SIMPLEPLOT
2-16 Supplement

BUSS Ltd.

SIMPLEPLOT 2-16 Supplement

Copyright © 2001 by

BUSS Ltd.
Business and Innovation Centre
Angel Way
Bradford
West Yorkshire
BD7 1BX
United Kingdom

Telephone 01274 841396
+44 1274 841396
Fax 01274 841388
+44 1274 841388

Web Site http://www.buss.co.uk/buss
Email info@buss.co.uk

All rights reserved.

First Edition, November 2001

Contents

List of Figures

Preface

SIMPLEPLOT is a library of FORTRAN subroutines for plotting graphs. A wide variety of graphs can be drawn as well as more general pictures and diagrams. Facilities are biased towards the graphical representation of data, in particular, scientific data.

SIMPLEPLOT was originally designed for programmers who wanted to draw pictures of their data with minimum programming effort. Although it still achieves this goal, SIMPLEPLOT has developed into a much more powerful tool for professional software engineers.

Six separate sections constitute the complete SIMPLEPLOT Mark 2:

- The basic package for conventional graph plotting – x-y plots and polar plots.
- Additional subroutines for 3-dimensional plotting – contour maps and surface pictures of 3-D data.
- Additional subroutines for presentation graphics – bar charts, histograms and pie charts.
- *SIMPLEPLOT Volumes* – perspective pictures of 4-dimensional data.
- *SIMPLEPLOT Maps* – for representing data based on geographical coordinate systems.
- *SIMPLEPLOT ViSualization* – for full colour modelling of functions of two, three and four variables.

SIMPLEPLOT-PLUS refers to a SIMPLEPLOT library which is made up from the first three sections and contains many additional facilities.

Graphics device interface

The SIMPLEPLOT library is independent of any single graphics system but, rather than describe SIMPLEPLOT as either device *independent* or device *dependent*, it is more appropriate to describe it as device *sensitive* with a device independent interface for the user. This means that the user is protected from having to know about the features of the target output device, but SIMPLEPLOT makes as much use of these features as possible.

The SIMPLEPLOT library is usually supplied with the BUSS *Single Entry Point* (S.E.P.) device driver system; this includes interfaces a large number of graphics devices, and the range of validated device drivers is constantly being extended. Moreover, SIMPLEPLOT can address graphics devices directly, or through a separate low-level graphics systems (*eg.* GKS, CGM) or graphics languages (*eg.* PostScript).

1. Introduction

The *SIMPLEPLOT 2-16 Supplement* consists a set of technical appendices for SIMPLEPLOT, some of which can also be found in earlier editions of the *SIMPLEPLOT Reference manual*. The *SIMPLEPLOT 2-16 Supplement* introduces the new facilities in SIMPLEPLOT version 2-16. For detailed explanations of all SIMPLEPLOT-PLUS and *SIMPLEPLOT ViSualization* subroutines, please refer to the *SIMPLEPLOT Reference manual* (8^{th} edition).

1.1 Overview

The *SIMPLEPLOT 2-16 Supplement* contains a number of technical appendices:

A. Example programs using the new SIMPLEPLOT 2-16 subroutines.

B. Brief description of new SIMPLEPLOT 2-16 subroutines.

D. Device interface – the Single Entry Point device driver system.

G. Graphic details – shading patterns, broken line styles, marker symbols and character sets.

L. Drawing logos with `BOXPG2` – supplement to explain new 2-16 subroutine.

M. Diagnostic messages.

S. Summary of all SIMPLEPLOT-PLUS and *SIMPLEPLOT ViSualization* subroutines.

T. Useful Tables

The *SIMPLEPLOT 2-16 Supplement* also contains this introduction and an index.

1.2 Software version

This manual is based on SIMPLEPLOT Mark 2, version 2-16.

1.3 Target audience

The *SIMPLEPLOT 2-16 Supplement* has been written with the following readers in mind:

- All users of SIMPLEPLOT who need details of subroutine arguments and behaviour, or specific information about the Single Entry Point device driver system.
- Existing users of SIMPLEPLOT who wish to update their knowledge (and programs) to include the new facilities of SIMPLEPLOT version 2-16.

All readers should be familiar with programming in FORTRAN on their host computer system.

1.4 Related documents

Related documents include

- The *SIMPLEPLOT Reference manual* (8^{th} edition) which contains full specifications for all subroutines in SIMPLEPLOT-PLUS and *SIMPLEPLOT ViSualization*, and in which *Host Specific Information* is also available.
- The *SIMPLEPLOT 2-13 Supplement* which gives details of the facilities introduced in SIMPLEPLOT version 2-13.
- The *SIMPLEPLOT Primer* which provides an introduction to SIMPLEPLOT, especially those facilities available for plotting 2-D data.
- The *SIMPLEPLOT 3-D* manual which describes how to draw representations of 3-D data including the new waterfall charts, stored curve coordinates and labelled curves.
- The *SIMPLEPLOT Volumes* manual which describes the *SIMPLEPLOT Volumes* subroutines for drawing perspective volumes of 3-D `REAL` arrays.
- The *SIMPLEPLOT Maps* manual which describes subroutines for plotting geographical data.
- The *SIMPLEPLOT ViSualization* manual which describes subroutines for perspective drawing of functions of two, three and four variables.

1.5 How to report problems

If you have any problems with SIMPLEPLOT software or its associated products and services please notify us on one of our Software Performance Report (SPR) forms. One of these should be sent out with every software kit – please photocopy it or contact us if you would like extra copies.

1.6 Conventions

The following conventions are used in this manual for subroutine specifications. All specifications are given in a similar format whether they are classified as graphics, specification or auxiliary subroutines.

- Numbers in square brackets, [1], [2], [3] and [4], within subroutine specifications relate to the use of SIMPLEPLOT with plotting devices which have program-selectable pens; these pointers may be reset as often as required (see `PEN` and `SETPNS`).
- Subroutines which take text strings as arguments are provided in two forms but both use FORTRAN 77 `CHARACTER` expressions. The only difference between them is that the subroutines which have the extra `INTEGER` argument interpret this number as the exact length of the text string; in this case,
  - There is no removal of trailing spaces.
  - Only the specified number of characters are used; this may lead to truncation of the string itself.
- Potential diagnostic messages are given in each subroutine specification with a number indicating the type of message and therefore at which diagnostic level it is issued. Additional information about individual messages is given in Appendix M.

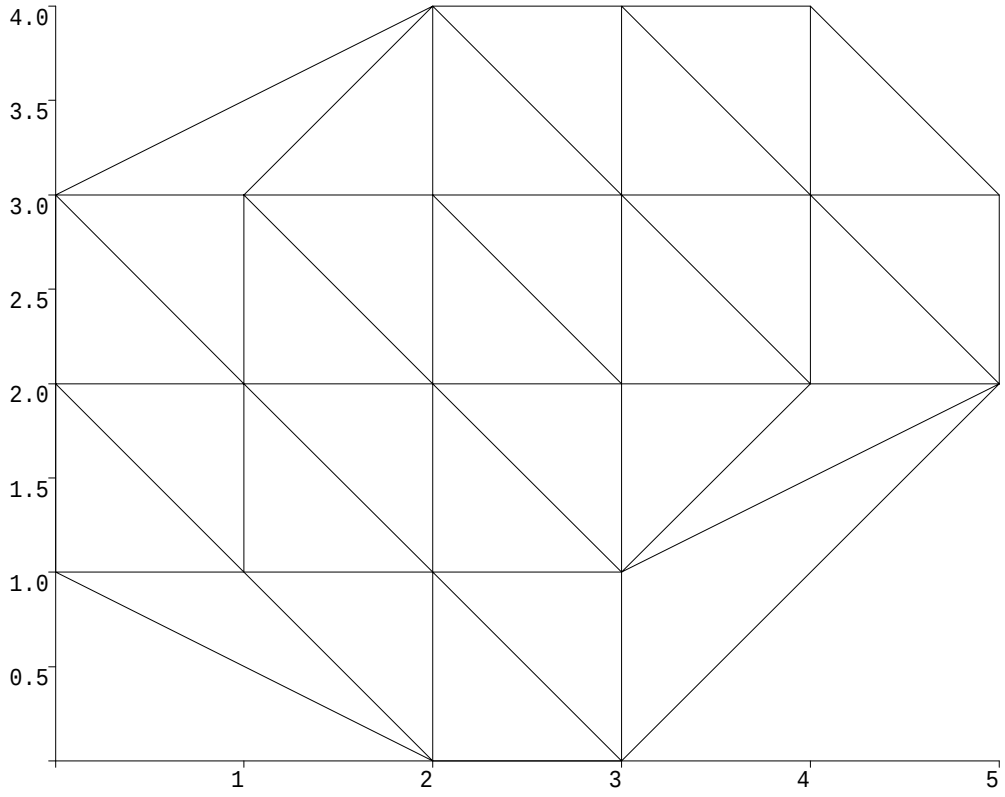
A. Example programs

The following examples of new and extended subroutines in SIMPLEPLOT version 2-16 are given below:

A.1 Simple Triangulation

This example illustrates how to:

- create a triangulation object,
- register triangulation data,
- triangulate data,
- retrieve triangles,
- delete triangulation object,
- display triangulated data.



Explanation of subroutines

`TRINEW(IHANDL)` allocates memory for a new Triangle object, and returns a handle to it (in `IHANDL`). `IHANDL` is negative if no new Triangle object can be created.

`TRIREG(IHANDL, XARR, YARR, NPTS, ISTAT)` registers the coordinates of `NPTS` points from `XARR` and `YARR` into the Triangle object accessed by `IHANDL`.

`TRIANG(IHANDL, NTRI, ISTAT)` triangulates the Triangle object accessed by `IHANDL`. The (x, y) points registered with `IHANDL` by the call to `TRIREG` are organized into non-overlapping triangular elements covering the data. `NTRI` returns the number of triangles generated. `ISTAT` returns a status code which is negative on failure.

`TRIOFF(IHANDL, IOFF, ISTAT)` specifies the base value (given by `IOFF`) subsequently used for point indices. `ISTAT` returns a status code, which is negative on error.

```
SUBROUTINE Sixt01  ! Triangulate & plot irregular data
  use spmodule
  INTEGER, PARAMETER :: NPTS = 22    ! # 'Random' points
  REAL, DIMENSION(NPTS) :: xArr, yArr
  INTEGER, PARAMETER :: MAX_TRIG = NPTS * 2 + 1
  INTEGER, DIMENSION(3, MAX_TRIG) :: iElArr
  INTEGER iHandle, iStatus, jStatus, nTri
  REAL xMin, xMax, yMin, yMax
  DATA xArr/0.0, 0.0, 0.0, 1.0, 2.0, 3.0, 3.0, 3.0, 4.0, 5.0, 5.0, 4.0, &
    4.0, 3.0, 2.0, 2.0, 1.0, 0.0, 1.0, 2.0, 2.0, 3.0/
  DATA yArr/3.0, 2.0, 1.0, 1.0, 0.0, 0.0, 1.0, 2.0, 2.0, 2.0, 3.0, 3.0, &
    4.0, 4.0, 4.0, 3.0, 3.0, 3.0, 2.0, 1.0, 2.0, 3.0/
  CALL TRINEW(iHandle)                   ! Create triangulation object
  IF(iHandle .LT. 0) RETURN
! ======== Register data & triangulate ================================
  CALL TRIREG(iHandle, xArr, yArr, NPTS, iStatus)
  CALL TRIANG(iHandle, nTri, iStatus)
  IF(iStatus.EQ.0) THEN                  ! retrieve elements if OK
    CALL TRIOFF(iHandle, 1, iStatus)
    CALL TRIGET(iHandle, 1, nTri, iElArr, iStatus)
  ENDIF
  CALL TRIDEL(iHandle, jStatus)          ! Delete triangulation object
  IF(iStatus.NE.0) RETURN                ! Previous error
! ======== Draw element structure =====================================
! ======== Find data range, & define isotropic scales =================
  CALL LIMEXC(xArr, NPTS, xMin, xMax)    ! Find data range
  CALL LIMEXC(yArr, NPTS, yMin, yMax)
  CALL EQSCAL(xMin, xMax, yMin, yMax, 0)  ! Define isotropic scales
! ======== Draw element structure =====================================
  CALL AXES7(' ', ' ')  ! Start picture & draw axes
  CALL ZELEMS(xArr, yArr, NPTS, iElArr, 3, nTri)
END SUBROUTINE Sixt01
```

Example 1. Simple Triangulation

`TRIGET(IHANDLE, IFROM, ISIZE, I2ARR, ISTAT)` extracts upto `ISIZE` triangular elements from the Triangle object, staring from triangle number `IFROM`, counting from 1. For each triangle, three point indicies are placed in `I2ARR`. By default, point indicies are counted from 1, but a call of `TRIOFF` can be used to change the base index. If `ISTAT` returns the number of triangles returned or a negative error code.

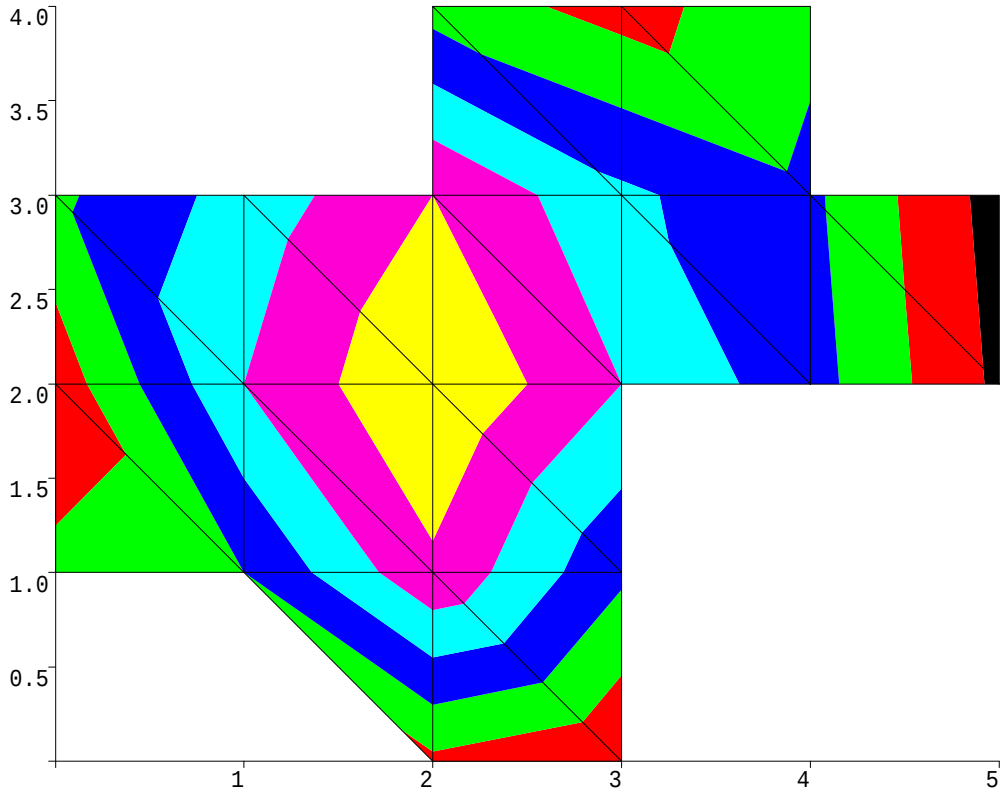
`TRIDEL(IHANDL, ISTAT)` deletes the Triangle object accessed by by `IHANDL`.

`ZELEMS(XARR, YARR, NTPS, I2ARR, NNODES, NELEMS)` draws the configuration of a set of area elements (without neighbours).

A.2 Triangulation specifying a boundary

This example illustrates how to:

- specify a single outer boundary
- display triangulated data and boundary.



Explanation of subroutines

`TRIFIX(IHANDL, IFIRST, NPTS, BCLOSE, ISTAT)` specifies that `NPTS` points starting from `IFIRST` define a fixed line. `BCLOSE` specifies whether the fixed line defines a closed boundary. When `BCLOSE` is true, the first point is joined to the last.

`ZSHDS(XARR, YARR, ZARR, NPTS, I2ARR, NNODES, NELEMS)` draws a shaded contour map from ungridded 3-D data.

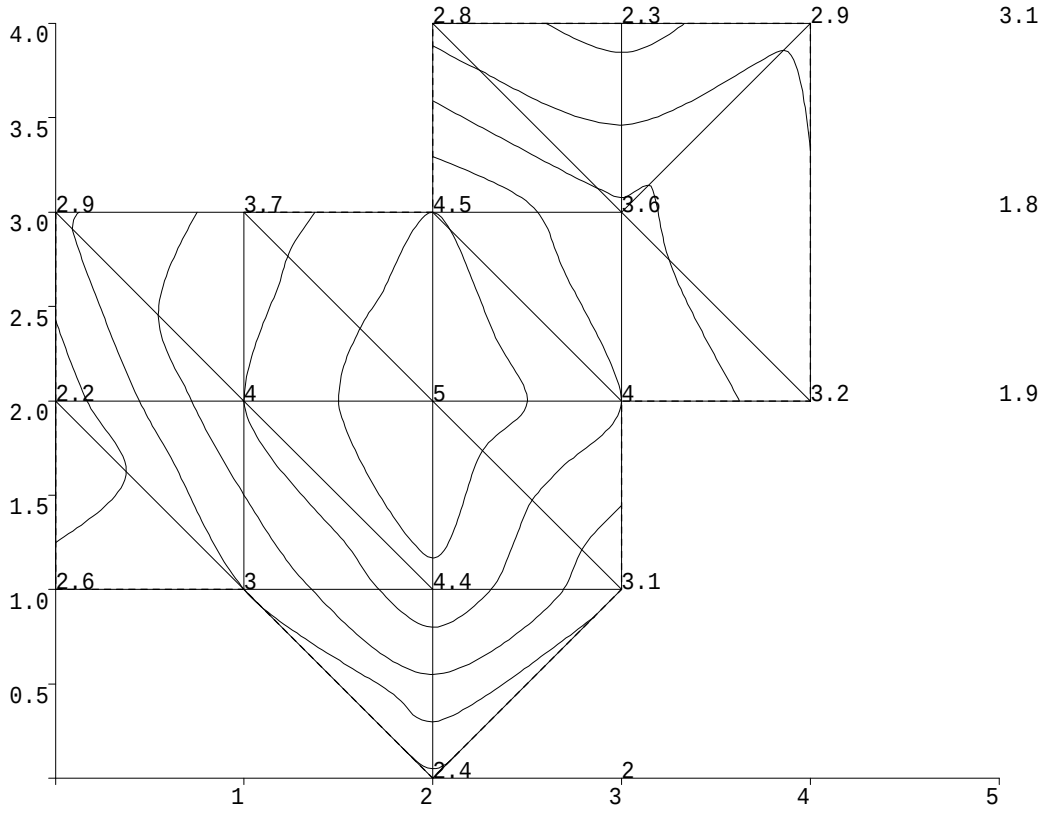
```
SUBROUTINE Sixt02  ! 1st 'NEDGE' points define perimeter
  use spmodule
  INTEGER, PARAMETER :: NPTS = 21   ! Total # points
  INTEGER, PARAMETER :: NEDGE = 17  ! # Perimeter points
  REAL, DIMENSION(NPTS) :: xArr, yArr, zArr
  INTEGER, PARAMETER :: MAX_TRIG = NPTS * 2 + 1
  INTEGER, DIMENSION(3, MAX_TRIG) :: iElArr
  INTEGER iHandle, iStatus, jStatus
  REAL xMin, xMax, yMin, yMax
  DATA xArr/0.0, 0.0, 0.0, 1.0, 2.0, 3.0, 3.0, 3.0, 4.0, 5.0, 5.0, &
    4.0, 4.0, 3.0, 2.0, 2.0, 1.0, 1.0, 2.0, 2.0, 3.0/
  DATA yArr/3.0, 2.0, 1.0, 1.0, 0.0, 0.0, 1.0, 2.0, 2.0, 2.0, 3.0, &
    3.0, 4.0, 4.0, 4.0, 3.0, 3.0, 2.0, 1.0, 2.0, 3.0/
  DATA zArr/2.9, 2.2, 2.6, 3.0, 2.4, 2.0, 3.1, 4.0, 3.2, 1.9, 1.8, &
    3.1, 2.9, 2.3, 2.8, 4.5, 3.7, 4.0, 4.4, 5.0, 3.6/
  CALL TRINEW(iHandle)                ! Create Triangulation object
  IF(iHandle .LT. 0) RETURN
! ======== Register data & define boundary ========================
  CALL TRIREG(iHandle, xArr, yArr, NPTS, iStatus)
  CALL TRIFIX(iHandle, 1, NEDGE, .TRUE., iStatus)
! ======== Triangulate: ===========================================
  CALL TRIANG(iHandle, nTri, iStatus)
  IF(iStatus.EQ.0) THEN               ! Triangulation OK
    CALL TRIOFF(iHandle, 1, iStatus)
    CALL TRIGET(iHandle, 1, nTri, iElArr, iStatus)
  ENDIF
  CALL TRIDEL(iHandle, jStatus)       ! Delete Triangulation object
  IF(iStatus.NE.0) RETURN     ! Previous error
! ======== Find data range, & define isotropic scales =============
  CALL LIMEXC(xArr, NPTS, xMin, xMax)
  CALL LIMEXC(yArr, NPTS, yMin, yMax)
  CALL EQSCAL(xMin, xMax, yMin, yMax, 0)
! ======== Draw solid contour plot ================================
  CALL AXES7(' ', ' ')  ! Start picture & draw axes
  CALL ZSHDS(xArr, yArr, zArr, NPTS, iElArr, 3, NTRI)
! ======== Superimpose element structure ==========================
  CALL ZELEMS(xArr, yArr, NPTS, iElArr, 3, NTRI)
END SUBROUTINE Sixt02
```

Example 2. Triangulation specifying a boundary

A.3 Triangulation specifying a boundary excluding data points

This example illustrates how to:

- define a boundary which omits some registered data points



Explanation of subroutines

`TRIFIX(IHANDL, IFIRST, NPTS, BCLOSE, ISTAT)` specifies that `NPTS` points starting from `IFIRST` define a fixed line. `BCLOSE` specifies whether the fixed line defines a closed boundary. When `BCLOSE` is true, the first point is joined to the last, and points excluded by the region are ignored.

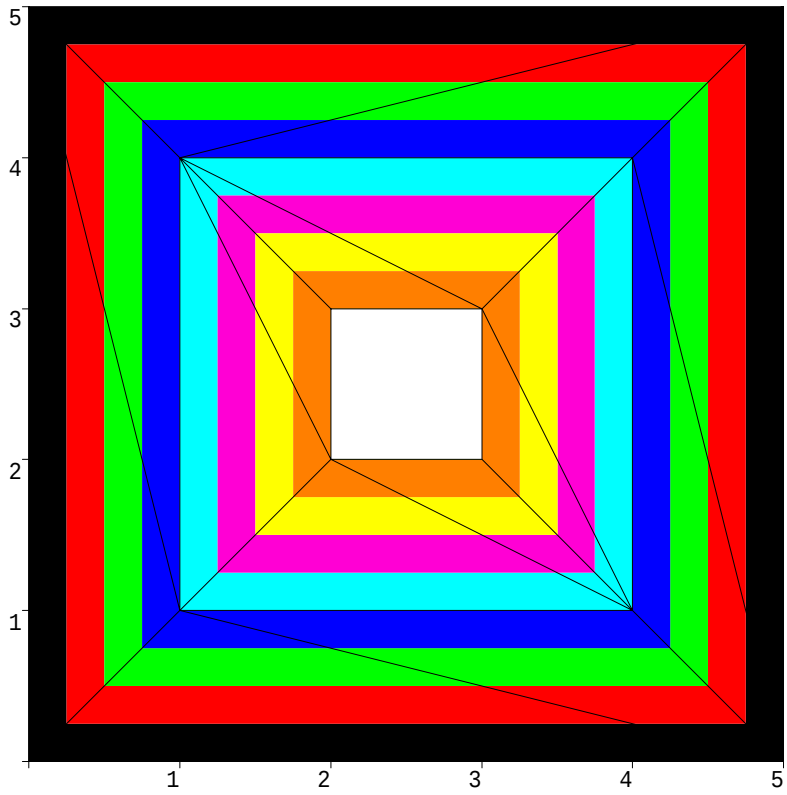
```
SUBROUTINE Sixt03  ! Define Internal edge
  use spmodule
  INTEGER, PARAMETER :: NPTS = 21  ! # Total # points
  INTEGER, PARAMETER :: NEDGE = 13 ! # Perimeter points
  REAL, DIMENSION(NPTS) :: xArr, yArr, zArr
  INTEGER, PARAMETER :: MAX_TRIG = NPTS * 2 + 1
  INTEGER, DIMENSION(3, MAX_TRIG) :: iElArr
  INTEGER iHandle, iStatus, jStatus, nTri
  INTEGER i
  REAL xMin, xMax, yMin, yMax
  CHARACTER*20 cReal
  DATA xArr/0.0, 0.0, 0.0, 1.0, 2.0, 3.0, 3.0, 4.0, 4.0, 3.0, 2.0, &
    2.0, 1.0, 1.0, 2.0, 2.0, 3.0, 3.0, 5.0, 5.0, 5.0/
  DATA yArr/3.0, 2.0, 1.0, 1.0, 0.0, 1.0, 2.0, 2.0, 4.0, 4.0, 4.0, &
    3.0, 3.0, 2.0, 1.0, 2.0, 3.0, 0.0, 2.0, 3.0, 4.0/
  DATA zArr/2.9, 2.2, 2.6, 3.0, 2.4, 3.1, 4.0, 3.2, 2.9, 2.3, 2.8, &
    4.5, 3.7, 4.0, 4.4, 5.0, 3.6, 2.0, 1.9, 1.8, 3.1/
  CALL TRINEW(iHandle)                ! Create Triangulation object
  IF(iHandle.LT.0) RETURN
! ======== Register data & define 2 boundaries ====================
  CALL TRIREG(iHandle, xArr, yArr, NPTS, iStatus)
  CALL TRIFIX(iHandle, 1, NEDGE, .TRUE., iStatus)
! ======== Triangulate. If OK, retrieve elements ==================
  CALL TRIANG(iHandle, nTri, iStatus)
  IF(iStatus.EQ.0) THEN
    CALL TRIOFF(iHandle, 1, iStatus)
    CALL TRIGET(iHandle, 1, nTri, iElArr, iStatus)
  ENDIF
  CALL TRIDEL(iHandle, jStatus)     ! Delete Triangulation object
  IF(iStatus.NE.0) RETURN           ! Previous error
! ======== Find data range, & define isotropic scales =============
  CALL LIMEXC(xArr, NPTS, xMin, xMax)
  CALL LIMEXC(yArr, NPTS, yMin, yMax)
  CALL EQSCAL(xMin, xMax, yMin, yMax, 0)
! ======== Draw data as line contour plot =========================
  CALL AXES7(' ', ' ')  ! Start picture & draw axes
  CALL ZCNTS(xArr, yArr, zArr, NPTS, iElArr, 3, nTri)
! ======== Draw element structure =================================
  CALL ZELEMS(xArr, yArr, NPTS, iElArr, 3, nTri)
! ======== Convert each z value to string, & draw =================
  DO, i = 1, NPTS
    CALL KREAL(zArr(i), cReal)      ! Return z value as string
    CALL CP7LB(xArr(i), yArr(i), cReal)
  ENDDO
! ======== Draw edges as double thick dotted line: ================
  CALL THCKMG('Lines', 2.0)
  CALL CVTYPE(3)                    ! Straight lines between points
  CALL BRKNCV(xArr, yArr, NEDGE, -1)
  CALL THCKMG('Lines', 1.0)         ! Reset
END SUBROUTINE Sixt03
```

Example 3. Triangulation specifying a boundary excluding data points

A.4 Triangulation specifying an inner and outer boundary

This example illustrates how to:

- define an inner and outer boundary



Explanation of subroutines

`TRIFIX(IHANDL, IFIRST, NPTS, BCLOSE, ISTAT)` specifies that `NPTS` points starting from `IFIRST` define a fixed line. `BCLOSE` specifies whether the fixed line defines a closed boundary. If closed boundaries are nested, the area between them defines the region to be triangulated.

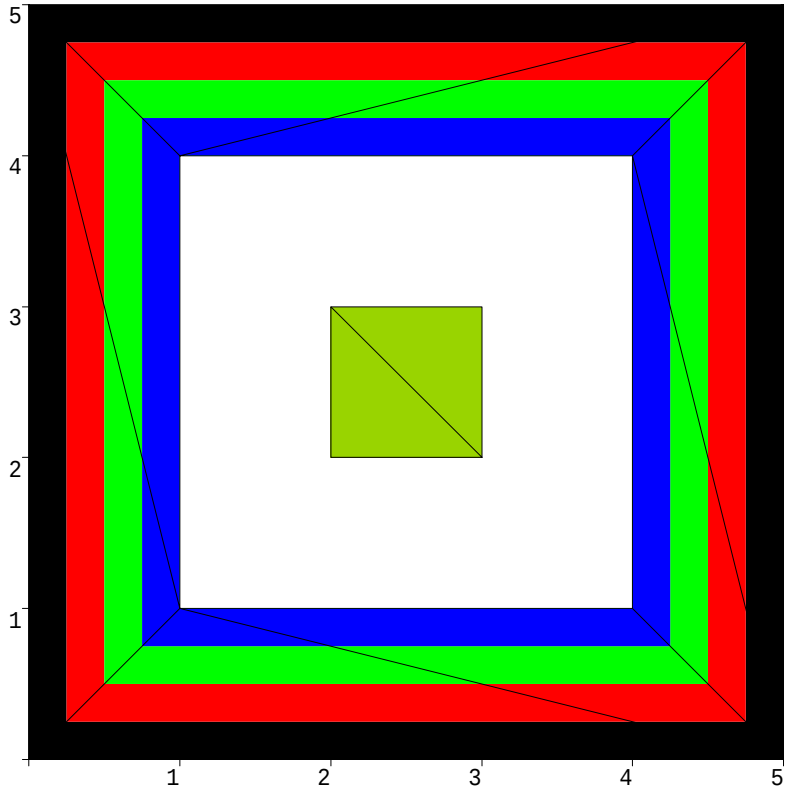
```
SUBROUTINE Sixt04  ! define inner & outer boundaries
  use spmodule
  INTEGER, PARAMETER :: NPTS = 15  ! # points
  REAL, DIMENSION(NPTS) :: xArr, yArr, zArr
  INTEGER, PARAMETER :: MAX_TRIG = NPTS * 2 + 1
  INTEGER, DIMENSION(3, MAX_TRIG) :: iElArr
  INTEGER iHandle, iStatus, jStatus
  REAL xMin, xMax, yMin, yMax
! Define outer & 2 inner rectangles:
  DATA xArr/0.0, 5.0, 5.0, 0.0, 0.0, &
           1.0, 4.0, 4.0, 1.0, 1.0, 2.0, 3.0, 3.0, 2.0, 2.0/
  DATA yArr/0.0, 0.0, 5.0, 5.0, 0.0, &
           1.0, 1.0, 4.0, 4.0, 1.0, 2.0, 2.0, 3.0, 3.0, 2.0/
  DATA zArr/0.0, 0.0, 0.0, 0.0, 0.0, &
           2.0, 2.0, 2.0, 2.0, 2.0, 4.0, 4.0, 4.0, 4.0, 4.0/
  CALL TRINEW(iHandle)             ! Create Triangulation object
  IF(iHandle .LT. 0) RETURN
! ======== Register data & define inner boundaries =============
  CALL TRIREG(iHandle, xArr, yArr, NPTS, iStatus)
  CALL TRIFIX(iHandle, 1, 5, .TRUE., iStatus)
  CALL TRIFIX(iHandle, 11, 5, .TRUE., iStatus)
! ======== Triangulate. If OK, return triangles ================
  CALL TRIANG(iHandle, nTri, iStatus)
  IF(iStatus.EQ.0) THEN
    CALL TRIOFF(iHandle, 1, iStatus)
    CALL TRIGET(iHandle, 1, nTri, iElArr, iStatus)
  ENDIF
  CALL TRIDEL(iHandle, jStatus)         ! Delete Triangulation object
  IF(iStatus .NE. 0) RETURN             ! Previous error
! ======== Superimpose element structure on solid contour plot
  CALL LIMEXC(xArr, NPTS, xMin, xMax)    ! Find data range
  CALL LIMEXC(yArr, NPTS, yMin, yMax)
  CALL EQSCAL(xMin, xMax, yMin, yMax, 0) ! Define isotropic scale
  CALL AXES7(' ', ' ')  ! Start picture & draw axes
  CALL ZSHDS(xArr, yArr, zArr, NPTS, iElArr, 3, nTri)
  CALL ZELEMS(xArr, yArr, NPTS, iElArr, 3, nTri)
END SUBROUTINE Sixt04
```

Example 4. Triangulation specifying an inner and outer boundary

A.5 Triangulation specifying nested boundaries

This example illustrates how to:

- define an island in a lake in an enclosing area.



Explanation of subroutines

`TRIFIX(IHANDL, IFIRST, NPTS, BCLOSE, ISTAT)` specifies that `NPTS` points starting from `IFIRST` define a fixed line. `BCLOSE` specifies whether the fixed line defines a closed boundary. If closed boundaries are nested, the area between them defines the region to be triangulated.

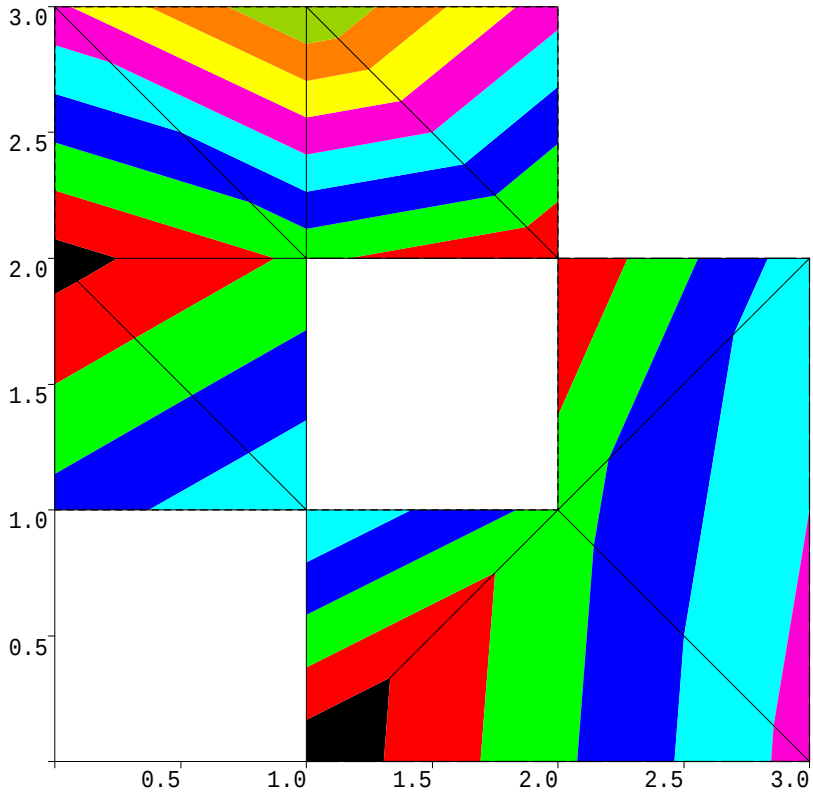
```
SUBROUTINE Sixt05  ! Define outer & 2 inner boundaries
  use spmodule
  INTEGER, PARAMETER :: NPTS = 12  ! # 'Random' points
  REAL, DIMENSION(NPTS) :: xArr, yArr, zArr
  INTEGER, PARAMETER :: MAX_TRIG = NPTS * 2 + 1
  INTEGER, DIMENSION(3, MAX_TRIG) :: iElArr
  INTEGER iHandle, iStatus, jStatus
  REAL xMin, xMax, yMin, yMax
  DATA xArr/0.0, 5.0, 5.0, 0.0, & ! Outer rectangle (x)
            1.0, 4.0, 4.0, 1.0, & ! Inner rectangle (x)
            2.0, 3.0, 3.0, 2.0/   ! Inner rectangle (x)
  DATA yArr/0.0, 0.0, 5.0, 5.0, & ! Outer rectangle (y)
            1.0, 1.0, 4.0, 4.0, & ! Inner rectangle (y)
            2.0, 2.0, 3.0, 3.0/   ! Inner rectangle (y)
  DATA zArr/0.0, 0.0, 0.0, 0.0, & ! Outer rectangle (z)
            2.0, 2.0, 2.0, 2.0, & ! Inner rectangle (z)
            4.0, 4.0, 4.0, 4.0/   ! Inner rectangle (z)
  CALL TRINEW(iHandle)
  IF(iHandle .LT. 0) RETURN
! ======== Register all points & define boundaries: =========
  CALL TRIREG(iHandle, xArr, yArr, NPTS, iStatus)
  CALL TRIFIX(iHandle, 1, 4, .TRUE., iStatus)
  CALL TRIFIX(iHandle, 5, 4, .TRUE., iStatus)
  CALL TRIFIX(iHandle, 9, 4, .TRUE., iStatus)
! ======== Triangulate. If OK, return triangles =============
  CALL TRIANG(iHandle, nTri, iStatus)
  IF(iStatus.EQ.0) THEN
    CALL TRIOFF(iHandle, 1, iStatus)
    CALL TRIGET(iHandle, 1, nTri, iElArr, iStatus)
  ENDIF
  CALL TRIDEL(iHandle, jStatus)
  IF(iStatus .NE. 0) RETURN              ! Previous error
! ======== Find data range, & define isotropic scales
  CALL LIMEXC(xArr, NPTS, xMin, xMax)    ! Find data range
  CALL LIMEXC(yArr, NPTS, yMin, yMax)
  CALL EQSCAL(xMin, xMax, yMin, yMax, 0) ! Define isotropic scales
  CALL AXES7(' ', ' ')                  ! Start picture & draw axes
  CALL ZSHDS(xArr, yArr, zArr, NPTS, iElArr, 3, nTri)
  CALL ZELEMS(xArr, yArr, NPTS, iElArr, 3, nTri)
END SUBROUTINE Sixt05
```

Example 5. Triangulation specifying nested boundaries

A.6 Triangulation specifying intersecting boundaries

This example illustrates how to:

- Define 2 intersecting boundaries



Explanation of subroutines

`TRIFIX(IHANDL, IFIRST, NPTS, BCLOSE, ISTAT)` specifies that `NPTS` points starting from IFIRST define a fixed line. `BCLOSE` specifies whether the fixed line defines a closed boundary. If closed boundaries overlap, the points of intersection between the regions must be data points registered by `TRIREG`.

```
SUBROUTINE Sixt06  ! Define intersecting boundaries
  use spmodule
  INTEGER, PARAMETER :: NPTS = 14 ! # points
  REAL, DIMENSION(NPTS) :: xArr, yArr, zArr
  INTEGER, PARAMETER :: MAX_TRIG = NPTS * 2 + 1
  INTEGER, DIMENSION(3, MAX_TRIG) :: iElArr
  INTEGER iHandle, iStatus, jStatus
  REAL xMin, xMax, yMin, yMax
  DATA xArr/0.0, 3.0, 0.0, 1.0, 2.0, 2.0, 2.0, 1.0, &
    0.0, 0.0, 1.0, 3.0, 3.0, 1.0/
  DATA yArr/0.0, 3.0, 1.0, 1.0, 1.0, 2.0, 3.0, 3.0, &
    3.0, 2.0, 0.0, 0.0, 2.0, 2.0/
  DATA zArr/2.9, 2.2, 2.6, 3.0, 2.4, 2.0, 3.1, 4.0, &
    3.2, 1.9, 1.8, 3.1, 2.9, 2.3/
  CALL TRINEW(iHandle)
  IF(iHandle .LT. 0) RETURN
! Register all points then 2 intersecting perimeters
  CALL TRIREG(iHandle, xArr, yArr, NPTS, iStatus)
  CALL TRIFIX(iHandle, 3, 8, .TRUE., iStatus)
  CALL TRIFIX(iHandle, 11, 4, .TRUE., iStatus)
! ======== Triangulate. If OK, return triangles ============
  CALL TRIANG(iHandle, nTri, iStatus)
  IF(iStatus.EQ.0) THEN
    CALL TRIOFF(iHandle, 1, iStatus)
    CALL TRIGET(iHandle, 1, nTri, iElArr, iStatus)
  ENDIF
  CALL TRIDEL(iHandle, jStatus)
  IF(iStatus.NE.0) RETURN  ! Previous error
! ======== Find data range, & define isotropic scales ======
  CALL LIMEXC(xArr, NPTS, xMin, xMax)
  CALL LIMEXC(yArr, NPTS, yMin, yMax)
  CALL EQSCAL(xMin, xMax, yMin, yMax, 0)
! ======= Superimpose lines on solid contours ==============
  CALL AXES7(' ', ' ')  ! Start picture & draw axes
  CALL ZSHDS(xArr, yArr, zArr, NPTS, iElArr, 3, nTri)
! ======== Draw triangles ==================================
  CALL ZELEMS(xArr, yArr, NPTS, iElArr, 3, nTri)
! ======== Draw boundaries in double-thick lines ===========
  CALL THCKMG('Lines', 2.0)
  CALL CVTYPE(3)       ! Join points with straight lines
  CALL BRKNCV(xArr(3), yArr(3), 8, -1);
  CALL BRKNCV(xArr(11), yArr(11), 4, 1);
  CALL THCKMG('Lines', 1.0)    ! Normal thickness lines
END SUBROUTINE Sixt06
```

Example 6. Triangulation specifying intersecting boundaries

B. Brief specifications of new 2-16 subroutines

This appendix gives brief formal specifications for the SIMPLEPLOT-PLUS introduced in SIMPLEPLOT version 2-16. Full specifications are given in the *SIMPLEPLOT Reference manual* (8^{th} edition).

SUBROUTINE TRIANG (IHANDL, NTRI, ISTAT)

Name

`TRIANG` – to triangulate a Triangle object.

Availability Section 2, released version 2-16.

Arguments

`IHANDL`	`INTEGER` expression	A handle acquired from `TRINEW`
`NTRI`	`INTEGER` variable	To receive number of triangles
`ISTAT`	`INTEGER` variable	To receive a status return

SUBROUTINE TRIDEL (IHANDL, ISTAT)

Name

`TRIDEL` – to delete a Triangle object.

Availability Section 2, released version 2-16.

Arguments

`IHANDL`	`INTEGER` expression	A handle acquired from `TRINEW`
`ISTAT`	`INTEGER` variable	To receive a status return

SUBROUTINE TRIFIX (IHANDL, IFROM, NPTS, BPERIM, ISTAT)

Name

`TRIFIX` – to specify a sequence of points to lie on a fixed line in a Triangle object.

Availability Section 2, released version 2-16.

Arguments

`IHANDL`	`INTEGER` expression	A handle acquired from `TRINEW`
`IFROM`	`INTEGER` expression	Index of 1st point on fixed line
`NPTS`	`INTEGER` expression	Number of points on fixed line
`BPERI`	`LOGICAL` expression	Whether the line is a perimeter
`ISTAT`	`INTEGER` variable	To receive a status return

SUBROUTINE TRIGET (IHANDL, IFROM, ISIZE, I2ARR, ISTAT)

Name

`TRIGET` – to extract triangles from a Triangle object.

Availability Section 2, released version 2-16.

Arguments

IHANDL	INTEGER expression	A handle acquired from TRINEW
IFROM	INTEGER expression	index(from 1) of 1st triangle extracted
ISIZE	INTEGER expression	second dimension of I2ARR
I2ARR	INTEGER 2-D array	To receive data element structure, I2ARR(3,ISIZE)
ISTAT	INTEGER variable	To receive a status return

SUBROUTINE TRINEW (IHANDL)

Name

TRINEW – to acquire a handle to a Triangle object.

Availability Section 2, released version 2-16.

Argument

IHANDL	INTEGER variable	To receive a handle

SUBROUTINE TRIOFF (IHANDL, IOFF, ISTAT)

Name

TRIOFF – to specify the base value subsequently used for point indices.

Availability Section 2, released version 2-16.

Arguments

IHANDL	INTEGER expression	A handle acquired from TRINEW
IOFF	INTEGER expression	Point index offset
ISTAT	INTEGER variable	To receive a status return

SUBROUTINE TRIREG (IHANDL, XARR, YARR, NPTS, ISTAT)

Name

TRIREG – to register the (x, y) points in a Triangle object.

Availability Section 2, released version 2-16.

Arguments

IHANDL	INTEGER expression	A handle acquired from TRINEW
XARR, YARR	REAL arrays	(x, y) coordinates of data values
NPTS	INTEGER expression	Number of data points
ISTAT	INTEGER variable	To receive a status return

D. Graphics Device Interface

This appendix outlines the operation of the BUSS Single Entry Point (S.E.P.) device driver system.

D.1 Introduction

The SIMPLEPLOT library is usually supplied with the Single Entry Point (S.E.P.) device driver system. This device driver system contains some generalized code plus a number of individual device drivers. Each device driver is a set of subroutines (used by SIMPLEPLOT but not used directly by the user program) which control the graphics display or plotting for individual devices. It is the device driver which determines which, if any, hardware characteristics which are used by SIMPLEPLOT:

- the number of colours/pens available;
- the availability of a hardware font, the size of hardware characters and whether hardware text can be rotated;
- the number of hardware shading patterns;
- the default layout of the SIMPLEPLOT page – drivers determine whether a device is classified as a *fixed page* device or a *packed page* device and whether a fixed page can be rotated.

The S.E.P. device driver system allows any combination of individual device drivers, each with one or more modes of operation, or *options*, to be available at run-time. Only a subset of the possible individual device drivers are supplied to each site according to the site's individual requirements. Other sites may well have a different set of individual device drivers.

D.2 Device identification

Each individual device driver attends to the needs of one particular device (or a small group of related devices) and supplies one or more *driver options*. These options are normally used to allow for small changes in the configuration of the device, such as whether A3 or A4 paper is placed in a plotter, or at which resolution a laser printer is required to operate. For each option the following information is specified:

- The absolute name
- The absolute number
- The site number
- The site name
- A brief description

Site names and numbers allow changes in equipment or device drivers to be masked in order to maintain a compatible user interface. Absolute names and numbers allow a specific driver option to be selected correctly, even when an application is moved from one site to another; neither the absolute name nor the absolute number is site configurable.

`QDEV` can be called to inquire the values of the *site number*, *absolute number*, *site name* and *description* for any driver option.

Option numbers

Site numbers are integers in the range 1...999. Initially, each option has a site number of its position in the table (*ie.* the first option will have a site number of 1, the second 2, *etc.*). Site numbers may be altered to better suit a site's requirements if required.

The absolute number is an integer in the range 1000–32767. It can be used to specify a particular option, regardless of local site configuration. The absolute number is assigned by BUSS and cannot be locally configured.

For example, an application may specifically require the use of a Colour PostScript printer. At one site, the CPS option may have a site number of 5, at another it may be 27. However, at all sites which support Colour PostScript, the absolute number 8720 will always select the CPS option.

Both the site number and the absolute number can be used as the argument to the SIMPLEPLOT subroutine `DEVNO`.

Option names

Each option has both a *site name* and an *absolute name* which correspond to the site number and absolute number respectively. By default, the site name is the same as the absolute name (which is fixed by BUSS) but the site name may be changed. Both the *site name* and the *absolute name* can be used as an argument to `DEVNAM` – absolute names are preceded by a slash (`/`). For example,

```
CALL DEVNAM('/MOTIF')
```

It is the site name which is displayed on the device selection menu but absolute names can be used to select an option if the specified name is preceded by a slash character.

D.3 Device selection

The S.E.P. device driver system is designed such that driver options can be selected by name or number from within the program (*via* `DEVNO` or `DEVNAM`) or selected at run-time. The S.E.P. system creates a table of available options from all the installed device drivers.

```
Site No  Abs No  Name
-------  ------  ----
      1:   6330: VT340               DEC VT340
      2:   6331: VT340_V21           DEC VT340 with V2.1 firmware
      3:   9100: MOTIF               OSF/Motif window
      4:   9110: MOTIF_FULL          OSF/Motif window, full size
      5:   8720: CPS                 Colour PostScript printer
      6:   8400: LN03R               DEC LN03R PostScript generator
      7:   1140: LASERWRITER         Apple LaserWriter PostScript generator
      8:   1160: LPS40               DEC PrintServer 40 PostScript generator
      9:   2550: HPLASER             HP Laser printer
     10:   6480: DESKJET             HP DeskJet printer
     11:   6490: DESKJET_DITHER      HP DeskJet printer. 15 dithered colours
     12:   6491: DESKJET_BASIC       HP DeskJet. 7 basic colours @150 dpi
     13:   6492: DESKJET_PLUS        HP DeskJet Plus. Monochrome @100 dpi
     14:   6493: DESKJET_BASIC_K     DeskJet. 7 basic colours @150 dpi CMYK
     15:   6494: DESKJET_DITHER_K    DeskJet. 15 dithered colours CMYK
     16:  11101: CGMBINARY           Computer Graphics Metafile (Binary)
     17:  10000: META                SIMPLEPLOT Metafile
     18:   8780: TIMAGE              BUSS Test Image
Enter NAME, NUMBER or <RETURN> for menu:
```

Figure D.1 S.E.P. device selection menu

If neither `DEVNO` nor `DEVNAM` has been called before plotting is started, the system prompts the user to select an option (unless altered by a configuration file) – a title will appear (identifying SIMPLEPLOT as the source of the request) followed by a user prompt. For example,

```
+-----------------------------------+
| SIMPLEPLOT Device Driver Selection |
+-----------------------------------+

Enter NAME, NUMBER or <RETURN> for menu:
```

At this point an option may be selected (by name or number) without having to list out the whole table of options, which could be quite extensive. To view the table of options (see Figure D.1) just press the RETURN key (this may be known as ENTER or NEXT or ACCEPT on some systems). If an unrecognized option is selected, an error message is output and the user is prompted again.

Interactive selection may also be specified by `CALL DEVNAM(' ')` or `CALL DEVNO(0)`.

D.4 Configuration of the device selection menu

A menu configuration file may be used to alter the contents of the option table which can be displayed at run-time as a menu. The configuration file may be used to alter:

- the site number, site name and/or description of an option,
- whether an option is available and/or visible (*ie.* offered in a menu),
- the default option,
- the position in which the option appears in the menu,
- how many lines of menu are generated before the user is prompted.

Please refer to Appendix H in the *SIMPLEPLOT Reference manual* for information about making your menu configuration file known to the Single Entry Point system. Some machines allow the use of an *auxiliary* menu configuration, which provides a method of specifying a small number of command lines specified without having to create a file to contain them. Where both file and auxiliary configuration are supplied, commands in the menu configuration file are interpreted before those supplied *via* auxiliary input. Typically the configuration file is used to specify the general form of the menu while the auxiliary input is used to make any small alterations, such as specifying the `DEFAULT` option for individual users.

D.4.1 Menu configuration commands

The following commands can be used to configure the interactive selection menu; commands are not case sensitive and are executed in the order they are supplied.

option-id identifies an option and may be either:

- an absolute number (in the range 1000...32767),
- a current site number (in the range 1...999),
- an absolute name (which starts with a slash character (/)),
- a current site name.

In order to reduce the interdependence of commands, it is advisable to use absolute names or numbers rather than site names or numbers.

! introduces a comment. Comments are terminated by end of line and may be placed on any line in the menu configuration file.

DEFAULT *option-id*

The `DEFAULT` command can be used to specify which option is to be used if neither `DEVNO` nor `DEVNAM` is called. The configuration file should normally contain only one `DEFAULT` command but, if more than one is included, the last one overrides previous ones. If the configuration file does not contain a valid `DEFAULT` command the system will assume that interactive device selection is required.

The `DEFAULT` command does not use the *option-id* immediately, but stores it until it is needed; this will be after the menu has been completely configured. If the *o*ption-id is omitted, a null default is set which results in an interactive menu (the same as if all `DEFAULT` commands were omitted).

Use of `DEVNO` or `DEVNAM` overrides any requested default.

DESCRIBE *option-id* "description"

The `DESCRIBE` command enables a new description to be given to an option.

HIDE *option-id*

The `HIDE` command can be used to specify that the option should not appear on the menu but should still be available for selection.

HIDEALL

The `HIDEALL` command can be used to specify that all options are to be hidden. This allows only required options to be visible, by hiding everything and then unhiding those required (using `UNHIDE` or `POSITION`).

NAME *option-id site-name*

The `NAME` command sets a *site-name* for the option which will then be displayed on the menu and can be used to select a name from the menu at run-time. Care should be taken to ensure that no two options are given the same name.

All names are converted to upper-case and should start with an alphabetic character which may be followed by any alphanumeric or underscore character up to a maximum overall length of 20 characters.

NOPAGEWAIT

The `NOPAGEWAIT` command can be used to specify that the options menu should not wait for user input until the whole of the menu has been displayed.

This command cancels any previous `PAGEWAIT` commands.

NUMBER *option-id site-number*

The `NUMBER` command sets a *site-number* for the option which will then be displayed on the menu and can be used to select a name from the menu at run-time. The site number must be in the range 1...999 and care should be taken to ensure that no two options are given the same number.

This command does not affect the absolute number of an option which cannot be changed.

PAGEWAIT *integer*

The `PAGEWAIT` command can be used to specify how many lines of characters may be used by the options menu, before waiting for user input. Since this value includes space for two lines of column titles, one line of user prompt and one line for the user response, values of four or less should not be specified.

Normally, the user is prompted with the message:

```
Enter NAME, NUMBER or <RETURN> for menu:
```

but when there is still a part of the menu that has not yet been displayed, the following message is issued instead:

```
Enter NAME, NUMBER or <RETURN> to continue:
```

The user is always prompted at the end of a complete menu.

`PAGEWAIT` cancels the effect of all previous `NOPAGEWAIT` and `PAGEWAIT` commands. If no valid `PAGEWAIT` command is supplied, the effect is the same as if a `NOPAGEWAIT` command is supplied – by default the menu does not wait until it has been output completely. For example, `PAGEWAIT 24` may be useful for long menus on 24-line terminals.

POSITION *option-id display-position*

The `POSITION` command can be used to specify a new position in the option table, where *d*isplay-position is a number which must be within the range 1...current size of menu. The size of the menu will increase by one if the option was previously hidden.

Please note that options which lie in between where the option is moved from and where it is moved to, move up or down the table by one position.

REMOVE *option-id*

The `REMOVE` command can be used to remove options from the option table. All options which followed the removed option in the table are moved one position upwards and the menu size is reduced by one.

RENUMBER

The `RENUMBER` command can be used to change the site number of every option to match its position in the menu. For the purpose of this command hidden options are considered to follow on from the end of the menu, but in no defined order. After executing this command, every option (visible or hidden) has a unique site number.

SET *option-id* [`#`*display-position*] [*site-number*] [*site-name*] "*description*"]

The `SET` command can be used to set one or more attribute at a time.

```
! Example Single Entry Point MENU configuration file
!
REMOVE /VT340               ! All our VT340s have V2.1 firmware
NAME /VT340_V21 VT340
HIDE /MOTIF_FULL
HIDE /LN03R
HIDE /LPS40
DESCRIBE /LASERWRITER "Mono PostScript printer"
REMOVE /TIMAGE
NAME /CPS PHASER
POSITION MOTIF 1
RENUMBER
NUMBER VT340 340            ! Note that "VT340" means "/VT340_V21"
SET 6480 500 DESKJET_500  "Deskjet 500 (mono)"
SET 6494 550 DESKJET_550C "Deskjet 550C (colour)"
PAGEWAIT 24
```

Figure D.2 Menu configuration file

UNHIDE *option-id*

The `UNHIDE` command can be used to specify that an option is to be displayed in the menu. The position at which this option appears within the menu is not predictable. An option may also be unhidden using the `POSITION` command to reposition it within the menu.

Please note that some S.E.P. device drivers supply options that are already hidden and this may cause an apparent discontinuity of site numbers. Such options are supplied for special purposes and unless otherwise recommended by BUSS, these should not normally be used.

D.4.2 An example menu configuration

The configuration file listed in Figure D.2 is used to illustrate how the device selection menu shown in Figure D.1 is modified to appear as illustrated in Figure D.3. These commands configure the device selection menu as follows:

- The original `VT340` option is completely removed (it now cannot be selected by any means) and the site name of the`VT340_V21` option is changed to `VT340` to replace the original `VT340` option.
- The option to fill the display with a SIMPLEPLOT Motif window (`MOTIF_FULL`) and the LN03R and LPS40 PostScript printer options are all hidden. They may be selected but will not be displayed.
- The `LASERWRITER` option will be displayed with a new description.
- The `TIMAGE` option will be moved.
- The Colour PostScript option will be offered as `PHASER`.
- The Motif option is placed at the top of the menu.
- All site numbers are reset; commands may be in upper, lower or mixed case as preferred.
- The `VT340_V21` options is given a more meaningful site number. Note that, as a result of the first two configuration commands of the file, the option-id `VT340` now refers to the `VT340_V21` option and not the original `VT340` option.
- The `SET` commands identify the `Deskjet` option by number, and give them a new name and description.
- When the menu is output by the SIMPLEPLOT Single Entry Point system, output will pause every 22 lines (or fewer if the menu is shorter) and the user will be prompted.

```
+-----------------------------------+
| SIMPLEPLOT Device Driver Selection |
+-----------------------------------+

Enter NAME, NUMBER or <RETURN> for menu:

Site No  Abs No  Name
-------  ------  ----
      1:   9100: MOTIF               OSF/Motif window
    340:   6331: VT340               DEC VT340 with V2.1 firmware
      3:   8720: PHASER              Colour PostScript printer
      4:   1140: LASERWRITER         Mono PostScript printer
      5:   2550: HPLASER             HP Laser printer
    500:   6480: DESKJET_500         Deskjet 500 (mono)
      7:   6490: DESKJET_DITHER      HP DeskJet printer. 15 dithered colours
      8:   6491: DESKJET_BASIC       HP DeskJet. 7 basic colours @150 dpi
      9:   6492: DESKJET_PLUS        HP DeskJet Plus. Monochrome @100 dpi
     10:   6493: DESKJET_BASIC_K     DeskJet. 7 basic colours @150 dpi CMYK
    550:   6494: DESKJET_550C        Deskjet 550C (colour)
     12:  11101: CGMBINARY           Computer Graphics Metafile (Binary)
     13:  10000: META                SIMPLEPLOT Metafile
Enter NAME, NUMBER or <RETURN> for menu:
```

Figure D.3 S.E.P. device selection menu after configuration

Please note that the *option-id* for each command must be valid when it is used. For all commands except the `DEFAULT` command, this is when the command is interpreted. However, the option-id specified by the `DEFAULT` command is used not when the command is interpreted, but when a selection from the menu is required, that is, *after* configuration. Thus, given the above example menu and configuration file, if the `DEFAULT MOTIF` command was removed, the command `DEFAULT 1` would have the same effect regardless of whether it is placed at the start of the file, the end of the file or even somewhere in the middle.

D.5 Configuration of device drivers

Some S.E.P. device drivers can be configured using *device driver configuration files*. In a similar way to menu configuration, there are two types of device driver configuration file – main configuration files and auxiliary configuration files. Main configuration files are text files held in the normal file-store. Auxiliary configuration files are usually text held in some operating system variable (*eg.* logical names, shell variables or environment variables) which provide a convenient way of passing a small amount of information (often just one line of text); on some systems, auxiliary configuration files may not be available.

If configuration is in use, the main configuration file is read first (unless otherwise stated in the individual device driver's information sheet), immediately followed by the auxiliary configuration file; no distinction is made internally between the two. Thus, the effect is the same as if the contents of the auxiliary configuration file were removed and appended to the end of the main configuration file. Please refer to Appendix H of the *SIMPLEPLOT Reference manual* for information about making your device driver configuration file known to the S.E.P. system.

A device driver verification and demonstration program, CHECK, is available which links only with the S.E.P. device driver system but not with the SIMPLEPLOT library. CHECK exercises all the features of the various device drivers and can be used to test any changes to device driver configuration files.

G. Graphic Details

This appendix illustrates the graphical details of SIMPLEPLOT.

G.1 Broken line patterns

`CTBRKN` and `SQBRKN` specify broken line patterns for contour curves and `BRKNBX`, `BRKNPT`, `LABCV7` and `LINEK7` draw lines with a specified broken line pattern. These subroutines identify the line pattern by values in the range −6 to +6. The number of software line patterns is unlimited but patterns beyond the usual range −6 . . . 6 have longer patterns and may not be easily distinguishable from one another. The number of hardware broken line patterns is also unlimited in theory but in practice there are fewer than are available in software. Figure G.1 illustrates the thirteen SIMPLEPLOT software broken line patterns.

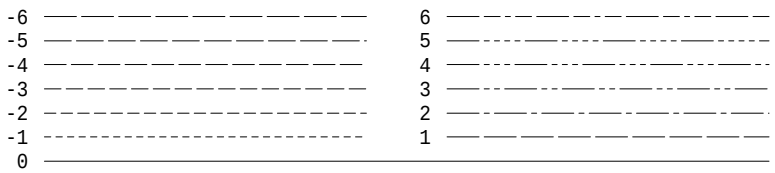


Figure G.1 SIMPLEPLOT software broken line patterns

Please refer to the specification of `BUNLPR` in the *SIMPLEPLOT Reference manual* for details of the cyclic order of broken line patterns when used as a bundled attribute for line drawing.

G.2 Shading patterns

Shading patterns are a function of colour (where available) and pattern. The precise details of the patterns depend on the output device but are always chosen to give distinct appearances.

SIMPLEPLOT uses hardware shading by default when possible. The availability and number of hardware shading patterns depend on the device you are using. On all devices SIMPLEPLOT resorts to software shading patterns when the hardware patterns have been exhausted. Figure G.2 illustrates hardware shading on a monochrome device with ten hardware shading patterns.

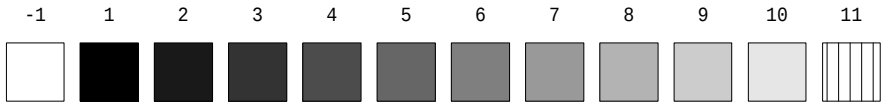


Figure G.2 Typical hardware shading patterns on a monochrome device

The patterns used for software shading consist of parallel hatching lines with adjustable angles and line separation. Figure G.3 illustrates software shading on a monochrome device. Software shading patterns consist of sets of equally-spaced parallel lines of one colour.

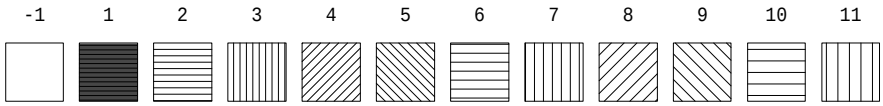


Figure G.3 Software shading patterns on a monochrome device

The default sequence for a monochrome device is as follows:

- Pattern −1 is empty but is outlined using the pen selected by `SHPEN`, or the pen currently selected for drawing lines (pen pointer 1).
- Pattern 0 (not illustrated) is as near to solid as the device permits, using background colour; if the device cannot draw in background colour, pattern 0 is equivalent to pattern −1.
- Pattern 1 is as near to solid as the device permits and is not affected by settings for the angle or separation of hatching lines.
- Patterns 2–5 use the four shading angles with a small line separation.
- Patterns 6–9 use the four shading angles with a larger line separation.
- *etc.*

If a very large pattern number is chosen, very large line separation is used.

By default, SIMPLEPLOT uses four shading angles, 0°, 90°, 45° and 135°; the number of colours used is set to the maximum number of colours available on the device, and the minimum separation between shading lines is set to the thickness of the lines drawn on the device. `SHDESC` can be called before a shading operation to specify alternatives for all these shading characteristics – the number of shading angles used is the number of different angles specified; the number of shading colours can be any positive value and the minimum separation between shading lines can be set to any value greater than or equal to the standard thickness of lines on the device.

G.3 Fonts

By default, SIMPLEPLOT uses the most appropriate hardware characters available on a graphics device to write text. In addition to hardware text, a set of simple software characters, proportionally spaced fonts (Hershey characters) and an adjustable fixed width font are available. Please note:

- *Hardware* fonts differ between graphics devices therefore lettering that fits comfortably on one device may be smaller or larger on another.

  Some devices have *adjustable hardware fonts*, which are indicated by negative numbers. Figure G.30 shows the PostScript font −9 as an example.

  See Appendix H of the *SIMPLEPLOT Reference manual* for the fonts available on your device.
- The *simple software* font is designed always to be clearly readable and may appear relatively larger on some low resolution graphics devices.
- Other software fonts are drawn independently of the resolution of the graphics device and may be illegible on some devices.

Figures G.4 to G.30 illustrate the character sets available.

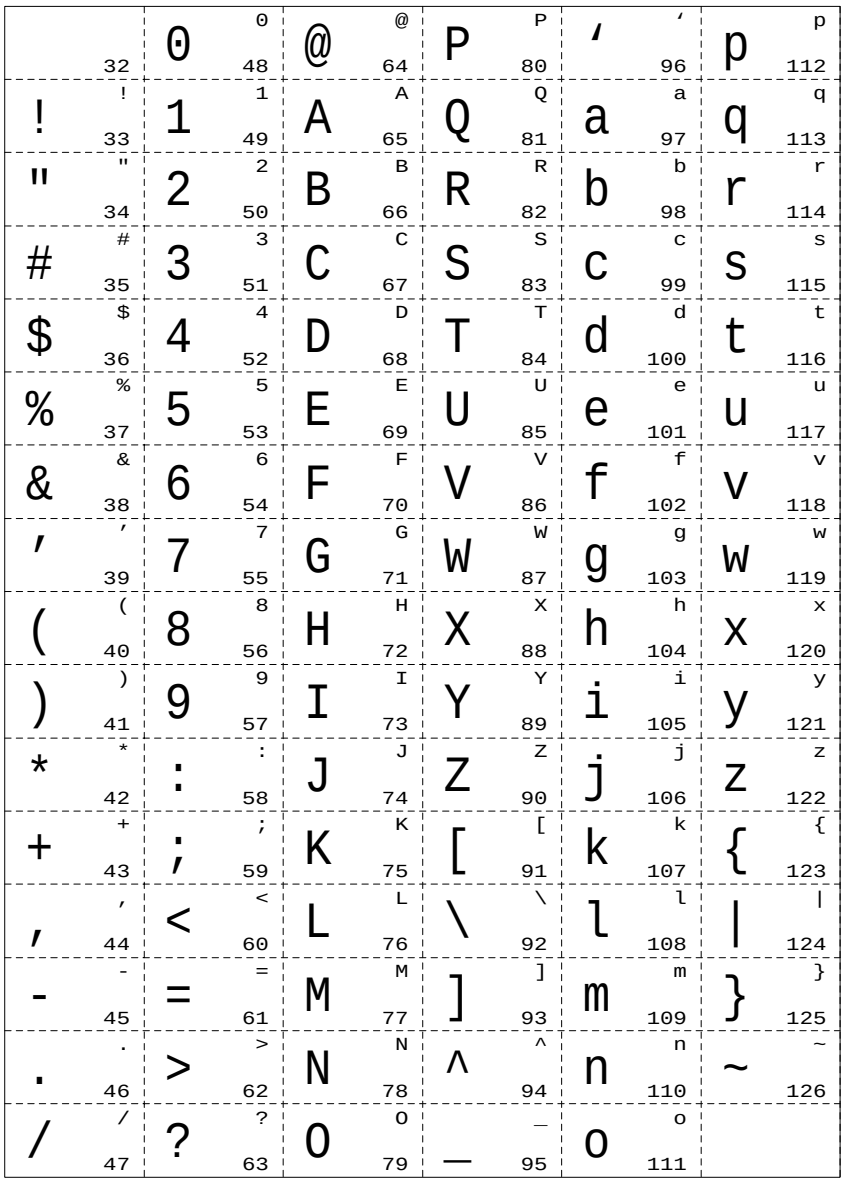


Figure G.4 `CHSET(0)` – Hardware

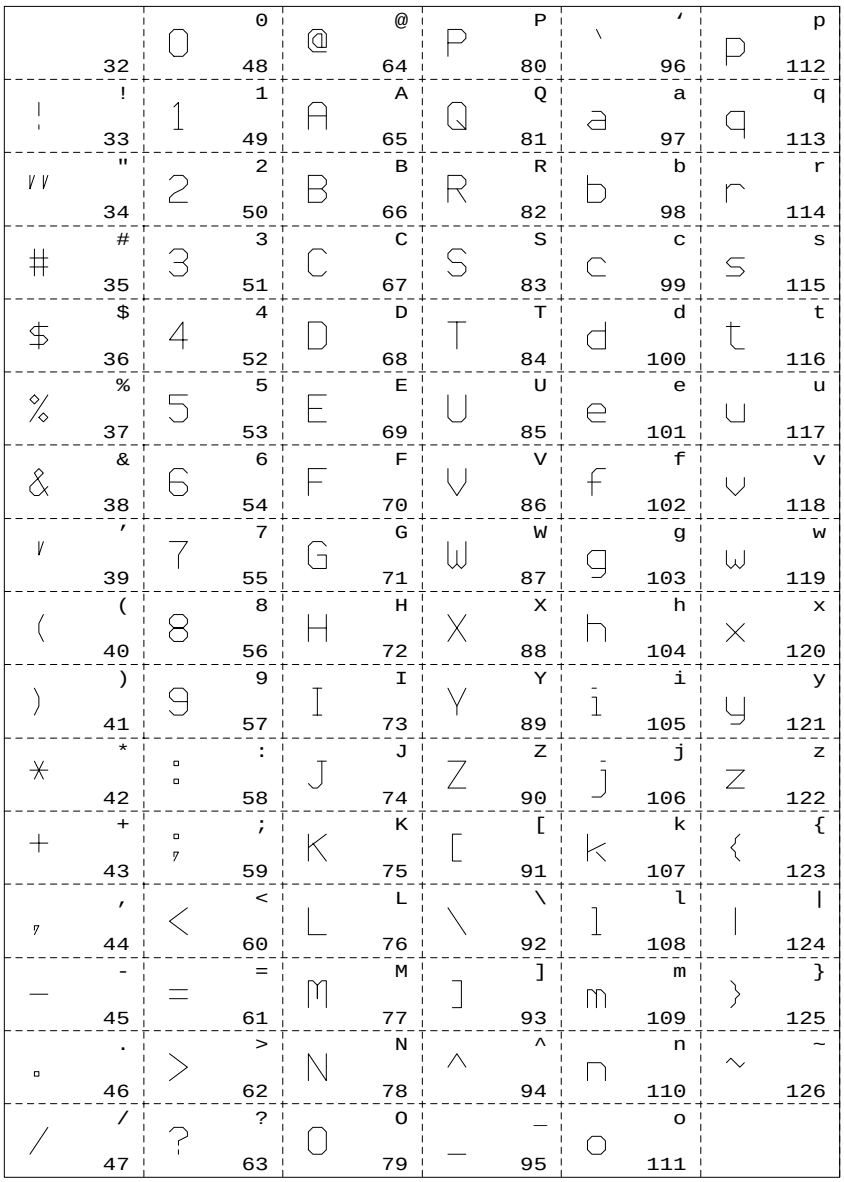


Figure G.5 `CHSET(1)` – Software

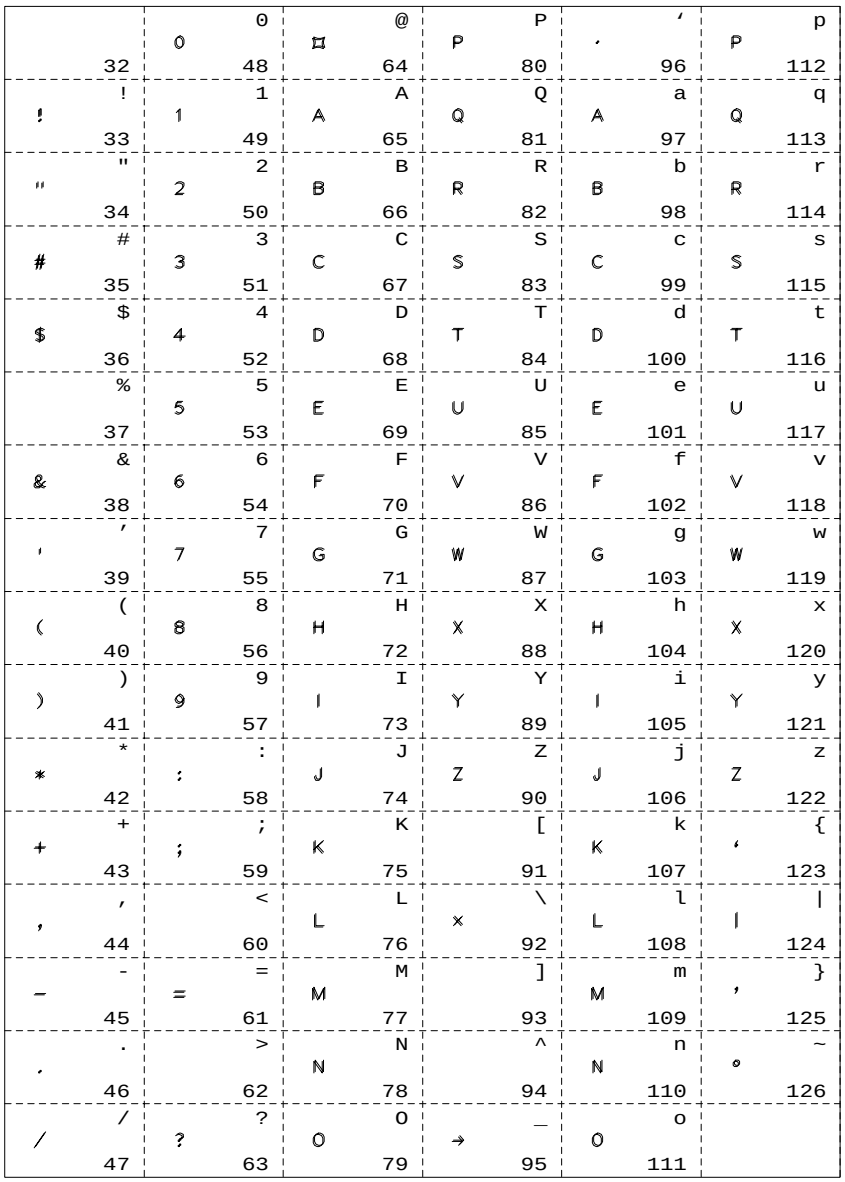


Figure G.6 `CHSET(2)` – Cartographic

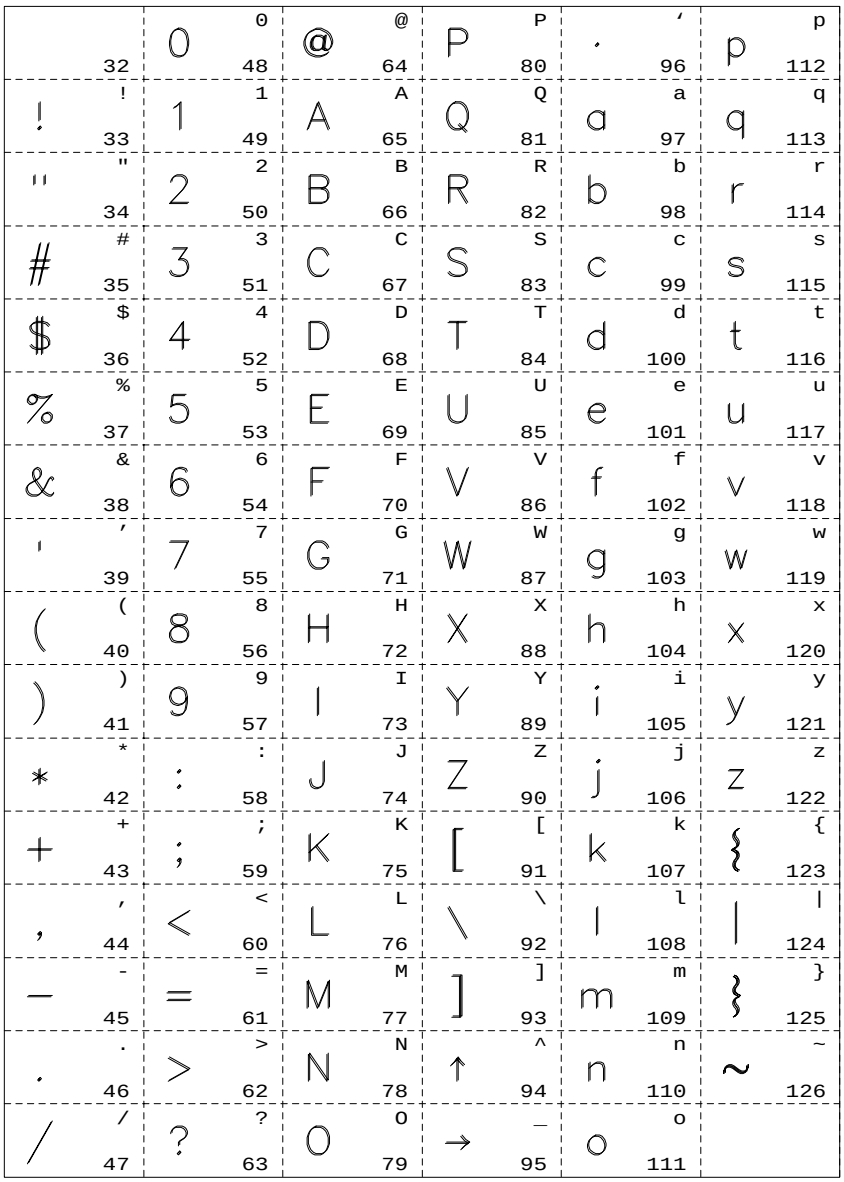


Figure G.7 `CHSET(3)` – Simplex Roman

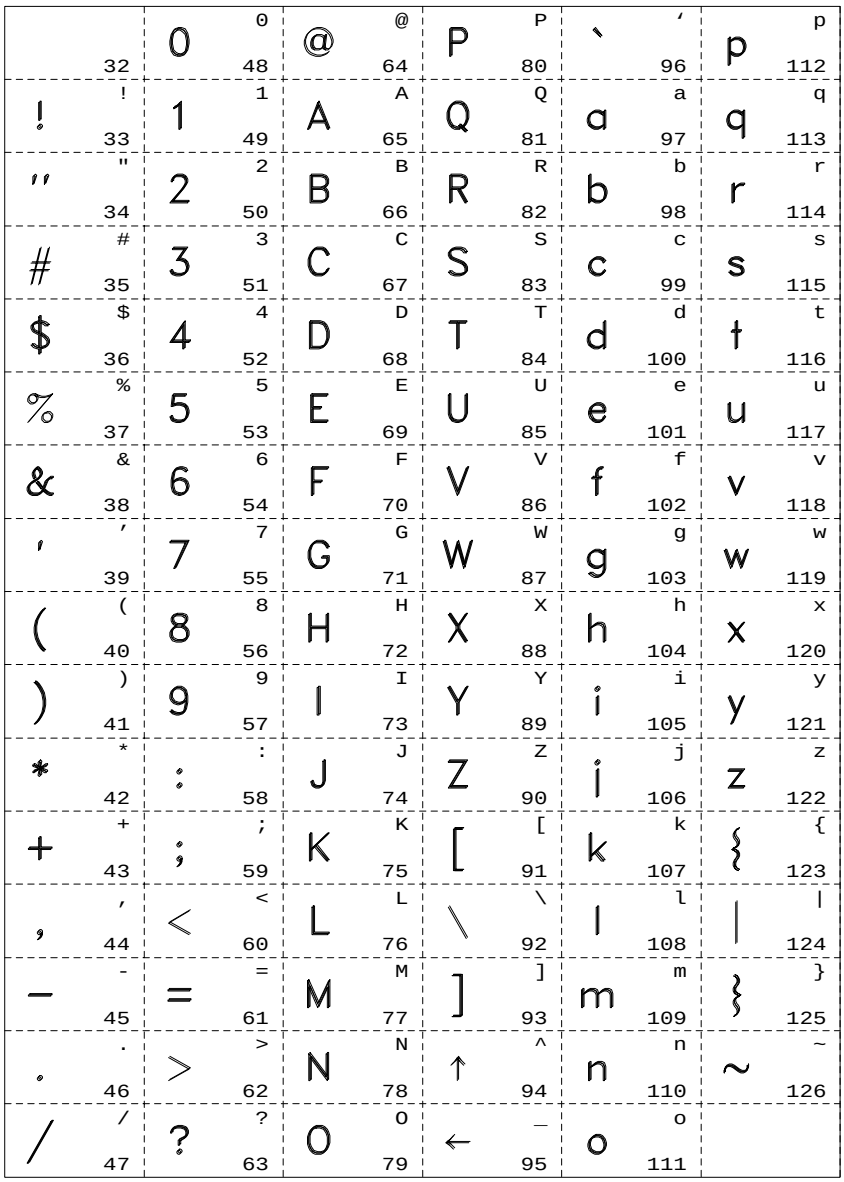


Figure G.8 `CHSET(4)` – Duplex Roman

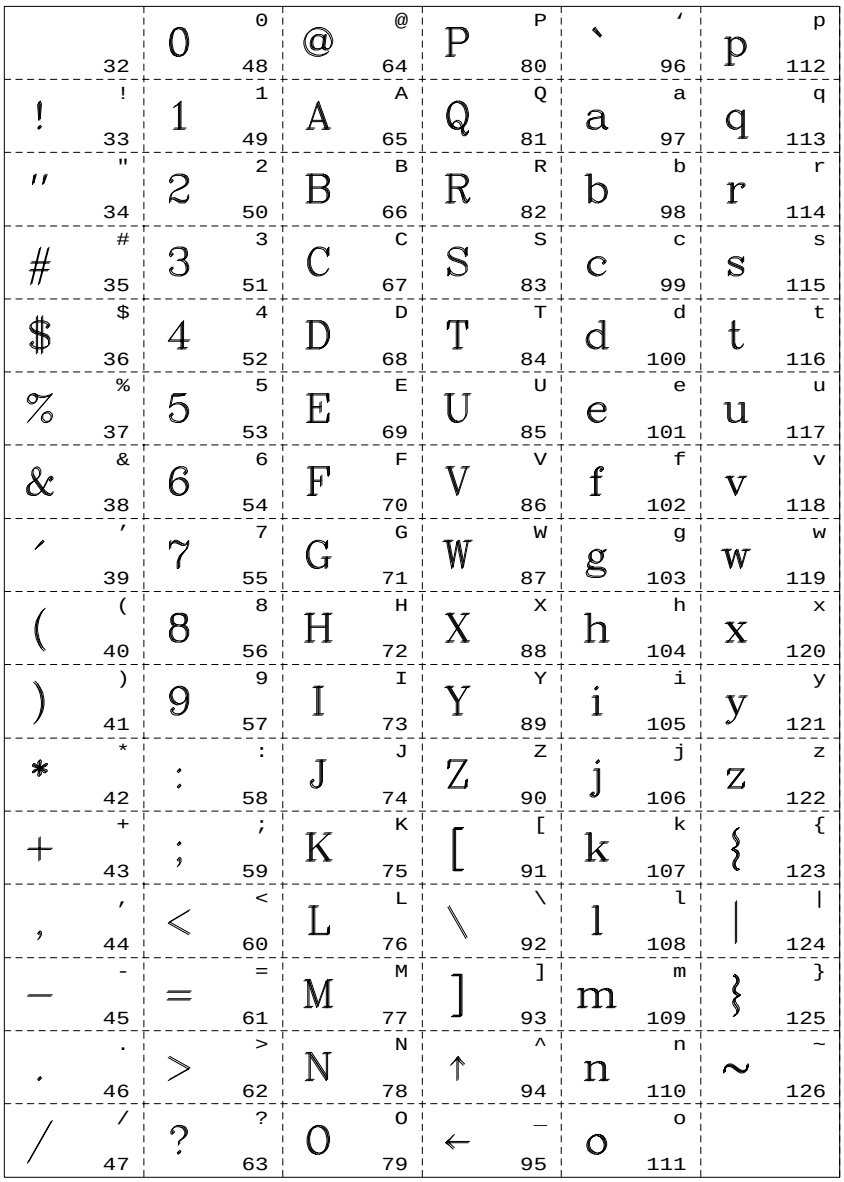


Figure G.9 `CHSET(5)` – Complex Roman

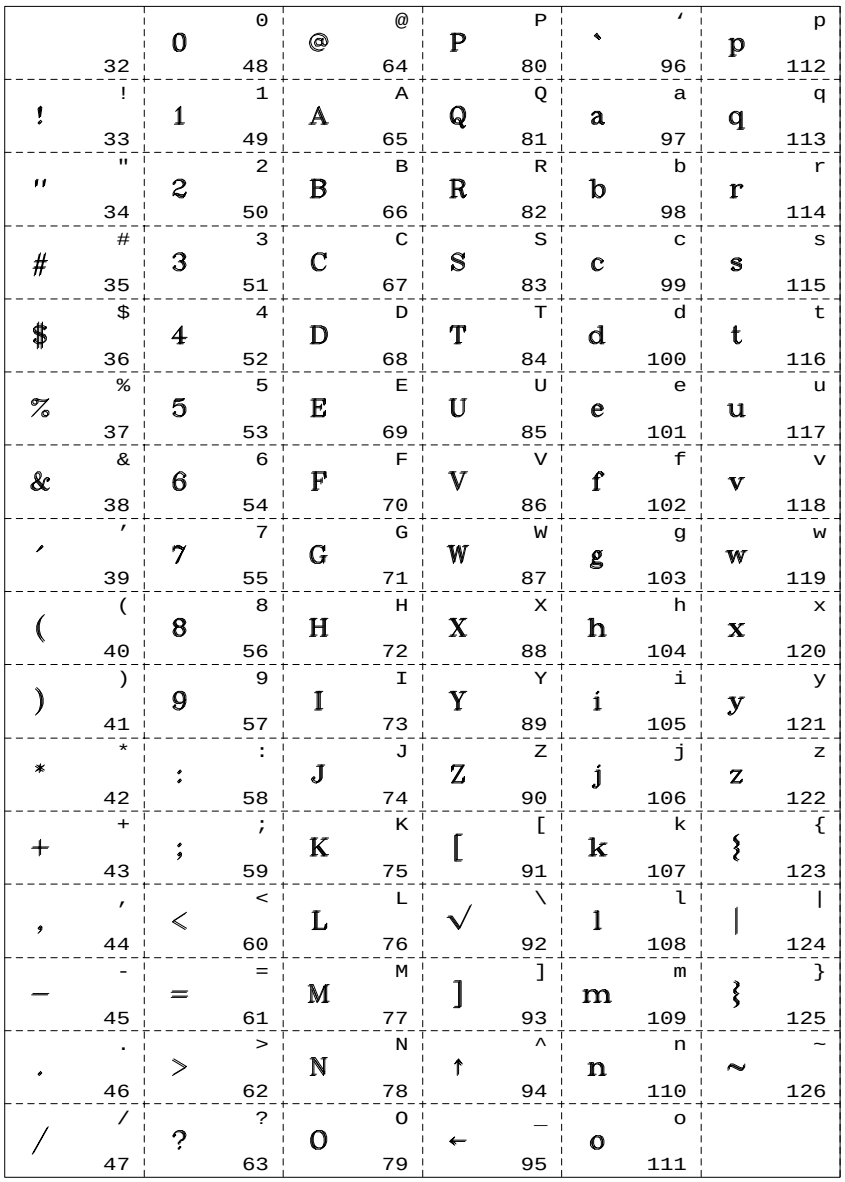


Figure G.10 `CHSET(6)` – Small Complex Roman

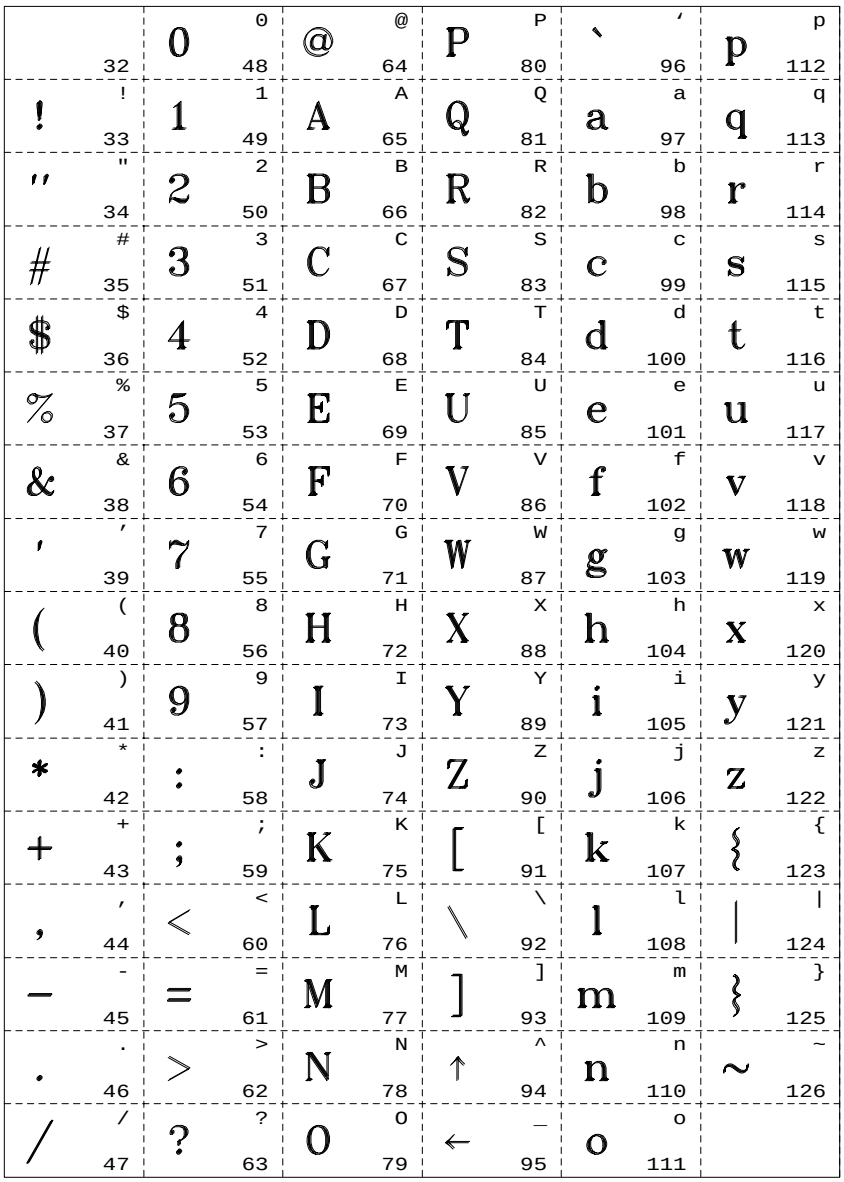


Figure G.11 `CHSET(7)` – Triplex Roman

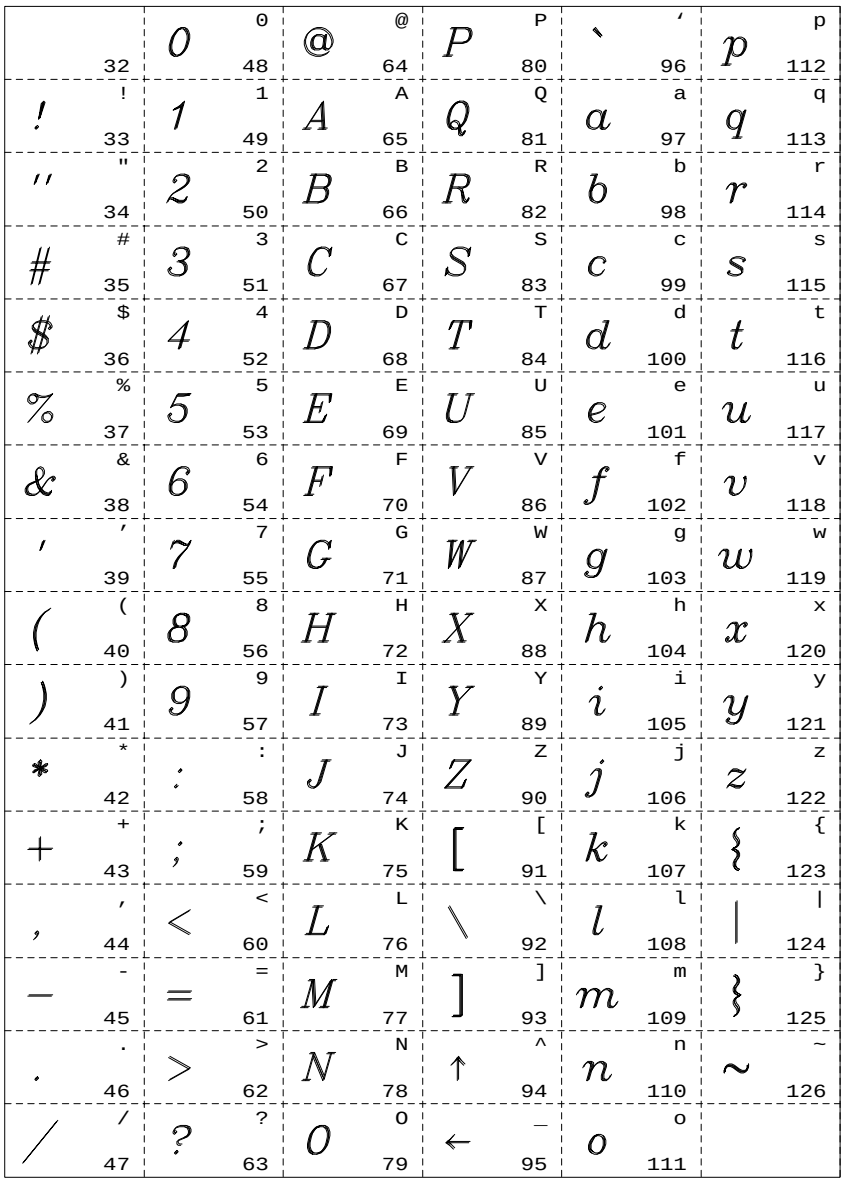


Figure G.12 `CHSET(8)` – Complex Italic

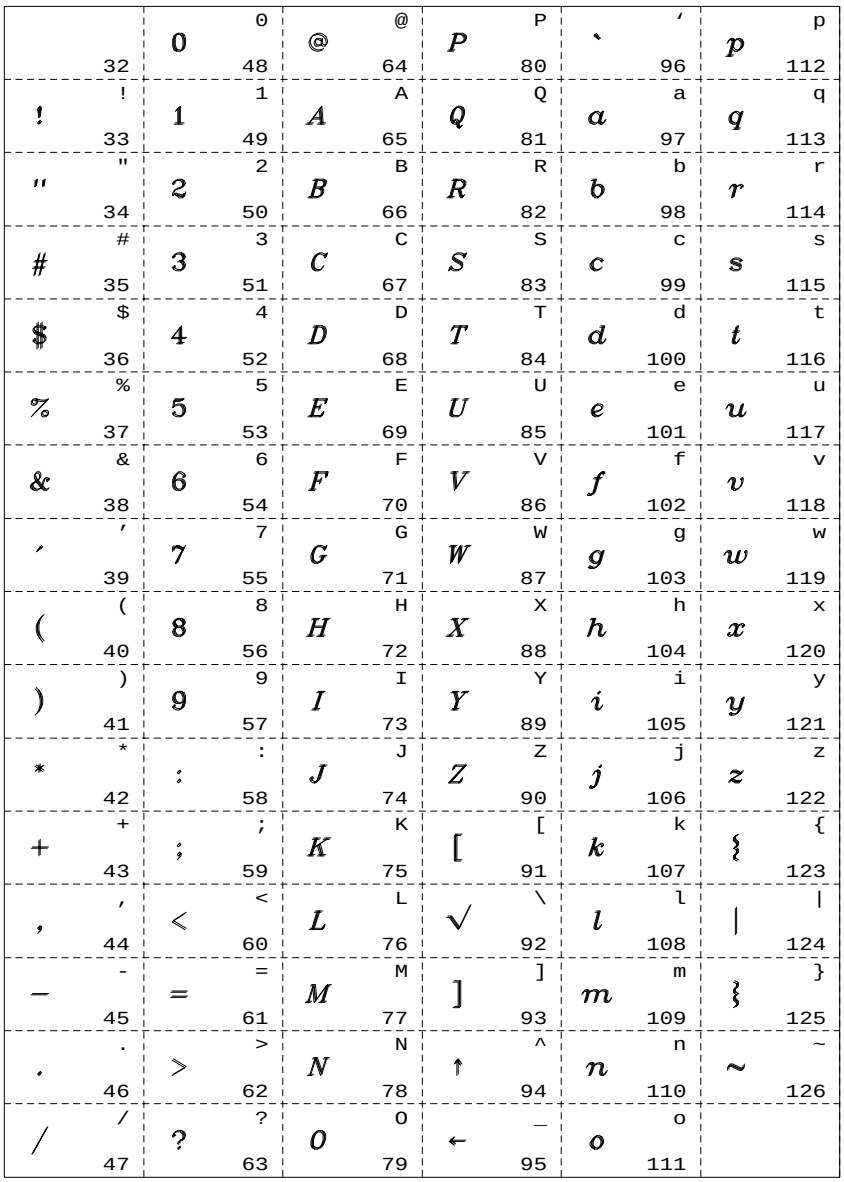


Figure G.13 `CHSET(9)` – Small Complex Italic

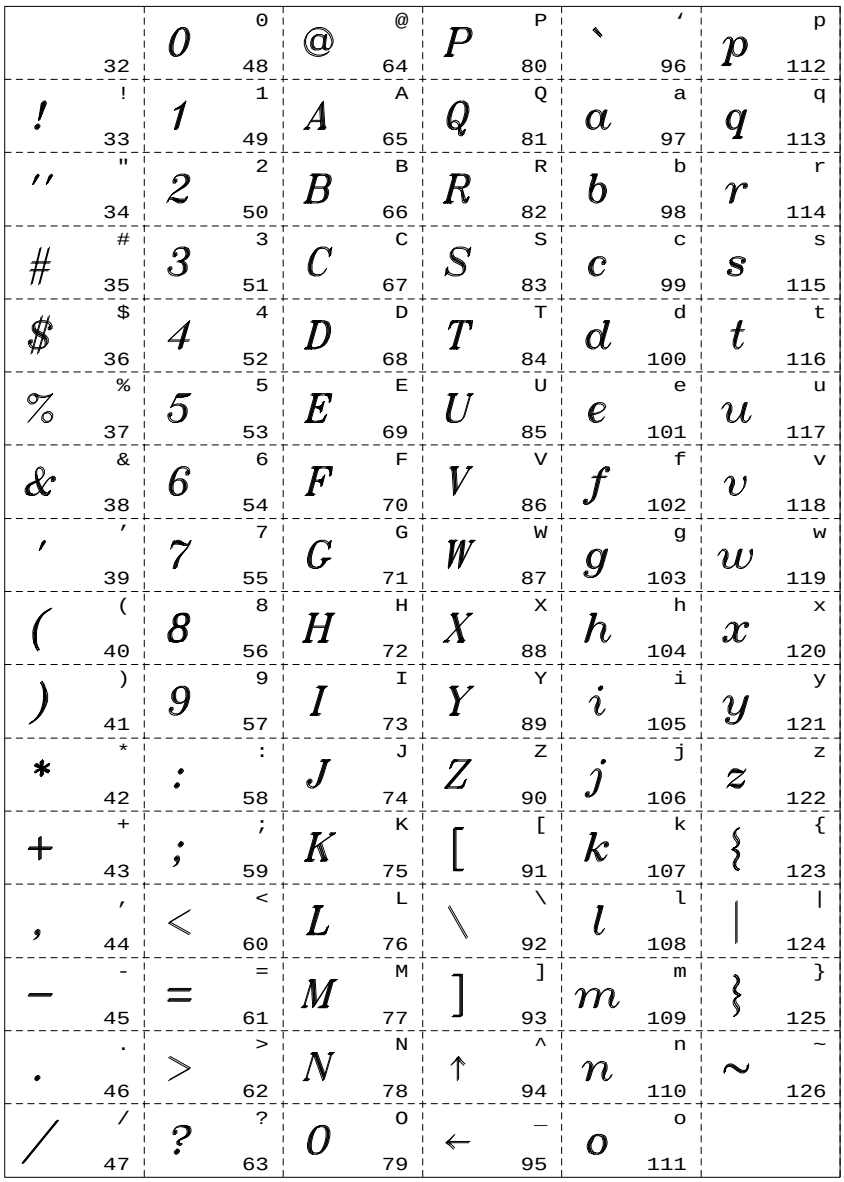


Figure G.14 `CHSET(10)` – Triplex Italic

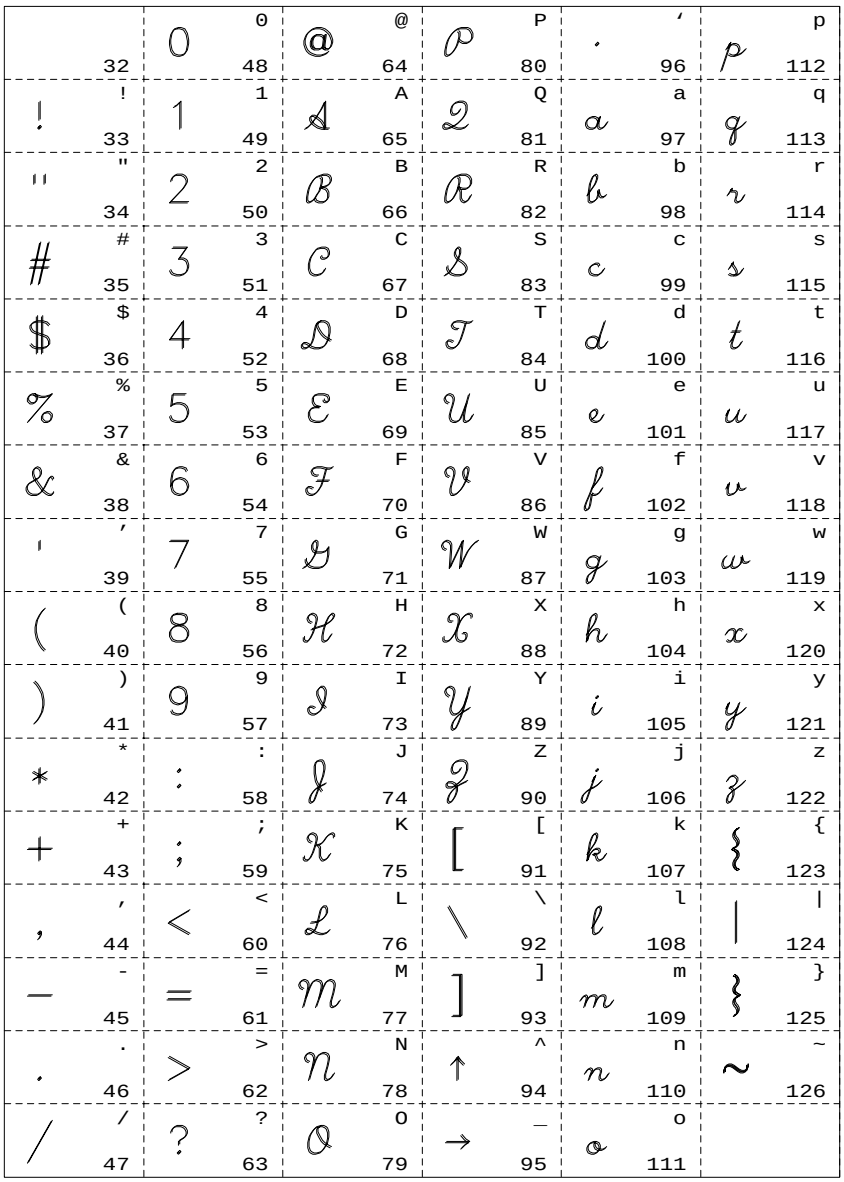


Figure G.15 `CHSET(11)` – Simplex Script

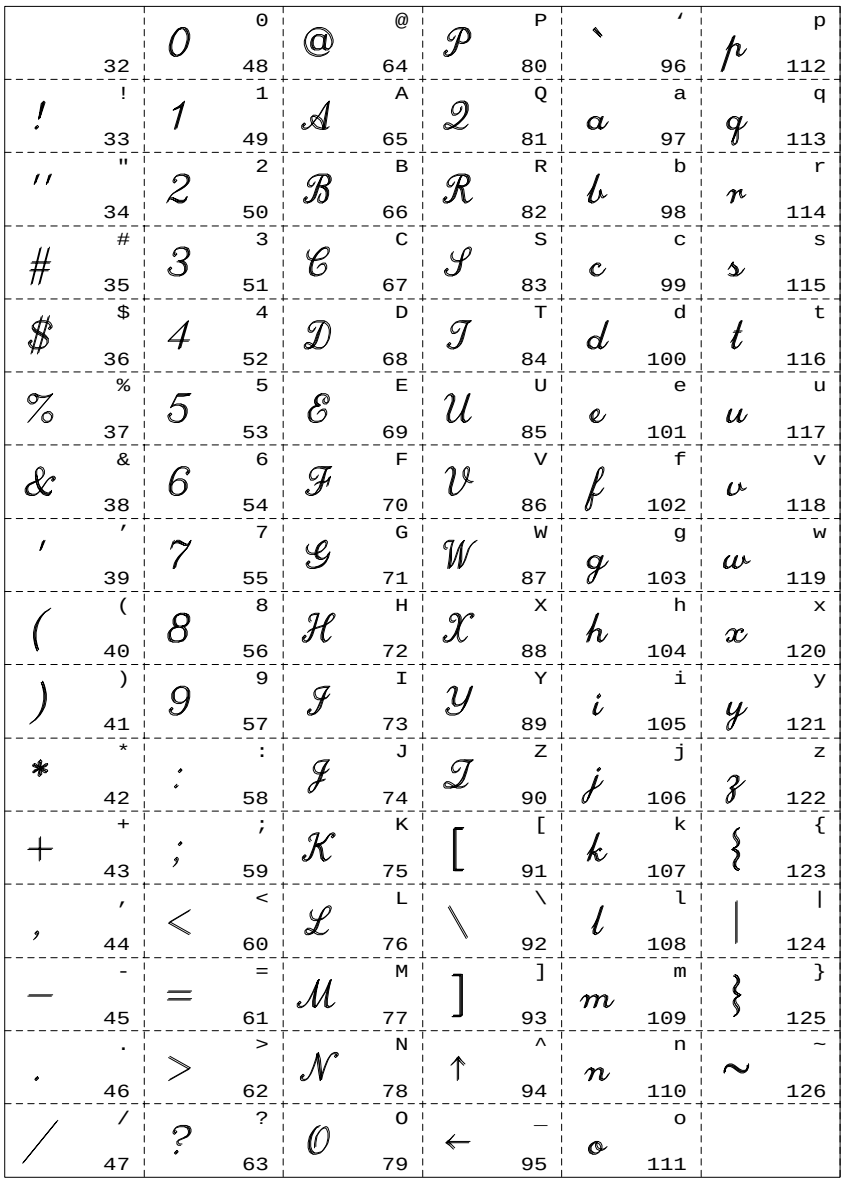


Figure G.16 `CHSET(12)` – Complex Script

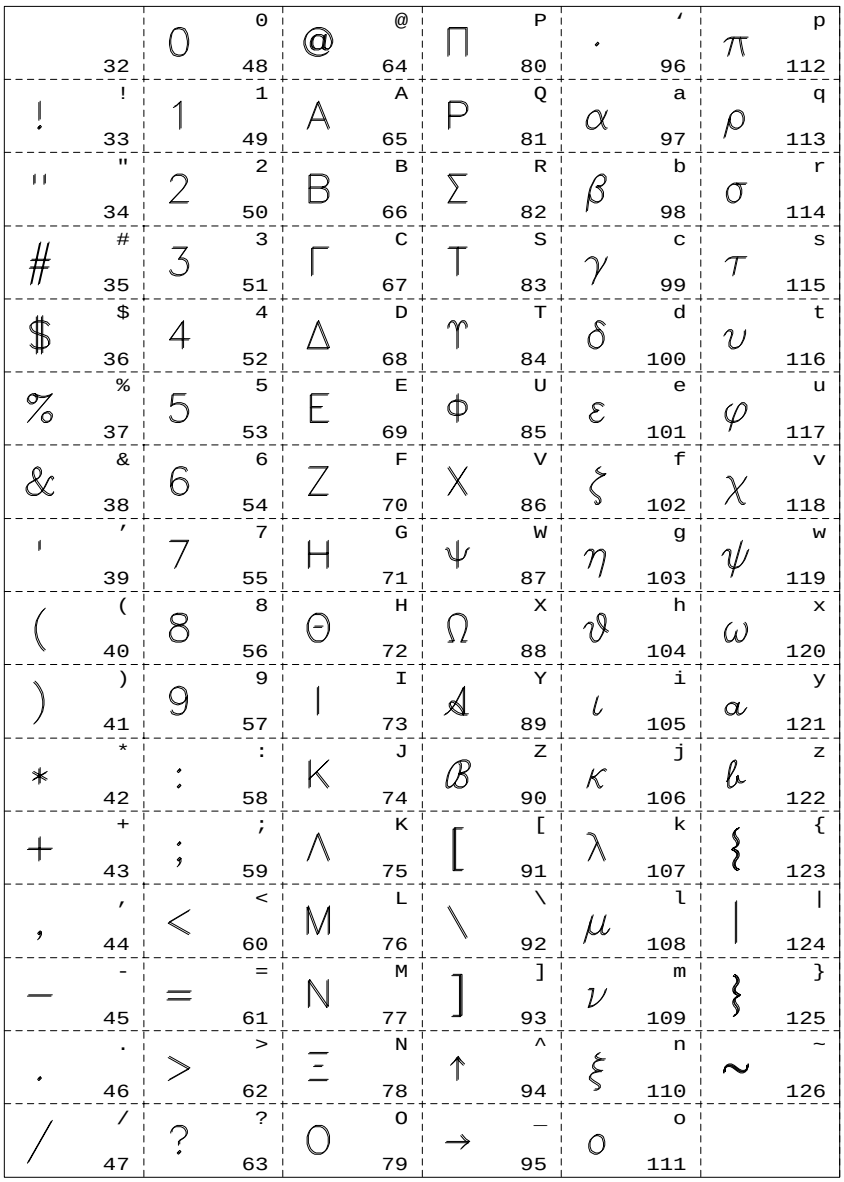


Figure G.17 `CHSET(13)` – Simplex Greek

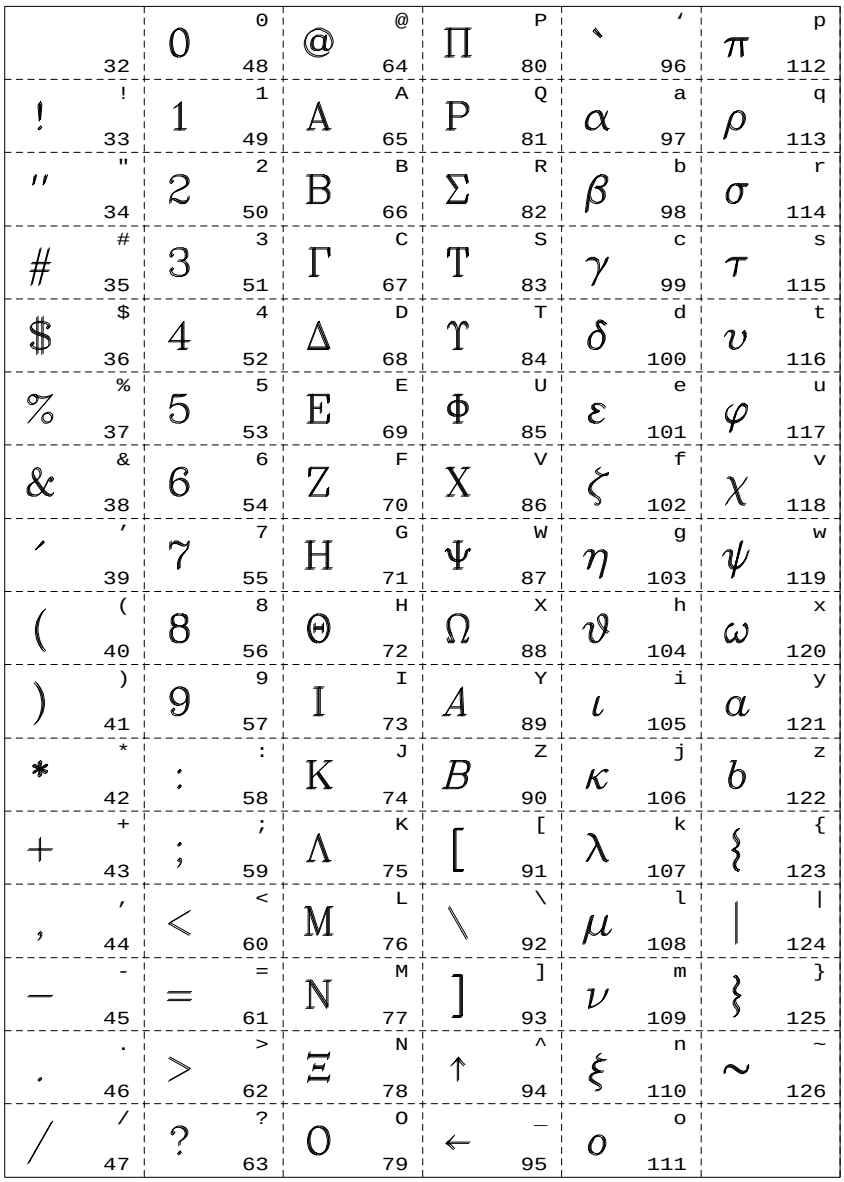


Figure G.18 `CHSET(14)` – Complex Greek

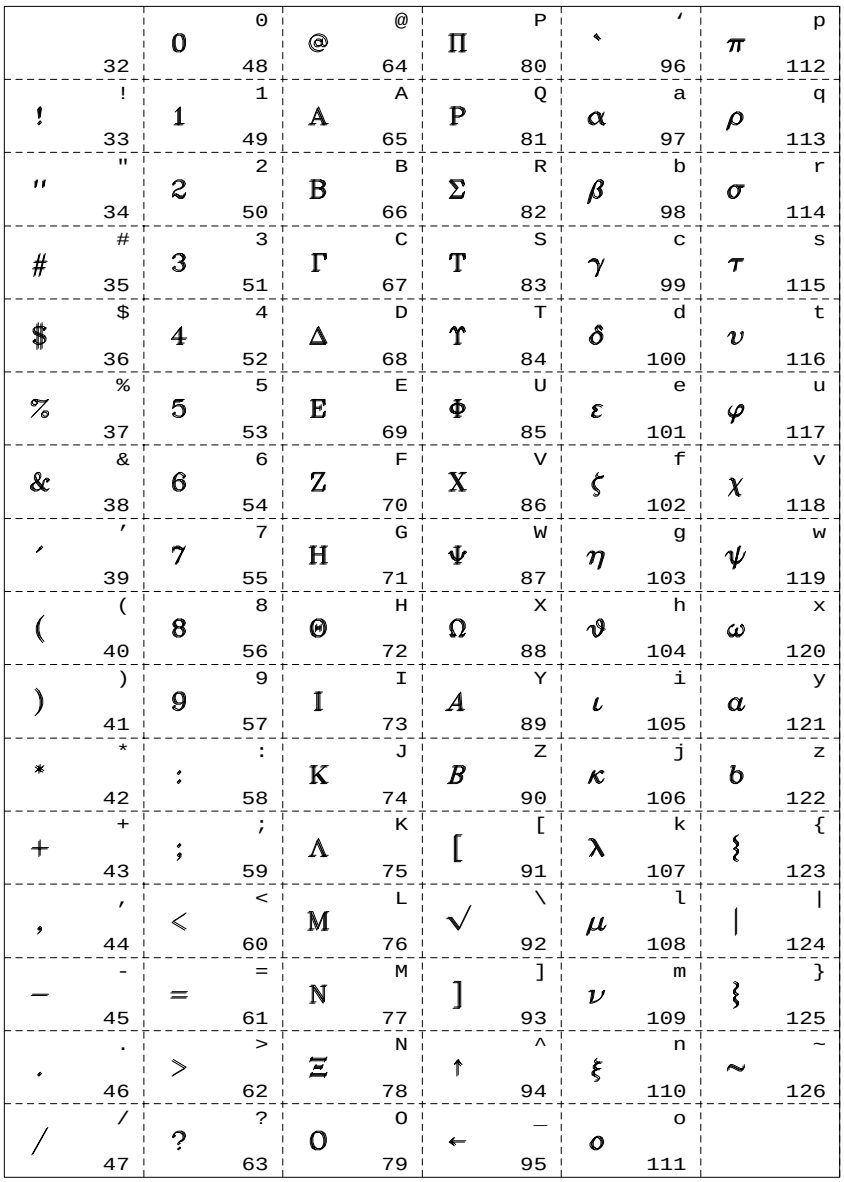


Figure G.19 `CHSET(15)` – Small Complex Greek

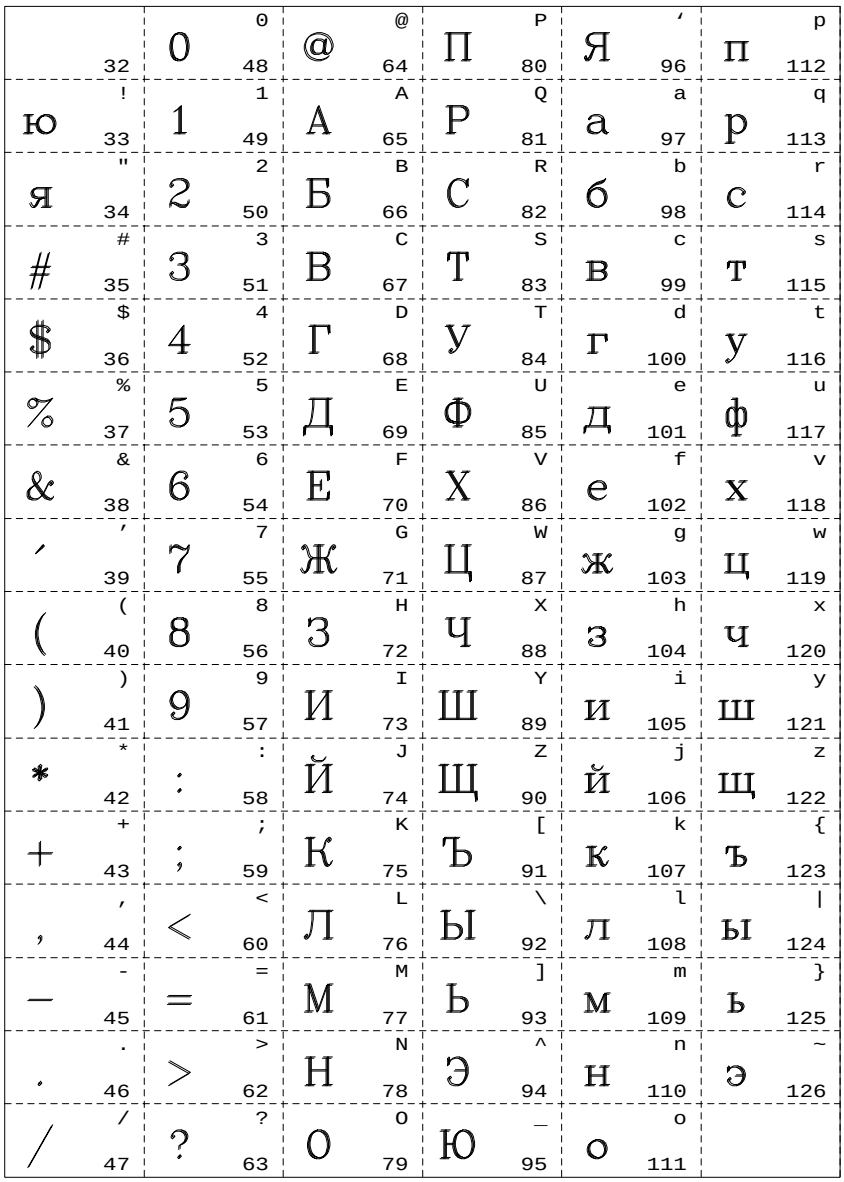


Figure G.20 `CHSET(16)` – Complex Cyrillic

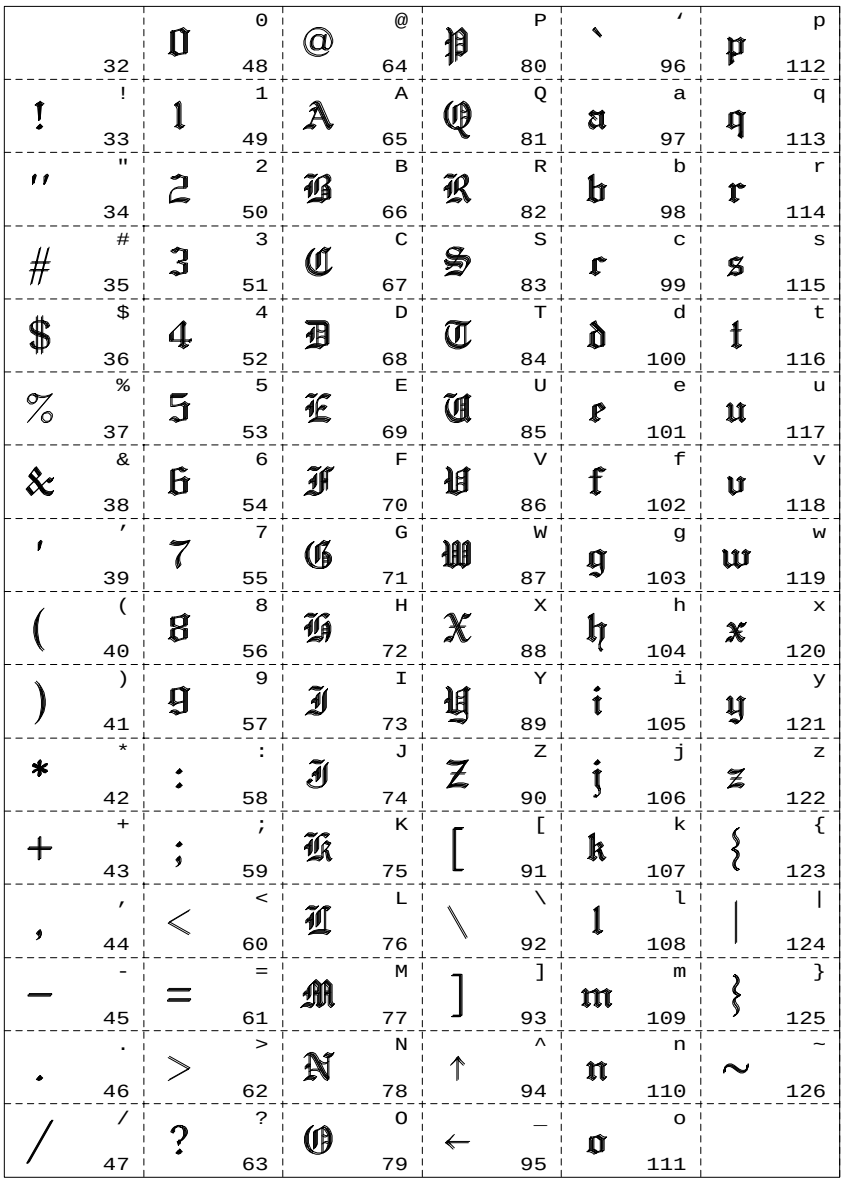


Figure G.21 `CHSET(17)` – English Gothic

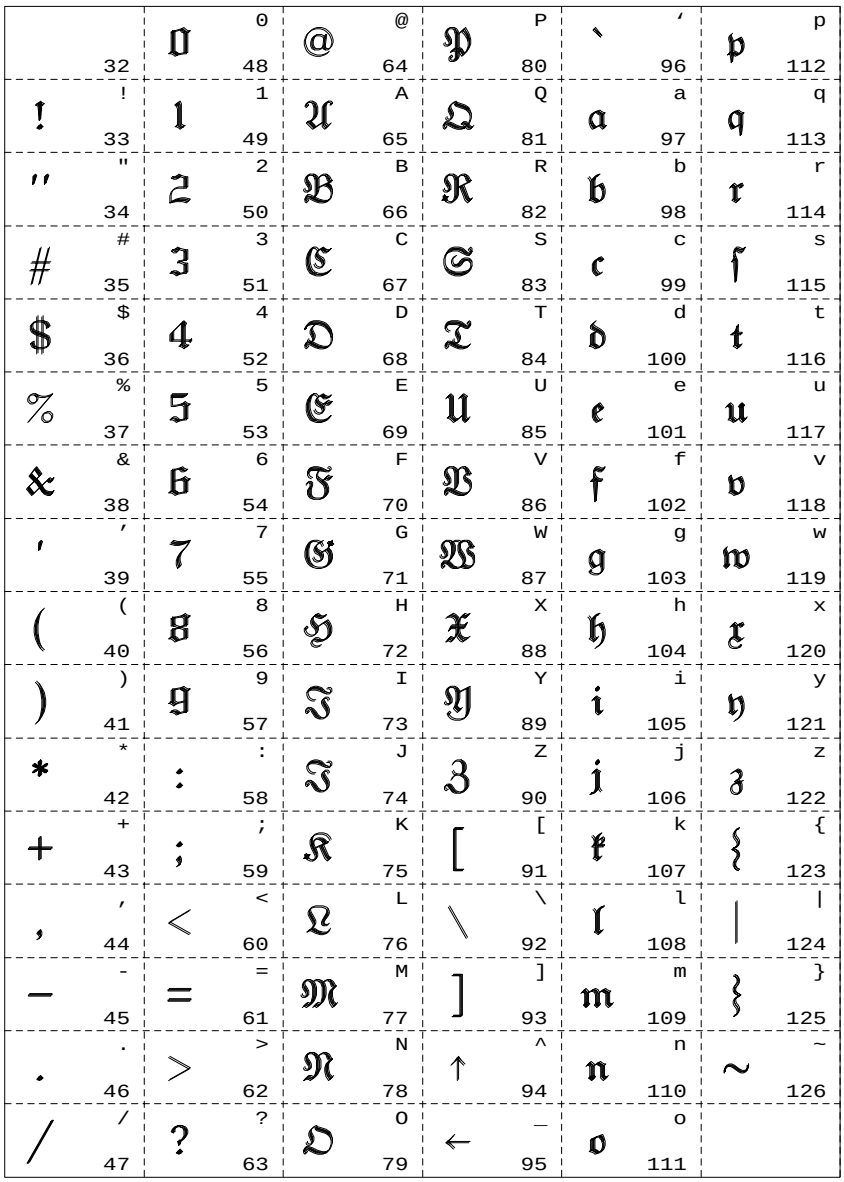


Figure G.22 `CHSET(18)` – German Gothic

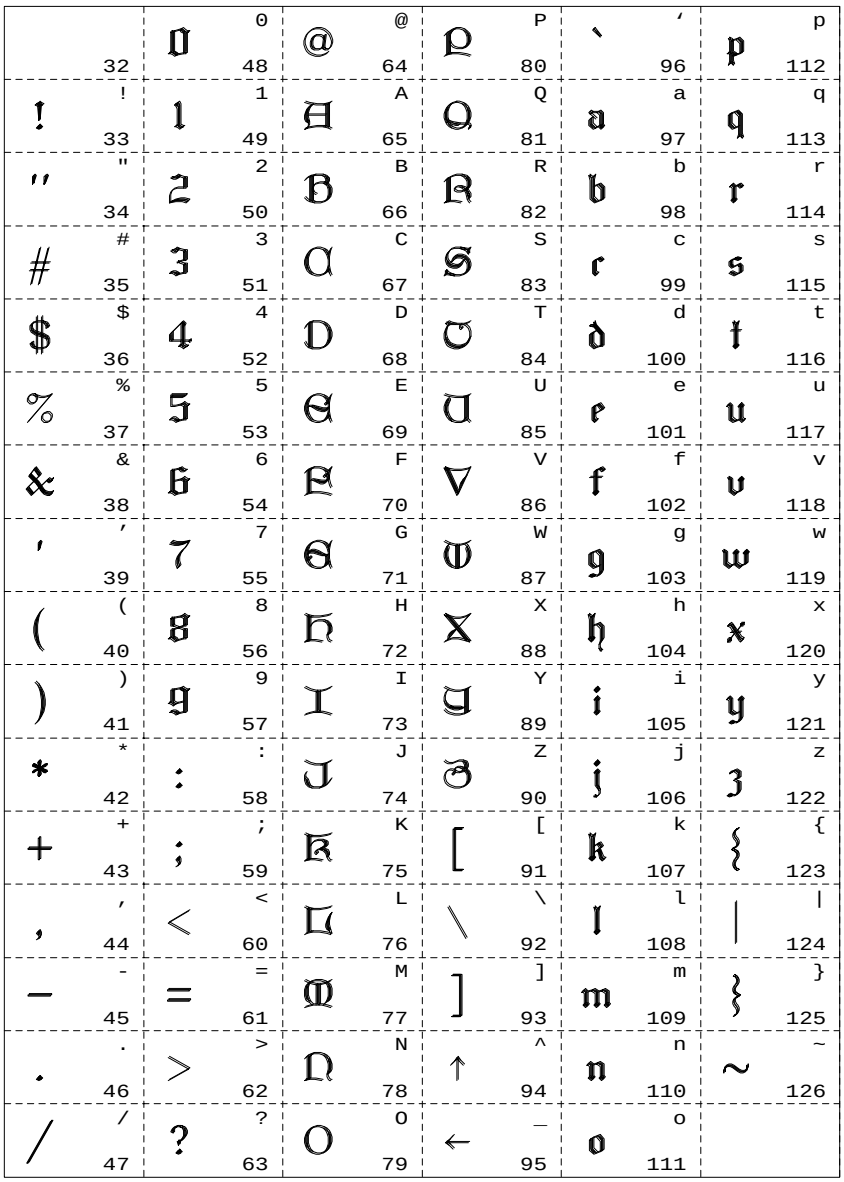


Figure G.23 `CHSET(19)` – Italian Gothic

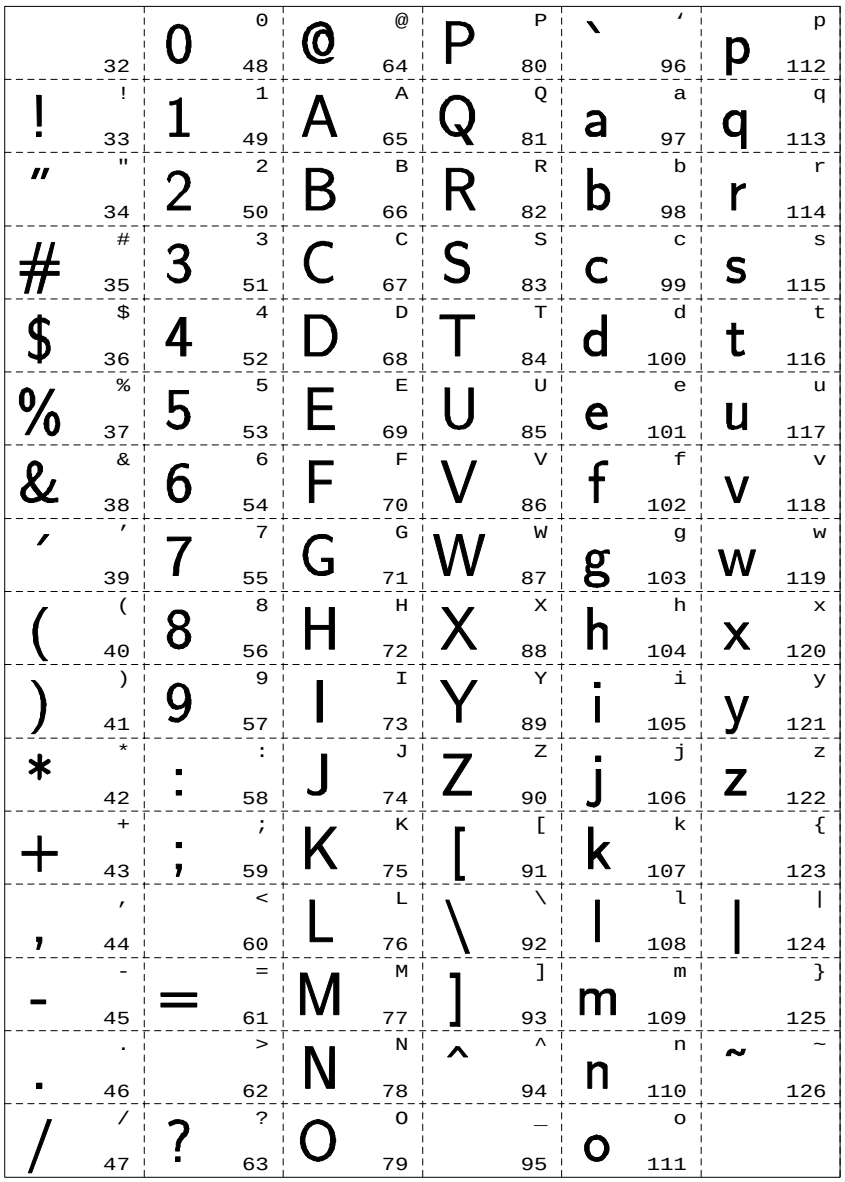


Figure G.24 `CHSET(20)` – Solid

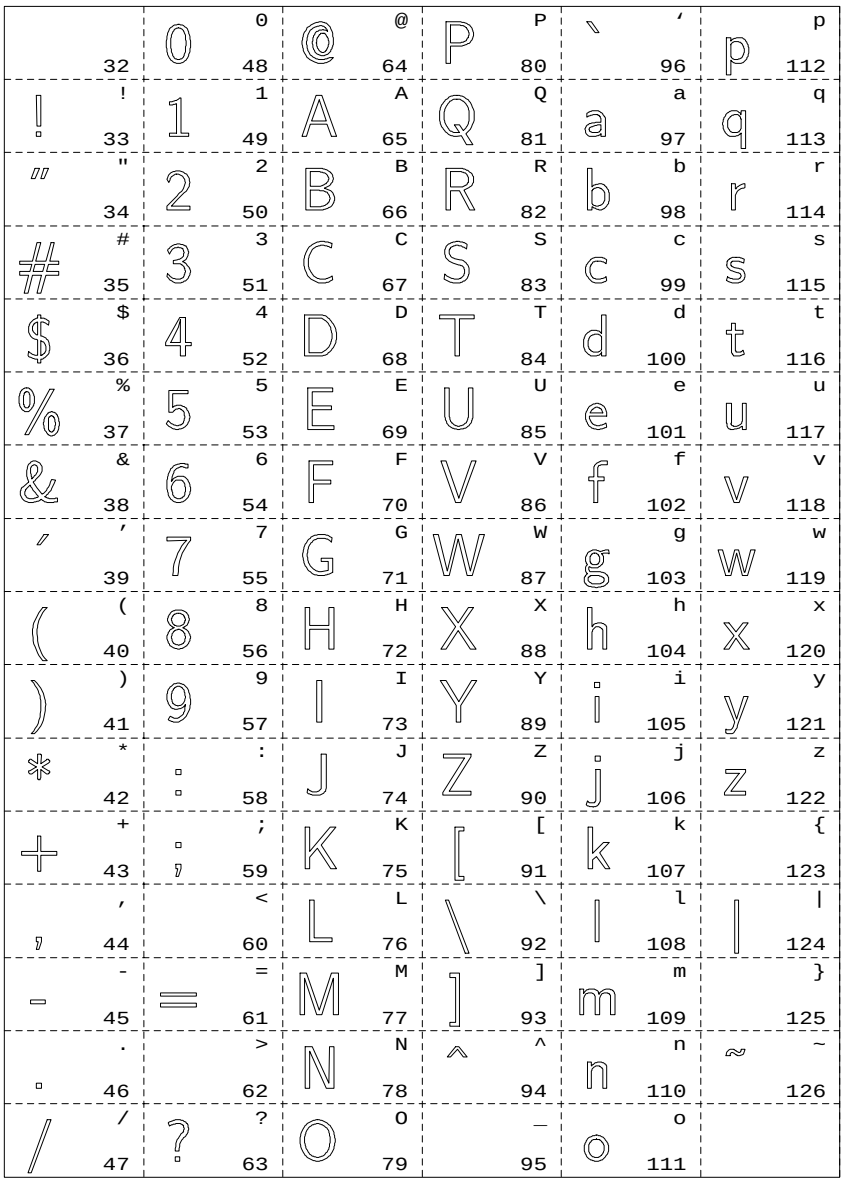


Figure G.25 `CHSET(21)` – Outline

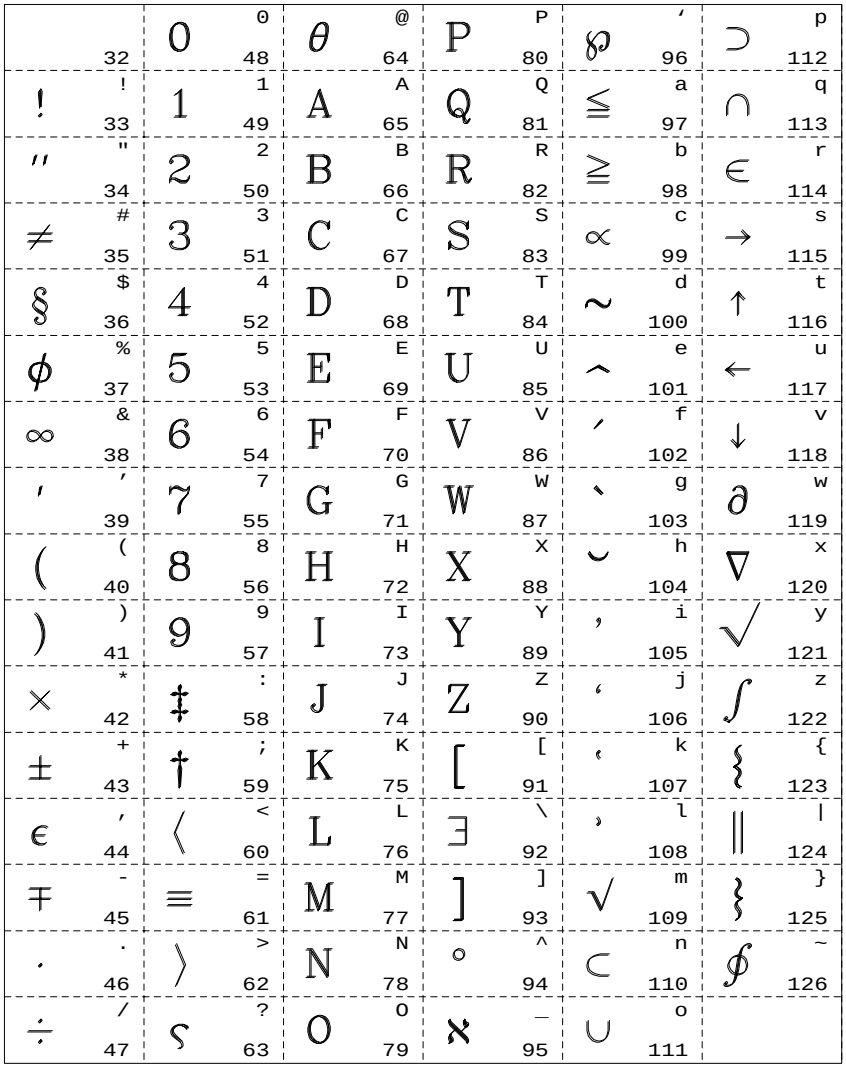


Figure G.26 `CHSET(22)` – Complex Maths

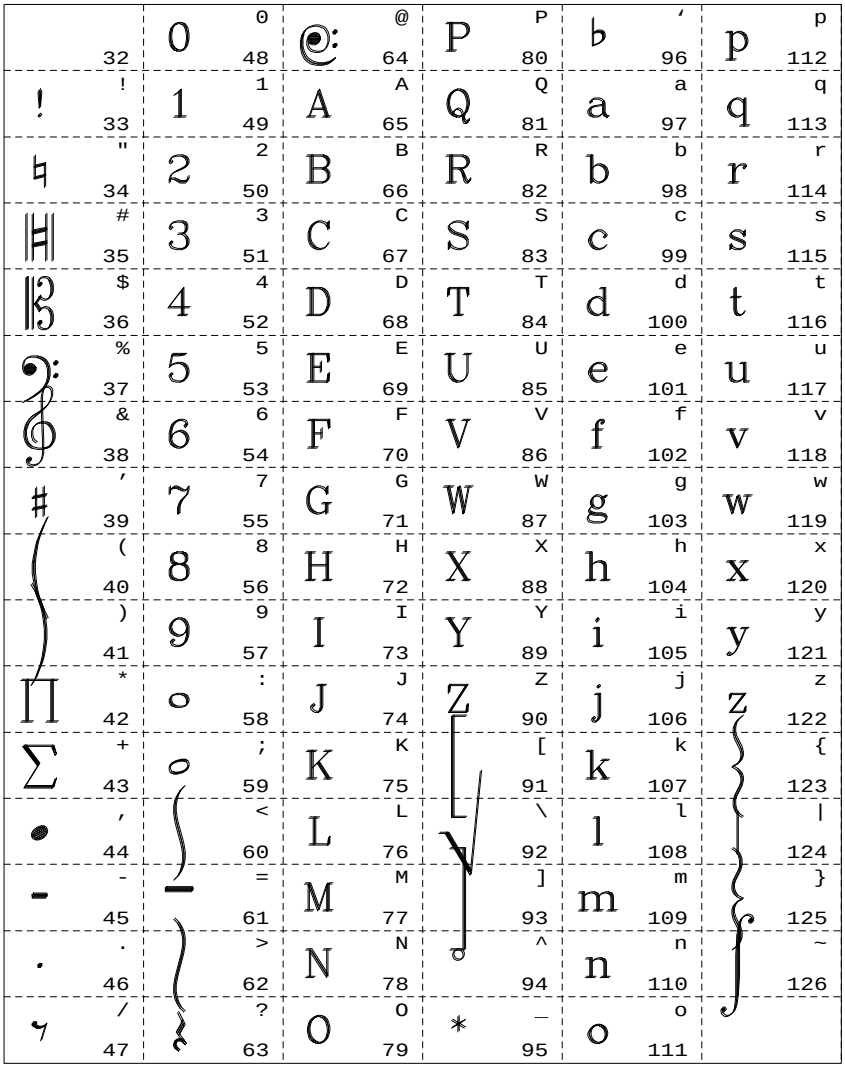


Figure G.27 `CHSET(23)` – Big Complex Maths

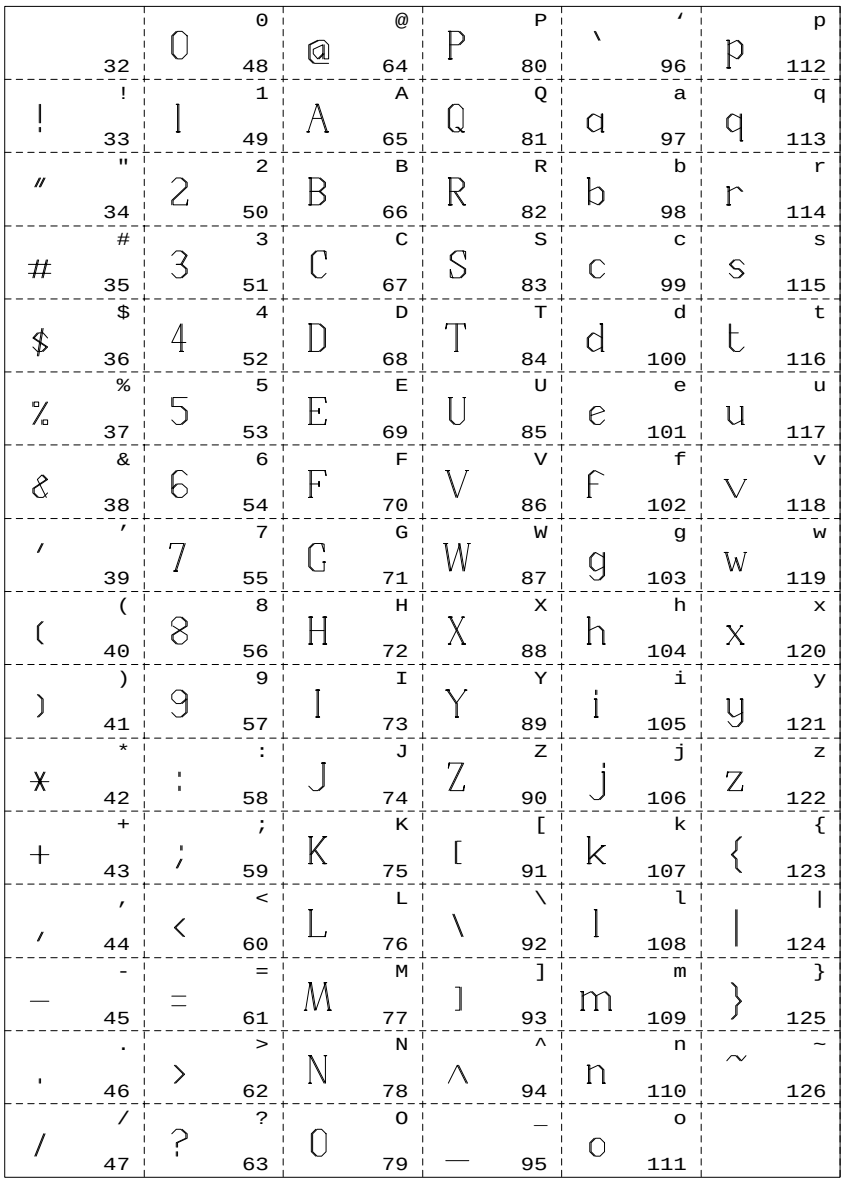


Figure G.28 `CHSET(51)` – Adjustable Ansi (#)

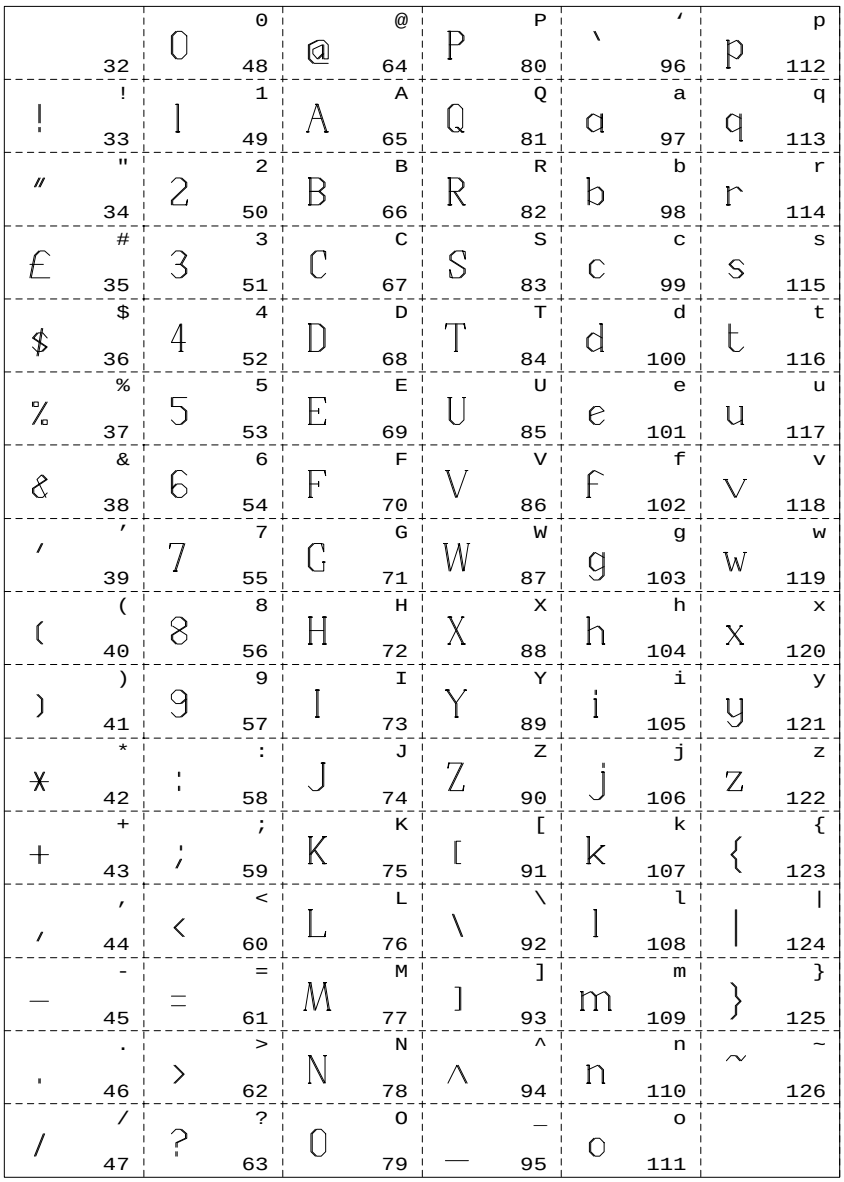


Figure G.29 `CHSET(52)` – Adjustable UK (#)

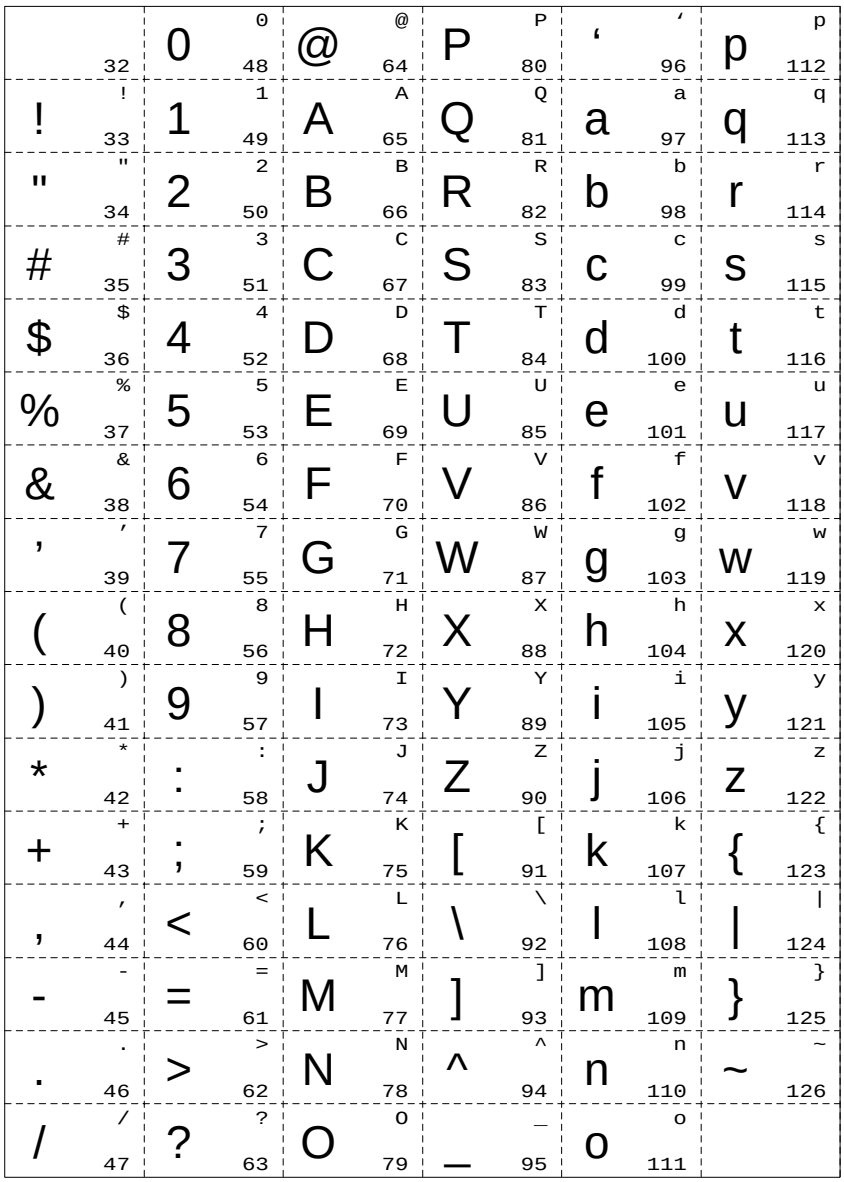


Figure G.30 `CHSET(-9)` – Alternative Hardware Font

G.4 Marker symbols

The standard SIMPLEPLOT marker symbols are identified in the range 0–16 and individual Hershey marker symbols can be selected from the range 17–96. A full range of software symbols is given in Figure G.31.

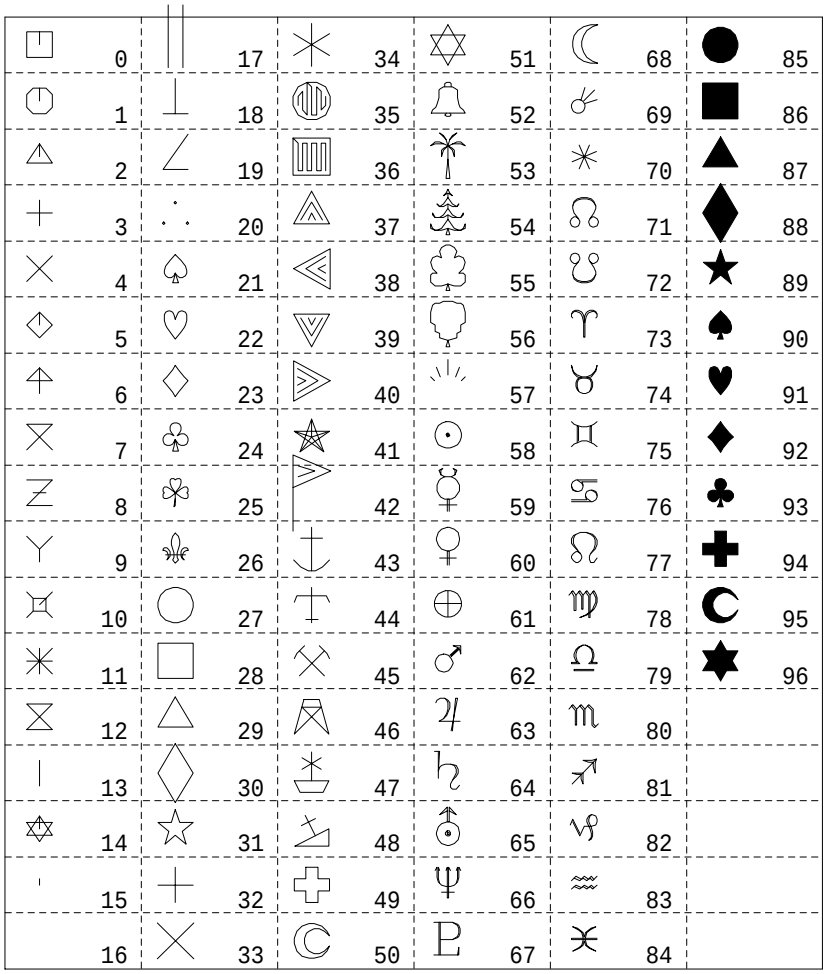


Figure G.31 SIMPLEPLOT marker symbols

L. Drawing logos with BOXPG2

By default, `CALL BOXPG2(2)` initiates page boundaries with the BUSS logo. Alternative formats can be provided *via* external data which are read from an online file. The radius of curvature, smoothness of the curved corners and thickness of the boundary line may all be specified, and an encoded logo may be placed in one of the corners in a break in the boundary.

The shape of the logo must be encoded as one or more curves and/or shaded areas. The curves and shaded areas must be described as a set of positive integer (x, y) coordinates, with each new start flagged by sign conventions on the coordinates.

To prepare logo data, sketch the shape(s) on squared paper, and read off integer coordinates to be joined by straight lines or circular arcs. The integers must be positive to accommodate the sign convention (see below), but any magnitude can be used. The same units must be used in x and y.

The size of the drawn logo is independent of the units in which it is encoded. A scaling factor is calculated to derive its drawn size from the page size combined with a controlling variable, also in the file.

L.1 Effect of `CALL BOXPG2(2)`

The Logical Name/Environmental Variable, `SIMPLE$LOGO` on OpenVMS systems, is examined. When it is defined, its value is treated as the name of a file from which additional details of the boundary are read. There are six free format READs from the file.

1. An `INTEGER` value to specify the number of steps to be used for the curve round each corner of the boundary line. Negative and zero values give square corners.
2. A `REAL` value between 0.0 and 0.5 to specify the curvature of the corners of the boundary. The radius of circular arcs is calculated by multiplying the smaller dimension of the page by the supplied value. Negative and zero values give square corners. Values greater than 0.5 are treated as 0.5, giving the maximum usable radius.
3. A `REAL` value between 0.0 and 1.0 to specify the thickness of border line in centimetres. Thickening is achieved by drawing multiple lines offset from each other. Negative and zero values give a single unthickened line. Values greater than 1.0 are treated as 1.0, but the thickness is not allowed to exceed half of the radius calculated from *2*.
4. A `REAL` value between 0.0 and 1.0 to specify the drawn logo size, *eg.* 0.1. The scaling factor for drawing the logo as large as possible without overlapping the curved corners is calculated, and multiplied by the supplied value. Negative and zero values give no logo. Values greater than 1.0 are treated as 1.0, giving the maximum possible size within the page.
5. An `INTEGER` value combining the number of (x, y) pairs with a coded indication of the corner to be used. The limiting number of points is 300:

*C*orner	*C*ode
Top Left	number of Pairs+30000
Top Right	−number of Pairs−30000
Bottom Left	number of Pairs
Bottom Right	−number of Pairs

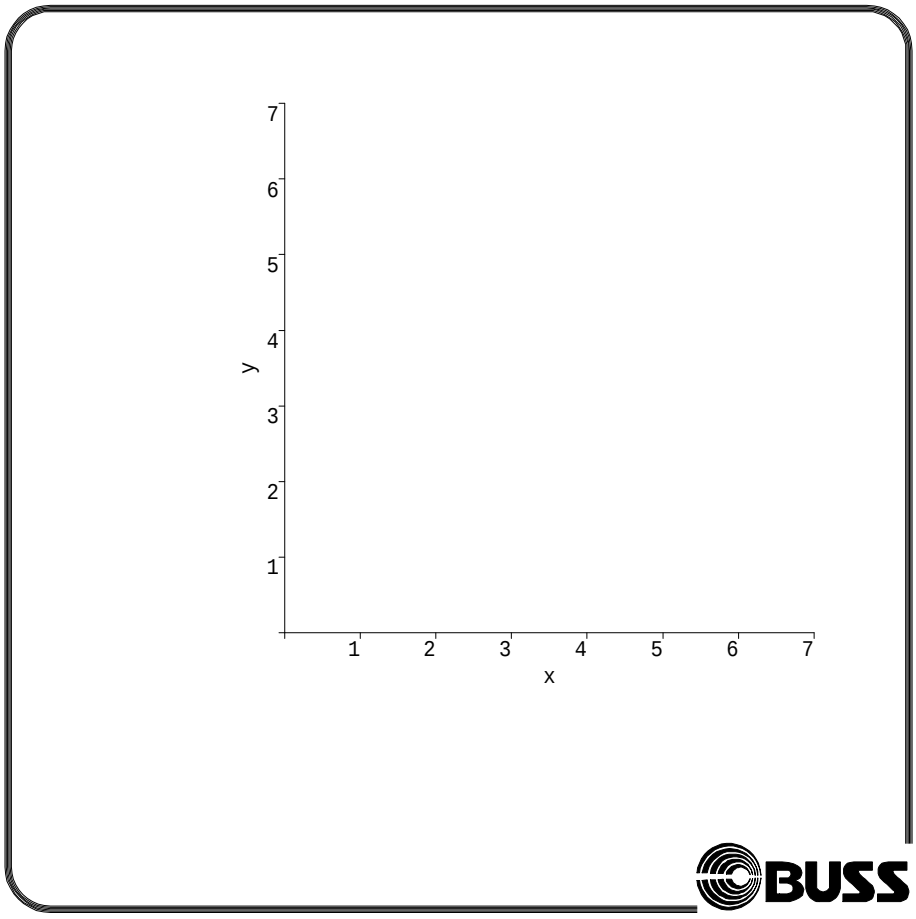
If the number of pairs is smaller than 2, no coordinates are read and no logo is drawn.

6. A sequence of INTEGERs specifying the shape of the logo as (x, y) coordinate pairs. The coordinates must be encoded as positive integers:
   - the x coordinate should be negated to signal the beginning of each curve.
   - both coordinates should be negated to signal the beginning of each shaded area formed by tracing the specified points, and then joining the last point to the first.
   - the y coordinate should be negated to signal the beginning of each closed curve formed by tracing the specified points, and then joining the last point to the first. When the preceding curve was a shaded area, the closed curve represents a hole.
   - a dummy (0,0) point signals that the next three points should be joined by a circular arc with step size controlled by *1*.
   - a dummy (0,1) point signals that the next three points should be combined with the preceding point to define a Bezier Curve drawn with step size controlled by *1*.

The logo is drawn in the requested corner, and the boundary line extends from near the logo, curves round the other three corners of the page, and back to the other side of the logo.

If the logical name is not defined, or some problem prevents the reading of the whole file, the BUSS logo is drawn. When the logo is null, the boundary line curves round all four corners.

Figure L.1 demonstrates the use of BOXPG2.



```
PROGRAM LOGO
CALL BOXPG2(2)              ! Specify logo (default = BUSS)
CALL AXES7('x', 'y')        ! Draw axes and start new picture
CALL ENDPLT                 ! Terminate plotting
END
```

Example 7. BUSS logo

M. Diagnostic Messages

SIMPLEPLOT issues diagnostic messages to provide information about progress, to warn of omissions and also for fatal errors. For example, a typical execution of a SIMPLEPLOT program which draws two pictures on separate pages, might produce the following diagnostics:

```
(SIMPLEPLOT Mark 2-16(000)F)
(DEVICE OPENED: LN03R)
(END OF PICTURE)
(END OF PICTURE)
(DEVICE CLOSED)
(SIMPLEPLOT CLOSED)
```

Please refer to Appendix H of the *SIMPLEPLOT Reference manual* for host specific information about how to control the destination of diagnostic messages.

List of Messages

The following list includes all diagnostic messages which can be produced by SIMPLEPLOT-PLUS subroutines at 2-16, and an explanation of why they occur. The number given after each message indicates the type of message, and therefore at which diagnostic level it is output.

(**x, y**)[2] – the (x, y) coordinates of a point to which plotting has been omitted.

(**x_1, y_1, x_2, y_2**)[2] – the coordinates of two points, (x_1, y_1) and (x_2, y_2), between which plotting has been omitted.

(>`999 omissions, no active page`)[1] – more then 999 omissions have accumulated before a page has been started.

(>`999 omissions, no active picture`)[1] – more then 999 omissions have accumulated before a picture has been started.

(>`9999 incomplete page tasks`)[1] – more than 9999 incomplete page tasks have accumulated by the end of the page.

(>`9999 incomplete picture tasks`)[1] – more than 9999 incomplete picture tasks have accumulated by the end of the picture.

(n `omissions, no active page`)[1] – n omissions have accumulated before a page has been started.

(n `omissions, no active picture`)[1] – n omissions have accumulated before a picture has been started.

(v_1 `requested`)[2] – the value, v_1, of a layout parameter requested (by `MARGIN` or `PERIPH`) which has had to be modified.

(v_2 `used`)[2] – the modified value, v_2, of a layout parameter requested (by `MARGIN` or `PERIPH`) which is actually used.

(x_1 y_1 `requested`)[2] – the values, x_1 and y_1, of layout parameters requested (by `PAGE`, `PICPOS` or `PICSIZ`) which have had to be modified.

(x_2 y_2 `used`)[2] – the modified values, x_2 and y_2, of layout parameters requested (by `PAGE`, `PICPOS` or `PICSIZ`) which are actually used.

(`AXLBAN not satisfied`)[2] – `AXLBAN` has been called to request a specific location for axis annotation but, due to layout restrictions, annotations have been placed elsewhere.

(AXLBTM cancelled: *XX*)[1] – AXLBTM has been called to restore the default components for time-date axis annotations.

(AXSBTK not satisfied)[2] – AXSBTK has been called to request a specific location for axis tick marks but, due to layout restrictions, they have been placed elsewhere.

(BAR CHART OMITTED: INVALID DIMENSIONS)[1] – the 2-D data array has not been given valid dimensions.

(BAR CHART OMITTED: NULL RANGE)[1] – all the values in the Z array used to draw a 3-D bar chart have been set to the base z level. This means that the z scale cannot be set..

(Caption truncated)[2] – the caption added to a key or caption area (using ADDCP7, BLNKKY, BOTHK7, FILLK7, LINEK7, MARKK7, PUTCP7 or SHDEK7) is too long to fit within the predefined area and has been truncated.

(Circle exceeds scales)[2] – the extent of a circle (drawn using BRKNCL, DRAWCL or SHDECL) exceeds the current picture scales.

(Constant data: default Z scale used)[1] – WFCHT or the first call of WFDRAW has a set of equal z values. This menas that there is no data to set the scales for each waterfall curve.

(Contour curve not all in range)[2] – the extent of a contour curve (drawn using *CONT) exceeds the current picture scales.

(Contour map not all in range)[2] – the extent of a set of contour curves (drawn using *CNTS) exceeds the current picture scales.

(Curve storage empty)[1] – QCURVE has been called when there are no stored curve coordinates to be retrieved.

(Data curve exceeds scales)[2] – the extent of a curve (drawn using BRKNAR, BRKNCV, DRAWAR, DRAWCV, LABCV7, SHDEA2, SHDEAR, SHDEC2 or SHDECV) exceeds the current picture scales.

(DATA GRID NOT MONOTONIC)[1] – a tartan grid (x, x-y or y) has been specified, or SQZVAL has been called, with non-monotonic values.

(DEVICE CLOSED)[1] – device closed following either a call to ENDPLT to terminate plotting or to DEVNO/DEVNAM or REDRAW to open another device.

(DEVICE OPENED: *device name*)[1] – communication with the selected device (for input or output) has started.

(Element boundary not all in range)[2] – the extent of an element boundary (drawn using ZEDGE or ZZEDGE) exceeds the current picture scales.

(Element not all in range)[2] – the extent of a single element (drawn using ZELEM) exceeds the current picture scales.

(Elements not all in range)[2] – the extent of a set of elements (drawn using ZELEMS or ZZELMS) exceeds the current picture scales.

(END OF GROUP)[1] – GROUP has been called and the specified configuration has been completed. The next picture will be on a new SIMPLEPLOT page.

(END OF PICTURE)[1] – the current picture has been completed either by an explicit call (ENDPIC, ENDPAG or ENDPLT) or because a new picture has been started.

(External character file not available)[1] – the Hershey file cannot be opened.

(External character file unreadable)[1] – the Hershey file was opened but is unreadable.

(Failed to read LOGO file)[1] – BOXPG2 has been unable to read an encoded LOGO file.

(Function curve exceeds scales)[2] – the extent of a function curve (drawn using BRKNFN, DRAWFN or SHDEFN) exceeds the current picture scales.

(Graphics input failed)[1] – GETXY has been called and graphics input has failed.

(Graphics input not available)[1] – GETXY has been called when the currently selected device has no graphics input.

(Grouping discontinued)[1] – GROUP has been called to restore the default picture configuration (one per page).

(HISTOGRAM OMITTED: ZERO INTERVAL)[1] – HSTPLT or HSTCUM has been called with DX = 0.

(`Inappropriate axis type: XX`)[2] – one of the axis subroutines has been called with an inappropriate type of axis. It is not possible to use the subroutine with `CHAXIS`=*XX*.

(`Inappropriate component type: XX`)[2] – `TIMLDA` has been called with an inappropriate type of component, `COMPNT`=*XX*.

(`Insufficient number of valid points`)[1] – `LABCV7` has been called with an array which contains fewer than two valid points (*ie.* not equal to the current no-data value).

(`INVALID ARGUMENT: ARRAY SIZE`)[1] – an array has not been given valid dimensions.

(`Invalid axis type: XX`)[1] – one of the axis subroutines has been called with an invalid type of axis. `CHAXIS`=*XX* is not a recognized axis type..

(`Invalid component type: XX`)[1] – `KTSTR`, `TIMFMT`, `TIMLDA` or `TIMLDC` has been called with an unrecognized component, `COMPNT`=*XX*.

(`Invalid external character file`)[1] – the Hershey file was opened and is readable but is not in the correct format.

(`Invalid shading area: XX`)[1] – `ISSPAT` has been called with an unrecognized type of shading area, `CHFACE`=*XX*.

(`Invalid time-date units: X`)[1] – one of the time-date subroutines (`KTREAL`, `KTTRI`) has been called with an unrecognizable units descriptor, `UNITS`=*X*.

(`Isometric axes inappropriate`)[2] – isometric axes have been requested (using `ISAXD7`) when the current picture is not a surface picture.

(`Isometric coordinates inappropriate`)[1] – `KISXY` has been called to convert isometric coordinates when the current picture is not a surface picture.

(`Key/caption area full`)[2] – there is no room for the caption/key entry (requested by `ADDCP7`, `BLNKKY`, `BOTHK7`, `FILLK7`, `LINEK7`, `MARKK7` or `SHDEK7`).

(`Key/caption attempted with no page`)[2] – a key or caption area has been defined (using `DEFCAP` or `DEFKEY` with `ITYPE`=1 or 2, `MPK7H`, `MPK7V` or `SHKEYS`) before a page has been started.

(`Key/caption attempted with no picture`)[2] – a key or caption area has been defined using picture-related positional descriptors before a picture has been started.

(`Key/caption height reduced`)[2] – the size of a key or caption area (defined using `DEFCAP`, `DEFKEY`, `MPK7H`, `MPK7V` or `SHKEYS`) has been reduced due to restrictions of the current page and/or picture size and the size of text.

(`Key/caption width reduced`)[2] – the width of a key or caption area (defined using `DEFCAP`, `DEFKEY`, `MPK7H`, `MPK7V` or `SHKEYS`) has been reduced due to restrictions of the current page and/or picture size and the size of text.

(`KPOSXY argument invalid`)[1] – `KPOSXY` has been called with an unrecognized argument.

(`KPOSXY called in wrong state`)[1] – `KPOSXY` has been called before a page or picture has been started.

(`KXYXY NOT VIABLE`)[1] – `KXYXY` has been called before the device is opened or before a page/picture has been started.

(`MAP KEY OMITTED: CONSTANT DATA`)[1] – a map key has been requested (using `MPK7H` or `MPK7V`) with the limits of the data range equal, and no z-scale specified elsewhere (by `SFZSCL`).

(`MAP KEY OMITTED: NULL RANGE`)[1] – a map key has been requested (using `MPK7H` or `MPK7V`) which would cover a null z range.

(`Margin modified`)[1] – the margin requested (using `MARGIN`) has had to be modified due to layout constraints.

(`Maximum no. of keys/captions reached`)[2] – the number of blanked or reserved key and caption areas (defined using `DEFCAP` or `DEFKEY` with `ITYPE`=2 or 3, or `MPK7H`, `MPK7V` or `SHKEYS`) has exceeded the maximum of 9.

(`Maximum no. of masked areas reached`)[1] – the number of masked areas used for individual marker symbols and text strings has exceeded the maximum of 50; subsequent marked areas take the place of the oldest areas and overdrawing of these areas may occur.

(`METAFILE ABANDONED: WRITE FAILURE`)[1] – a metafile has been set up but while the completed page was being copied, a `WRITE` failure occurred.

(`MOVEON cancelled`)[1] – `MOVEON` has been called to restore the default picture positioning.

(`No active key/caption area`)[2] – there is no defined area for the caption/key entry (requested by `ADDCP7`, `BLNKKY`, `BOTHK7`, `FILLK7`, `LINEK7`, `MARKK7`, `PUTCP7` or `SHDEK7`).

(`No current waterfall Z scale`)[1] – `KWZVAL` or `QWZSCL` has been called when there are no existing waterfall scales.

(`No room for key/caption area`)[2] – The space available for a key or caption area (defined using `DEFCAP`, `DEFKEY`, `MPK7H`, `MPK7V` or `SHKEYS`) is not even sufficient for one line of text containing only one character.

(`No. of incomplete page tasks=` n)[1] – n incomplete page tasks have accumulated by the end of the page.

(`No. of incomplete picture tasks=` n)[1] – n incomplete picture tasks have accumulated by the end of the picture.

(`None of title will fit in`)[2] – The space available for a title is not wide enough for a single character.

(`Null box, nothing drawn`)[2] – the extent of a box (drawn using `BRKNBX`, `DRAWBX` or `SHDEBX`) exceeds the current picture scales.

(`OWNFIL ignored: invalid argument`)[1] – `OWNFIL` has been called with an unrecognized argument.

(`PAGE cancelled`)[1] – `PAGE` has been called to restore the default page size.

(`PAGE COPIED TO METAFILE`)[1] – a metafile has been set up and the completed page drawn by the program has been copied to the metafile.

(`PAGE ignored: invalid argument`)[1] – `PAGE` has been called with a negative argument.

(`PAGE OMITTED FROM METAFILE`)[1] – a metafile has been set up but the page drawn by the program has been omitted because the program called `MTMISS` during the production of the page.

(`Page Size modified`)[1] – the page size requested (using `PAGE`) has had to be modified due to layout constraints.

(`PAGPOS cancelled`)[1] – `PAGPOS` has been called to restore the default page positioning.

(`Parametric function exceeds scales`)[2] – the extent of a parametric function curve (drawn using `BRKNPR`, `DRAWPR` or `SHDEPR`) exceeds the current picture scales.

(`Periphery modified`)[1] – the size of the periphery requested (using `PERIPH`) has had to be modified due to layout constraints.

(`PICPOS cancelled`)[1] – `PICPOS` has been called to restore the default picture positioning.

(`PICRST: argument invalid`)[1] – `PICRST` has been called with an argument outside the valid range 1 to 10.

(`PICSAV: argument invalid`)[1] – `PICSAV` has been called with an argument outside the valid range 1 to 10.

(`PICSIZ cancelled`)[1] – `PICSIZ` has been called to restore the default picture size.

(`PICSIZ ignored: invalid argument`)[1] – `PICSIZ` has been called with a negative argument.

(`Picture Position modified`)[1] – the picture position requested (using `PICPOS`) has had to be modified due to layout constraints.

(`Picture Size modified`)[1] – the picture size requested (using `PICSIZ`) has had to be modified due to layout constraints.

(`PIE CHART OMITTED: LESS THAN 1 SEGMENT`)[1] – a pie chart has been requested with `NSEGS` < 1.

(`PIE CHART OMITTED: -VE AND +VE VALUES`)[1] – `PIECHT` has been called with both negative and positive segment values.

(`POLAR OMITTED: ZERO RADIUS`)[1] – `POLAR`/`POLAR7` has been called with zero value for maximum radial scale value.

(`Range >=100; linear scale used`)[1] – `SCALES`, `XSCALE` or `YSCALE` has been called for a normal probability scale which exceeds 100.

(`QIBRFL invalid dimensions`)[1] – The z array for `QIBRFL` has been allocated invalid dimensions.

(`Range through 0; linear scale used`)[1] – `SCALES`, `XSCALE` or `YSCALE` has been called for a non-linear scale which includes zero.

(`Read failed on external character file`)[1] – the Hershey file has been read from but a subsequent read has failed.

(`REDRAWING ABANDONED`)[1] – a subroutine other than `MTDRAW` or `MTCLOS` has been called and interrupted the line-by-line redrawing process.

(`Requested point unavailable`)[1] – `CVLBPS` has been specified a reference point for `LABCV7` which cannot be used.

(`RESNXT ignored, area not reserved`)[1] – `RESNXT` has been called specifying a position for which there is no corresponding reserved area.

(`Shaded contour not all in range`)[2] – the extent of a shaded contour region (drawn using `*SHAD`) exceeds the current picture scales.

(`Shaded contours not all in range`)[2] – the extent of a shaded contour map (drawn using `*SHDS`) exceeds the current picture scales.

(`Shading lines too fragmented`)[1] – if a very complex shaded area is to be filled using software shading, circumstances can arise when the shading line must be broken up into more pieces than can be stored; subsequent shading of the area is abandoned.

(`SIMPLEPLOT CLOSED`)[1] – is issued by `ENDPLT` and indicates that all activity by SIMPLEPLOT has finished and all associated files have been closed.

(`SIMPLEPLOT Mark 2-16(`*nnn*`)X`)[1] – indicates that SIMPLEPLOT is in use; it is issued by the first call to any SIMPLEPLOT subroutine except `DIAGLV` with `ILEVEL`=0, `OWNFIL` and `IOCHAN`.

(`START OF GROUP`)[1] – `GROUP` has been called and the next picture will be the first in a group formation.

(`String too short for component`)[1] – `KTSTR` has been called with a string variable which is not large enough to hold the converted time or date.

(`String too short for INTEGER`)[1] – `KNUMB` has been called with a string variable which is not large enough to hold the converted `INTEGER`.

(`String too short for REAL`)[1] – `KREAL` has been called with a string variable which is not large enough to hold the converted `REAL` number.

(`Surface label inappropriate`)[2] – `SFLAB` has been called when there is no current z scale (for a contour map or a surface picture) to be labelled.

(`Surface not all within picture`)[2] – the extent of a surface (drawn using `*SURF` exceeds the current picture scales.

(`SURFACE OMITTED: CONSTANT DATA`)[1] – surface drawing has been attempted for an array containing values which are all the same.

(`Surface section not all in range`)[2] – the extent of a cross-sectional curve (drawn using `*CUT`) exceeds the current picture scales.

(`Symbol clipped`)[2] – a marker symbol (drawn by `CP7PT`, `MARKAR`, `MARKCV` or `MARKPT`) has exceeded the current clipping window (the picture or the page).

(`Symbol spills over boundary`)[2] – a marker symbol (drawn by `CP7PT`, `MARKAR`, `MARKCV` or `MARKPT`) is centred on the edge of the current picture.

(`Text clipped`)[2] – a caption (drawn by `CP7LB`, `CP7PT`, `CP7XC` or `CP7YC`) or title has exceeded the current clipping window (the picture or the page).

(`Text label unavailable: `*XX*)[2] – `TIMFMT` has been called with an unrecognized type of label, `LBTYPE`=XX.

(`Text omitted: too many lines`)[2] – the key/caption area is not full but the caption/key entry contains more lines than can be accommodated (*ie.* the caption includes active escape sequences to insert new lines).

(`Text truncated on curve label`)[2] – the label of a labelled curve would have extended beyond the end of the curve and has therefore been truncated.

(`Title attempted with no page`)[2] – a title has been requested before a page has been started.

(`Title attempted with no picture`)[2] – a title has been requested using picture-related positional descriptors before a picture has been started.

(`Title omitted below bottom`)[2] – an additional title has been requested towards the bottom of the SIMPLEPLOT page but there is not enough room beneath the existing line(s) of title.

(`Title too tall`)[2] – The space available for a title is not even sufficient for one line of text.

(`Title truncated`)[2] – a title is too long to fit within the limiting area (which depends on the position) and has been truncated.

(`Too many axis intervals: default used`)[1] – the axis interval (specified using `AXSBDV`) is less than $1.0^{-4} \times$ axis range.

(`Too many contours: default used`)[1] – `SFEQZ` or `SFEQZD` has been called to specify a contour interval less than $1.0^{-4} \times$ contour range.

(`Too many points in waterfall curve`)[1] – `WFCHT` or `WFDRAW` has been called with more than 1024 points.

(`Too many vectors in shading`)[1] – occurs only when a very complex shaded area contains too many points; subsequent shading of the area is abandoned.

(`Too many waterfall curves on picture`)[1] – `WFDRAW` has been called when the picture already contains number of curves specified by `WFNCVS`.

(`Triangulation data duplicated`)[1] – data cannot be triangulated because of duplicated data (*eg.* (x, y, z_1) and (x, y, z_2)).

(`Triangulation failed`)[1] – triangulation process has failed; try again with `ZZORDN` to normalize data first.

(`Triangulation impossible`)[1] – data cannot be triangulated, *eg.* points are co-linear.

(`Triangulation incomplete`)[1] – maximum size of element array is too small to store all elements.

(`Unclosed shaded area`)[1] – `SHSTOP` has been called but you have tried to shade an incomplete shape.

(`USER SHADING ABANDONED`)[1] – `SHSTRT` has been called but the user-defined shape has not been completed before a subroutine has been called which is not one of those allowed for drawing the shape.

(`Waterfall chart not all in range`)[2] – the extent of a waterfall chart (drawn using `WFCHT`) exceeds the current picture scales.

(`Waterfall curve not all in range`)[2] – the extent of a waterfall curve (drawn using `WFDRAW`) exceeds the current picture scales.

(`WATERFALL OMITTED: INVALID DIMENSIONS`)[1] – 2-D array has not been given valid dimensions.

S. Subroutine Summary

This appendix summarizes the SIMPLEPLOT subroutines described in this manual. Please refer to subroutine specifications for full details of subroutine arguments. These subroutines represent only a subset of the complete SIMPLEPLOT library which is made up of seven separate sections:

1. The basic package for conventional graph plotting.
2. Additional subroutines for three-dimensional plotting.
4. Additional facilities for presentation graphics.
5. *SIMPLEPLOT Volumes* – for plotting three-dimensional `REAL` arrays.
6. *SIMPLEPLOT Maps* – for representing data based on geographical coordinate systems.
7. *SIMPLEPLOT ViSualization* – for perspective drawing of functions of two, three and four variables.

\+ Extra facilities which are available with a SIMPLEPLOT library that has at least Sections 1, 2 and 4.

The numbers marked against the subroutine name indicate in which section of the library the subroutine is included. Only sections 1, 2, 4, + and 7 are listed in this manual.

S.1 Starting a new picture and/or page

`AXES7`[1]	start a new picture and draw axes
`FNSURF`[2]	start 3-D picture and draw surface of user-defined function
`ISBRFL`[+]	start an isometric picture and fill a 3-D barchart
`ISNEW`[2]	start a new isometric picture
`NEWPAG`[+]	start new SIMPLEPLOT page without starting a new picture
`NEWPIC`[1]	start a new picture
`POLAR7`[1]	specify scales, start a picture, draw polar axes
`RGSURF`[2]	start 3-D picture and draw surface (data on regular grid)
`VSNEW`[7]	start a new *SIMPLEPLOT ViSualization* picture
`XSURF`[2]	start 3-D picture and draw surface (x-specified grid)
`XYSURF`[2]	start 3-D picture and draw surface (x-y specified grid)
`YSURF`[2]	start 3-D picture and draw surface (y-specified grid)
`ZSURF`[2]	start picture and draw isometric surface (ungridded data)
`ZZSURF`[2]	start 3-D picture and draw surface (ungridded data with neighbours)

S.2 2-D coordinate system and scales

`COORDS`[1]	change interpretation of coordinates
`EQSCAL`[1]	specify similar linear scales for Cartesian/polar pictures
`FNAREA`[2]	specify plotting ranges for 3-D functions
`KSCALE`[1]	convert scale limits such that they span whole subdivisions
`POLRNG`[+]	specify numerical angular scale for polar plotting
`POLZER`[1]	specify convention for polar coordinates
`QVSCAL`[7]	inquire the scale limits in SIMPLEPLOT user coordinates
`SCALES`[1]	specify both horizontal and vertical scales
`XSCALE`[1]	specify horizontal scale
`YSCALE`[1]	specify vertical scale

S.3 2-D data manipulation

LIMEXC[1]	find the exclusive range of values in an array
LIMIDX[1]	find index of maximum and minimum values in an array
LIMINC[1]	find the inclusive range of values in an array
NODATA[1]	specify REAL value to represent missing data
POLIN[1]	transfer polynomial coefficients to SIMPLEPLOT
POLOUT[1]	transfer polynomial coefficients from SIMPLEPLOT
POLY[1]	evaluate a polynomial function

S.4 Plotting data sets

BRKNAR[4]	draw a curve from function values in array with a broken line
BRKNCV[1]	draw a curve from arrays with a broken line
BRKNFN[1]	draw a user defined function with a broken line
BRKNPR[4]	draw a user defined parametric function with a broken line
DRAWAR[4]	draw a curve from function values in an array with solid line
DRAWCV[1]	draw a curve from arrays of coordinates with solid line
DRAWFN[1]	draw a user defined function with solid line
DRAWPR[4]	draw a user defined parametric function with solid line
LABCV7[4]	draw a labelled curve
MARKAR[4]	draw a set of symbols from function values in an array
MARKCV[1]	draw a set of symbols from arrays of coordinates
SHDEA2[4]	shade between 2 curves defined by function values in 2 arrays
SHDEAR[4]	shade a curve from function values in an array
SHDEC2[4]	shade area between 2 curves
SHDECV[4]	shade curve from arrays of coordinates
SHDEFN[4]	shade user defined function
SHDEPR[4]	shade user defined parametric function
VSPGFL[7]	fill a planar polygon with the current fill index
VSPGTN[7]	interpolate colour from user data within a planar polygon
VSPLDR[7]	draw a polyline through a succession of (x, y, z) data points
VSPLTN[7]	interpolate colour from user data on a polyline
VSPMC[7]	specify the colour index for subsequent markers
VSPMDR[7]	draw a set of markers in a *SIMPLEPLOT ViSualization* image
VSPMMG[7]	specify the scaling factor for subsequent markers

S.5 Bar charts, histograms and pie charts

BARCHT[4]	draw a bar chart from data held in a 2-D array
BARDIR[4]	specify the direction of bar chart scales
BARFMT[4]	specify how bars occupy the space available
BARRNG[4]	specify the range of bar chart numerical scale
BARTYP[4]	specify type of bar chart to be drawn by BARCHT
HSTCLC[1]	calculate frequencies from raw data
HSTCUM[1]	draw a cumulative histogram from frequency data
HSTDIR[4]	specify direction of scales for histograms drawn by HSTGRM
HSTGRM[4]	draw a histogram of raw data and axes
HSTPLT[1]	draw a histogram from frequency data
HSTRNG[4]	specify ranges of scales for histograms drawn by HSTGRM
HSTSHD[4]	specify shading pattern for histogram boxes
HSTTYP[4]	specify type of histogram to be drawn by HSTGRM
ISBRBZ[+]	specify the base z value for 3-D barcharts

`ISBRFC`[+] specify colour sequences on subsequent 3-D barcharts
`ISBRFL`[+] start an isometric picture and fill a 3-D barchart
`ISBRSZ`[+] specify the widths of bars
`ISBRUP`[+] specify the stacking direction for 3-D barcharts
`PIBOXL`[4] specify whether to box pie chart labels
`PIDIAM`[4] specify reduced diameter of pie charts
`PIECHT`[4] draw a pie chart from an array of values
`PIEXPL`[4] specify explosion of pie chart segments
`PIINCL`[4] specify contents of pie chart labels
`PIOMIT`[4] specify segments to be omitted from pie charts
`PIPOSL`[4] specify positions for pie chart labels
`PITILT`[4] specify tilt of pie charts
`QIBRFL`[+] inquire the z scale required by a 3-D barchart
`QVBRFL`[7] inquire the z scale required by a 3-D barchart
`QSPLOT`[1] inquire the current version of SIMPLEPLOT
`SGEXPL`[4] specify one pie chart segment to be exploded
`SGOMIT`[4] specify one pie chart segment to be omitted
`VSBLFL`[7] fill a 3-D block on a *SIMPLEPLOT ViSualization* picture
`VSBLTN`[7] tint a 3-D block on a *SIMPLEPLOT ViSualization* picture
`VSBRBZ`[7] specify the base z value for 3-D barcharts
`VSBRFC`[7] specify colour sequences on subsequent 3-D barcharts
`VSBRFL`[7] fill a *SIMPLEPLOT ViSualization* 3-D barchart
`VSBRSZ`[7] specify the widths of bars
`VSBRTN`[7] tint a *SIMPLEPLOT ViSualization* 3-D barchart
`VSBRUP`[7] specify the stacking direction for 3-D barcharts

S.6 Point-by-point plotting

`ARROW`[4] draw an arrow
`BREAK`[1] force break between joined points
`BRKNBX`[1] draw a box with a broken line
`BRKNCL`[1] draw a circle with a broken line
`BRKNPT`[1] draw a straight line to a specified point with a broken line
`DRAWBX`[1] draw a box with solid line
`DRAWCL`[1] draw a circle with solid line
`DRAWLN`[1] draw a straight line from a specified point to zero
`JOINPT`[1] draw a straight line to a specified point
`MARKPT`[1] draw a marker symbol at a specified point
`RANGE`[1] draw a line indicating a range of values
`SHDEBX`[4] shade a box
`SHDECL`[4] shade a circle

S.7 Annotation labelling at user coordinates

`CP7LB`[4] draw a caption at a specified point
`CP7PT`[1] draw a symbol annotated with a caption at a point
`CP7XC`[4] draw a caption centred horizontally between points
`CP7YC`[4] draw a caption centred vertically between points
`FIGFMT`[1] specify format for `REAL` numbers
`FIGSGN`[1] specify format for signs of `REAL` numbers
`KNUMB`[1] convert `INTEGER` to text string
`KREAL`[1] convert `REAL` to text string
`KXYXY`[+] convert coordinate from one set of units to another

`LABANG`[4] specify angle of labels
`LABJST`[4] specify justification of labels

S.8 Axes

`AXCLR`[1] specify level of automatically generated axis labels
`AXCRSS`[1] specify the intersection of an axis
`AXES7`[1] start a new picture and draw axes
`AXGRID`[1] specify style and level of grids at axis subdivisions
`AXIS7`[1] draw an axis
`AXLAB7`[1] draw an individual axis annotation label
`AXLBAN`[4] specify style of axis annotation labels
`AXLBGP`[4] specify level of annotation at axis intersections
`AXLBJS`[4] specify justification of axis annotation labels
`AXLBLV`[4] specify level of additional axis annotation
`AXLBSL`[4] specify slope of axis annotation labels
`AXLOCN`[4] specify location of an axis w.r.t. the picture
`AXMAJ`[4] draw a major tick mark on an axis
`AXMIN`[4] draw a minor tick mark on an axis
`AXRNGE`[1] specify the sub-range of axis
`AXSBDV`[1] specify the axis subdivision interval
`AXSBMN`[4] specify the minor axis subdivision interval
`AXSBTK`[4] specify the style of axis subdivision tick marks
`AXSUBS`[4] specify number of major/minor subdivisions on an axis
`AXTXT7`[4] draw a set of axis annotation labels
`ISAXD7`[+] draw *x-y-z* axes on a surface picture
`ISAXES`[+] specify whether *x-y-z* axes are drawn on surface pictures
`ISAXFC`[+] specify colours for filled rear planes
`ISAXGC`[+] specify the colour indices for grid lines on the rear planes
`POLAR7`[1] specify scales, start a picture, draw polar axes
`VSAXDR`[7] draw 3-D axes on a *SIMPLEPLOT ViSualization* picture
`VSAXFC`[7] specify colours for filled rear planes
`VSAXGC`[7] specify the colour indices for grid lines on the rear planes

S.9 Time and date based data

`AXLBSP`[+] specify separators between components of time-date axis labels
`AXLBTM`[+] specify label components of time-date axis annotations
`AXLBTP`[+] specify numeric or time-date axis annotations
`KTREAL`[+] convert external time-date triple to internal `REAL` form
`KTSTR`[+] convert one component of internal time-date value to text
`KTTRI`[+] convert internal time-date value to external triple form
`TIMFMT`[+] specify format for one component of time-date data
`TIMLDA`[+] specify set of labels for a component of time-date axis labels
`TIMLDC`[+] specify prefix/suffix for time-date axis labels

S.10 Titles, keys and captions

`ADDCP7`[1] draw a caption in defined area
`ADDJST`[4] specify justification of captions/key entries
`ADDTOP`[4] reset key/caption pointer to top of defined area
`BLNKKY`[1] leave blank line in key/caption area

`BOTHK7`[1] draw key to broken line pattern and marker symbol
`BOXCAP`[4] specify whether captions are to be boxed
`BOXKY`[4] specify whether keys are to be boxed
`DEFCAP`[1] define an area for captions
`DEFKEY`[1] define an area for keys
`DEFKYW`[4] specify width of samples in a key box
`DEFPOS`[+] specify page coordinates of movable positional descriptor `'M'`
`FILLK7`[+] draw key to fast area-fill characters
`KPOSXY`[+] convert a descriptive position into page coordinates
`LINEK7`[1] draw key to broken line pattern
`MARKK7`[1] draw a key to a marker symbol
`MPK7H`[+] draw a complete horizontal key to a shaded contour map
`MPK7V`[+] draw a complete vertical key to a shaded contour map
`PUTCP7`[4] draw caption positioned by row and column in defined area
`QKYCAP`[+] inquire size of key/caption area to fit a set of labels
`RESCLR`[4] cancel all reserved areas
`RESNXT`[4] specify reserved area to be used next
`SHDEK7`[4] draw a key to a shading pattern
`SHKEYS`[4] draw a complete key to a sequence of shading patterns
`TITLE7`[1] draw a caption as a title to a picture, group or page
`VSK7H`[7] draw a complete horizontal key to a tinted plot
`VSK7V`[7] draw a complete vertical key to a tinted plot
`VSKEYS`[7] draw a complete key with distinct shading patterns

S.11 Drawing characteristics for 2-D graph plotting

`ARHEAD`[4] specify proportional size of arrow head
`ARSHFT`[4] specify whether arrows are to include shaft
`CVLBJS`[4] specify justification of lettering on a labelled curve
`CVLBPS`[4] specify reference point for label on a labelled curve
`CVTYPE`[1] specify curve drawing method for 2-D curves
`FNRNGE`[1] specify range for evaluating 2-D functions, $y = f(x)$
`NODATA`[1] specify `REAL` value to represent missing data
`SHTYPE`[4] specify shading boundary for unclosed shaded curves

S.12 General plotting characteristics

`BUNLPR`[1] specify order of precedence of bundled line-drawing attributes
`CHSET`[1] specify source of character fonts
`CLIPLV`[+] specify the clipping region for all plotting
`FFSET`[+] specify source of fast area-fill characters
`LINSET`[1] specify source of broken line patterns
`MKSET`[1] specify source of marker symbols
`RIDDLE`[1] specify level of elimination of redundant points in polylines
`SHSET`[4] specify source of shading patterns
`THCKMG`[1] specify magnification factor for line thickness

S.13 Text and symbol attributes

`CHDESC`[4] specify all details of adjustable characters
`CHMASK`[4] specify character masking
`CHUNDL`[4] specify underlining of text

`MKMASK`[4] specify whether to mask marker symbols
`MKSIZE`[4] specify the size of marker symbols
`QSTR7`[+] inquire physical height and width of text string
`STRCHS`[+] specify special characters for embedded text commands
`STRLV`[+] specify level of interpretation of embedded text commands
`TEXTAR`[4] specify aspect ratio of text
`TEXTLF`[1] specify depth of line spacing for multiple lines of text
`TEXTMG`[1] specify magnification of text
`TEXTMN`[1] specify minimum character width
`TEXTSL`[4] specify slant of hardware text
`TEXTSZ`[1] specify fixed character width of text
`THCKMG`[1] specify magnification factor for line thickness

S.14 Layout

`BOXGRP`[4] specify whether boxes required around groups of pictures
`BOXPG2`[1] specify how the boundaries of pages are to be drawn
`BOXPIC`[4] specify whether boxes required around pictures
`FITPAG`[4] specify interpretation of `PAGE` on fixed page device
`GROUP`[1] specify group layout of pictures
`MARGDV`[1] specify distribution of picture margins
`MARGIN`[1] specify overall size of picture margins
`MOVEON`[4] specify whether to superimpose pictures
`PAGE`[1] specify size of SIMPLEPLOT page
`PAGMRG`[4] specify size of individual peripheral margins
`PAGPOS`[4] specify position of page within the device space
`PAGVW`[4] specify orientation of page
`PAPINC`[1] specify length of paper required (drum plotter *etc.*)
`PERIM`[1] draw rectangular perimeter around current picture
`PERIPH`[4] specify overall size of peripheral margin
`PGFULL`[1] specify whether pages are to be filled
`PICMRG`[4] specify size of individual picture margins
`PICPOS`[4] specify position of pictures
`PICRST`[1] reset the current picture to the dimensions saved by `PICSAV`
`PICSAV`[1] save the size, position and scales of the current picture
`PICSIZ`[1] specify size of pictures
`QPAGE`[+] inquire current device page size
`QPIC`[1] inquire the position and size of the current picture

S.15 Pen/bundle control

`BUNLPR`[1] specify order of precedence of bundled line-drawing attributes
`ONEPEN`[1] specify a single colour of a multi-coloured picture
`PEN`[1] specify single pen for all drawing
`PENHLS`[+] specify palette using HLS colour definition
`PENRGB`[+] specify palette using RGB colour definition
`PENRST`[+] reset palette
`SETPNS`[1] specify pens for four pen pointers
`SHPEN`[4] specify pen number for monochrome shading
`SQPEN`[+] specify sequence of pens
`THCKMG`[1] specify magnification factor for line thickness
`VSCHLS`[7] set hue, lightness and saturation for the palette
`VSCRGB`[7] specify red, green and blue levels for the palette

`VSCRNG`[7] specify a graded range of target colours in the palette
`WFPNS`[+] specify pens for 4 pen pointers on waterfall curves

S.16 Shading control

`FFSET`[+] specify source of fast area-fill characters
`FFSIZE`[+] specify size of fast area-fill characters
`MPTYPE`[+] specify method of area-fill used by `*SHDS`
`RASTER`[+] specify shading technique for shaded surface pictures
`SHANGS`[4] specify angles for software shading lines
`SHCOLS`[4] specify the number of shading colours
`SHDESC`[4] specify all the details of software shading patterns
`SHEDGE`[+] specify whether to draw a shading boundary
`SHGAP`[4] specify the minimum gap between software shading lines
`SHPATT`[4] specify one of a sequence of shading patterns
`SHPEN`[4] specify pen number for monochrome shading
`SHSET`[4] specify source of shading patterns
`SHSTOP`[+] terminate user-defined shape
`SHSTRT`[+] start definition of user-defined shape
`SHTYPE`[4] specify shading boundary for unclosed shaded curves
`SHUNIF`[4] specify or cancel boustrophedon shading lines
`SQSHAD`[4] specify sequence of shading patterns
`VSEDGC`[7] specify the colour index for the edge of filled areas
`VSEDGV`[7] specify whether the edge of polygons are to be drawn
`VSFILC`[7] specify the colour index for area fill subroutines
`VSLINC`[7] specify the colour index to be used by line drawing subroutines
`VSPMC`[7] specify the colour index for subsequent markers
`VSUTOC`[7] define mapping from user data values to colour indices

S.17 Inquiry subroutines

`QCURVE`[+] inquire coordinates of a stored curve or curve segments
`QDEV`[+] inquire details of available devices
`QFFSIZ`[+] inquire physical size of current fast area-fill characters
`QIBRFL`[+] inquire the z scale required by a 3-D barchart
`QKYCAP`[+] inquire size of key/caption area to fit a set of labels
`QNODAT`[1] inquire current missing data value
`QPAGE`[+] inquire current device page size
`QPIC`[1] inquire the position and size of the current picture
`QPIXLS`[+] inquire number of pixels on current device
`QSFLAB`[2] inquire surface picture values
`QSTAT`[+] inquire status of the last SIMPLEPLOT call
`QSTR7`[+] inquire physical height and width of text string
`QVBRFL`[7] inquire the z scale required by a 3-D barchart
`QV3DLM`[7] inquire the 3-D limits for plottable data
`QVDPLM`[7] inquire the range of depths for the current picture
`QVSCAL`[7] inquire the scale limits in SIMPLEPLOT user coordinates
`QWZSCL`[+] inquire limits of waterfall z scale

S.18 Conversion subroutines

`KISXY`[2] convert coordinate on an isometric picture to (x, y)

`KNUMB`[1] convert `INTEGER` to text string
`KPOSXY`[+] convert a descriptive position into page coordinates
`KREAL`[1] convert `REAL` to text string
`KSCALE`[1] convert scale limits such that they span whole subdivisions
`KTREAL`[+] convert external time-date triple to internal `REAL` form
`KTSTR`[+] convert one component of internal time-date value to text
`KTTRI`[+] convert internal time-date value to external triple form
`KVXYD`[7] convert a 3-D point (x, y, z) to (x, y) and depth
`KWZVAL`[+] convert z value on curve to waterfall label scale
`KXYXY`[+] convert coordinate from one set of units to another
`KZRG`[2] convert ungridded data without neighbours to regular
`KZZRG`[2] convert ungridded data with neighbours to regular

S.19 Device and job control

`DDDATA`[+] pass device driver specific information to a device driver
`DEVNAM`[+] specify device by absolute or site name
`DEVNO`[1] specify device by absolute or site number
`DIAGDD`[1] specify level of device driver diagnostics
`DIAGLV`[+] specify level of diagnostics
`DIAGOP`[1] issue a user-defined diagnostic message
`ENDPLT`[1] terminate plotting, close graphics device and SIMPLEPLOT
`INITBL`[1] cancel all current blanked areas.
`INITSP`[1] reset all defaults
`IOCHAN`[+] specify value for an internal SIMPLEPLOT I/O channel
`OWNFIL`[+] specify filename, I/O channel and conditions for use
`PENHLS`[+] specify palette using HLS colour definition
`PENRGB`[+] specify palette using RGB colour definition
`PENRST`[+] reset palette
`QDEV`[+] inquire details of available devices
`QPAGE`[+] inquire current device page size
`QPIC`[1] inquire the position and size of the current picture
`QPIXLS`[+] inquire number of pixels on current device

S.20 Interactive programming

`ENDPAG`[4] finish current group/page
`ENDPIC`[4] finish current picture
`GETXY`[4] get coordinate of point via graphics input
`NEWPAG`[+] start new SIMPLEPLOT page without starting a new picture
`OUTBUF`[4] flush internal graphics buffer to update picture
`OWNIO`[4] specify whether terminal to be used for graphics or user I/O
`OWNNEW`[4] specify whether user in control between pages
`SCLEAR`[4] clear graphics screen
`VSOUT`[7] flush the raster image

S.21 Metafile subroutines

`MTCLOS`[+] terminate metafile redrawing
`MTCOMM`[+] send a comment to the metafile
`MTDRAW`[+] redraw string of metafile commands
`MTFILE`[+] start metafile output

`MTMISS`$^+$ omit current page from metafile
`MTONLY`$^+$ suppress device output while producing metafile
`MTOPEN`$^+$ start metafile redrawing
`REDRAW`$^+$ redraw complete metafile

S.22 Plotting three-dimensional data

X *r*egular *Y* *r*egular *Z* *m*atrix	*r*egular *r*egular *m*atrix	*a*rray *r*egular *m*atrix	*r*egular *a*rray *m*atrix	*a*rray *a*rray *m*atrix	*a*rray *a*rray *a*rray	*r*egular *r*egular *f*unction
*S*urface	*RGSURF*^{2}	*XSURF*^{2}	*YSURF*^{2}	*XYSURF*^{2}	*ZZSURF*^{2}	*FNSURF*^{2}
*C*ontour map	*RGCNTS*^{2}	*XCNTS*^{2}	*YCNTS*^{2}	*XYCNTS*^{2}	*ZZCNTS*^{2}	*FNCNTS*^{2}
*C*ontour line	*RGCONT*^{2}	*XCONT*^{2}	*XCONT*^{2}	*XYCONT*^{2}	*ZZCONT*^{2}	*FNCONT*^{2}
*C*ross-section	*RGCUT*^{2}	*XCUT*^{2}	*YCUT*^{2}	*XYCUT*^{2}	*ZCUT*^{2}	*FNCUT*^{2}
*I*nterpolation	*RGCALC*^{2}	*XCALC*^{2}	*YCALC*^{2}	*XYCALC*^{2}	*ZZCALC*^{2}	*N*/A
*S*haded map	*RGSHDS*$^+$	*XSHDS*$^+$	*YSHDS*$^+$	*XYSHDS*$^+$	*ZZSHDS*$^+$	*FNSHDS*$^+$
*S*haded region	*RGSHAD*$^+$	*XSHAD*$^+$	*YSHAD*$^+$	*XYSHAD*$^+$	*ZZSHAD*$^+$	*FNSHAD*$^+$
*W*aterfall *C*hart	*WFCHT*$^+$ *WFDRAW*$^+$		*C*onvert data *t*o regular grid			

S.23 Perspective surfaces

`VSRG`^{7} draw a surface picture from a 2-D array
`VSRGU`^{7} draw contours on a surface picture from 2 functions of 2 variables
`VSXYZ`^{7} draw a surface picture from three 2-D arrays
`VSXYZU`^{7} draw contours on a surface picture from three 2-D arrays
`VSZ`^{7} draw a surface from data structured into elements
`VSZU`^{7} draw contours on a surface from data structured into elements

S.24 Ungridded 3-D data manipulation

`KZRG`^{2} convert ungridded data without neighbours to regular
`KZZRG`^{2} convert ungridded data with neighbours to regular
`VSZ`^{7} draw a surface from data structured into elements
`VSZU`^{7} draw contours on a surface from data structured into elements
`ZEDGE`^{2} draw periphery of data (without neighbours)
`ZELEM`^{2} draw outline of a single element
`ZELEMS`^{2} draw element outlines without neighbours
`ZNEIGH`^{2} generate neighbour array
`ZNUMB`^{2} specify node and element numbering
`ZORDEN`^{2} structure data into elements (normalized)
`ZORDER`^{2} structure data into elements
`ZZEDGE`^{2} draw periphery of data (with neighbours)
`ZZELMS`^{2} draw element outlines (with neighbours)
`ZZORDN`^{2} structure data into elements, neighbours (normalized)
`ZZORDR`^{2} structure data into elements and neighbours
`TRINEW`^{2} to acquire a handle to a Triangle object
`TRIREG`^{2} to register the (x, y) points in a Triangle object
`TRIANG`^{2} to triangulate a Triangle object
`TRIOFF`^{2} to specify the base value subsequently used for point indices
`TRIGET`^{2} to extract triangles from a Triangle object
`TRIDEL`^{2} to delete a Triangle object

`TRIFIX`[2] to specify a sequence of points to lie on a fixed line in a Triangle object

S.25 Surface pictures

`ISANG`[2] specify angle for representing surface pictures
`ISAXD7`[+] draw *x*-*y*-*z* axes on a surface picture
`ISAXES`[+] specify whether *x*-*y*-*z* axes are drawn on surface pictures
`ISAXFC`[+] specify colours for filled rear planes
`ISAXGC`[+] specify the colour indices for grid lines on the rear planes
`ISBRBZ`[+] specify the base *z* value for 3-D barcharts
`ISBRFC`[+] specify colour sequences on subsequent 3-D barcharts
`ISBRFL`[+] start an isometric picture and fill a 3-D barchart
`ISBRSZ`[+] specify the widths of bars
`ISBRUP`[+] specify the stacking direction for 3-D barcharts
`ISCURV`[2] specify smoothness for crosshatch/cascade lines on surfaces
`ISDIAG`[2] specify whether to include *x*-*y*-*z* arrows on surface pictures
`ISFULL`[2] specify whether surfaces are to fill available space
`ISMESH`[2] specify fineness of detail for surface pictures
`ISRISE`[2] specify width:height proportions for surface pictures
`ISSHAD`[+] specify method of shading for isometric surfaces
`ISSKRT`[+] specify type of skirt on a shaded surface picture
`ISSPAT`[+] specify shading pattern of skirt, backdrop or base of shaded surface
`ISTYPE`[2] specify type of surface picture
`ISVIEW`[2] specify view point for surface pictures
`ISYUP`[2] specify direction of change of *y* for isometric pictures
`QIBRFL`[+] inquire the *z* scale required by a 3-D barchart
`RASTER`[+] specify shading technique for shaded surface pictures
`SFEQZ`[2] specify spacing of contours
`SFEQZD`[2] specify intervals for contour drawing
`SFLIMS`[2] specify *z* plotting range without changing scale
`SFMESH`[+] specify mesh for contour curves and surface pictures
`SFZSCL`[2] specify *z*-scale for surface pictures and contour maps

S.26 Contour maps and cross sections

`CTBRKN`[2] specify broken line pattern for contour curves
`CTCURV`[2] specify curve type for contours
`CTHOLD`[+] specify whether coordinates of subsequent contours should be stored
`CTLABS`[2] specify frequency of contour labels
`CTNUMB`[2] specify contour labelling
`MPTYPE`[+] specify method of area-fill used by `*SHDS`
`QCURVE`[+] inquire coordinates of a stored curve or curve segments
`SFEQZ`[2] specify spacing of contours
`SFEQZD`[2] specify intervals for contour drawing
`SFLIMS`[2] specify *z* plotting range without changing scale
`SFMESH`[+] specify mesh for contour curves and surface pictures
`SFZSCL`[2] specify *z*-scale for surface pictures and contour maps
`SQBRKN`[+] specify sequence of broken line patterns
`SQPEN`[+] specify sequence of pens
`SQSHAD`[4] specify sequence of shading patterns
`SQZLAB`[+] specify sequence of contour labels
`SQZVAL`[+] specify sequence of contour levels
`VSINIT`[7] reset *SIMPLEPLOT ViSualization* defaults

`WFINIT`$^{+}$ reset default waterfall characteristics

S.27 Waterfall charts

`KWZVAL`$^{+}$ convert z value on curve to waterfall label scale
`QWZSCL`$^{+}$ inquire limits of waterfall z scale
`WFCHT`$^{+}$ draw waterfall chart on the current 2-D picture
`WFDRAW`$^{+}$ draw an individual waterfall curve
`WFEQL`$^{+}$ specify equally-spaced waterfall label scale
`WFEQN`$^{+}$ specify equally-spaced waterfall numeric scale
`WFINIT`$^{+}$ reset default waterfall characteristics
`WFNCVS`$^{+}$ specify number of waterfall curves
`WFNSCL`$^{+}$ specify limits of numeric waterfall scale
`WFPNS`$^{+}$ specify pens for 4 pen pointers on waterfall curves
`WFSTEP`$^{+}$ specify displacement between waterfall curves
`WFZLEV`$^{+}$ specify level on waterfall curve for pen change
`WFZSCL`$^{+}$ specify z scale of waterfall charts

S.28 Additional 3-D facilities

`ISAXD7`$^{+}$ draw x-y-z axes on a surface picture
`KISXY`^{2} convert coordinate on an isometric picture to (x, y)
`LIMSFN`^{2} find the range of 3-D function
`NODATA`^{1} specify `REAL` value to represent missing data
`QNODAT`^{1} inquire current missing data value
`QSFLAB`^{2} inquire surface picture values
`SFLAB`^{2} draw caption of range of values displayed
`VS3DLM`^{7} set the 3-D Limiting Box for plottable data
`VSEQX`^{7} specify the equally-spaced x values for gridded 3-D data
`VSEQY`^{7} specify the equally-spaced y values for gridded 3-D data
`VSFIT`^{7} specify which axis scales are measured in similar units
`VSFITP`^{7} specify the ratio between axis scale lengths
`VSFULL`^{7} specify how a projected 3-D object is to fill the picture
`VSMAG`^{7} magnify coordinates along any or all of the x-y-z axes
`VSROT`^{7} rotate coordinates around one of the x-y-z axes
`VSTRAN`^{7} translate coordinates along any or all of the x-y-z axes
`VSVRTP`^{7} set the Viewing Position in terms of a radius and 2 angles
`VSVXYZ`^{7} set the Viewing Position in terms of (x, y, z)

T. Useful Tables

T.1 Axis types

CHAXIS is an argument to the following subroutines:

AXCLR, AXCRSS, AXGRID, AXIS7, AXLAB7, AXLBAN, AXLBGP, AXLBJS, AXLBLV, AXLBSL, AXLBSP, AXLBTM, AXLBTP, AXLOCN, AXMAJ, AXMIN, AXRNGE, AXSBDV, AXSBMN, AXSBTK, AXSUBS, AXTXT7 and VSROT.

CHAXIS	*T*ype of axis
'XC'	Cartesian x-axis
'YC'	Cartesian y-axis
'RP'	Polar r-axis (radial)
'AP'	Polar θ-axis (angular)
'XI'	Isometric x-axis
'YI'	Isometric y-axis
'ZI'	Isometric z-axis
'NB'	Bar chart numeric axis
'LB'	Bar chart label axis
'NW'	Waterfall chart numeric axis
'LW'	Waterfall chart label axis
'X3'	3-D x-axis (*SIMPLEPLOT ViSualization* only)
'Y3'	3-D y-axis (*SIMPLEPLOT ViSualization* only)
'Z3'	3-D z-axis (*SIMPLEPLOT ViSualization* only)
'U3'	3-D u-axis (*SIMPLEPLOT ViSualization* only)

T.2 Availability of axis facilities

Subroutine	XC	YC	RP	AP	XI	YI	ZI	NB	LB	NW	LW	X3	Y3	Z3	U3
AXCLR	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√
AXCRSS	√	√	√	√	x	x	x	x	x	√	√	x	x	x	x
AXGRID	√	√	√	√	x	x	x	√	√	√	√	x	x	x	x
AXIS7	√	√	√	√	√	√	√	√	√	√	√	√	√	√	x
AXLAB7	√	√	√	√	√	√	√	√	x	√	√	√	√	√	x
AXLBAN	√	√	√	√	√	√	√	√	√	√	√	*U*	*U*	*U*	x
AXLBGP	√	√	√	x	√	√	√	√	x	√	√	√	√	√	√
AXLBJS	√	√	√	x	x	x	√	√	x	√	√	x	x	x	x
AXLBLV	*NL*	*NL*	x	√	x	x	x	x	x	x	x	x	x	x	x
AXLBSL	√	√	√	x	√	√	√	√	√	√	√	x	x	x	x
AXLBSP	√	√	√	√	√	√	√	√	x	√	√	√	√	√	x
AXLBTM	√	√	√	√	√	√	√	√	x	√	√	√	√	√	x
AXLBTP	√	√	√	√	√	√	√	√	x	√	√	√	√	√	x
AXLOCN	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√
AXMAJ	√	√	√	√	√	√	√	√	x	√	√	√	√	√	x
AXMIN	√	√	√	√	√	√	√	√	x	√	√	√	√	√	x
AXRNGE	√	√	√	√	√	√	√	√	x	√	√	√	√	√	√
AXSBDV	*L*	*L*	√	√	√	√	√	√	x	√	√	√	√	√	√
AXSBMN	*L*	*L*	√	√	√	√	√	√	x	√	√	√	√	√	√
AXSBTK	√	√	√	√	√	√	√	√	√	√	√	*U*	*U*	*U*	*U*
AXSUBS	*L*	*L*	√	x	√	√	√	√	√	√	√	√	√	√	√
AXTXT7	√	√	√	√	√	√	√	√	x	√	√	√	√	√	x

NL: non-linear scales only
L: Linear scales only
U: Preceding, Following and Inside unavailable

T.3 Direction of axes

CHAXIS is an argument to the subroutines listed above, but the direction of axes is of particular relevance to the following routines:

AXIS7, AXLBAN, AXLBJS, AXLOCN and AXSBTK.

The *direction* of an axis is defined as follows:

CHAXIS	*Direction of axis*
'XC'	Left to right of picture
'YC'	Bottom to top of picture
'RP'	Center to outside of polar chart
'AP'	see THZERO
'XI'/'YI'	from front to back of surface picture
'ZI'	Bottom to top
'NB'/'LB'	as x-axis or y-axis depending on orientation
'NW'/'LW'	as x-axis or y-axis

'P'receding and 'F'ollowing are positions relative to the direction of an axis and describe positions unambiguously on *all* types of axis. For example, on Cartesian axes, 'P' refers to the left of a horizontal

axis and the bottom of a vertical axis; `'F'` refers to the right of a horizontal axis and the top of a vertical axis.

T.4 Position of Keys and Captions

VCHAR and HCHAR are arguments to the following subroutines:

DEFCAP, DEFKEY, KPOSXY, MPK7H, MPK7V, RESNXT, SHKEYS, TITLE7, VSK7H, VSK7V and VSKEYS.

VCHAR	*V*ertical position	HCHAR	*H*orizontal position
'N'	North (highest on page)	'W'	West (far left of page)
'H'	Higher than group		
'O'	Over picture	'P'	Preceding picture
'T'	Top of picture	'L'	Left of picture
'C'	Centre of picture	'C'	Centre of picture
'B'	Bottom of picture	'R'	Right of picture
'U'	Under picture	'F'	Following picture
'L'	Lower than group		
'S'	South (lowest on page)	'E'	East (far right of page)

('N'orth,'C'entre), ('S'outh,'C'entre) and all combinations with HCHAR = 'E'ast or 'W'est refer to positions relative to the page. Other combinations with VCHAR = 'N'orth, 'S'outh, 'H'igher or 'L'ower refer to positions relative to the group of pictures. All other combinations refer to positions on the picture.

T.5 Justification within a key or caption area

VJST and HJST are arguments to the following subroutines:

DEFPOS and KPOSXY.

VJST	*V*ertical Justification
'T'	Top of title, key or caption area
'C'	Halfway between 'T' and 'B'
'B'	Bottom of title, key or caption area
'D'	Default – bottom of title, key or caption area

HJST	*H*orizontal Justification
'L'	At the left of the title, key or caption area
'C'	Halfway between 'L' and 'R'
'R'	At the right of the title, key or caption area
'D'	Default – left of title, key or caption area

T.6 Area associated with key/caption

ITYPE is an argument to the following subroutines:

DEFCAP and DEFKEY.

`ITYPE` *associated area*

1 this area is associated only with the current picture/page which must have already been started.

2 a blanked area which behaves as `ITYPE`=1 except that subsequent drawing on the current page excludes any drawing from this area except that which is specifically aimed at it; it is best called after `NEWPIC` but before the main picture is drawn (this may be slower because SIMPLEPLOT must check for overlapping).

3 a reserved area which is kept free from all drawing except that which is specifically aimed at it; each time a new picture/page is started, a suitable character size is calculated for each reserved area; it must be called before `NEWPIC` (or `NEWPAG`) but does nothing until a new picture (or page) is started.
Reserved areas can be cancelled by another call to `DEFCAP` or `DEFKEY` with `ITYPE`=3, the same values of `VCHAR` and `HCHAR`, but with `NROWS` and/or `NCOLMS` equal to zero.

T.7 Time/date component

`COMPNT` is an argument to the following subroutines:

`AXLBTM` (values of `COMP1`, `COMP2` and `COMP3`), `TIMFMT` and `TIMLDC`.

`COMPNT`	*Meaning*	*Default*`CHTYPE`
`'YE'`	Year	`'y'`
`'MO'`	Month	`'M'`
`'WE'`	Week number	`'W'`
`'DM'`	Day of the month	`'D'`
`'DW'`	Day of the week	`'w'`
`'DE'`	Days elapsed	`'d'`
`'HO'`	Hour	`'h'`
`'MI'`	Minute	`''`
`'SE'`	Second	`'"'`

A subset of these arguments are available for `TIMLDA`. The default arrays of labels stored by SIMPLEPLOT are as follows:

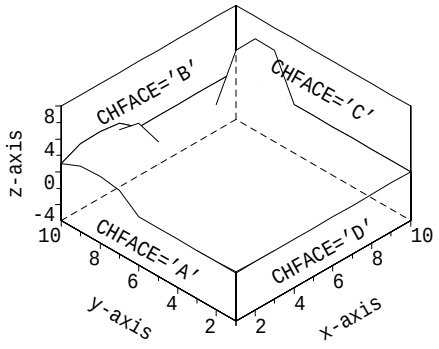
`'DW'`	`'DM'`	`'MO'`	`'HO'`
Sunday	1st	January	Midnight
Monday	2nd	February	1am
Tuesday	3rd	March	2am
Wednesday	4th	April	3am
Thursday	5th	May	⋮
Friday	6th	June	11am
Saturday	7th	July	Noon
	8th	August	1pm
	9th	September	2pm
	10th	October	3pm
	11th	November	⋮
	etc.	December	

T.8 Faces of skirts, backdrops and bases

`CHFACE` is an argument to the following subroutine:

`ISSPAT`.

`CHFACE`	*C*omponent	
`'AI'`	inner skirt, face A	inner skirts, `'*I'`
`'BI'`	inner skirt, face B	
`'CI'`	inner skirt, face C	
`'DI'`	inner skirt, face D	
`'AO'`	outer skirt, face A	outer skirts, `'*O'`
`'BO'`	outer skirt, face B	
`'CO'`	outer skirt, face C	
`'DO'`	outer skirt, face D	
`'AB'`	backdrop, face A	backdrops, `'*B'`
`'BB'`	backdrop, face B	
`'CB'`	backdrop, face C	
`'DB'`	backdrop, face D	
`'GI'`	inner base	bases, `'G*'`
`'GO'`	outer base	
`'ND'`	skirts dropped from no-data values	



T.9 Representation of pen colours

HUE, BRIGHT and SAT are arguments of PENHLS and VSCHLS; RED, GREEN and BLUE are arguments of PENRGB and VSCRGB.

Colour	HUE	BRIGHT	SAT	RED	GREEN	BLUE
Black	any	0.0	any	0.0	0.0	0.0
White	any	1.0	any	1.0	1.0	1.0
Red	120.0	0.5	1.0	1.0	0.0	0.0
Green	240.0	0.5	1.0	0.0	1.0	0.0
Blue	0.0	0.5	1.0	0.0	0.0	1.0
Cyan	300.0	0.5	1.0	0.0	1.0	1.0
Magenta	60.0	0.5	1.0	1.0	0.0	1.0
Yellow	180.0	0.5	1.0	1.0	1.0	0.0
Orange	150.0	0.5	1.0	1.0	0.5	0.0
Pink	120.0	0.8	1.0	1.0	0.6	0.6
Grey	any	0.5	0.0	0.5	0.5	0.5

Index