  C5 00074           MULL3   24(DESC_ADR_LCL), LAST_INDEX, R1                    ;
                51           54  C0 0007A           ADDL2   DIM2, R1                                            ;
                          04 B041  D5 0007D           TSTL    @4(DESC_ADR_LCL)[R1]                                ;
                             06  14 00081           BGTR    5$                                                  ;
            04  AE           01  D0 00083 4$:       MOVL    #1, ROW_POS                                         ; 1169
                             13  11 00087           BRB     6$                                                  ;
                50           55  D0 00089 5$:       MOVL    R5, DESC_ADR_LCL                                    ; 1172
      51     0C AC        18  A0  C5 0008C           MULL3   24(DESC_ADR_LCL), LAST_INDEX, R1                    ;
                51           54  C0 00092           ADDL2   DIM2, R1                                            ;
 04   AE        0B        04 B041  C3 00095           SUBL3   @4(DESC_ADR_LCL)[R1], #11, ROW_POS                  ;
                50 00000000'   00  D0 0009C 6$:       MOVL    Y_AXIS_LINE, R0                                     ; 1177
                6E        02 A044  3E 000A3           MOVAW   2(R0)[DIM2], COLUMN_POS                             ;
                             5E  DD 000A8           PUSHL   SP                                                  ; 1182
                          08  AE  9F 000AA           PUSHAB  ROW_POS                                             ;
                          10  AE  9F 000AD           PUSHAB  DESC                                                ;
                68           03  FB 000B0           CALLS   #3, LIB$PUT_SCREEN                                  ;
                05           50  E8 000B3           BLBS    RTN_STATUS, 7$                                      ;
                             50  DD 000B6           PUSHL   RTN_STATUS                                          ;
                67           01  FB 000B8           CALLS   #1, LIB$SIGNAL                                      ;
                50           55  D0 000BB 7$:       MOVL    R5, DESC_ADR_LCL                                    ; 1187
      51        56        18  A0  C5 000BE           MULL3   24(DESC_ADR_LCL), R6, R1                            ;
                51           54  C0 000C3           ADDL2   DIM2, R1                                            ;
                          04 B041  D5 000C6           TSTL    @4(DESC_ADR_LCL)[R1]                                ;
                             0C  14 000CA           BGTR    8$                                                  ;
            10  AE      2020  8F  3C 000CC           MOVZWL  #8224, BUFFER                                       ; 1191
            08  AE           02  B0 000D2           MOVW    #2, DESC                                            ; 1192
                             2A  11 000D6           BRB     9$                                                  ; 1187
                50           55  D0 000D8 8$:       MOVL    R5, DESC_ADR_LCL                                    ; 1198
      51        56        18  A0  C5 000DB           MULL3   24(DESC_ADR_LCL), R6, R1                            ;
                51           54  C0 000E0           ADDL2   DIM2, R1                                            ;
                          04 B041  DF 000E3           PUSHAL  @4(DESC_ADR_LCL)[R1]                                ;
                          0C  AE  9F 000E7           PUSHAB  DESC                                                ;
                          10  AE  9F 000EA           PUSHAB  DESC                                                ;
                     00000000'   00  9F 000ED           PUSHAB  CTRSTR_DESC                                         ;
        00000000G  00           04  FB 000F3           CALLS   #4, SYS$FAOL                                        ;
                05           50  E8 000FA           BLBS    RTN_STATUS, 9$                                      ;
                             50  DD 000FD           PUSHL   RTN_STATUS                                          ;
                67           01  FB 000FF           CALLS   #1, LIB$SIGNAL                                      ;
                50           55  D0 00102 9$:       MOVL    R5, DESC_ADR_LCL                                    ; 1203
      51        56        18  A0  C5 00105           MULL3   24(DESC_ADR_LCL), R6, R1                            ;
                51           54  C0 0010A           ADDL2   DIM2, R1                                            ;
                0A        04 B041  D1 0010D           CMPL    @4(DESC_ADR_LCL)[R1], #10                           ;
                             11  18 00112           BGEQ    10$                                                 ;
                50           55  D0 00114           MOVL    R5, DESC_ADR_LCL                                    ; 1205
```

```
                    51          56      18   A0  C5 00117          MULL3   24(DESC_ADR_LCL), R6, R1                    ;
                                51          54  C0 0011C          ADDL2   DIM2, R1                                    ;
                                        04 B041 D5 0011F          TSTL    @4(DESC_ADR_LCL)[R1]                        ;
                                            06  14 00123          BGTR    11$                                         ;
                        04  AE                 01  D0 00125 10$:  MOVL    #1, ROW_POS                                 ; 1207
                                            12  11 00129          BRB     12$                                         ;
                                50              55  D0 0012B 11$:  MOVL    R5, DESC_ADR_LCL                            ; 1210
                    51          56      18   A0  C5 0012E          MULL3   24(DESC_ADR_LCL), R6, R1                    ;
                                51          54  C0 00133          ADDL2   DIM2, R1                                    ;
                04  AE          0B      04 B041 C3 00136          SUBL3   @4(DESC_ADR_LCL)[R1], #11, ROW_POS          ;
                                            5E  DD 0013D 12$:      PUSHL   SP                                          ; 1215
                                        08  AE  9F 0013F          PUSHAB  ROW_POS                                     ;
                                        10  AE  9F 00142          PUSHAB  DESC                                        ;
                                68              03  FB 00145      CALLS   #3, LIB$PUT_SCREEN                          ;
                                05              50  E8 00148      BLBS    RTN_STATUS, 13$                             ;
                                                50  DD 0014B      PUSHL   RTN_STATUS                                  ;
                                67              01  FB 0014D      CALLS   #1, LIB$SIGNAL                              ;
        FEDB                    54          01      28   A5  F1 00150 13$:  ACBL    40(R5), #1, DIM2, 2$                ; 1150
                                                        04 00157          RET                                         ; 1217

; Routine Size:  344 bytes,    Routine Base:  $CODE$ + 024C
```

```
;   938          1218  1 %SBTTL 'Create a surface graph'
;   939          1219  1 ROUTINE PLOT_SURFACE_GRAPH( GRAPH_TYPE, XY_ARRAY_DESC, Y_HIGH, Y_LOW, Y_INCR,
;   940          1220  1                             Y_LABEL_DESC, NEW_SURFACE_GRAPH, SHADE_ARRAY_DESC )
;   941          1221  1     : NOVALUE =
;   942          1222  1
;   943          1223  1 !++
;   944          1224  1 !
;   945          1225  1 ! FUNCTIONAL DESCRIPTION:
;   946          1226  1 !
;   947          1227  1 !       PLOT_SURFACE_GRAPH produces a "table" from the xy-array.  For each
;   948          1228  1 !       row in the array, each value in the row is printed.  If the terminal
;   949          1229  1 !       understands REGIS, then shade the background.
;   950          1230  1 !
;   951          1231  1 ! FORMAL PARAMETERS:
;   952          1232  1 !
;   953          1233  1 !       GRAPH_TYPE       :  type of table being produced (increasing or
;   954          1234  1 !                           decreasing values
;   955          1235  1 !       XY_ARRAY_DESC    :  address of xy-array descriptor
;   956          1236  1 !       Y_HIGH           :  high value of y-axis numeric label
;   957          1237  1 !       Y_LOW            :  low value of numeric label
;   958          1238  1 !       Y_INCR           :  increment of numeric labels
;   959          1239  1 !       NEW_SURFACE_GRAPH :  set if complete new graph will be written
;   960          1240  1 !       SHADE_ARRAY_DESC  :  Descriptor for array containing shading info
;   961          1241  1 !
;   962          1242  1 ! IMPLICIT INPUTS:
;   963          1243  1 !
;   964          1244  1 !       Y_AXIS_LINE   :  column which contains the y-axis line.  This value
;   965          1245  1 !                        is considered for non-REGIS terminals only.
;   966          1246  1 !       DEVICE_FLAGS  :  a longword of flags which are set or cleared by
;   967          1247  1 !                        LIB$SCREEN_INFO depending on the terminal
;   968          1248  1 !                        characteristics.
;   969          1249  1 !
;   970          1250  1 ! IMPLICIT OUTPUTS:
;   971          1251  1 !
;   972          1252  1 !       Y_AXIS_LINE   :  column which contains the y-axis line.  This value
;   973          1253  1 !                        is only output for REGIS terminals.
;   974          1254  1 !
;   975          1255  1 ! ROUTINE VALUE:
;   976          1256  1 !
;   977          1257  1 !       None
;   978          1258  1 !
;   979          1259  1 ! COMPLETION CODES:
;   980          1260  1 !
;   981          1261  1 !       None
;   982          1262  1 !
;   983          1263  1 ! SIDE EFFECTS:
;   984          1264  1 !
;   985          1265  1 !       None
;   986          1266  1 !
;   987          1267  1 !--
;   988          1268  1
;   989          1269  2 BEGIN
;   990          1270  2
;   991          1271  2 MAP
;   992          1272  2     XY_ARRAY_DESC : REF BBLOCK; !Descriptor for the data array
;   993          1273  2
;   994          1274  2
```

```
;  995          1275  2 LOCAL
;  996          1276  2     XY_ARRAY                :                   !Array to hold the data to be plotted
;  997          1277  2             REF TWO_DIM_ARRAY[ LONG, UNSIGNED ],
;  998          1278  2     SEPARATOR_CHAR      : WORD,  !Character for visually separating bands of values
;  999          1279  2     RTN_STATUS,                  !Status returned from external calls
; 1000          1280  2     MAX_ROWS,                    !Number of rows in the surface graph
; 1001          1281  2     DIM1,                        !Index for first dimension of xy-array
; 1002          1282  2     DIM2,                        !Index for second dimension of xy-array
; 1003          1283  2     ROW_LABEL,                   !Label for y-axis value
; 1004          1284  2     CURRENT_VALUE,               !Current value in row, for comparisons
; 1005          1285  2     REPEAT_COUNT,                !Number of values equal to current value
; 1006          1286  2     ROW_POS,                     !Current row on graph
; 1007          1287  2     COLUMN_POS                   !Column postion for output
; 1008          1288  2     ;
```

```
;  1010          1289  2      !Determine the number of rows in the graph.
;  1011          1290  2      !
;  1012          1291  2      IF SURFACE_GRAPH_LEN LEQ ((.Y_HIGH - .Y_LOW) / .Y_INCR)
;  1013          1292  2      THEN
;  1014          1293  2          MAX_ROWS = SURFACE_GRAPH_LEN - 1
;  1015          1294  2      ELSE
;  1016          1295  2          MAX_ROWS = ((.Y_HIGH - .Y_LOW) / .Y_INCR);
;  1017          1296  2
;  1018          1297  2
;  1019          1298  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  1020          1299  2      THEN
;  1021          1300  3          BEGIN
;  1022          1301  3
;  1023          1302  3          !Choose separator character.  Set up colors for this graph.
;  1024          1303  3          !
;  1025          1304  3          SEPARATOR_CHAR = NULL;
;  1026          1305  3          PUT_REGIS( COLOR_SET_RYG );
;  1027          1306  3          END
;  1028          1307  3
;  1029          1308  2      ELSE
;  1030          1309  2
;  1031          1310  2          !Choose the appropriate separator character
;  1032          1311  2          !
;  1033          1312  2          IF .GRAPH_TYPE EQL GRF$_SRF_DECREASING
;  1034          1313  2          THEN
;  1035          1314  2              SEPARATOR_CHAR = '\'
;  1036          1315  2          ELSE
;  1037          1316  2              SEPARATOR_CHAR = '/';
;  1038          1317  2
;  1039          1318  2
;  1040          1319  2      IF .NEW_SURFACE_GRAPH
;  1041          1320  2      THEN
;  1042          1321  3          BEGIN
;  1043          1322  3
;  1044          1323  3          !Erase the screen to give the graph a "clean slate".
;  1045          1324  3          !
;  1046          1325  3          ERASE_PAGE();
;  1047          1326  3
;  1048          1327  3
;  1049          1328  3          !Draw and label the y-axis
;  1050          1329  3          !
;  1051          1330  3          DRAW_Y_AXIS( .MAX_ROWS+1, .GRAPH_TYPE, .Y_HIGH, .Y_INCR );
;  1052          1331  3          LABEL_Y_AXIS( .Y_LABEL_DESC, .MAX_ROWS+1 );
;  1053          1332  3
;  1054          1333  3
;  1055          1334  3          !Draw and label the x-axis
;  1056          1335  3          !
;  1057          1336  3          DRAW_X_AXIS( .MAX_ROWS+1 );
;  1058          1337  2          END;
;  1059          1338  2
;  1060          1339  2
;  1061          1340  2      !For each row:
;  1062          1341  2      !
;  1063          1342  2      INCR DIM1 FROM 0 TO  .MAX_ROWS
;  1064          1343  2      DO
;  1065          1344  3          BEGIN
;  1066          1345  3
```

```
;  1067          1346  3          !Reinitialize values for this row. If terminal understands regis,
;  1068          1347  3          !move the cursor to the beginning of the this row.  Shade the row
;  1069          1348  3          !according to the information passed in the shade array.
;  1070          1349  3          !
;  1071          1350  3          ROW_POS = .MAX_ROWS - .DIM1 + 1;
;  1072          1351  3          IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  1073          1352  3          THEN
;  1074          1353  4              BEGIN
;  1075          1354  4              MOVE_CURSOR_REGIS( GRF$_REGIS_POS, X_ORIGIN,
;  1076          1355  4                                 (.ROW_POS - 1) * UNIT_HEIGHT,
;  1077          1356  4                                 NO_VALUE, NO_VALUE );
;  1078          1357  4              SHADE_ROW_REGIS( .DIM1, (.ROW_POS * UNIT_HEIGHT), .SHADE_ARRAY_DESC);
;  1079          1358  4              MOVE_CURSOR_REGIS( GRF$_REGIS_POS, X_ORIGIN,
;  1080          1359  4                                 (.ROW_POS - 1) * UNIT_HEIGHT + 1,
;  1081          1360  4                                 NO_VALUE, NO_VALUE );
;  1082          1361  3              END;
;  1083          1362  3
;  1084          1363  3
;  1085          1364  3
;  1086          1365  3          !For each element in the row:
;  1087          1366  3          !
;  1088          1367  3          DIM2 = 0;
;  1089          1368  3          WHILE .DIM2 LEQ .XY_ARRAY_DESC[ DSC$L_U2 ]
;  1090          1369  3          DO
;  1091          1370  4              BEGIN
;  1092          1371  4
;  1093          1372  4              !Save the current xy-value for comparisons; determine the position
;  1094          1373  4              !in the row.
;  1095          1374  4              !
;  1096          1375  4              CURRENT_VALUE = .XY_ARRAY[ .DIM1, .DIM2, .XY_ARRAY_DESC ];
;  1097          1376  4              COLUMN_POS    = .DIM2;
;  1098          1377  4
;  1099          1378  4
;  1100          1379  4              !Compare the current value to all subsequent values in the row
;  1101          1380  4              !until a value which is not equal is encountered.  Keep a count
;  1102          1381  4              !of the equal values, so the right number can be printed in the
;  1103          1382  4              !graph.
;  1104          1383  4              !
;  1105          1384  4              DIM2 = .DIM2 + 1;
;  1106          1385  4              WHILE .DIM2 LEQ .XY_ARRAY_DESC[ DSC$L_U2 ]
;  1107          1386  4                    AND
;  1108          1387  5                  (.CURRENT_VALUE EQL .XY_ARRAY[ .DIM1, .DIM2, .XY_ARRAY_DESC ]
;  1109          1388  5                    OR
;  1110          1389  5                   .XY_ARRAY[ .DIM1, .DIM2, .XY_ARRAY_DESC] LEQ 0 )
;  1111          1390  4              DO
;  1112          1391  4                  DIM2 = .DIM2 + 1;
;  1113          1392  4
;  1114          1393  4
;  1115          1394  4              !Calculate the number of equal values and write them out.
;  1116          1395  4              !
;  1117          1396  4              REPEAT_COUNT = .DIM2 - .COLUMN_POS;
;  1118          1397  4              PUT_ROW_SEGMENT( .CURRENT_VALUE, .REPEAT_COUNT, .COLUMN_POS,
;  1119          1398  4                               .ROW_POS,       .SEPARATOR_CHAR, .DIM1,
;  1120          1399  4                               .SHADE_ARRAY_DESC);
;  1121          1400  4
;  1122          1401  3              END;                            !DIM2
;  1123          1402  2          END;                                !ROW_COUNT
```

```
;  1124          1403  2
;  1125          1404  2
;  1126          1405  1 END;                                    ! End of routine PLOT_SURFACE_GRAPH


                                        0FFC 00000 PLOT_SURFACE_GRAPH:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 1219
                50        0C    AC        10    AC  C3 00002          SUBL3   Y_LOW, Y_HIGH, R0          ; 1291
                                50        14    AC  C6 00008          DIVL2   Y_INCR, R0
                                13              50  D1 0000C          CMPL    R0, #19
                                                05  19 0000F          BLSS    1$
                                55              12  D0 00011          MOVL    #18, MAX_ROWS              ; 1293
                                                03  11 00014          BRB     2$
                                53              50  D0 00016 1$:      MOVL    R0, MAX_ROWS               ; 1295
             1D 00000000'       00              02  E1 00019 2$:      BBC     #2, DEVICE_FLAGS, 3$       ; 1298
                                                5B  B4 00021          CLRW    SEPARATOR_CHAR             ; 1304
                                   00000000'    00  9F 00023          PUSHAB  COLOR_SET_RYG              ; 1305
                00000000G       00              01  FB 00029          CALLS   #1, LIB$PUT_OUTPUT
                                1A              50  E8 00030          BLBS    RTN_STATUS, 5$
                                                50  DD 00033          PUSHL   RTN_STATUS
                00000000G       00              01  FB 00035          CALLS   #1, LIB$SIGNAL
                                                0F  11 0003C          BRB     5$                         ; 1298
                                02        04    AC  D1 0003E 3$:      CMPL    GRAPH_TYPE, #2             ; 1312
                                                06  12 00042          BNEQ    4$
                                5B        5C    8F  9B 00044          MOVZBW  #92, SEPARATOR_CHAR        ; 1314
                                                03  11 00048          BRB     5$
                                5B              2F  B0 0004A 4$:      MOVW    #47, SEPARATOR_CHAR        ; 1316
                                31        1C    AC  E9 0004D 5$:      BLBC    NEW_SURFACE_GRAPH, 6$      ; 1319
                00000000V       00              00  FB 00051          CALLS   #0, ERASE_PAGE             ; 1325
                                                14  AC  DD 00058      PUSHL   Y_INCR                     ; 1330
                                                0C  AC  DD 0005B      PUSHL   Y_HIGH
                                                04  AC  DD 0005E      PUSHL   GRAPH_TYPE
                                                01  A3  9F 00061      PUSHAB  1(MAX_ROWS)
                00000000V       00              04  FB 00064          CALLS   #4, DRAW_Y_AXIS
                                                01  A3  9F 0006B      PUSHAB  1(MAX_ROWS)                ; 1331
                                                18  AC  DD 0006E      PUSHL   Y_LABEL_DESC
                00000000V       00              02  FB 00071          CALLS   #2, LABEL_Y_AXIS
                                                01  A3  9F 00078      PUSHAB  1(MAX_ROWS)                ; 1336
                00000000V       00              01  FB 0007B          CALLS   #1, DRAW_X_AXIS
                                56        08    AC  D0 00082 6$:      MOVL    XY_ARRAY_DESC, R6          ; 1368
                                52              01  CE 00086          MNEGL   #1, DIM1                   ; 1398
                                              00B4  31 00089          BRW     12$
          55                    53              52  C3 0008C 7$:      SUBL3   DIM1, MAX_ROWS, R5         ; 1350
                                54        01    A5  9E 00090          MOVAB   1(R5), ROW_POS
          42 00000000'          00              02  E1 00094          BBC     #2, DEVICE_FLAGS, 8$       ; 1351
                                7E              01  CE 0009C          MNEGL   #1, -(SP)                  ; 1354
                                7E              01  CE 0009F          MNEGL   #1, -(SP)
                                55        FF    A4  9E 000A2          MOVAB   -1(R4), R5                 ; 1355
                                55              12  C4 000A6          MULL2   #18, R5
                                                55  DD 000A9          PUSHL   R5
                                7E        BF    8F  9A 000AB          MOVZBL  #191, -(SP)                ; 1354
                                                7E  D4 000AF          CLRL    -(SP)
                00000000V       00              05  FB 000B1          CALLS   #5, MOVE_CURSOR_REGIS
                                                20  AC  DD 000B8      PUSHL   SHADE_ARRAY_DESC           ; 1357
```

```
                7E              54              12  C5 000BB          MULL3   #18, ROW_POS, -(SP)                                     ;
                                                52  DD 000BF          PUSHL   DIM1                                                    ;
    00000000V   00                              03  FB 000C1          CALLS   #3, SHADE_ROW_REGIS                                     ;
                7E                              01  CE 000C8          MNEGL   #1, -(SP)                                               ; 1358
                7E                              01  CE 000CB          MNEGL   #1, -(SP)                                               ;
                                            01  A5  9F 000CE          PUSHAB  1(R5)                                                   ; 1359
                7E                          BF  8F  9A 000D1          MOVZBL  #191, -(SP)                                             ; 1358
                                                7E  D4 000D5          CLRL    -(SP)                                                   ;
    00000000V   00                              05  FB 000D7          CALLS   #5, MOVE_CURSOR_REGIS                                   ;
                                                57  D4 000DE  8$:      CLRL    DIM2                                                    ; 1367
          28    A6                              57  D1 000E0  9$:      CMPL    DIM2, 40(R6)                                            ; 1368
                                                5A  14 000E4          BGTR    12$                                                     ;
                50                              56  D0 000E6          MOVL    R6, DESC_ADR_LCL                                        ; 1375
51              52                          18  A0  C5 000E9          MULL3   24(DESC_ADR_LCL), DIM1, R1                              ;
                51                              57  C0 000EE          ADDL2   DIM2, R1                                                ;
                59                          04 B041 D0 000F1          MOVL    @4(DESC_ADR_LCL)[R1], CURRENT_VALUE                     ;
                58                              57  D0 000F6          MOVL    DIM2, COLUMN_POS                                        ; 1376
                                                57  D6 000F9  10$:      INCL    DIM2                                                    ; 1384
          28    A6                              57  D1 000FB          CMPL    DIM2, 40(R6)                                            ; 1385
                                                23  14 000FF          BGTR    11$                                                     ;
                50                              56  D0 00101          MOVL    R6, DESC_ADR_LCL                                        ; 1387
51              52                          18  A0  C5 00104          MULL3   24(DESC_ADR_LCL), DIM1, R1                              ;
                51                              57  C0 00109          ADDL2   DIM2, R1                                                ;
          04 B041                               59  D1 0010C          CMPL    CURRENT_VALUE, @4(DESC_ADR_LCL)[R1]                     ;
                                                E6  13 00111          BEQL    10$                                                     ;
                50                              56  D0 00113          MOVL    R6, DESC_ADR_LCL                                        ; 1389
51              52                          18  A0  C5 00116          MULL3   24(DESC_ADR_LCL), DIM1, R1                              ;
                51                              57  C0 0011B          ADDL2   DIM2, R1                                                ;
                                            04 B041 D5 0011E          TSTL    @4(DESC_ADR_LCL)[R1]                                    ;
                                                D5  15 00122          BLEQ    10$                                                     ;
5A              57                              58  C3 00124  11$:      SUBL3   COLUMN_POS, DIM2, REPEAT_COUNT                          ; 1396
                                            20  AC  DD 00128          PUSHL   SHADE_ARRAY_DESC                                        ; 1399
                                                52  DD 0012B          PUSHL   DIM1                                                    ; 1398
                7E                              5B  3C 0012D          MOVZWL  SEPARATOR_CHAR, -(SP)                                   ;
                                                54  DD 00130          PUSHL   ROW_POS                                                 ;
                                                58  DD 00132          PUSHL   COLUMN_POS                                              ; 1397
                7E                              59  7D 00134          MOVQ    CURRENT_VALUE, -(SP)                                    ;
    00000000V   00                              07  FB 00137          CALLS   #7, PUT_ROW_SEGMENT                                     ;
                                                A0  11 0013E          BRB     9$                                                      ; 1368
      FF46              52                    01  53  F1 00140  12$:      ACBL    MAX_ROWS, #1, DIM1, 7$                          ; 1342
                                                    04 00146          RET                                                             ; 1405

; Routine Size:  327 bytes,    Routine Base:  $CODE$ + 03A4
```

```
;  1128          1406  1 %SBTTL 'Erase the screen or page'
;  1129          1407  1 ROUTINE ERASE_PAGE : NOVALUE =
;  1130          1408  1
;  1131          1409  1 !++
;  1132          1410  1 !
;  1133          1411  1 ! FUNCTIONAL DESCRIPTION:
;  1134          1412  1 !
;  1135          1413  1 !       Starting in the first row and column erase the page.
;  1136          1414  1 !
;  1137          1415  1 ! FORMAL PARAMETERS:
;  1138          1416  1 !
;  1139          1417  1 !       None
;  1140          1418  1 !
;  1141          1419  1 ! IMPLICIT INPUTS:
;  1142          1420  1 !
;  1143          1421  1 !       None
;  1144          1422  1 !
;  1145          1423  1 ! IMPLICIT OUTPUTS:
;  1146          1424  1 !
;  1147          1425  1 !       None
;  1148          1426  1 !
;  1149          1427  1 ! ROUTINE VALUE:
;  1150          1428  1 !
;  1151          1429  1 !       None
;  1152          1430  1 !
;  1153          1431  1 ! COMPLETION CODES:
;  1154          1432  1 !
;  1155          1433  1 !       None
;  1156          1434  1 !
;  1157          1435  1 ! SIDE EFFECTS:
;  1158          1436  1 !
;  1159          1437  1 !       None
;  1160          1438  1 !
;  1161          1439  1 !--
;  1162          1440  1
;  1163          1441  2 BEGIN
;  1164          1442  2
;  1165          1443  2 LOCAL
;  1166          1444  2     RTN_STATUS,                !Status returned from RTL call
;  1167          1445  2     ROW_POS,                   !Row to start erasing from
;  1168          1446  2     COLUMN_POS                 !Column to start erasing from
;  1169          1447  2     ;
```

```
;  1171          1448  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  1172          1449  2      THEN
;  1173          1450  2
;  1174          1451  2          !Erase screen using REGIS command.
;  1175          1452  2          !
;  1176          1453  2          PUT_REGIS( %ASCID 'S(e)' )
;  1177          1454  2
;  1178          1455  2      ELSE
;  1179          1456  2          BEGIN
;  1180          1457  2
;  1181          1458  2          !Erase the screen to give the graph a "clean slate".
;  1182          1459  2          !
;  1183          1460  2          ROW_POS    = FIRST_ROW;
;  1184          1461  2          COLUMN_POS = FIRST_COLUMN;
;  1185          1462  4          IF NOT ( RTN_STATUS = LIB$ERASE_PAGE( ROW_POS, COLUMN_POS ))
;  1186          1463  3              THEN
;  1187          1464  3
;  1188          1465  3              !An error has occured while erasing the page.
;  1189          1466  3              !
;  1190          1467  3              SIGNAL( .RTN_STATUS );
;  1191          1468  3
;  1192          1469  2          END;
;  1193          1470  2
;  1194          1471  2      RETURN;
;  1195          1472  1  END;                                    ! End of routine ERASE_PAGE


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                 29  65  28  53  00168 P.AAN:  .ASCII  \S(e)\                                    ;
                                   010E0004  0016C P.AAM:  .LONG   17694724                                      ;
                                   00000000' 00170         .ADDRESS P.AAN                                        ;


                                                        .PSECT  $CODE$,NOWRT,2

                                             0000 00000 ERASE_PAGE:
                                                        .WORD   Save nothing                                    ; 1407
                                      5E            08  C2 00002         SUBL2   #8, SP                        ;
                          0F 00000000'  00            02  E1 00005         BBC     #2, DEVICE_FLAGS, 1$          ; 1448
                                        00000000'     00  9F 0000D         PUSHAB  P.AAM                         ; 1453
                             00000000G  00            01  FB 00013         CALLS   #1, LIB$PUT_OUTPUT            ;
                                                      13  11 0001A         BRB     2$                            ;
                                   04   AE            01  D0 0001C 1$:     MOVL    #1, ROW_POS                   ; 1460
                                        6E            01  D0 00020         MOVL    #1, COLUMN_POS                ; 1461
                                                      5E  DD 00023         PUSHL   SP                            ; 1462
                                              08      AE  9F 00025         PUSHAB  ROW_POS                       ;
                             00000000G  00            02  FB 00028         CALLS   #2, LIB$ERASE_PAGE            ;
                                        09            50  E8 0002F 2$:     BLBS    RTN_STATUS, 3$                ;
                                                      50  DD 00032         PUSHL   RTN_STATUS                    ; 1467
                             00000000G  00            01  FB 00034         CALLS   #1, LIB$SIGNAL                ;
                                                         04 0003B 3$:      RET                                   ; 1472

; Routine Size:  60 bytes,    Routine Base:  $CODE$ + 04EB
```

```
;  1196          1473  1
```

```
;  1198          1474  1 %SBTTL 'Draw y-axis for surface graph'
;  1199          1475  1 ROUTINE DRAW_Y_AXIS ( GRAPH_LEN, GRAPH_TYPE, Y_HIGH, Y_INCR ) : NOVALUE =
;  1200          1476  1
;  1201          1477  1 !++
;  1202          1478  1 !
;  1203          1479  1 ! FUNCTIONAL DESCRIPTION:
;  1204          1480  1 !
;  1205          1481  1 !       DRAW_Y_AXIS draws the y-axis line and writes the numeric labels.
;  1206          1482  1 !
;  1207          1483  1 ! FORMAL PARAMETERS:
;  1208          1484  1 !
;  1209          1485  1 !       GRAPH_LEN       :  number of rows in the graph
;  1210          1486  1 !       GRAPH_TYPE      :  type of graph being drawn
;  1211          1487  1 !       Y_HIGH          :  high value for y-axis numeric labels
;  1212          1488  1 !       Y_INCR          :  increment for y-axis numeric labels
;  1213          1489  1 !
;  1214          1490  1 ! IMPLICIT INPUTS:
;  1215          1491  1 !
;  1216          1492  1 !       Y_AXIS_LINE  :  column which contains the y-axis line.
;  1217          1493  1 !       DEVICE_FLAGS :  a longword of flags which are set or cleared by
;  1218          1494  1 !                       LIB$SCREEN_INFO depending on the terminal
;  1219          1495  1 !                       characteristics.
;  1220          1496  1 !
;  1221          1497  1 ! IMPLICIT OUTPUTS:
;  1222          1498  1 !
;  1223          1499  1 !       None
;  1224          1500  1 !
;  1225          1501  1 ! ROUTINE VALUE:
;  1226          1502  1 !
;  1227          1503  1 !       None
;  1228          1504  1 !
;  1229          1505  1 ! COMPLETION CODES:
;  1230          1506  1 !
;  1231          1507  1 !       None
;  1232          1508  1 !
;  1233          1509  1 ! SIDE EFFECTS:
;  1234          1510  1 !
;  1235          1511  1 !       Output to the terminal
;  1236          1512  1 !
;  1237          1513  1 !--
;  1238          1514  1
;  1239          1515  2 BEGIN
;  1240          1516  2
;  1241          1517  2 LOCAL
;  1242          1518  2     ROW_POS,                           !Row where label is to be placed
;  1243          1519  2     COLUMN_POS,                        !Column in which the label starts
;  1244          1520  2     ROW_LABEL,                         !Numeric label for y-axis
;  1245          1521  2     X_AXIS_LINE,                       !Location of x-axis line
;  1246          1522  2     CTRSTR_DESC,                       !Address of control string
;  1247          1523  2     BUFFER                  :          !Buffer for building the output
;  1248          1524  2         VECTOR[ MAX_PAGE_WIDTH, BYTE ],
;  1249          1525  2     DESC                    :          !Descriptor for BUFFER
;  1250          1526  2         BBLOCK[ DSC$K_Z_BLN ]
;  1251          1527  2     ;
;  1252          1528  2
;  1253          1529  2 BIND
;  1254          1530  2     CTRSTR_DESC1 = %ASCID'!2UB',
```

```
; 1255          1531  2     CTRSTR_DESC2 = %ASCID'!UL'
; 1256          1532  2     ;
```

```
;  1258     1533  2      !Inititalize variables
;  1259     1534  2      !
;  1260     1535  2      (BUFFER)                = '    ';
;  1261     1536  2      DESC[ DSC$B_CLASS ]     = DSC$K_CLASS_Z;
;  1262     1537  2      DESC[ DSC$B_DTYPE ]     = DSC$K_DTYPE_Z;
;  1263     1538  2      DESC[ DSC$A_POINTER ] = BUFFER;
;  1264     1539  2
;  1265     1540  2      IF .GRAPH_TYPE EQL GRF$_LINE
;  1266     1541  2      THEN
;  1267     1542  2          CTRSTR_DESC = CTRSTR_DESC1
;  1268     1543  2      ELSE
;  1269     1544  2          CTRSTR_DESC = CTRSTR_DESC2;
;  1270     1545  2
;  1271     1546  2
;  1272     1547  2      !If this is a regis terminal, draw the axis line
;  1273     1548  2      !
;  1274     1549  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  1275     1550  2      THEN
;  1276     1551  3          BEGIN
;  1277     1552  3
;  1278     1553  3          !Position the cursor at the begining of the vector.
;  1279     1554  3          !
;  1280     1555  3          MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .Y_AXIS_LINE, FIRST_ROW,
;  1281     1556  3                             NO_VALUE, NO_VALUE );
;  1282     1557  3
;  1283     1558  3          !Draw a vector finishing at the the x-axis line.
;  1284     1559  3          !
;  1285     1560  3          X_AXIS_LINE = UNIT_HEIGHT * (.GRAPH_LEN ) + AXIS_SHIFT;
;  1286     1561  3          MOVE_CURSOR_REGIS( GRF$_REGIS_VCTR, NO_VALUE, .X_AXIS_LINE,
;  1287     1562  3                             NO_VALUE, NO_VALUE );
;  1288     1563  2          END;
;  1289     1564  2
;  1290     1565  2
;  1291     1566  2      !For each row of the graph:
;  1292     1567  2      !
;  1293     1568  2      INCR ROW_COUNT FROM 0 TO .GRAPH_LEN - 1
;  1294     1569  2      DO
;  1295     1570  3          BEGIN
;  1296     1571  3
;  1297     1572  3          !Initialize variables for this row
;  1298     1573  3          !
;  1299     1574  3          DESC[ DSC$W_LENGTH ] = MAX_PAGE_WIDTH;
;  1300     1575  3          ROW_POS = .ROW_COUNT + 1;
;  1301     1576  3
;  1302     1577  3          IF .GRAPH_TYPE EQL GRF$_LINE
;  1303     1578  3          THEN
;  1304     1579  3              ROW_LABEL = .GRAPH_LEN - .ROW_COUNT
;  1305     1580  3          ELSE
;  1306     1581  3              ROW_LABEL = .Y_HIGH - (.Y_INCR * .ROW_COUNT);
;  1307     1582  3
;  1308     1583  3
;  1309     1584  3          !If this is a line graph or an "even" row, put the numeric label along
;  1310     1585  3          !the y-axis line.  This is for readability... too many numeric
;  1311     1586  3          !labels make it hard to read the graph.
;  1312     1587  3          !
;  1313     1588  3          IF .GRAPH_TYPE EQL GRF$_LINE
;  1314     1589  3                  OR
```

```
;  1315          1590  3          (.ROW_COUNT MOD 2) EQL 0
;  1316          1591  3          THEN
;  1317          1592  4              TRANSLATE_VALUE( .CTRSTR_DESC, DESC[ DSC$W_LENGTH ], DESC, ROW_LABEL )
;  1318          1593  3          ELSE
;  1319          1594  3              DESC[ DSC$W_LENGTH ] = 0;
;  1320          1595  3
;  1321          1596  3
;  1322          1597  3          !If this is the first label for the line graph, an "*" is really
;  1323          1598  3          !desired to denote that the numbers in this row are large. Replace
;  1324          1599  3          !the numeric label with a "*".
;  1325          1600  3          !
;  1326          1601  3          IF .GRAPH_TYPE EQL GRF$_LINE AND .ROW_COUNT EQL 0
;  1327          1602  3          THEN
;  1328          1603  4              BEGIN
;  1329          1604  4              (BUFFER)                = ' *';
;  1330          1605  4              DESC[ DSC$W_LENGTH ] = SEPARATOR_WIDTH;
;  1331          1606  3              END;
;  1332          1607  3
;  1333          1608  3
;  1334          1609  3          !Determine column to start the string in and write it out.
;  1335          1610  3          !
;  1336          1611  3          IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  1337          1612  3          THEN
;  1338          1613  4              BEGIN
;  1339          1614  4              COLUMN_POS = .Y_AXIS_LINE - (.DESC[ DSC$W_LENGTH ]+1) * CHAR_WIDTH;
;  1340          1615  4              MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS,
;  1341          1616  4                                 (.ROW_POS - 1) * UNIT_HEIGHT + 1,
;  1342          1617  4                                 NO_VALUE, NO_VALUE );
;  1343          1618  4              PUT_REGIS_TEXT( DESC, LIGHT_INTENSITY );
;  1344          1619  4              END
;  1345          1620  4
;  1346          1621  3          ELSE
;  1347          1622  4              BEGIN
;  1348          1623  4
;  1349          1624  4              !Add the y-axis line delimiter to the output string.
;  1350          1625  4              !
;  1351          1626  4              BUFFER[ .DESC[ DSC$W_LENGTH ] ] = '|';
;  1352          1627  4              DESC[ DSC$W_LENGTH ]            = .DESC[ DSC$W_LENGTH ] + 1;
;  1353          1628  4
;  1354          1629  4              COLUMN_POS = .Y_AXIS_LINE - .DESC[ DSC$W_LENGTH ] + 1;
;  1355          1630  4              PUT_TEXT( DESC, ROW_POS, COLUMN_POS );
;  1356          1631  3              END;
;  1357          1632  3
;  1358          1633  2          END;
;  1359          1634  1 END;                                  ! End of routine DRAW_Y_AXIS


                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                                  42  55  32  21  00174 P.AAP:  .ASCII  \'?UB\                                    ;
                                    010E0004  00178 P.AAO:  .LONG   17694724                                     ;
                                    00000000' 0017C         .ADDRESS P.AAP                                       ;
                                  00  4C  55  21  00180 P.AAR:  .ASCII  \!UL\<0>                                  ;
                                    010E0003  00184 P.AAQ:  .LONG   17694723                                     ;
                                    00000000' 00188         .ADDRESS P.AAR                                       ;
```

```
                                CTRSTR_DESC1=       P.AAO
                                CTRSTR_DESC2=       P.AAQ

                                                    .PSECT  $CODE$,NOWRT,2

                                01FC 00000 DRAW_Y_AXIS:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8                       ; 1475
                             58 00000000G   00  9E 00002    MOVAB   LIB$SIGNAL, R8                          ;
                             57 00000000V   00  9E 00009    MOVAB   MOVE_CURSOR_REGIS, R7                   ;
                             56 00000000'   00  9E 00010    MOVAB   DEVICE_FLAGS, R6                        ;
                             5E     FF68    CE  9E 00017    MOVAB   -152(SP), SP                            ;
                          14 AE 20202020    8F  D0 0001C    MOVL    #538976288, BUFFER                      ; 1535
                                     0E     AE  B4 00024    CLRW    DESC+2                                  ; 1537
                          10 AE      14     AE  9E 00027    MOVAB   BUFFER, DESC+4                          ; 1538
                                            55  D4 0002C    CLRL    R5                                      ; 1540
                                     08     AC  D5 0002E    TSTL    GRAPH_TYPE                              ;
                                            0B  12 00031    BNEQ    1$                                      ;
                                            55  D6 00033    INCL    R5                                      ;
                             54 00000000'   00  9E 00035    MOVAB   CTRSTR_DESC1, CTRSTR_DESC               ; 1542
                                            07  11 0003C    BRB     2$                                      ;
                             54 00000000'   00  9E 0003E 1$: MOVAB   CTRSTR_DESC2, CTRSTR_DESC               ; 1544
                       28                   66  02  E1 00045 2$: BBC     #2, DEVICE_FLAGS, 3$                    ; 1549
                                            7E  01  CE 00049    MNEGL   #1, -(SP)                               ; 1555
                                            7E  01  CE 0004C    MNEGL   #1, -(SP)                               ;
                                                01  DD 0004F    PUSHL   #1                                      ;
                                     04     A6  DD 00051        PUSHL   Y_AXIS_LINE                             ;
                                            7E  D4 00054        CLRL    -(SP)                                   ;
                                         67     05  FB 00056    CALLS   #5, MOVE_CURSOR_REGIS                   ;
                                50       04 AC      12  C5 00059    MULL3   #18, GRAPH_LEN, R0                  ; 1560
                                                50  05  C0 0005E    ADDL2   #5, X_AXIS_LINE                     ;
                                                7E  01  CE 00061    MNEGL   #1, -(SP)                           ; 1561
                                                7E  01  CE 00064    MNEGL   #1, -(SP)                           ;
                                                50  DD 00067        PUSHL   X_AXIS_LINE                         ;
                                                7E  01  CE 00069    MNEGL   #1, -(SP)                           ;
                                                01  DD 0006C        PUSHL   #1                                  ;
                                             67     05  FB 0006E    CALLS   #5, MOVE_CURSOR_REGIS               ;
                                             53     01  CE 00071 3$: MNEGL   #1, ROW_COUNT                   ; 1568
                                                 00C9   31  00074    BRW     12$                             ;
                                         0C  AE 84      8F  9B 00077 4$: MOVZBW  #132, DESC                  ; 1574
                                         08  AE 01      A3  9E 0007C    MOVAB   1(R3), ROW_POS                  ; 1575
                                             07         55  E9 00081    BLBC    R5, 5$                          ; 1577
                                50       04 AC          53  C3 00084    SUBL3   ROW_COUNT, GRAPH_LEN, ROW_LABEL ; 1579
                                                        0A  11 00089    BRB     6$                              ;
                                50       10 AC          53  C5 0008B 5$: MULL3   ROW_COUNT, Y_INCR, R0          ; 1581
                                6E       0C AC          50  C3 00090    SUBL3   R0, Y_HIGH, ROW_LABEL           ;
                                             0E         55  E8 00095 6$: BLBS    R5, 7$                          ; 1588
                          7E    00                 53      01  7A 00098    EMUL    #1, ROW_COUNT, #0, -(SP)    ; 1590
                          50    50                 8E      02  7B 0009D    EDIV    #2, (SP)+, R0, R0           ;
                                                        50  D5 000A2    TSTL    R0                              ;
                                                        1B  12 000A4    BNEQ    8$                              ;
                                                        5E  DD 000A6 7$: PUSHL   SP                              ; 1592
                                                    10  AE  9F 000A8    PUSHAB  DESC                            ;
                                                    14  AE  9F 000AB    PUSHAB  DESC                            ;
                                                        54  DD 000AE    PUSHL   CTRSTR_DESC                     ;
                                    00000000G       00      04  FB 000B0    CALLS   #4, SYS$FAOL                ;
                                                    0A      50  E8 000B7    BLBS    RTN_STATUS, 9$              ;
```

```
                                                50  DD 000BA          PUSHL   RTN_STATUS                                        ;
                                        68      01  FB 000BC          CALLS   #1, LIB$SIGNAL                                   ;
                                                03  11 000BF          BRB     9$                                               ; 1588
                                            0C  AE  B4 000C1 8$:      CLRW    DESC                                             ; 1594
                                        0E      55  E9 000C4 9$:      BLBC    R5, 10$                                          ; 1601
                                                53  D5 000C7          TSTL    ROW_COUNT                                        ;
                                                0A  12 000C9          BNEQ    10$                                              ;
                                    14  AE  2A20    8F  3C 000CB      MOVZWL  #10784, BUFFER                                   ; 1604
                                    0C  AE          02  B0 000D1      MOVW    #2, DESC                                         ; 1605
                                        52      04  A6  D0 000D5 10$:  MOVL    Y_AXIS_LINE, R2                                  ; 1614
                                        50      0C  AE  3C 000D9      MOVZWL  DESC, R0                                         ;
                                31      66          02  E1 000DD      BBC     #2, DEVICE_FLAGS, 11$                            ; 1611
                                51      50          09  C5 000E1      MULL3   #9, R0, R1                                       ; 1614
                                51      52          51  C3 000E5      SUBL3   R1, R2, R1                                       ;
                                    04  AE      F7  A1  9E 000E9      MOVAB   -9(R1), COLUMN_POS                               ;
                                        7E          01  CE 000EE      MNEGL   #1, -(SP)                                        ; 1615
                                        7E          01  CE 000F1      MNEGL   #1, -(SP)                                        ;
                                51  10  AE          12  C5 000F4      MULL3   #18, ROW_POS, R1                                 ; 1616
                                            EF      A1  9F 000F9      PUSHAB  -17(R1)                                          ;
                                            10  AE  DD  000FC        PUSHL   COLUMN_POS                                       ; 1615
                                                7E  D4  000FF        CLRL    -(SP)                                            ;
                                        67          05  FB 00101      CALLS   #5, MOVE_CURSOR_REGIS                            ;
                                                33  DD  00104        PUSHL   #51                                              ; 1618
                                            10  AE  9F  00106        PUSHAB  DESC                                             ;
                            00000000V   00          02  FB 00109      CALLS   #2, PUT_REGIS_TEXT                               ;
                                                2E  11  00110        BRB     12$                                              ; 1611
                                14  AE40     7C  8F  90 00112 11$:  MOVB    #124, BUFFER[R0]                                 ; 1626
                                            0C  AE  B6  00118        INCW    DESC                                             ; 1627
                                        51  0C  AE  3C  0011B        MOVZWL  DESC, R1                                         ; 1629
                                51      52          51  C3 0011F      SUBL3   R1, R2, R1                                       ;
                                    04  AE      01  A1  9E 00123      MOVAB   1(R1), COLUMN_POS                                ;
                                            04  AE  9F  00128        PUSHAB  COLUMN_POS                                       ; 1630
                                            0C  AE  9F  0012B        PUSHAB  ROW_POS                                          ;
                                            14  AE  9F  0012E        PUSHAB  DESC                                             ;
                            00000000G   00          03  FB 00131      CALLS   #3, LIB$PUT_SCREEN                               ;
                                        05          50  E8 00138      BLBS    RTN_STATUS, 12$                                  ;
                                                    50  DD 0013B      PUSHL   RTN_STATUS                                       ;
                                        68          01  FB 0013D      CALLS   #1, LIB$SIGNAL                                   ;
                                01      53      04  AC  F2 00140 12$:  AOBLSS  GRAPH_LEN, ROW_COUNT, 13$                       ; 1568
                                                    04  00145        RET                                                      ; 1634
                                                FF2E    31  00146 13$:  BRW     4$                                               ; 1568

; Routine Size:  329 bytes,    Routine Base:  $CODE$ + 0527
```

```
;  1361          1635  1 %SBTTL 'Label the y-axis for a surface graph'
;  1362          1636  1 ROUTINE LABEL_Y_AXIS( Y_LABEL_DESC, GRAPH_LEN ) : NOVALUE
;  1363          1637  1
;  1364          1638  1 !++
;  1365          1639  1 !
;  1366          1640  1 ! FUNCTIONAL DESCRIPTION:
;  1367          1641  1 !
;  1368          1642  1 !       Draw the y-axis and write the numeric labels.
;  1369          1643  1 !
;  1370          1644  1 ! FORMAL PARAMETERS:
;  1371          1645  1 !
;  1372          1646  1 !       Y_LABEL_DESC  :  Address of descriptor for descriptive labels
;  1373          1647  1 !       GRAPH_LEN     :  The number of rows in the graph
;  1374          1648  1 !
;  1375          1649  1 ! IMPLICIT INPUTS:
;  1376          1650  1 !
;  1377          1651  1 !       Y_AXIS_LINE   :  column which contains the y-axis line.
;  1378          1652  1 !       DEVICE_FLAGS  :  a longword of flags which are set or cleared by
;  1379          1653  1 !                        LIB$SCREEN_INFO depending on the terminal
;  1380          1654  1 !                        characteristics.
;  1381          1655  1 !
;  1382          1656  1 ! IMPLICIT OUTPUTS:
;  1383          1657  1 !
;  1384          1658  1 !       None
;  1385          1659  1 !
;  1386          1660  1 ! ROUTINE VALUE:
;  1387          1661  1 !
;  1388          1662  1 !       None
;  1389          1663  1 !
;  1390          1664  1 ! COMPLETION CODES:
;  1391          1665  1 !
;  1392          1666  1 !       None
;  1393          1667  1 !
;  1394          1668  1 ! SIDE EFFECTS:
;  1395          1669  1 !
;  1396          1670  1 !       None
;  1397          1671  1 !
;  1398          1672  1 !--
;  1399          1673  1
;  1400          1674  2 BEGIN
;  1401          1675  2
;  1402          1676  2 LITERAL
;  1403          1677  2     LABEL_WIDTH = 15
;  1404          1678  2     ;
;  1405          1679  2
;  1406          1680  2 LOCAL
;  1407          1681  2     ROW_POS,                          !Row where label is to be placed
;  1408          1682  2     COLUMN_POS,                       !Column in which the label starts
;  1409          1683  2     STRING_LEN,                       !Length of remaining string
;  1410          1684  2     STRING_START,                     !Address of first character
;  1411          1685  2     CURRENT_LEN,                      !Length of remaining string
;  1412          1686  2     WORD_START,                       !Address of first character in word
;  1413          1687  2     WORD_END,                         !Address of last character in word
;  1414          1688  2     WORD_COUNT,                       !Number of words in label
;  1415          1689  2     DESC                              !Descriptor for output buffer
;  1416          1690  2         BBLOCK [ DSC$K_Z_BLN ]
;  1417          1691  2     ;
```

```
; 1418          1692  2
; 1419          1693  2 MAP
; 1420          1694  2     Y_LABEL_DESC        :                   .Descriptor for y-axis labels
; 1421          1695  2         REF BBLOCK
; 1422          1696  2     ;
```

```
;  1424     1697  2      !Initialize variables
;  1425     1698  2      !
;  1426     1699  2      DESC[ DSC$B_CLASS ]   = DSC$K_CLASS_Z;
;  1427     1700  2      DESC[ DSC$B_DTYPE ]   = DSC$K_DTYPE_Z;
;  1428     1701  2      STRING_LEN   = .Y_LABEL_DESC[ DSC$W_LENGTH ];
;  1429     1702  2      CURRENT_LEN  = .STRING_LEN;
;  1430     1703  2      STRING_START = .Y_LABEL_DESC[ DSC$A_POINTER ];
;  1431     1704  2      WORD_START   = .STRING_START;
;  1432     1705  2      WORD_END     = .STRING_START;
;  1433     1706  2      WORD_COUNT   = 0;
;  1434     1707  2
;  1435     1708  2
;  1436     1709  2      !Count the number of words in the label
;  1437     1710  2      !
;  1438     1711  2      UNTIL .CURRENT_LEN GTRU .STRING_LEN
;  1439     1712  2      DO
;  1440     1713  3          BEGIN
;  1441     1714  3
;  1442     1715  3          !Find the start of the next word.
;  1443     1716  3          !
;  1444     1717  3          WORD_START  = CH$FIND_NOT_CH( .CURRENT_LEN, .WORD_END, ' ');
;  1445     1718  3          CURRENT_LEN = .STRING_LEN - ( .WORD_START - .STRING_START );
;  1446     1719  3
;  1447     1720  3
;  1448     1721  3          IF .WORD_START EQL 0
;  1449     1722  3          THEN
;  1450     1723  3
;  1451     1724  3              !There are no more words in the string. Finish processing.
;  1452     1725  3              !
;  1453     1726  3              EXITLOOP;
;  1454     1727  3
;  1455     1728  3          !Find the end of the next word.
;  1456     1729  3          !
;  1457     1730  3          WORD_END    = CH$FIND_CH( .CURRENT_LEN, .WORD_START, ' ');
;  1458     1731  3          CURRENT_LEN = .STRING_LEN - ( .WORD_END - .STRING_START );
;  1459     1732  3
;  1460     1733  3
;  1461     1734  3 !        IF (.WORD_END - .WORD_START + 1) GTR LABEL_WIDTH
;  1462     1735  3 !        THEN
;  1463     1736  3 !
;  1464     1737  3 !            !Word is too longer than the maximum width allowed (LABEL_WIDTH)
;  1465     1738  3 !            !
;  1466     1739  3 !            SIGNAL( EDF$_INTSWERR, 1, INVALID_LABEL );
;  1467     1740  3
;  1468     1741  3
;  1469     1742  3          !Count the new word.
;  1470     1743  3          WORD_COUNT = .WORD_COUNT + 1;
;  1471     1744  2          END;
;  1472     1745  2
;  1473     1746  2
;  1474     1747  2 !    IF .WORD_COUNT GTR (.GRAPH_LEN / 2) OR .WORD_COUNT EQL 0
;  1475     1748  2 !    THEN
;  1476     1749  2 !
;  1477     1750  2 !        !There are more words than lines to write them on or there are none.
;  1478     1751  2 !        !
;  1479     1752  2 !        SIGNAL( EDF$_INTSWERR, 1, INVALID_LABEL );
;  1480     1753  2
```

```
;  1481          1754  2
;  1482          1755  2      !Put the label to the screen.  Determine in which row and column the
;  1483          1756  2      !first word of the label is to be positioned.
;  1484          1757  2      !
;  1485          1758  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  1486          1759  2      THEN
;  1487          1760  3          COLUMN_POS = .Y_AXIS_LINE - ( LABEL_WIDTH  * CHAR_WIDTH )
;  1488          1761  2      ELSE
;  1489          1762  2          COLUMN_POS = .Y_AXIS_LINE - LABEL_WIDTH;
;  1490          1763  2
;  1491          1764  2
;  1492          1765  3      IF (.COLUMN_POS LSSU 1)
;  1493          1766  2      THEN
;  1494          1767  2          COLUMN_POS = 1;
;  1495          1768  2
;  1496          1769  2
;  1497          1770  2      ROW_POS = (.GRAPH_LEN - .WORD_COUNT) / 2;
;  1498          1771  3      IF NOT ((.ROW_POS MOD 2) EQL 0)
;  1499          1772  2      THEN
;  1500          1773  2          ROW_POS = .ROW_POS - 1;
;  1501          1774  2
;  1502          1775  2
;  1503          1776  2      !Extract next word from string and print it in the proper place.
;  1504          1777  2      !
;  1505          1778  2      CURRENT_LEN = .Y_LABEL_DESC[ DSC$W_LENGTH ];
;  1506          1779  2      WORD_START  = .STRING_START;
;  1507          1780  2      WORD_END    = .STRING_START;
;  1508          1781  2
;  1509          1782  2
;  1510          1783  2      DECR LOOP_INDEX FROM .WORD_COUNT TO 1 DO
;  1511          1784  3          BEGIN
;  1512          1785  3
;  1513          1786  3          !Find the start c' the next word.
;  1514          1787  3          !
;  1515          1788  3          WORD_START  = CH$FIND_NOT_CH( .CURRENT_LEN, .WORD_END, ' ');
;  1516          1789  3          CURRENT_LEN = .STRING_LEN - ( .WORD_START - .STRING_START );
;  1517          1790  3
;  1518          1791  3
;  1519          1792  3          !Find the end of the next word.
;  1520          1793  3          !
;  1521          1794  3          WORD_END    = CH$FIND_CH( .CURRENT_LEN, .WORD_START, ' ');
;  1522          1795  3          CURRENT_LEN = .STRING_LEN - ( .WORD_END - .STRING_START );
;  1523          1796  3
;  1524          1797  3          IF .WORD_END EQL 0
;  1525          1798  3          THEN
;  1526          1799  3              WORD_END = .STRING_START + .Y_LABEL_DESC[ DSC$A_POINTER ];
;  1527          1800  3
;  1528          1801  3
;  1529          1802  3          !Adjust Output buffer descriptor to point to the current word.
;  1530          1803  3          !
;  1531          1804  3          DESC[ DSC$W_LENGTH ]  = .WORD_END - .WORD_START;
;  1532          1805  3          DESC[ DSC$A_POINTER ] = .WORD_START;
;  1533          1806  3
;  1534          1807  3
;  1535          1808  3          !Write this part of the label on the graph and decide where next
;  1536          1809  3          !word should be written.
;  1537          1810  3          !
```

```
;  1538          1811  3          IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  1539          1812  3          THEN
;  1540          1813  4              BEGIN
;  1541          1814  4              MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS,
;  1542          1815  4                                 (.ROW_POS - 1) * UNIT_HEIGHT,
;  1543          1816  4                                NO_VALUE, NO_VALUE );
;  1544          1817  4              PUT_REGIS_TEXT( DESC, LIGHT_INTENSITY );
;  1545          1818  4              END
;  1546          1819  4
;  1547          1820  3          ELSE
;  1548          1821  3              PUT_TEXT( DESC, ROW_POS, COLUMN_POS );
;  1549          1822  3
;  1550          1823  3
;  1551          1824  3          ROW_POS = .ROW_POS + 2;
;  1552          1825  2          END;                                !Print loop
;  1553          1826  1 END;                                       ! End of routine LABEL_Y_AXIS



                                  03FC 00000 LABEL_Y_AXIS:
                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                    ; 1636
                59 00000000'   00   9E 00002    MOVAB   DEVICE_FLAGS, R9                               ;
                5E             10   C2 00009    SUBL2   #16, SP                                        ;
                         0A    AE   B4 0000C    CLRW    DESC+2                                         ; 1700
                54       04    AC   D0 0000F    MOVL    Y_LABEL_DESC, R4                               ; 1701
                58             64   3C 00013    MOVZWL  (R4), STRING_LEN                               ;
                55             58   D0 00016    MOVL    STRING_LEN, CURRENT_LEN                        ; 1702
                52       04    A4   D0 00019    MOVL    4(R4), STRING_START                            ; 1703
                57             52   D0 0001D    MOVL    STRING_START, WORD_START                       ; 1704
                56             52   D0 00020    MOVL    STRING_START, WORD_END                         ; 1705
                               53   D4 00023    CLRL    WORD_COUNT                                     ; 1706
                58             55   D1 00025 1$: CMPL    CURRENT_LEN, STRING_LEN                        ; 1711
                               2E   1A 00028    BGTRU   4$                                             ;
                66             55   20 3B 0002A SKPC    #32, CURRENT_LEN, (WORD_END)                   ; 1717
                               02   12 0002E    BNEQ    2$                                             ;
                               51   D4 00030    CLRL    R1                                             ;
                57             51   D0 00032 2$: MOVL    R1, WORD_START                                 ;
                50             52   57 C3 00035 SUBL3   WORD_START, STRING_START, R0                   ; 1718
                55             50   58 C1 00039 ADDL3   STRING_LEN, R0, CURRENT_LEN                   ;
                               57   D5 0003D    TSTL    WORD_START                                     ; 1721
                               17   13 0003F    BEQL    4$                                             ;
                67             55   20 3A 00041 LOCC    #32, CURRENT_LEN, (WORD_START)                 ; 1730
                               02   12 00045    BNEQ    3$                                             ;
                               51   D4 00047    CLRL    R1                                             ;
                56             51   D0 00049 3$: MOVL    R1, WORD_END                                   ;
                50             52   56 C3 0004C SUBL3   WORD_END, STRING_START, R0                     ; 1731
                55             50   58 C1 00050 ADDL3   STRING_LEN, R0, CURRENT_LEN                   ;
                               53   D6 00054    INCL    WORD_COUNT                                     ; 1743
                               CD   11 00056    BRB     1$                                             ; 1711
                50       04    A9   D0 00058 4$: MOVL    Y_AXIS_LINE, R0                                ; 1760
                07             69   02 E1 0005C BBC     #2, DEVICE_FLAGS, 5$                           ; 1758
                               6E FF79 C0 9E 00060 MOVAB   -135(R0), COLUMN_POS                        ; 1760
                               04   11 00065    BRB     6$                                             ;
                               6E   F1 A0 9E 00067 5$: MOVAB   -15(R0), COLUMN_POS                     ; 1762
                                    03 12 0006B 6$: BNEQ    7$                                         ; 1765
```

```
                                    6E        01 D0 0006D        MOVL    #1, COLUMN_POS                          ; 1767
                   50      08       AC        53 C3 00070 7$:    SUBL3   WORD_COUNT, GRAPH_LEN, R0               ; 1770
              04   AE               50        02 C7 00075        DIVL3   #2, R0, ROW_POS                         ;
        7E         00      04       AE        01 7A 0007A        EMUL    #1, ROW_POS, #0, -(SP)                  ; 1771
        50         50               8E        02 7B 00080        EDIV    #2, (SP)+, R0, R0                       ;
                                              50 D5 00085        TSTL    R0                                      ;
                                              03 13 00087        BEQL    8$                                      ;
                                    04        AE D7 00089        DECL    ROW_POS                                 ; 1773
                                    55        64 3C 0008C 8$:    MOVZWL  (R4), CURRENT_LEN                       ; 1778
                                    57        52 D0 0008F        MOVL    STRING_START, WORD_START                ; 1779
                                    56        52 D0 00092        MOVL    STRING_START, WORD_END                  ; 1780
                                              53 D6 00095        INCL    LOOP_INDEX                              ; 1783
                                        0084  31 00097           BRW     15$                                     ;
                   66               55        20 3B 0009A 9$:    SKPC    #32, CURRENT_LEN, (WORD_END)            ; 1788
                                              02 12 0009E        BNEQ    10$                                     ;
                                              51 D4 000A0        CLRL    R1                                      ;
                                    57        51 D0 000A2 10$:   MOVL    R1, WORD_START                          ;
                   50               52        57 C3 000A5        SUBL3   WORD_START, STRING_START, R0            ; 1789
                   55               50        58 C1 000A9        ADDL3   STRING_LEN, R0, CURRENT_LEN             ;
                   67               55        20 3A 000AD        LOCC    #32, CURRENT_LEN, (WORD_START)          ; 1794
                                              02 12 000B1        BNEQ    11$                                     ;
                                              51 D4 000B3        CLRL    R1                                      ;
                                    56        51 D0 000B5 11$:   MOVL    R1, WORD_END                            ;
                   50               52        56 C3 000B8        SUBL3   WORD_END, STRING_START, R0              ; 1795
                   55               50        58 C1 000BC        ADDL3   STRING_LEN, R0, CURRENT_LEN             ;
                                              56 D5 000C0        TSTL    WORD_END                                ; 1797
                                              05 12 000C2        BNEQ    12$                                     ;
                   56               52  04    A4 C1 000C4        ADDL3   4(R4), STRING_START, WORD_END           ; 1799
              08   AE               56        57 A3 000C9 12$:   SUBW3   WORD_START, WORD_END, DESC              ; 1804
                           0C       AE        57 D0 000CE        MOVL    WORD_START, DESC+4                      ; 1805
                   29               69        02 E1 000D2        BBC     #2, DEVICE_FLAGS, 13$                   ; 1811
                                    7E        01 CE 000D6        MNEGL   #1, -(SP)                               ; 1814
                                    7E        01 CE 000D9        MNEGL   #1, -(SP)                               ;
                   50      0C       AE        01 C3 000DC        SUBL3   #1, ROW_POS, R0                         ; 1815
                   7E               50        12 C5 000E1        MULL3   #18, R0, -(SP)                          ;
                                        0C    AE DD 000E5        PUSHL   COLUMN_POS                              ; 1814
                                              7E D4 000E8        CLRL    -(SP)                                   ;
                       00000000V    00        05 FB 000EA        CALLS   #5, MOVE_CURSOR_REGIS                   ;
                                              33 DD 000F1        PUSHL   #51                                     ; 1817
                                        0C    AE 9F 000F3        PUSHAB  DESC                                    ;
                       00000000V    00        02 FB 000F6        CALLS   #2, PUT_REGIS_TEXT                      ;
                                              1B 11 000FD        BRB     14$                                     ; 1811
                                              5E DD 000FF 13$:   PUSHL   SP                                      ; 1821
                                        08    AE 9F 00101        PUSHAB  ROW_POS                                 ;
                                        10    AE 9F 00104        PUSHAB  DESC                                    ;
                       00000000G    00        03 FB 00107        CALLS   #3, LIB$PUT_SCREEN                      ;
                                    09        50 E8 0010E        BLBS    RTN_STATUS, 14$                         ;
                                              50 DD 00111        PUSHL   RTN_STATUS                              ;
                       00000000G    00        01 FB 00113        CALLS   #1, LIB$SIGNAL                          ;
                           04       AE        02 C0 0011A 14$:   ADDL2   #2, ROW_POS                             ; 1824
                                    01        53 F5 0011E 15$:   SOBGTR  LOOP_INDEX, 16$                         ; 1783
                                                 04 00121        RET                                             ; 1826
                                        FF75  31 00122 16$:      BRW     9$                                      ; 1783

; Routine Size:  293 bytes,    Routine Base:  $CODE$ + 0670
```

```
;  1555          1827  1 %SBTTL 'Write a portion of a graph'
;  1556          1828  1 ROUTINE PUT_ROW_SEGMENT( CURRENT_VALUE, REPEAT_COUNT,   COLUMN_POS,
;  1557          1829  1                          ROW_POS,       SEPARATOR_CHAR, DIM1,
;  1558          1830  1                          SHADE_ARRAY_DESC ) : NOVALUE =
;  1559          1831  1
;  1560          1832  1 !++
;  1561          1833  1 !
;  1562          1834  1 ! FUNCTIONAL DESCRIPTION:
;  1563          1835  1 !
;  1564          1836  1 !       PUT_ROW_SEGMENT places a requested number of repetions of the current
;  1565          1837  1 !       value in a buffer and put the buffer to the terminal in the proper
;  1566          1838  1 !       place.
;  1567          1839  1 !
;  1568          1840  1 ! FORMAL PARAMETERS:
;  1569          1841  1 !
;  1570          1842  1 !       CURRENT_VALUE   : Current value to be printed in the row segment
;  1571          1843  1 !       REPEAT_COUNT    : Number of times the value is to be printed
;  1572          1844  1 !       COLUMN_POS      : Column in which row segment starts
;  1573          1845  1 !       ROW_POS         : Row in which row segment is to be written
;  1574          1846  1 !       SEPARATOR_CHAR  : Character used to delimit "good" section
;  1575          1847  1 !       DIM1            : Index into SHADE_ARRAY
;  1576          1848  1 !       SHADE_ARRAY_DESC: Descriptor for the array of shading information
;  1577          1849  1 !
;  1578          1850  1 ! IMPLICIT INPUTS:
;  1579          1851  1 !
;  1580          1852  1 !       DEVICE_FLAGS :  a longword of flags which are set or cleared by
;  1581          1853  1 !                       LIB$SCREEN_INFO depending on the terminal
;  1582          1854  1 !                       characteristics.
;  1583          1855  1 !
;  1584          1856  1 ! IMPLICIT OUTPUTS:
;  1585          1857  1 !
;  1586          1858  1 !       None
;  1587          1859  1 !
;  1588          1860  1 ! ROUTINE VALUE:
;  1589          1861  1 !
;  1590          1862  1 !       None
;  1591          1863  1 !
;  1592          1864  1 ! COMPLETION CODES:
;  1593          1865  1 !
;  1594          1866  1 !       None
;  1595          1867  1 !
;  1596          1868  1 ! SIDE EFFECTS:
;  1597          1869  1 !
;  1598          1870  1 !       Output to the terminal.
;  1599          1871  1 !
;  1600          1872  1 !--
;  1601          1873  1
;  1602          1874  2 BEGIN
;  1603          1875  2
;  1604          1876  2 LITERAL
;  1605          1877  2     TWO_DIGITS = 10             !First two digit number
;  1606          1878  2     ;
;  1607          1879  2
;  1608          1880  2 LOCAL
;  1609          1881  2     SHADE_ARRAY         :       !Array containing shading values
;  1610          1882  2         REF TWO_DIM_ARRAY[ LONG, UNSIGNED ],
;  1611          1883  2     BUFFER              :       !Buffer for building the output
```

```
;  1612          1884  2          VECTOR[ MAX_PAGE_WIDTH, BYTE ],
;  1613          1885  2      DESC                :                !Descriptor for BUFFER
;  1614          1886  2          BBLOCK[ DSC$K_Z_BLN ],
;  1615          1887  2      REGION_FOUND        :  BYTE  !Flag -- set if we need to delimit "good" area
;  1616          1888  2      ;
;  1617          1889  2
;  1618          1890  2  BIND
;  1619          1891  2      CTRSTR_DESC = %ASCID'!2UB'
;  1620          1892  2      ;
```

```
;  1622          1893  2     !Inititalize variables
;  1623          1894  2     !
;  1624          1895  2     (BUFFER)                = '    ';
;  1625          1896  2     DESC[ DSC$B_CLASS ]     = DSC$K_CLASS_Z;
;  1626          1897  2     DESC[ DSC$B_DTYPE ]     = DSC$K_DTYPE_Z;
;  1627          1898  2     DESC[ DSC$A_POINTER ] = BUFFER;
;  1628          1899  2     DESC[ DSC$W_LENGTH  ] = SEPARATOR_WIDTH;
;  1629          1900  2     REGION_FOUND            = FALSE;
;  1630          1901  2
;  1631          1902  2
;  1632          1903  2     !If this is an invalid value, then use blanks to fill the row segment.
;  1633          1904  2     !Otherwise, translate the value into ASCII.
;  1634          1905  2     !
;  1635          1906  2     IF .CURRENT_VALUE LEQ 0
;  1636          1907  2     THEN
;  1637          1908  2         (BUFFER) = '  '
;  1638          1909  2     ELSE
;  1639        P 1910  2         TRANSLATE_VALUE( CTRSTR_DESC, DESC[ DSC$W_LENGTH ], DESC,
;  1640          1911  2                          CURRENT_VALUE );
;  1641          1912  2
;  1642          1913  2
;  1643          1914  2     !If the start of this row segement is also the start of a "good" section,
;  1644          1915  2     !then flag it so that the end of the "good" section can be looked for.
;  1645          1916  2     !
;  1646          1917  2     IF .SEPARATOR_CHAR NEQ NULL
;  1647          1918  2                AND
;  1648          1919  2        .CURRENT_VALUE LSS TWO_DIGITS
;  1649          1920  2                AND
;  1650          1921  2        .SHADE_ARRAY[.DIM1, .COLUMN_POS, .SHADE_ARRAY_DESC]
;  1651          1922  3         EQL (LIGHT_INTENSITY - '0')
;  1652          1923  2     THEN
;  1653          1924  2         REGION_FOUND = TRUE;
;  1654          1925  2
;  1655          1926  2
;  1656          1927  2     !For each repition of the value:
;  1657          1928  2     !
;  1658          1929  3     INCR LOOP_COUNT FROM 1 TO (.REPEAT_COUNT - 1)
;  1659          1930  2     DO
;  1660          1931  3         BEGIN
;  1661          1932  3
;  1662          1933  3         !Copy the ASCII value into the next available position in the buffer.
;  1663          1934  3         !
;  1664          1935  3         CH$MOVE( SEPARATOR_WIDTH, BUFFER,
;  1665          1936  3                  (BUFFER + (.LOOP_COUNT * SEPARATOR_WIDTH)));
;  1666          1937  3         DESC[ DSC$W_LENGTH ] = .DESC[ DSC$W_LENGTH ] + SEPARATOR_WIDTH;
;  1667          1938  3
;  1668          1939  3
;  1669          1940  3         !If the end of the "good" area is encountered, put in a delimiter.
;  1670          1941  3         !
;  1671          1942  3         IF .REGION_FOUND
;  1672          1943  3                 AND
;  1673          1944  3            .SHADE_ARRAY[.DIM1, (.COLUMN_POS + .LOOP_COUNT), .SHADE_ARRAY_DESC]
;  1674          1945  4            NEQ (LIGHT_INTENSITY - '0')
;  1675          1946  3         THEN
;  1676          1947  4             BEGIN
;  1677          1948  4             BUFFER[ .DESC[ DSC$W_LENGTH ] - 2 ] = .SEPARATOR_CHAR;
;  1678          1949  4             REGION_FOUND = FALSE;
```

```
;  1679     1950  3              END;
;  1680     1951  3
;  1681     1952  2          END;
;  1682     1953  2
;  1683     1954  2
;  1684     1955  2      !If this is the beginning or the end of the "good" area of the graph,
;  1685     1956  2      !place a visual separator in the front of the buffer.  If separator is
;  1686     1957  2      !more than one byte in length, this won't work -- it only moves one
;  1687     1958  2      !byte into the buffer at the speci ied position.
;  1688     1959  2      !
;  1689     1960  2      IF .SEPARATOR_CHAR NEQ NULL
;  1690     1961  3                 AND
;  1691     1962  2         .CURRENT_VALUE LSS TWO_DIGITS
;  1692     1963  2                 AND
;  1693     1964  3         (.SHADE_ARRAY[.DIM1, .COLUMN_POS, .SHADE_ARRAY_DESC]
;  1694     1965  4          EQL (LIGHT_INTENSITY - '0')
;  1695     1966  3                 OR
;  1696     1967  3         .SHADE_ARRAY[.DIM1, .COLUMN_POS - 1, .SHADE_ARRAY_DESC]
;  1697     1968  3          EQL (LIGHT_INTENSITY - '0'))
;  1698     1969  2      THEN
;  1699     1970  2          BUFFER[ 0 ] = .SEPARATOR_CHAR;
;  1700     1971  2
;  1701     1972  2
;  1702     1973  2      !If the end of the "good" region has not been encountered, then assume
;  1703     1974  2      !the region ends with the row segment. Put the separator at the end.
;  1704     1975  2      !
;  1705     1976  2      IF .REGION_FOUND
;  1706     1977  2      THEN
;  1707     1978  3          BEGIN
;  1708     1979  3          BUFFER[ .DESC[DSC$W_LENGTH] ] = .SEPARATOR_CHAR;
;  1709     1980  3          DESC[ DSC$W_LENGTH ] = .DESC[ DSC$W_LENGTH ] + 1;
;  1710     1981  2          END;
;  1711     1982  2
;  1712     1983  2
;  1713     1984  2      !Put the buffer out to the terminal.
;  1714     1985  2      !
;  1715     1986  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  1716     1987  2      THEN
;  1717     1988  2          PUT_REGIS_TEXT( DESC, BACKGROUND_INTENSITY )
;  1718     1989  2      ELSE
;  1719     1990  3          BEGIN
;  1720     1991  3          COLUMN_POS = (.COLUMN_POS + 1) * SEPARATOR_WIDTH + .Y_AXIS_LINE;
;  1721     1992  3          PUT_TEXT( DESC, ROW_POS, COLUMN_POS );
;  1722     1993  2          END;
;  1723     1994  2
;  1724     1995  2      RETURN;
;  1725     1996  1  END;                                   ! End of routine PUT_ROW_SEGMENT


                                                         .PSECT  $PLIT$,NOWRT,NOEXE,2

                                  42  55  32  21  0018C P.AAT:  .ASCII  \!2UB\                    ;
                                        010E0004  00190 P.AAS:  .LONG   17694724                  ;
                                        00000000' 00194         .ADDRESS P.AAT                    ;

                                                              CTRSTR_DESC=        P.AAS
```

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

                                                00DC 00000 PUT_ROW_SEGMENT:
                                                           .WORD   Save R2,R3,R4,R5,R6,R7                  ; 1828
                    57 00000000G     00    9E 00002        MOVAB   LIB$SIGNAL, R7
                    5E       FF78    CE    9E 00009        MOVAB   -136(SP), SP
                04  AE 20202020     8F    D0 0000E        MOVL    #538976288, BUFFER                      ; 1895
                                     02    DD 00016        PUSHL   #2                                      ; 1899
                04  AE         08    AE    9E 00018        MOVAB   BUFFER, DESC+4                          ; 1898
                                     55    94 0001D        CLRB    REGION_FOUND                            ; 1900
                                 04  AC    D5 0001F        TSTL    CURRENT_VALUE                           ; 1906
                                     08    14 00022        BGTR    1$
                08  AE       2020     8F    3C 00024        MOVZWL  #8224, BUFFER                           ; 1908
                                     1E    11 0002A        BRB     2$
                                 04  AC    9F 0002C 1$:    PUSHAB  CURRENT_VALUE                           ; 1911
                                 04  AE    9F 0002F        PUSHAB  DESC
                                 08  AE    9F 00032        PUSHAB  DESC
                            00000000'    00    9F 00035        PUSHAB  CTRSTR_DESC
            00000000G       00           04    FB 0003B        CALLS   #4, SYS$FAOL
                            05           50    E8 00042        BLBS    RTN_STATUS, 2$
                                         50    DD 00045        PUSHL   RTN_STATUS
                            67           01    FB 00047        CALLS   #1, LIB$SIGNAL
                            54       14  AC    D0 0004A 2$:    MOVL    SEPARATOR_CHAR, R4              ; 1917
                                         56    D4 0004E        CLRL    R6
                                         54    D5 00050        TSTL    R4
                                         20    13 00052        BEQL    3$
                                         56    D6 00054        INCL    R6
                                0A   04  AC    D1 00056        CMPL    CURRENT_VALUE, #10              ; 1919
                                         18    18 0005A        BGEQ    3$
                                50   1C  AC    D0 0005C        MOVL    SHADE_ARRAY_DESC, DESC_ADR_LCL  ; 1921
            51      18          AC   18  A0    C5 00060        MULL3   24(DESC_ADR_LCL), DIM1, R1
                                51   0C  AC    C0 00066        ADDL2   COLUMN_POS, R1
                                03   04  B041  D1 0006A        CMPL    @4(DESC_ADR_LCL)[R1], #3        ; 1922
                                         03    12 0006F        BNEQ    3$
                                55           01    90 00071        MOVB    #1, REGION_FOUND                ; 1924
                                             52    D4 00074 3$:    CLRL    LOOP_COUNT                      ; 1929
                                             2F    11 00076        BRB     5$
                            08  AE42       08    AE    B0 00078 4$:    MOVW    BUFFER, BUFFER[LOOP_COUNT]  ; 1936
                                6E           02    A0 0007E        ADDW2   #2, DESC                        ; 1937
                                23           55    E9 00081        BLBC    REGION_FOUND, 5$                ; 1942
                                50   1C  AC    D0 00084        MOVL    SHADE_ARRAY_DESC, DESC_ADR_LCL  ; 1944
            51      18          AC   18  A0    C5 00088        MULL3   24(DESC_ADR_LCL), DIM1, R1
            53                  52   0C  AC    C1 0008E        ADDL3   COLUMN_POS, LOOP_COUNT, R3
                                51           53    C0 00093        ADDL2   R3, R1
                                03   04  B041  D1 00096        CMPL    @4(DESC_ADR_LCL)[R1], #3        ; 1945
                                         0A    13 0009B        BEQL    5$
                                50           6E    3C 0009D        MOVZWL  DESC, R0                        ; 1948
                            06  AE40       54    90 000A0        MOVB    R4, BUFFER-2[R0]
                                             55    94 000A5        CLRB    REGION_FOUND                    ; 1949
                CC              52   08  AC    F2 000A7 5$:    AOBLSS  REPEAT_COUNT, LOOP_COUNT, 4$    ; 1929
                                38           56    E9 000AC        BLBC    R6, 7$                          ; 1960
                                0A   04  AC    D1 000AF        CMPL    CURRENT_VALUE, #10              ; 1962
                                         32    18 000B3        BGEQ    7$
                                50   1C  AC    D0 000B5        MOVL    SHADE_ARRAY_DESC, DESC_ADR_LCL  ; 1964
            51      18          AC   18  A0    C5 000B9        MULL3   24(DESC_ADR_LCL), DIM1, R1
```

```
                         51      0C   AC  C0 000BF          ADDL2   COLUMN_POS, R1
                         03      04 B041  D1 000C3          CMPL    @4(DESC_ADR_LCL)[R1], #3                    ;1965
                                    19    13 000C8          BEQL    6$
                         50      1C   AC  D0 000CA          MOVL    SHADE_ARRAY_DESC, DESC_ADR_LCL              ;1967
                 51      18      AC   18  A0 C5 000CE       MULL3   24(DESC_ADR_LCL), DIM1, R1
                         51      0C   AC  C0 000D4          ADDL2   COLUMN_POS, R1
                         50      04 B041  DE 000D8          MOVAL   @4(DESC_ADR_LCL)[R1], R0
                         03      FC   A0  D1 000DD          CMPL    -4(R0), #3                                  ;1968
                                    04    12 000E1          BNEQ    7$
                      08 AE              54  90 000E3 6$:    MOVB    R4, BUFFER                                  ;1970
                         0A              55  E9 000E7 7$:    BLBC    REGION_FOUND, 8$                            ;1976
                         50              6E  3C 000EA       MOVZWL  DESC, R0                                    ;1979
                      08 AE40            54  90 000ED       MOVB    R4, BUFFER[R0]
                                         6E  B6 000F2       INCW    DESC                                        ;1980
                 0D 00000000'    00      02  E1 000F4 8$:    BBC     #2, DEVICE_FLAGS, 9$                        ;1986
                                         30  DD 000FC       PUSHL   #48                                         ;1988
                                   04    AE  9F 000FE       PUSHAB  DESC
                    00000000V    00      02  FB 00101       CALLS   #2, PUT_REGIS_TEXT
                                             04 00108       RET
                         50      0C   AC  D0 00109 9$:    MOVL    COLUMN_POS, R0                              ;1991
                         51 00000000'  00  D0 0010D       MOVL    Y_AXIS_LINE, R1
                      0C   AC        02 A140  3E 00114       MOVAW   2(R1)[R0], COLUMN_POS
                                   0C   AC  9F 0011A       PUSHAB  COLUMN_POS                                  ;1992
                                   10   AC  9F 0011D       PUSHAB  ROW_POS
                                   08   AE  9F 00120       PUSHAB  DESC
                    00000000G    00      03  FB 00123       CALLS   #3, LIB$PUT_SCREEN
                         05              50  E8 0012A       BLBS    RTN_STATUS, 10$
                                         50  DD 0012D       PUSHL   RTN_STATUS
                         67              01  FB 0012F       CALLS   #1, LIB$SIGNAL
                                             04 00132 10$:    RET                                             ;1996

; Routine Size:  307 bytes,    Routine Base:  $CODE$ + 0795
```

```
;  1727          1997  1 %SBTTL 'Position the cursor on a given pixel address'
;  1728          1998  1 ROUTINE MOVE_CURSOR_REGIS (COMMAND_TYPE, X_POS, Y_POS, INTENSITY, SHADE_LINE)
;  1729          1999  1         : NOVALUE =
;  1730          2000  1
;  1731          2001  1 !++
;  1732          2002  1 !
;  1733          2003  1 ! FUNCTIONAL DESCRIPTION:
;  1734          2004  1 !
;  1735          2005  1 !
;  1736          2006  1 !       MOVE_CURSOR_REGIS builds REGIS commands to move the cursor to a
;  1737          2007  1 !       specified position, draw a vector from the current postion to a
;  1738          2008  1 !       another postion, or draw a vector with shading to a relative line.
;  1739          2009  1 !       Send the command to the terminal.
;  1740          2010  1 !
;  1741          2011  1 !       >>>NOTE: all pixel addresses are absolute, except the line to be
;  1742          2012  1 !                shaded which is relative to the current postion.
;  1743          2013  1 !
;  1744          2014  1 ! FORMAL PARAMETERS:
;  1745          2015  1 !
;  1746          2016  1 !       COMMAND_TYPE : code to indicate if a postion or vector command is
;  1747          2017  1 !                      being built
;  1748          2018  1 !       X_POS        : x-value address of pixel
;  1749          2019  1 !       Y_POS        : y-value address of pixel
;  1750          2020  1 !       INTENSITY    : intensity of the shaded area
;  1751          2021  1 !       SHADE_LINE   : relative pixel address of the shade line
;  1752          2022  1 !
;  1753          2023  1 ! IMPLICIT INPUTS:
;  1754          2024  1 !
;  1755          2025  1 !       None
;  1756          2026  1 !
;  1757          2027  1 ! IMPLICIT OUTPUTS:
;  1758          2028  1 !
;  1759          2029  1 !       None
;  1760          2030  1 !
;  1761          2031  1 ! ROUTINE VALUE:
;  1762          2032  1 !
;  1763          2033  1 !       None
;  1764          2034  1 !
;  1765          2035  1 ! COMPLETION CODES:
;  1766          2036  1 !
;  1767          2037  1 !       None
;  1768          2038  1 !
;  1769          2039  1 ! SIDE EFFECTS:
;  1770          2040  1 !
;  1771          2041  1 !       None
;  1772          2042  1 !
;  1773          2043  1 !--
;  1774          2044  1
;  1775          2045  2 BEGIN
;  1776          2046  2
;  1777          2047  2 BIND
;  1778          2048  2     POS_CMD_START  = UPLIT( 'P[' ),      !Command to move cursor
;  1779          2049  2     COMMAND_MID    = UPLIT( ',' ),       !X and Y value separator
;  1780          2050  2     COMMAND_END    = UPLIT( ']' ),       !Command terminator
;  1781          2051  2     VCTR_CMD_START = UPLIT( 'V(W(I3))[][' ),     !Vector command start
;  1782          2052  2     SHADE_CMD_1    = UPLIT( 'V(W (S1 S[,' ),     !First part of shade command
;  1783          2053  2     SHADE_CMD_2    = UPLIT( '] I3))[][+' )       !Remainder of command start
```

```
2       ;
2
2
2 LITERAL
2       POS_CMD_START_LEN  = %CHARCOUNT( 'P[' ),
2       CMD_MID_LEN        = %CHARCOUNT( ',' ),
2       CMD_END_LEN        = %CHARCOUNT( ']' ),
2       VCTR_CMD_START_LEN = %CHARCOUNT( 'V(W(I3))[][ ' ),
2       SHADE_CMD_1_LEN    = %CHARCOUNT( 'V(W (S1 S[,' ),
2       SHADE_CMD_2_LEN    = %CHARCOUNT( '] I3))[][+' ),
2       INTENSITY_LOC      = 7                 !Distance to intensity indicator from end
2       ;
2
2 LOCAL
2       TEMP_DESC :                            !Record temporary changes in buffer here
2           BBLOCK[ DSC$K_Z_BLN ]
2           INITIAL ( 0, 0 ),
2       BUFFER    :                            !Buffer to build REGIS command
2           VECTOR[ MAX_PAGE_WIDTH, BYTE ],
2       DESC      :                            !Descriptor for the buffer
2           BBLOCK[ DSC$K_Z_BLN ]
2       ;
2
2 BIND
2       CTRSTR = %ASCID '!UL'
2       ;
```

```
;  1812          2081  2
;  1813          2082  2      !Initialize descriptor
;  1814          2083  2      !
;  1815          2084  2      DESC[ DSC$B_CLASS ]   = DSC$K_CLASS_Z;
;  1816          2085  2      DESC[ DSC$B_DTYPE ]   = DSC$K_DTYPE_Z;
;  1817          2086  2      DESC[ DSC$A_POINTER ] = BUFFER;
;  1818          2087  2
;  1819          2088  2
;  1820          2089  2      !Place start of command string in buffer
;  1821          2090  2      !
;  1822          2091  2      SELECTONE .COMMAND_TYPE OF
;  1823          2092  2          SET
;  1824          2093  2
;  1825          2094  2          !Positon cursor
;  1826          2095  2          !
;  1827          2096  2          [ GRF$_REGIS_POS ] :
;  1828          2097  3              BEGIN
;  1829          2098  3              CH$MOVE( POS_CMD_START_LEN, POS_CMD_START, BUFFER);
;  1830          2099  3              DESC[ DSC$W_LENGTH ] = POS_CMD_START_LEN;
;  1831          2100  2              END;
;  1832          2101  2
;  1833          2102  2          !Draw vector
;  1834          2103  2          !
;  1835          2104  2          [ GRF$_REGIS_VCTR ] :
;  1836          2105  3              BEGIN
;  1837          2106  3              CH$MOVE( VCTR_CMD_START_LEN, VCTR_CMD_START, BUFFER);
;  1838          2107  3              DESC[ DSC$W_LENGTH ] = VCTR_CMD_START_LEN;
;  1839          2108  2              END;
;  1840          2109  2
;  1841          2110  2          !Draw a shaded vector
;  1842          2111  2          !
;  1843          2112  2          [ GRF$_REGIS_SHADE ] :
;  1844          2113  3              BEGIN
;  1845          2114  3              CH$MOVE( SHADE_CMD_1_LEN, SHADE_CMD_1, BUFFER);
;  1846          2115  3              DESC[ DSC$W_LENGTH ] = SHADE_CMD_1_LEN;
;  1847          2116  3
;  1848          2117  3
;  1849          2118  3              !Translate shade line address into ASCII and place in command.
;  1850          2119  3              !
;  1851          2120  3              TEMP_DESC[ DSC$W_LENGTH ] = MAX_PAGE_WIDTH - .DESC[ DSC$W_LENGTH ];
;  1852          2121  3              TEMP_DESC[ DSC$A_POINTER ] =
;  1853          2122  3                                  .DESC[ DSC$A_POINTER ] + .DESC[ DSC$W_LENGTH ];
;  1854        P 2123  3              TRANSLATE_VALUE( CTRSTR, TEMP_DESC[ DSC$W_LENGTH ], TEMP_DESC,
;  1855          2124  3                               SHADE_LINE );
;  1856          2125  3              DESC[ DSC$W_LENGTH ] =
;  1857          2126  3                                  .DESC[ DSC$W_LENGTH ] + .TEMP_DESC[ DSC$W_LENGTH ];
;  1858          2127  3
;  1859          2128  3
;  1860          2129  3              CH$MOVE( SHADE_CMD_2_LEN, SHADE_CMD_2,
;  1861          2130  3                       BUFFER[ .DESC[ DSC$W_LENGTH ]  ]);
;  1862          2131  3              DESC[ DSC$W_LENGTH ] = .DESC[ DSC$W_LENGTH ] + SHADE_CMD_2_LEN;
;  1863          2132  3
;  1864          2133  3
;  1865          2134  3              !Place desired intensity in command string.
;  1866          2135  3              !
;  1867          2136  3              BUFFER[ .DESC[ DSC$W_LENGTH ] - INTENSITY_LOC ] =
;  1868          2137  3                          .INTENSITY;
```

```
;  1869     2138  2              END;
;  1870     2139  2          TES;
;  1871     2140  2
;  1872     2141  2
;  1873     2142  2      !If there is an x value, translate it and put it in the command string.
;  1874     2143  2      !
;  1875     2144  2      IF .X_POS NEQ NO_VALUE
;  1876     2145  2      THEN
;  1877     2146  3          BEGIN
;  1878     2147  3
;  1879     2148  3          TEMP_DESC[ DSC$W_LENGTH ]  = MAX_PAGE_WIDTH - .DESC[ DSC$W_LENGTH ];
;  1880     2149  3          TEMP_DESC[ DSC$A_POINTER ] =
;  1881     2150  3                  .DESC[ DSC$A_POINTER ] + .DESC[ DSC$W_LENGTH ];
;  1882     2151  3          TRANSLATE_VALUE( CTRSTR, TEMP_DESC[ DSC$W_LENGTH ], TEMP_DESC, X_POS );
;  1883     2152  3          DESC[ DSC$W_LENGTH ]       =
;  1884     2153  3                  .DESC[ DSC$W_LENGTH ] + .TEMP_DESC[ DSC$W_LENGTH ];
;  1885     2154  2          END;
;  1886     2155  2
;  1887     2156  2
;  1888     2157  2      !Put comma in to separate x and y values.  If separator is more than one
;  1889     2158  2      !byte this will not work!!!
;  1890     2159  2      !
;  1891     2160  2      BUFFER[ .DESC[ DSC$W_LENGTH ] ] = .COMMAND_MID;
;  1892     2161  2      DESC[ DSC$W_LENGTH ]            = .DESC[ DSC$W_LENGTH ] + CMD_MID_LEN;
;  1893     2162  2
;  1894     2163  2
;  1895     2164  2      !If there is a y value, translate it and put it in the command string.
;  1896     2165  2      !
;  1897     2166  2      IF .Y_POS NEQ NO_VALUE
;  1898     2167  2      THEN
;  1899     2168  3          BEGIN
;  1900     2169  3
;  1901     2170  3          TEMP_DESC[ DSC$W_LENGTH ]  = MAX_PAGE_WIDTH - .DESC[ DSC$W_LENGTH ];
;  1902     2171  3          TEMP_DESC[ DSC$A_POINTER ] =
;  1903     2172  3                  .DESC[ DSC$A_POINTER ] + .DESC[ DSC$W_LENGTH ];
;  1904     2173  3          TRANSLATE_VALUE( CTRSTR, TEMP_DESC[ DSC$W_LENGTH ], TEMP_DESC, Y_POS );
;  1905     2174  3          DESC[ DSC$W_LENGTH ]       =
;  1906     2175  3                  .DESC[ DSC$W_LENGTH ] + .TEMP_DESC[ DSC$W_LENGTH ];
;  1907     2176  2          END;
;  1908     2177  2
;  1909     2178  2
;  1910     2179  2      !Finish command string.
;  1911     2180  2      !
;  1912     2181  2      BUFFER[ .DESC[ DSC$W_LENGTH ] ] = .COMMAND_END;
;  1913     2182  2      DESC[ DSC$W_LENGTH ]            = .DESC[ DSC$W_LENGTH ] + CMD_END_LEN;
;  1914     2183  2
;  1915     2184  2
;  1916     2185  2      !Write the command string.
;  1917     2186  2      !
;  1918     2187  2      PUT_REGIS( DESC );
;  1919     2188  2
;  1920     2189  2
;  1921     2190  2      RETURN;
;  1922     2191  2
;  1923     2192  1 END;                                    ! End of routine MOVE_CURSOR_REGIS
```

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

                                                00 00 5B 50 00198 P.AAU:  .ASCII  \P[\<0><0>
                                                00 00 00 2C 0019C P.AAV:  .ASCII  \,\<0><0><0>
                                                00 00 00 5D 001A0 P.AAW:  .ASCII  \]\<0><0><0>
                                00 5B 5D 5B 29 29 33 49 28 57 28 56 001A4 P.AAX:  .ASCII  \V(W(I3))[][\<0>
                                00 2C 5B 53 20 31 53 28 20 57 28 56 001B0 P.AAY:  .ASCII  \V(W (S1 S[,\<0>
                                00 00 2B 5B 5D 5B 29 29 33 49 20 5D 001BC P.AAZ:  .ASCII  \] I3))[][+\<0><0>
                                                00 4C 55 21 001C8 P.ABB:  .ASCII  \!UL\<0>
                                                      010E0003  001CC P.ABA:  .LONG   17694723
                                                     00000000'  001D0        .ADDRESS P.ABB

                                                                POS_CMD_START=      P.AAU
                                                                COMMAND_MID=        P.AAV
                                                                COMMAND_END=        P.AAW
                                                                VCTR_CMD_START=     P.AAX
                                                                SHADE_CMD_1=        P.AAY
                                                                SHADE_CMD_2=        P.AAZ
                                                                CTRSTR=             P.ABA

                                                        .PSECT  $CODE$,NOWRT,2

                                                     01FC 00000 MOVE_CURSOR_REGIS:
                                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8                       ; 1998
                                        58 00000000G   00  9E 00002         MOVAB   SYS$FAOL, R8                        ;
                                        57 00000000G   00  9E 00009         MOVAB   LIB$SIGNAL, R7                      ;
                                        56 00000000'   00  9E 00010         MOVAB   CTRSTR, R6                          ;
                                        5E     FF6C    CE  9E 00017         MOVAB   -148(SP), SP                        ;
                                                 F8    AD  7C 0001C         CLRQ    TEMP_DESC                           ; 2045
                                                 02    AE  B4 0001F         CLRW    DESC+2                              ; 2085
                                           04    AE       08  AE  9E 00022         MOVAB   BUFFER, DESC+4                      ; 2086
                                                 50       04  AC  D0 00027         MOVL    COMMAND_TYPE, R0            ; 2091
                                                          0A  12  0002B         BNEQ    1$                          ; 2096
                                           08    AE       CC  A6  B0 0002D         MOVW    POS_CMD_START, BUFFER       ; 2098
                                                 6E           02  B0 00032         MOVW    #2, DESC                    ; 2099
                                                              5E  11 00035         BRB     4$                          ; 2091
                                                 01           50  D1 00037 1$:      CMPL    R0, #1                      ; 2104
                                                              0B  12 0003A         BNEQ    2$                          ;
                                     08    AE       D8  A6           0C  28 0003C         MOVC3   #12, VCTR_CMD_START, BUFFER ; 2106
                                                 6E           0C  B0 00042         MOVW    #12, DESC                   ; 2107
                                                              4E  11 00045         BRB     4$                          ; 2091
                                                 02           50  D1 00047 2$:      CMPL    R0, #2                      ; 2112
                                                              49  12 0004A         BNEQ    4$                          ;
                                     08    AE       E4  A6           0B  28 0004C         MOVC3   #11, SHADE_CMD_1, BUFFER    ; 2114
                                                 6E           0B  B0 00052         MOVW    #11, DESC                   ; 2115
                                     F8    AD     0084  8F           6E  A3 00055         SUBW3   DESC, #132, TEMP_DESC       ; 2120
                                                 50           6E  3C 0005C         MOVZWL  DESC, R0                    ; 2122
                                                 FC    AD       04 BE40  9E 0005F         MOVAB   @DESC+4[R0], TEMP_DESC+4    ;
                                                          14    AC  9F 00065         PUSHAB  SHADE_LINE                  ; 2124
                                                          F8    AD  9F 00068         PUSHAB  TEMP_DESC                   ;
                                                          F8    AD  9F 0006B         PUSHAB  TEMP_DESC                   ;
                                                                56  DD 0006E         PUSHL   R6                          ;
                                                 68           04  FB 00070         CALLS   #4, SYS$FAOL                ;
                                                 05           50  E8 00073         BLBS    RTN_STATUS, 3$              ;
                                                              50  DD 00076         PUSHL   RTN_STATUS                  ;
```

```
                     67           01 FB 00078          CALLS   #1, LIB$SIGNAL                       ;
                     6E     F8    AD A0 0007B 3$:      ADDW2   TEMP_DESC, DESC                      ; 2126
                     50           6E 3C 0007F          MOVZWL  DESC, R0                             ; 2130
       08 AE40   F0  A6           0A 28 00082          MOVC3   #10, SHADE_CMD_2, BUFFER[R0]         ;
                     6E           0A A0 00089          ADDW2   #10, DESC                            ; 2131
                     50           6E 3C 0008C          MOVZWL  DESC, R0                             ; 2136
            01 AE40        10     AC 90 0008F          MOVB    INTENSITY, BUFFER-7[R0]              ; 2137
          FFFFFFFF   8F    08     AC D1 00095 4$:      CMPL    X_POS, #-1                           ; 2144
                                  2A 13 0009D          BEQL    6$                                   ;
       F8    AD    0084   8F      6E A3 0009F          SUBW3   DESC, #132, TEMP_DESC                ; 2148
                     50           6E 3C 000A6          MOVZWL  DESC, R0                             ; 2150
                 FC  AD    04 BE40 9E 000A9            MOVAB   @DESC+4[R0], TEMP_DESC+4             ;
                           08     AC 9F 000AF          PUSHAB  X_POS                                ; 2151
                           F8     AD 9F 000B2          PUSHAB  TEMP_DESC                            ;
                           F8     AD 9F 000B5          PUSHAB  TEMP_DESC                            ;
                                  56 DD 000B8          PUSHL   R6                                   ;
                     68           04 FB 000BA          CALLS   #4, SYS$FAOL                         ;
                     05           50 E8 000BD          BLBS    RTN_STATUS, 5$                       ;
                                  50 DD 000C0          PUSHL   RTN_STATUS                           ;
                     67           01 FB 000C2          CALLS   #1, LIB$SIGNAL                       ;
                     6E     F8    AD A0 000C5 5$:      ADDW2   TEMP_DESC, DESC                      ; 2153
                     50           6E 3C 000C9 6$:      MOVZWL  DESC, R0                             ; 2160
            08 AE40         D0    A6 90 000CC          MOVB    COMMAND_MID, BUFFER[R0]              ;
                                  6E B6 000D2          INCW    DESC                                 ; 2161
          FFFFFFFF   8F    0C     AC D1 000D4          CMPL    Y_POS, #-1                           ; 2166
                                  2A 13 000DC          BEQL    8$                                   ;
       F8    AD    0084   8F      6E A3 000DE          SUBW3   DESC, #132, TEMP_DESC                ; 2170
                     50           6E 3C 000E5          MOVZWL  DESC, R0                             ; 2172
                 FC  AD    04 BE40 9E 000E8            MOVAB   @DESC+4[R0], TEMP_DESC+4             ;
                           0C     AC 9F 000EE          PUSHAB  Y_POS                                ; 2173
                           F8     AD 9F 000F1          PUSHAB  TEMP_DESC                            ;
                           F8     AD 9F 000F4          PUSHAB  TEMP_DESC                            ;
                                  56 DD 000F7          PUSHL   R6                                   ;
                     68           04 FB 000F9          CALLS   #4, SYS$FAOL                         ;
                     05           50 E8 000FC          BLBS    RTN_STATUS, 7$                       ;
                                  50 DD 000FF          PUSHL   RTN_STATUS                           ;
                     67           01 FB 00101          CALLS   #1, LIB$SIGNAL                       ;
                     6E     F8    AD A0 00104 7$:      ADDW2   TEMP_DESC, DESC                      ; 2175
                     50           6E 3C 00108 8$:      MOVZWL  DESC, R0                             ; 2181
            08 AE40         D4    A6 90 0010B          MOVB    COMMAND_END, BUFFER[R0]              ;
                                  6E B6 00111          INCW    DESC                                 ; 2182
                                  5E DD 00113          PUSHL   SP                                   ; 2187
          00000000G  00           01 FB 00115          CALLS   #1, LIB$PUT_OUTPUT                   ;
                     05           50 E8 0011C          BLBS    RTN_STATUS, 9$                       ;
                                  50 DD 0011F          PUSHL   RTN_STATUS                           ;
                     67           01 FB 00121          CALLS   #1, LIB$SIGNAL                       ;
                                     04 00124 9$:      RET                                          ; 2192

; Routine Size:  293 bytes,    Routine Base:  $CODE$ + 08C8

; 1924         2193  1
```

```
;  1926          2194  1 %SBTTL 'Put REGIS text to the terminal'
;  1927          2195  1 ROUTINE PUT_REGIS_TEXT ( DESC, INTENSITY ) : NOVALUE =
;  1928          2196  1
;  1929          2197  1 !++
;  1930          2198  1 !
;  1931          2199  1 ! FUNCTIONAL DESCRIPTION:
;  1932          2200  1 !
;  1933          2201  1 !       Build a REGIS text command and put it out.  This procedure assumes the
;  1934          2202  1 !       cursor is already at the desired position.
;  1935          2203  1 !
;  1936          2204  1 ! FORMAL PARAMETERS:
;  1937          2205  1 !
;  1938          2206  1 !       DESC      :  address of descriptor for test string to be put out
;  1939          2207  1 !       INTENSITY :  intensity at which text is to be written
;  1940          2208  1 !
;  1941          2209  1 ! IMPLICIT INPUTS:
;  1942          2210  1 !
;  1943          2211  1 !       None
;  1944          2212  1 !
;  1945          2213  1 ! IMPLICIT OUTPUTS:
;  1946          2214  1 !
;  1947          2215  1 !       None
;  1948          2216  1 !
;  1949          2217  1 ! ROUTINE VALUE:
;  1950          2218  1 !
;  1951          2219  1 !       None
;  1952          2220  1 !
;  1953          2221  1 ! COMPLETION CODES:
;  1954          2222  1 !
;  1955          2223  1 !       None
;  1956          2224  1 !
;  1957          2225  1 ! SIDE EFFECTS:
;  1958          2226  1 !
;  1959          2227  1 !       None
;  1960          2228  1 !
;  1961          2229  1 !--
;  1962          2230  1
;  1963          2231  2 BEGIN
;  1964          2232  2
;  1965          2233  2 MAP
;  1966          2234  2     DESC : REF BBLOCK                   !Descriptor for text to be output
;  1967          2235  2     ;
;  1968          2236  2
;  1969          2237  2 BIND
;  1970          2238  2     COMMAND_START = UPLIT( %STRING( 'T( W(i3))', %CHAR(39) )),
;  1971          2239  2     COMMAND_END   = UPLIT( %STRING( %CHAR(39)) )
;  1972          2240  2     ;
;  1973          2241  2
;  1974          2242  2 LITERAL
;  1975          2243  2     CMD_START_LEN = %CHARCOUNT( %STRING( 'T( W(i3))', %CHAR(39) )),
;  1976          2244  2     CMD_END_LEN   = %CHARCOUNT( %STRING (%CHAR(39)) ),
;  1977          2245  2     INTENSITY_LOC = 4
;  1978          2246  2     ;
;  1979          2247  2
;  1980          2248  2 LOCAL
;  1981          2249  2     CMD_BUFFER :                        !Buffer to build REGIS command
;  1982          2250  2         VECTOR[ MAX_PAGE_WIDTH, BYTE ],
```

```
;  1983          2251  2     CMD_DESC    :
;  1984          2252  2         BBLOCK[ DSC$K_Z_BLN ]
;  1985          2253  2     ;
;  1986          2254  2
```

.Descriptor for the buffer

```
;  1988          2255  2      !Initialize.
;  1989          2256  2      !
;  1990          2257  2      CMD_DESC[ DSC$B_CLASS ]   = DSC$K_CLASS_Z;
;  1991          2258  2      CMD_DESC[ DSC$B_DTYPE ]   = DSC$K_DTYPE_Z;
;  1992          2259  2      CMD_DESC[ DSC$A_POINTER ] = CMD_BUFFER;
;  1993          2260  2
;  1994          2261  2
;  1995          2262  2      !Build start of command string.
;  1996          2263  2      !
;  1997          2264  2      CH$MOVE( CMD_START_LEN, COMMAND_START, CMD_BUFFER );
;  1998          2265  2      CMD_DESC[ DSC$W_LENGTH ]  = CMD_START_LEN;
;  1999          2266  2      CMD_BUFFER[ .CMD_DESC[DSC$W_LENGTH] - INTENSITY_LOC ] = .INTENSITY;
;  2000          2267  2
;  2001          2268  2
;  2002          2269  2      !Copy text to be written into command buffer.
;  2003          2270  2      !
;  2004          2271  2      CH$MOVE( .DESC[ DSC$W_LENGTH ], .DESC[ DSC$A_POINTER ],
;  2005          2272  2               (CMD_BUFFER + .CMD_DESC[ DSC$W_LENGTH ] ));
;  2006          2273  2      CMD_DESC[ DSC$W_LENGTH ] =
;  2007          2274  2               .CMD_DESC[ DSC$W_LENGTH ] + .DESC[ DSC$W_LENGTH ];
;  2008          2275  2
;  2009          2276  2
;  2010          2277  2      !Put command terminator at end of string. WARNING!!! If more than one
;  2011          2278  2      !character is used as a terminator this won't work.  This only places one
;  2012          2279  2      !byte in the buffer.
;  2013          2280  2      !
;  2014          2281  2      CMD_BUFFER[ .CMD_DESC[ DSC$W_LENGTH ] ] = .COMMAND_END;
;  2015          2282  2      CMD_DESC[ DSC$W_LENGTH ] = .CMD_DESC[ DSC$W_LENGTH ] + CMD_END_LEN;
;  2016          2283  2
;  2017          2284  2
;  2018          2285  2      !Write the command string out.
;  2019          2286  2      !
;  2020          2287  2      PUT_REGIS( CMD_DESC );
;  2021          2288  2
;  2022          2289  2
;  2023          2290  2      RETURN;
;  2024          2291  1 END;                                  ! End of routine PUT_REGIS_TEXT


                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

          00  00  27  29  29  33  69  28  57  20  28  54  001D4 P.ABC:  .ASCII  \T( W(i3))'\<0><0>                          ;
                                          00  00  00  27  001E0 P.ABD:  .ASCII  \'\<0><0><0>                            ;

                                                              COMMAND_START=      P.ABC
                                                              COMMAND_END=        P.ABD


                                                              .PSECT  $CODE$,NOWRT,2

                                                  007C 00000 PUT_REGIS_TEXT:
                                                              .WORD   Save R2,R3,R4,R5,R6                                 ; 2195
                                  5E      FF74    CE   9E 00002         MOVAB   -140(SP), SP                                ;
                                            02    AE   B4 00007         CLRW    CMD_DESC+2                                  ; 2258
                                      04    AE      08    AE   9E 0000A         MOVAB   CMD_BUFFER, CMD_DESC+4              ; 2259
                  08    AE 00000000'    00            0A   28 0000F         MOVC3   #10, COMMAND_START, CMD_BUFFER              ; 2264
```

```
                                 6E              0A  B0 00018          MOVW    #10, CMD_DESC                                   ; 2265
                                 50              6E  3C 0001B          MOVZWL  CMD_DESC, R0                                    ; 2266
                           04 AE40         08    AC  90 0001E          MOVB    INTENSITY, CMD_BUFFER-4[R0]                     ;
                                 56        04    AC  D0 00024          MOVL    DESC, R6                                        ; 2271
                     08 AE40     04        B6    66  28 00028          MOVC3   (R6), @4(R6), CMD_BUFFER[R0]                    ; 2272
                                 6E              66  A0 0002F          ADDW2   (R6), CMD_DESC                                  ; 2274
                                 50              6E  3C 00032          MOVZWL  CMD_DESC, R0                                    ; 2281
                           08 AE40 00000000'     00  90 00035          MOVB    COMMAND_END, CMD_BUFFER[R0]                     ;
                                                 6E  B6 0003E          INCW    CMD_DESC                                        ; 2282
                                                 5E  DD 00040          PUSHL   SP                                              ; 2287
                     00000000G   00              01  FB 00042          CALLS   #1, LIB$PUT_OUTPUT                              ;
                                 09              50  E8 00049          BLBS    RTN_STATUS, 1$                                  ;
                                                 50  DD 0004C          PUSHL   RTN_STATUS                                      ;
                     00000000G   00              01  FB 0004E          CALLS   #1, LIB$SIGNAL                                  ;
                                                     04 00055 1$:      RET                                                     ; 2291

; Routine Size:  86 bytes,    Routine Base:  $CODE$ + 09ED
```

```
;  2026          2292  1 %SBTTL 'Shade a row of the graph'
;  2027          2293  1 ROUTINE SHADE_ROW_REGIS ( DIM1, SHADE_LINE, SHADE_ARRAY_DESC ) : NOVALUE =
;  2028          2294  1
;  2029          2295  1 !++
;  2030          2296  1 !
;  2031          2297  1 ! FUNCTIONAL DESCRIPTION:
;  2032          2298  1 !
;  2033          2299  1 !       Determine the length of the vector to be drawn, where the shading
;  2034          2300  1 !       line is to be located and the intesity of the shaded area.
;  2035          2301  1 !
;  2036          2302  1 ! FORMAL PARAMETERS:
;  2037          2303  1 !
;  2038          2304  1 !       DIM1            : index for the current row of the shade array
;  2039          2305  1 !       SHADE_LINE      : realtive location of the shade line
;  2040          2306  1 !       SHADE_ARRAY_DESC: descriptor for the array of shading information
;  2041          2307  1 !
;  2042          2308  1 ! IMPLICIT INPUTS:                                                                    :
;  2043          2309  1 !
;  2044          2310  1 !       None
;  2045          2311  1 !                                                                                     ;
;  2046          2312  1 ! IMPLICIT OUTPUTS:                                                                   ;
;  2047          2313  1 !                                                                                     ;
;  2048          2314  1 !       None                                                                          ;
;  2049          2315  1 !
;  2050          2316  1 ! ROUTINE VALUE:
;  2051          2317  1 !
;  2052          2318  1 !       None
;  2053          2319  1 !
;  2054          2320  1 ! COMPLETION CODES:                                                                   ;
;  2055          2321  1 !                                                                                     ;
;  2056          2322  1 !       None                                                                          ;
;  2057          2323  1 !                                                                                     ;
;  2058          2324  1 ! SIDE EFFECTS:                                                                       ;
;  2059          2325  1 !                                                                                     ;
;  2060          2326  1 !       None                                                                          ;
;  2061          2327  1 !                                                                                     ;
;  2062          2328  1 !--                                                                                   ;
;  2063          2329  1
;  2064          2330  2 BEGIN
;  2065          2331  2
;  2066          2332  2 MAP
;  2067          2333  2     SHADE_ARRAY_DESC : REF BBLOCK       !Descriptor for the shading array             ;
;  2068          2334  2     ;                                                                                 ;
;  2069          2335  2                                                                                     ;
;  2070          2336  2 LOCAL                                                                               ;
;  2071          2337  2     SHADE_ARRAY    :                    !Array of intesities for shading              ;
;  2072          2338  2         REF TWO_DIM_ARRAY[ LONG, UNSIGNED ],                                        ;
;  2073          2339  2     DIM2,                               !Index for second dimension of xy-array       ;
;  2074          2340  2     CURRENT_VALUE,                      !Current value in row, for comparisons        ;
;  2075          2341  2     REPEAT_COUNT,                       !Number of values equal to current value      ;
;  2076          2342  2     ROW_POS,                            !Current row on graph                         ;
;  2077          2343  2     COLUMN_POS,                         !Column postion for output                    ;
;  2078          2344  2     VECTOR_LEN                          !Location of x-origin when working with a VT125;
;  2079          2345  2     ;                                                                                 ;
;  2080          2346  2                                                                                     ;
;  2081          2347  2                                                                                     ;
```

```
;  2083          2348  2      !For each element in the row:
;  2084          2349  2      !
;  2085          2350  2      DIM2 = 0;
;  2086          2351  2      WHILE .DIM2 LEQ .SHADE_ARRAY_DESC[ DSC$L_U2 ]
;  2087          2352  2      DO
;  2088          2353  3          BEGIN
;  2089          2354  3
;  2090          2355  3          !Save the current xy-value for comparisons; determine the position
;  2091          2356  3          !in the row.
;  2092          2357  3          !
;  2093          2358  3          CURRENT_VALUE = .SHADE_ARRAY[ .DIM1, .DIM2, .SHADE_ARRAY_DESC ];
;  2094          2359  3          COLUMN_POS    = .DIM2;
;  2095          2360  3
;  2096          2361  3
;  2097          2362  3          !Compare the current value to all subsequent values in the row
;  2098          2363  3          !until a value which is not equal is encountered.  Keep a count
;  2099          2364  3          !of the equal values, so the right number can be printed in the
;  2100          2365  3          !graph.
;  2101          2366  3          !
;  2102          2367  3          DIM2 = .DIM2 + 1;
;  2103          2368  3          WHILE .DIM2 LEQ .SHADE_ARRAY_DESC[ DSC$L_U2 ]
;  2104          2369  3                AND
;  2105          2370  3              .CURRENT_VALUE EQL .SHADE_ARRAY[ .DIM1, .DIM2, .SHADE_ARRAY_DESC ]
;  2106          2371  3          DO
;  2107          2372  3              DIM2 = .DIM2 + 1;
;  2108          2373  3
;  2109          2374  3
;  2110          2375  3          !Calculate the number of equal values and draw a shade line of the
;  2111          2376  3          !appropriate length (number-of-equal-values * unit-width).
;  2112          2377  3          !
;  2113          2378  3          REPEAT_COUNT = .DIM2 - .COLUMN_POS;
;  2114          2379  3          VECTOR_LEN   = .REPEAT_COUNT * UNIT_WIDTH;
;  2115          2380  3          MOVE_CURSOR_REGIS( GRF$_REGIS_SHADE, .VECTOR_LEN, NO_VALUE,
;  2116          2381  3                             .CURRENT_VALUE + '0', .SHADE_LINE );
;  2117          2382  3
;  2118          2383  2      END;                          !DIM2
;  2119          2384  1 END;                                    ! End of routine SHADE_ROW_REGIS



                                    00FC 00000 SHADE_ROW_REGIS:
                                                       .WORD   Save R2,R3,R4,R5,R6,R7                          ; 2293
                                          53  D4 00002         CLRL    DIM2                                    ; 2350
                               52         0C  AC  D0 00004         MOVL    SHADE_ARRAY_DESC, R2                    ; 2351
                          28   A2             53  D1 00008 1$:     CMPL    DIM2, 40(R2)                            ;
                                              4B  14 0000C         BGTR    4$                                      ;
                               51             52  D0 0000E         MOVL    R2, DESC_ADR_LCL                        ; 2358
                     50   04   AC        18   A1  C5 00011         MULL3   24(DESC_ADR_LCL), DIM1, R0              ;
                               50             53  C0 00017         ADDL2   DIM2, R0                                ;
                               54        04 B140  D0 0001A         MOVL    @4(DESC_ADR_LCL)[R0], CURRENT_VALUE     ;
                               56             53  D0 0001F         MOVL    DIM2, COLUMN_POS                        ; 2359
                                              53  D6 00022 2$:     INCL    DIM2                                    ; 2367
                          28   A2             53  D1 00024         CMPL    DIM2, 40(R2)                            ; 2368
                                              13  14 00028         BGTR    3$                                      ;
                               50             52  D0 0002A         MOVL    R2, DESC_ADR_LCL                        ; 2370
```

```
                51      04    AC        18    A0  C5 0002D           MULL3   24(DESC_ADR_LCL), DIM1, R1                      ;
                        51                    53  C0 00033           ADDL2   DIM2, R1                                        ;
                      04 B041                 54  D1 00036           CMPL    CURRENT_VALUE, @4(DESC_ADR_LCL)[R1]             ;
                                              E5  13 0003B           BEQL    2$                                              ;
                57              53            56  C3 0003D 3$:       SUBL3   COLUMN_POS, DIM2, REPEAT_COUNT                  ; 2378
                55              57            12  C5 00041           MULL3   #18, REPEAT_COUNT, VECTOR_LEN                   ; 2379
                                          08  AC  DD 00045           PUSHL   SHADE_LINE                                      ; 2381
                                          30  A4  9F 00048           PUSHAB  48(CURRENT_VALUE)                               ;
                              7E                01  CE C0048           MNEGL   #1, -(SP)                                       ; 2380
                                                55  DD 0004E           PUSHL   VECTOR_LEN                                      ;
                                                02  DD 00050           PUSHL   #2                                              ;
                          FE2E  CF              05  FB 00052           CALLS   #5, MOVE_CURSOR_REGIS                           ;
                                                AF  11 00057           BRB     1$                                              ; 2351
                                                    04 00059 4$:       RET                                                     ; 2384

; Routine Size:  90 bytes,    Routine Base:  $CODE$ + 0A43
```

```
;  2121          2385  1 %SBTTL 'Draw a portion of the bar graph'
;  2122          2386  1 ROUTINE DRAW_BARS_REGIS ( Y_VALUE, X_VALUE, REPEAT_COUNT ) : NOVALUE =
;  2123          2387  1
;  2124          2388  1 !++
;  2125          2389  1 !
;  2126          2390  1 ! FUNCTIONAL DESCRIPTION:
;  2127          2391  1 !
;  2128          2392  1 !       Draw the bars for the current row segment of the REGIS line graph.
;  2129          2393  1 !
;  2130          2394  1 ! FORMAL PARAMETERS:
;  2131          2395  1 !
;  2132          2396  1 !       Y_VALUE      :  The row in which contains the top of the bar.
;  2133          2397  1 !       X_VALUE      :  The first column of this row segment
;  2134          2398  1 !       REPEAT_COUNT :  The number of bars to be drawn
;  2135          2399  1 !
;  2136          2400  1 ! IMPLICIT INPUTS:
;  2137          2401  1 !
;  2138          2402  1 !       None
;  2139          2403  1 !
;  2140          2404  1 ! IMPLICIT OUTPUTS:
;  2141          2405  1 !
;  2142          2406  1 !       None
;  2143          2407  1 !
;  2144          2408  1 ! ROUTINE VALUE:
;  2145          2409  1 !
;  2146          2410  1 !       None
;  2147          2411  1 !
;  2148          2412  1 ! COMPLETION CODES:
;  2149          2413  1 !
;  2150          2414  1 !       None
;  2151          2415  1 !
;  2152          2416  1 ! SIDE EFFECTS:
;  2153          2417  1 !
;  2154          2418  1 !       None
;  2155          2419  1 !
;  2156          2420  1 !--
;  2157          2421  1
;  2158          2422  2 BEGIN
;  2159          2423  2
;  2160          2424  2 LITERAL
;  2161          2425  2     BAR_GAP = 2                !Number of ixels between bars in hte graph
;  2162          2426  2     ;
;  2163          2427  2
;  2164          2428  2 LOCAL
;  2165          2429  2     ROW_POS,                   !Y value for a pixel address
;  2166          2430  2     COLUMN_POS,                !X value for a pixel address
;  2167          2431  2     SHADE_LINE                 !Line to which bar is shaded
;  2168          2432  2     ;
```

```
2       !Initialize values.
2       !
2       ROW_POS    = ( .Y_VALUE - 1 ) * UNIT_HEIGHT;
2       COLUMN_POS = ( .X_VALUE * UNIT_WIDTH ) + X_ORIGIN + 1 ;
2       SHADE_LINE = LINE_GRAPH_LEN * UNIT_HEIGHT;
2
2
2       !Move the cursor to the starting postion.
2       !
2       MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS, .ROW_POS,
2                          NO_VALUE, NO_VALUE );
2
2
2       !For each bar in the line graph:
2       !
2       DECR LOOP_COUNT FROM (.REPEAT_COUNT - 1) TO 0
2       DO
3           BEGIN
3
3           !Draw a vector; shade it to the bottom of the graph.
3           !
3           MOVE_CURSOR_REGIS( GRF$_REGIS_SHADE, (UNIT_WIDTH - BAR_GAP), NO_VALUE,
3                              DARK_INTENSITY, .SHADE_LINE );
3
3
3           !Move the cursor over, to create spaces between the bars.
3           !
3           COLUMN_POS = .COLUMN_POS + UNIT_WIDTH;
3           MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS, NO_VALUE,
3                              NO_VALUE, NO_VALUE );
2           END;
2
1 END;                                          ! End of routine DRAW_BARS_REGIS
```

```
                                003C 00000 DRAW_BARS_REGIS:
                                                   .WORD   Save R2,R3,R4,R5                        ; 2386
                55      FE25    CF    9E 00002     MOVAB   MOVE_CURSOR_REGIS, R5                   ;
  50       04   AC              01    C3 00007     SUBL3   #1, Y_VALUE, R0                         ; 2435
                50              12    C4 0000C     MULL2   #18, ROW_POS                            ;
  52       08   AC              12    C5 0000F     MULL3   #18, X_VALUE, R2                        ; 2436
                52      00C0    C2    9E 00014     MOVAB   192(R2), COLUMN_POS                     ;
                54        B4    8F    9A 00019     MOVZBL  #180, SHADE_LINE                        ; 2437
                7E              01    CE 0001D     MNEGL   #1, -(SP)                               ; 2442
                7E              01    CE 00020     MNEGL   #1, -(SP)                               ;
                                50    DD 00023     PUSHL   ROW_POS                                 ;
                                52    DD 00025     PUSHL   COLUMN_POS                              ;
                                7E    D4 00027     CLRL    -(SP)                                   ;
                65              05    FB 00029     CALLS   #5, MOVE_CURSOR_REGIS                   ;
                53        0C    AC    D0 0002C     MOVL    REPEAT_COUNT, LOOP_COUNT                ; 2455
                                21    11 00030     BRB     2$                                      ;
                                54    DD 00032 1$: PUSHL   SHADE_LINE                              ;
                                31    DD 00034     PUSHL   #49                                     ; 2454
```

```
           7E        01  CE 00036         MNEGL   #1, -(SP)                    ;
                     10  DD 00039         PUSHL   #16                          ;
                     02  DD 0003B         PUSHL   #2                           ;
           65        05  FB 0003D         CALLS   #5, MOVE_CURSOR_REGIS        ;
           52        12  C0 00040         ADDL2   #18, COLUMN_POS              ; 2460
           7E        01  CE 00043         MNEGL   #1, -(SP)                    ; 2461
           7E        01  CE 00046         MNEGL   #1, -(SP)                    ;
           7E        01  CE 00049         MNEGL   #1, -(SP)                    ;
                     52  DD 0004C         PUSHL   COLUMN_POS                   ;
                     7E  D4 0004E         CLRL    -(SP)                        ;
           65        05  FB 00050         CALLS   #5, MOVE_CURSOR_REGIS        ;
           DC        53  F4 00053 2$:     SOBGEQ  LOOP_COUNT, 1$               ; 2448
                         04 00056         RET                                  ; 2466

; Routine Size:  87 bytes,    Routine Base:  $CODE$ + 0A9D

; 2204          2467  1
```

```
;  2206          2468  1 %SBTTL 'Format the x-axis headers'
;  2207          2469  1 ROUTINE DRAW_X_AXIS ( GRAPH_LEN ) : NOVALUE =
;  2208          2470  1
;  2209          2471  1 !++
;  2210          2472  1 !
;  2211          2473  1 ! FUNCTIONAL DESCRIPTION:
;  2212          2474  1 !
;  2213          2475  1 !       Draw the x-axis line and then write the x-axis numeric and
;  2214          2476  1 !       descriptive labels.  These are the same for all graphs.
;  2215          2477  1 !
;  2216          2478  1 ! FORMAL PARAMETERS:
;  2217          2479  1 !
;  2218          2480  1 !       GRAPH_LEN       :  the number of rows in a the graph
;  2219          2481  1 !
;  2220          2482  1 ! IMPLICIT INPUTS:
;  2221          2483  1 !
;  2222          2484  1 !       Y_AXIS_LINE  :  column which contains the y-axis line.
;  2223          2485  1 !       DEVICE_FLAGS :  a longword of flags which are set or cleared by
;  2224          2486  1 !                       LIB$SCREEN_INFO depending on the terminal
;  2225          2487  1 !                       characteristics.
;  2226          2488  1 !
;  2227          2489  1 ! IMPLICIT OUTPUTS:
;  2228          2490  1 !
;  2229          2491  1 !       None
;  2230          2492  1 !
;  2231          2493  1 ! ROUTINE VALUE:
;  2232          2494  1 !
;  2233          2495  1 !       None
;  2234          2496  1 !
;  2235          2497  1 ! COMPLETION CODES:
;  2236          2498  1 !
;  2237          2499  1 !       None
;  2238          2500  1 !
;  2239          2501  1 ! SIDE EFFECTS:
;  2240          2502  1 !
;  2241          2503  1 !       None
;  2242          2504  1 !
;  2243          2505  1 !--
;  2244          2506  1
;  2245          2507  2 BEGIN
;  2246          2508  2
;  2247          2509  2 LITERAL
;  2248          2510  2     INCR_VAL    = 5,            !The increment between x_value labels
;  2249          2511  2     TWO_VALUES = 2              !For formatting last x-value label
;  2250          2512  2     ;
;  2251          2513  2
;  2252          2514  2 LOCAL
;  2253          2515  2     RTN_STATUS,                 !Status returned from external calls
;  2254          2516  2     ROW_POS,                    !Row postion for output
;  2255          2517  2     COLUMN_POS,                 !Column postion for output
;  2256          2518  2     ROW_INCR,                   !Distance to next row
;  2257          2519  2     COLUMN_INCR,                !Distance to next column
;  2258          2520  2     Y_ORIGIN,                   !Left most column in graph
;  2259          2521  2     CHARACTER_WIDTH,            !Width of character, for calculations
;  2260          2522  2     PLOT_WIDTH,                 !The width of the current graph
;  2261          2523  2     BUFFER              :       !Buffer for building the output
;  2262          2524  2         VECTOR[ MAX_PAGE_WIDTH, BYTE ],
```

```
;  2263     2525  2     DESC                    :       !Descriptor for BUFFER
;  2264     2526  2         BBLOCK[ DSC$K_Z_BLN ]
;  2265     2527  2     ;
;  2266     2528  2
;  2267     2529  2 BIND
;  2268     2530  2                                     !Header for the x-axis
;  2269     2531  2     X_AXIS_HEADER  = %ASCID'Bucket Size (number of blocks)',
;  2270     2532  2                                     !Numeric label for the x-axis
;  2271     2533  2     X_AXIS_LABEL_N = PLIT ( %ASCID' 1',
;  2272     2534  2                             %ASCID' 5',
;  2273     2535  2                             %ASCID'10',
;  2274     2536  2                             %ASCID'15',
;  2275     2537  2                             %ASCID'20',
;  2276     2538  2                             %ASCID'25',
;  2277     2539  2                             %ASCID'30',
;  2278     2540  2                             %ASCID'32'
;  2279     2541  2                           ) : VECTOR,
;  2280     2542  2                                     !Formatting for the x-axis line
;  2281     2543  2     AXIS_LINE      = PLIT ( %ASCID'+- + - - - + - - - - + - - - - + - - - - + - - - - + - - - - + - - - - + - +'
;  2282     2544  2                           ) : VECTOR;
```

```
;  2284          2545  2      !Initialize values for distance between rows and columns.  Draw x-axis
;  2285          2546  2      !line.
;  2286          2547  2      !
;  2287          2548  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  2288          2549  2      THEN
;  2289          2550  2          BEGIN
;  2290          2551  2
;  2291          2552  2
;  2292          2553  2          !Initialize.
;  2293          2554  2          !
;  2294          2555  2          ROW_INCR    = 3;
;  2295          2556  2          COLUMN_INCR = 0;
;  2296          2557  2          Y_ORIGIN        = .Y_AXIS_LINE + AXIS_SHIFT;
;  2297          2558  2          CHARACTER_WIDTH = CHAR_WIDTH;
;  2298          2559  2
;  2299          2560  2
;  2300          2561  2          !Position x-axis line. Draw it.
;  2301          2562  2          !
;  2302          2563  2          ROW_POS    = UNIT_HEIGHT * ( .GRAPH_LEN ) + AXIS_SHIFT;
;  2303          2564  2          COLUMN_POS = .Y_AXIS_LINE;
;  2304          2565  2
;  2305          2566  2
;  2306          2567  2          MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS, .ROW_POS );
;  2307          2568  2          COLUMN_POS = .Y_ORIGIN + ( MAX_BUCKET_SIZE * UNIT_WIDTH );
;  2308          2569  2          MOVE_CURSOR_REGIS( GRF$_REGIS_VCTR, .COLUMN_POS, NO_VALUE );
;  2309          2570  2          END
;  2310          2571  2
;  2311          2572  2      ELSE
;  2312          2573  2          BEGIN
;  2313          2574  2
;  2314          2575  2          !Initialize.
;  2315          2576  2          !
;  2316          2577  2          ROW_INCR    = 1;
;  2317          2578  2          COLUMN_INCR = SEPARATOR_WIDTH;
;  2318          2579  2          Y_ORIGIN        = .Y_AXIS_LINE;
;  2319          2580  2          CHARACTER_WIDTH = 1;
;  2320          2581  2
;  2321          2582  2
;  2322          2583  2          !Position x-axis line. Draw it.
;  2323          2584  2          !
;  2324          2585  2          ROW_POS    = .GRAPH_LEN + 1;
;  2325          2586  2          COLUMN_POS = .Y_ORIGIN;
;  2326          2587  2          PUT_TEXT( .AXIS_LINE, ROW_POS, COLUMN_POS );
;  2327          2588  2          END;
;  2328          2589  2
;  2329          2590  2
;  2330          2591  2      !Move cursor to the beginning of the next row, put out first numeric label.
;  2331          2592  2      !
;  2332          2593  2      ROW_POS         = .ROW_POS + .ROW_INCR;
;  2333          2594  2      COLUMN_POS      = .Y_ORIGIN + .COLUMN_INCR;
;  2334          2595  2
;  2335          2596  2
;  2336          2597  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;  2337          2598  2      THEN
;  2338          2599  3          BEGIN
;  2339          2600  3          MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS, .ROW_POS );
;  2340          2601  3          PUT_REGIS_TEXT( .X_AXIS_LABEL_N[ 0 ], LIGHT_INTENSITY );
```

```
;2341     2602  3          ROW_INCR    =  CHAR_HEIGHT + SEPARATOR_WIDTH;
;2342     2603  3          COLUMN_INCR =  UNIT_WIDTH;
;2343     2604  3          END
;2344     2605  3
;2345     2606  2      ELSE
;2346     2607  2          PUT_TEXT ( .X_AXIS_LABEL_N[ 0 ], ROW_POS, COLUMN_POS );
;2347     2608  2
;2348     2609  2
;2349     2610  2      !For each subsequent numeric label: positon cursor and write the label.
;2350     2611  2      !
;2351     2612  2      INCR DESC_PTR FROM 1  TO (.(X_AXIS_LABEL_N - FULLWORD) - TWO_VALUES )
;2352     2613  2      DO
;2353     2614  2
;2354     2615  3          BEGIN
;2355     2616  3          COLUMN_POS = .Y_ORIGIN + ( .COLUMN_INCR * INCR_VAL * .DESC_PTR );
;2356     2617  3          IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;2357     2618  3          THEN
;2358     2619  4              BEGIN
;2359     2620  4              COLUMN_POS = .COLUMN_POS - UNIT_WIDTH;
;2360     2621  4              MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS, NO_VALUE);
;2361     2622  4              PUT_REGIS_TEXT( .X_AXIS_LABEL_N[ .DESC_PTR ], LIGHT_INTENSITY );
;2362     2623  4              END
;2363     2624  4
;2364     2625  3          ELSE
;2365     2626  3              PUT_TEXT( .X_AXIS_LABEL_N[ .DESC_PTR ], ROW_POS, COLUMN_POS );
;2366     2627  2          END;
;2367     2628  2
;2368     2629  2
;2369     2630  2      !Position and write the last numeric label.
;2370     2631  2      !
;2371     2632  2      COLUMN_POS = .COLUMN_POS + .COLUMN_INCR * TWO_VALUES;
;2372     2633  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;2373     2634  2      THEN
;2374     2635  3          BEGIN
;2375     2636  3          MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS, NO_VALUE );
;2376     2637  3          PUT_REGIS_TEXT( .X_AXIS_LABEL_N[ .(X_AXIS_LABEL_N - FULLWORD) - 1 ],
;2377     2638  3                          LIGHT_INTENSITY );
;2378     2639  3          END
;2379     2640  3
;2380     2641  2      ELSE
;2381   P 2642  2          PUT_TEXT( .X_AXIS_LABEL_N[ .( X_AXIS_LABEL_N - FULLWORD ) - 1 ],
;2382     2643  2                    ROW_POS, COLUMN_POS);
;2383     2644  2
;2384     2645  2
;2385     2646  2      !Put the descriptive labels out.
;2386     2647  2      !
;2387     2648  2      ROW_POS = .ROW_POS + .ROW_INCR;
;2388     2649  2      PLOT_WIDTH  = MAX_BUCKET_SIZE * .COLUMN_INCR;
;2389     2650  2      COLUMN_POS = (.PLOT_WIDTH - (.X_AXIS_HEADER<0,8> * .CHARACTER_WIDTH) )/2
;2390     2651  2                 + .Y_ORIGIN;
;2391     2652  2
;2392     2653  2
;2393     2654  2      IF .DEVICE_FLAGS[ SCR$V_REGIS ]
;2394     2655  2      THEN
;2395     2656  3          BEGIN
;2396     2657  3          MOVE_CURSOR_REGIS( GRF$_REGIS_POS, .COLUMN_POS, .ROW_POS );
;2397     2658  3          PUT_REGIS_TEXT( X_AXIS_HEADER, LIGHT_INTENSITY );
```

```
;  2398          2659  3          END
;  2399          2660  3
;  2400          2661  2      ELSE
;  2401          2662  2          PUT_TEXT( X_AXIS_HEADER, ROW_POS, COLUMN_POS );
;  2402          2663  2
;  2403          2664  1 END;                                    ! End of routine DRAW_X_AXIS


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

75  6E  28  20  65  7A  69  53  20  74  65  6B  63  75  42  001E4 P.ABF:  .ASCII  \Bucket Size (number of blocks)\<0><0>
29  73  6B  63  6F  6C  62  20  66  6F  20  72  65  62  6D  001F3
                                                    00  00  00202
                                                  010E001E  00204 P.ABE:  .LONG   17694750
                                                  00000000'  00208        .ADDRESS P.ABF
                                              00  00  31  20  0020C P.ABI:  .ASCII  \ 1\<0><0>
                                                  010E0002  00210 P.ABH:  .LONG   17694722
                                                  00000000'  00214        .ADDRESS P.ABI
                                              00  00  35  20  00218 P.ABK:  .ASCII  \ 5\<0><0>
                                                  010E0002  0021C P.ABJ:  .LONG   17694722
                                                  00000000'  00220        .ADDRESS P.ABK
                                              00  00  30  31  00224 P.ABM:  .ASCII  \10\<0><0>
                                                  010E0002  00228 P.ABL:  .LONG   17694722
                                                  00000000'  0022C        .ADDRESS P.ABM
                                              00  00  35  31  00230 P.ABO:  .ASCII  \15\<0><0>
                                                  010E0002  00234 P.ABN:  .LONG   17694722
                                                  00000000'  00238        .ADDRESS P.ABO
                                              00  00  30  32  0023C P.ABQ:  .ASCII  \20\<0><0>
                                                  010E0002  00240 P.ABP:  .LONG   17694722
                                                  00000000'  00244        .ADDRESS P.ABQ
                                              00  00  35  32  00248 P.ABS:  .ASCII  \25\<0><0>
                                                  010E0002  0024C P.ABR:  .LONG   17694722
                                                  00000000'  00250        .ADDRESS P.ABS
                                              00  00  30  33  00254 P.ABU:  .ASCII  \30\<0><0>
                                                  010E0002  00258 P.ABT:  .LONG   17694722
                                                  00000000'  0025C        .ADDRESS P.ABU
                                              00  00  32  33  00260 P.ABW:  .ASCII  \32\<0><0>
                                                  010E0002  00264 P.ABV:  .LONG   17694722
                                                  00000000'  00268        .ADDRESS P.ABW
                                                  00000008  0026C         .LONG   8
00000000' 00000000' 00000000' 00000000' 0000000C' 00000000'  00270 P.ABG:  .ADDRESS P.ABH, P.ABJ, P.ABL, P.ABN, P.ABP, -
                                                  00000000'  00288              P.ABR, P.ABT, P.ABV
20  2D  20  2B  20  2D  20  2D  20  2D  20  2B  20  2D  2B  00290 P.ABZ:  .ASCII  \+- + - - - + - - - - - + - - - - - + - - - -\
2D  20  2D  20  2D  20  2D  20  2B  20  2D  20  2D  20  2D  0029F
                    2D  20  2D  20  2D  20  2D  20  2B  20  002AE
20  2D  20  2B  20  2D  20  2D  20  2D  20  2D  20  2D  20  002B8        .ASCII  \ + - - - - - + - - - - - + - - +\<0><0>
        00  00  2B  20  2D  20  2B  20  2D  20  2D  20  2D  002C7
                                                  010E0042  002D4 P.ABY:  .LONG   17694786
                                                  00000000'  002D8        .ADDRESS P.ABZ
                                                  00000001  002DC         LONG   1
                                                  00000000'  002E0 P.ABX:  .ADDRESS P.ABY

                                                  X_AXIS_HEADER=      P.ABE
                                                  X_AXIS_LABEL_N=     P.ABG
                                                  AXIS_LINE=          P.ABX
```

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

                                        0FFC 00000 DRAW_X_AXIS:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11            ; 2469
                                5B 00000000'    00  9E 00002            MOVAB   DEVICE_FLAGS, R11       ;
                                5A        FDC7  CF  9E 00009            MOVAB   MOVE_CURSOR_REGIS, R10  ;
                                59 00000000'    00  9E 0000E            MOVAB   X_AXIS_LABEL_N-4, R9    ;
                                5E        FF6C  CE  9E 00015            MOVAB   -148(SP), SP            ;
                                50          04  AB  D0 0001A            MOVL    Y_AXIS_LINE, R0         ; 2557
                          36                6B    02  E1 0001E            BBC     #2, DEVICE_FLAGS, 1$    ; 2548
                                57                  03  D0 00022            MOVL    #3, ROW_INCR            ; 2555
                                                    52  D4 00025            CLRL    COLUMN_INCR             ; 2556
                                53          05  A0  9E 00027            MOVAB   5(R0), Y_ORIGIN         ; 2557
                                58                  09  D0 0002B            MOVL    #9, CHARACTER_WIDTH     ; 2558
                          51    04          AC        12  C5 0002E            MULL3   #18, GRAPH_LEN, R1      ; 2563
                                04          AE  A1  9E 00033            MOVAB   5(R1), ROW_POS          ;
                                6E                  50  D0 00038            MOVL    R0, COLUMN_POS          ; 2564
                                                04  AE  DD 0003B            PUSHL   ROW_POS                 ; 2567
                                                04  AE  DD 0003E            PUSHL   COLUMN_POS              ;
                                                    7E  D4 00041            CLRL    -(SP)                   ;
                                6A                  03  FB 00043            CALLS   #3, MOVE_CURSOR_REGIS   ;
                                6E        0240      C3  9E 00046            MOVAB   576(R3), COLUMN_POS     ; 2568
                                7E                  01  CE 0004B            MNEGL   #1, -(SP)               ; 2569
                                                04  AE  DD 0004E            PUSHL   COLUMN_POS              ;
                                                    01  DD 00051            PUSHL   #1                      ;
                                6A                  03  FB 00053            CALLS   #3, MOVE_CURSOR_REGIS   ;
                                                    30  11 00056            BRB     2$                      ; 2548
                                57                  01  D0 00058 1$:        MOVL    #1, ROW_INCR            ; 2577
                                52                  02  D0 0005B            MOVL    #2, COLUMN_INCR         ; 2578
                                53                  50  D0 0005E            MOVL    R0, Y_ORIGIN            ; 2579
                                58                  01  D0 00061            MOVL    #1, CHARACTER_WIDTH     ; 2580
                    04  AE          04      AC        01  C1 00064            ADDL3   #1, GRAPH_LEN, ROW_POS  ; 2585
                                6E                  53  D0 0006A            MOVL    Y_ORIGIN, COLUMN_POS    ; 2586
                                                    5E  DD 0006D            PUSHL   SP                      ; 2587
                                                08  AE  9F 0006F            PUSHAB  ROW_POS                 ;
                                                74  A9  DD 00072            PUSHL   AXIS_LINE               ;
                              00000000G         00        03  FB 00075            CALLS   #3, LIB$PUT_SCREEN      ;
                                                09        50  E8 0007C            BLBS    RTN_STATUS, 2$          ;
                                                          50  DD 0007F            PUSHL   RTN_STATUS              ;
                              00000000G         00        01  FB 00081            CALLS   #1, LIB$SIGNAL          ;
                                    04          AE        57  C0 00088 2$:        ADDL2   ROW_INCR, ROW_POS       ; 2593
                                6E              53        52  C1 0008C            ADDL3   COLUMN_INCR, Y_ORIGIN, COLUMN_POS ; 2594
                                1D              6B        02  E1 00090            BBC     #2, DEVICE_FLAGS, 3$    ; 2597
                                                04  AE  DD 00094            PUSHL   ROW_POS                 ; 2600
                                                04  AE  DD 00097            PUSHL   COLUMN_POS              ;
                                                    7E  D4 0009A            CLRL    -(SP)                   ;
                                6A                  03  FB 0009C            CALLS   #3, MOVE_CURSOR_REGIS   ;
                                                    33  DD 0009F            PUSHL   #51                     ; 2601
                                                04  A9  DD 000A1            PUSHL   X_AXIS_LABEL_N          ;
                                  0125      CA        02  FB 000A4            CALLS   #2, PUT_REGIS_TEXT      ;
                                57                  11  D0 000A9            MOVL    #17, ROW_INCR           ; 2602
                                52                  12  D0 000AC            MOVL    #18, COLUMN_INCR        ; 2603
                                                    1B  11 000AF            BRB     4$                      ; 2597
                                                    5E  DD 000B1 3$:        PUSHL   SP                      ; 2607
                                                08  AE  9F 000B3            PUSHAB  ROW_POS                 ;
                                                04  A9  DD 000B6            PUSHL   X_AXIS_LABEL_N          ;
```

```
                    00000000G  00        03  FB 000B9          CALLS   #3, LIB$PUT_SCREEN                       ;
                               09        50  E8 000C0          BLBS    RTN_STATUS, 4$                          ;
                                         50  DD 000C3          PUSHL   RTN_STATUS                              ;
                    00000000G  00        01  FB 000C5          CALLS   #1, LIB$SIGNAL                          ;
                         56    69        02  C3 000CC 4$:      SUBL3   #2, X_AXIS_LABEL_N-4, R6                ; 2612
                                         54  D4 000D0          CLRL    DESC_PTR                                ; 2616
                                         47  11 000D2          BRB     7$                                      ;
                         50    52        54  C5 000D4 5$:      MULL3   DESC_PTR, COLUMN_INCR, R0               ;
                               50        05  C4 000D8          MULL2   #5, R0                                  ;
                         6E    50        53  C1 000DB          ADDL3   Y_ORIGIN, R0, COLUMN_POS                ;
                               55  04 A944  DE 000DF          MOVAL   X_AXIS_LABEL_N[DESC_PTR], R5            ; 2622
                         19    6B        02  E1 000E4          BBC     #2, DEVICE_FLAGS, 6$                    ; 2617
                               6E        12  C2 000E8          SUBL2   #18, COLUMN_POS                         ; 2620
                               7E        01  CE 000EB          MNEGL   #1, -(SP)                               ; 2621
                                   04    AE  DD 000EE          PUSHL   COLUMN_POS                              ;
                                         7E  D4 000F1          CLRL    -(SP)                                   ;
                               6A        03  FB 000F3          CALLS   #3, MOVE_CURSOR_REGIS                   ;
                                         33  DD 000F6          PUSHL   #51                                     ; 2622
                                         65  DD 000F8          PUSHL   (R5)                                    ;
                             0125  CA    02  FB 000FA          CALLS   #2, PUT_REGIS_TEXT                      ;
                                         1A  11 000FF          BRB     7$                                      ; 2617
                                         5E  DD 00101 6$:      PUSHL   SP                                      ; 2626
                                   08    AE  9F 00103          PUSHAB  ROW_POS                                 ;
                                         65  DD 00106          PUSHL   (R5)                                    ;
                    00000000G  00        03  FB 00108          CALLS   #3, LIB$PUT_SCREEN                      ;
                               09        50  E8 0010F          BLBS    RTN_STATUS, 7$                          ;
                                         50  DD 00112          PUSHL   RTN_STATUS                              ;
                    00000000G  00        01  FB 00114          CALLS   #1, LIB$SIGNAL                          ;
                         B5    54        56  F3 0011B 7$:      AOBLEQ  R6, DESC_PTR, 5$                        ; 2612
                                     9E42  3F 0011F          PUSHAW  @COLUMN_POS[COLUMN_INCR]                ; 2632
                         1A    6B        02  E1 00122          BBC     #2, DEVICE_FLAGS, 8$                    ; 2633
                               7E        01  CE 00126          MNEGL   #1, -(SP)                               ; 2636
                                   04    AE  DD 00129          PUSHL   COLUMN_POS                              ;
                                         7E  D4 0012C          CLRL    -(SP)                                   ;
                               6A        03  FB 0012E          CALLS   #3, MOVE_CURSOR_REGIS                   ;
                                         33  DD 00131          PUSHL   #51                                     ; 2637
                               50        69  D0 00133          MOVL    X_AXIS_LABEL_N-4, R0                    ;
                                       6940  DD 00136          PUSHL   X_AXIS_LABEL_N-4[R0]                    ;
                             0125  CA    02  FB 00139          CALLS   #2, PUT_REGIS_TEXT                      ;
                                         1E  11 0013E          BRB     9$                                      ; 2633
                                         5E  DD 00140 8$:      PUSHL   SP                                      ; 2643
                                   08    AE  9F 00142          PUSHAB  ROW_POS                                 ;
                               50        69  D0 00145          MOVL    X_AXIS_LABEL_N-4, R0                    ;
                                       6940  DD 00148          PUSHL   X_AXIS_LABEL_N-4[R0]                    ;
                    00000000G  00        03  FB 0014B          CALLS   #3, LIB$PUT_SCREEN                      ;
                               09        50  E8 00152          BLBS    RTN_STATUS, 9$                          ;
                                         50  DD 00155          PUSHL   RTN_STATUS                              ;
                    00000000G  00        01  FB 00157          CALLS   #1, LIB$SIGNAL                          ;
                         04    AE        57  C0 0015E 9$:      ADDL2   ROW_INCR, ROW_POS                       ; 2648
                         50    52        05  78 00162          ASHL    #5, COLUMN_INCR, PLOT_WIDTH             ; 2649
                               51  98    A9  9A 00166          MOVZBL  X_AXIS_HEADER, R1                       ; 2650
                               51        58  C4 0016A          MULL2   CHARACTER_WIDTH, R1                     ;
                               50        51  C2 0016D          SUBL2   R1, R0                                  ;
                               50        02  C6 00170          DIVL2   #2, R0                                  ;
                         6E    50        53  C1 00173          ADDL3   Y_ORIGIN, R0, COLUMN_POS                ; 2651
                         16    6B        02  E1 00177          BBC     #2, DEVICE_FLAGS, 10$                   ; 2654
                                   04    AE  DD 0017B          PUSHL   ROW_POS                                 ; 2657
```

```
                          04   AE  DD 0017E          PUSHL   COLUMN_POS                    ;
                               7E  D4 00181          CLRL    -(SP)                         ;
               6A              03  FB 00183          CALLS   #3, MOVE_CURSOR_REGIS         ;
                               33  DD 00186          PUSHL   #51                           ; 2658
                          98   A9  9F 00188          PUSHAB  X_AXIS_HEADER                 ;
          0125 CA              02  FB 0018B          CALLS   #2, PUT_REGIS_TEXT            ;
                                   04 00190          RET                                   ; 2654
                               5E  DD 00191 10$:     PUSHL   SP                            ; 2662
                          08   AE  9F 00193          PUSHAB  ROW_POS                       ;
                          98   A9  9F 00196          PUSHAB  X_AXIS_HEADER                 ;
     00000000G 00              03  FB 00199          CALLS   #3, LIB$PUT_SCREEN            ;
               09              50  E8 001A0          BLBS    RTN_STATUS, 11$               ;
                               50  DD 001A3          PUSHL   RTN_STATUS                    ;
     00000000G 00              01  FB 001A5          CALLS   #1, LIB$SIGNAL                ;
                                   04 001AC 11$:     RET                                   ; 2664

; Routine Size:  429 bytes,    Routine Base:  $CODE$ + 0AF4


; 2404          2665  1
; 2405          2666  1 END
; 2406          2667  0 ELUDOM



                                                     .EXTRN  LIB$SIGNAL

;                              PSECT SUMMARY
;
;       Name                   Bytes                  Attributes
;
;   $PLIT$                       740  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $OWN$                         12  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $CODE$                      3233  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       13         0         581         00:01.0



;                              COMMAND QUALIFIERS

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

; Size:         3233 code + 752 data bytes
```

```
; Run Time:         01:00.4
; Elapsed Time:     02:03.6
; Lines/CPU Min:     2651
; Lexemes/CPU-Min: 17403
; Memory Used:  213 pages
; Compilation Complete
```

EDFSDLMSG
LIS

EDFMAIN
LIS

EDFSHOW
LIS

EDFGRF
LIS

EDFMSG
LIS