SUN/88.6

Starlink Project
Starlink User Note 88.6

P C T Rees, M J Bly, D L Terrett

28 June 1999

NCAR — Graphics Utilities v1.3 User's Guide

Contents

1 Introduction

The National Center for Atmospheric Research in Boulder, Colorado has, for many years, distributed a extensive suite of high level graphics utilities. When NCAR converted these utilities to run over the ISO standard GKS, Starlink was used as a test site. Starlink was subsequently given permission to release the NCAR utilities as part of the Starlink Software Collection. The NCAR utilities offer an alternative high level graphics package to PGPLOT (SUN/15) for use within Starlink applications. The NCAR package is arguably more powerful than PGPLOT in functionality and flexibility, but it does require a little more time to master. Starlink provides a set of extensions to the NCAR graphics utilities, SNX (SUN/90), to make the use of NCAR more accessible to the beginner. For lower level work, SGS (SUN/85) or GKS (SUN/83) should be used.

The utilities provided by NCAR are:

AUTOGRAPH — Draws and annotates curves or families of curves.

CONRAN — Contours irregularly spaced data labelling the contour lines.

CONRAQ — Like CONRAN, but faster because it has no labelling capability.

CONRAS — Like CONRAN, but slower because lines are smoothed and crowded lines are removed.

CONREC — Contours two dimensional arrays, labelling the contour lines.

CONRECQCK — Like CONREC, but faster because it has no labelling capability.

CONRECSUPR — Like CONREC, but slower because lines are smoothed and crowded lines are removed.

DASHCHAR — Software dashed line package with labelling capability.

DASHLINE — Like DASHCHAR, but faster because it has no labelling capability.

DASHSMTH — Like DASHCHAR, but slower because lines are smoothed.

DASHSUPR — Like DASHCHAR, but slower because lines are smoothed and crowded lines are removed.

EZMAP — Plots continental and/or national and US state boundaries in one of nine map projections.

GRIDAL — Package for drawing graph paper, axis, *etc.*

HAFTON — Draws halftone (greyscale) picture from a two dimensional array. (*N.B.* The GKS cell array function GCA should be used on devices with a true greyscale capability.)

HSTGRM — Plots histograms.

ISOSRF — Plots iso-values surfaces (with hidden lines removed) from a three dimensional array.

ISOSRFHR — Plots iso-values surfaces (with hidden lines removed) from a high resolution three dimensional array.

PWRITX — Plots high quality software characters.

PWRITY — Plots simple software characters.

PWRZI — Draws characters in three-space, for use with the ISOSRF utility.

PWRZS — Draws characters in three-space, for use with the SRFACE utility.

PWRZT — Draws characters in three-space, for use with the THREED utility.

SRFACE — Plots a three dimensional display of a surface (with hidden lines removed) from a two dimensional array.

STRMLN — Plots a representation of a vector flow of any field for which planar vector components are given on a regular rectangular lattice. [1]

THREED — Provides a three space line drawing capability.

VELVCT — Draws a two dimensional velocity field by drawing arrows from the data locations.

The NCAR graphics utilities are released exactly as supplied by NCAR except that the default behaviour of AUTOGRAPH and HSTGRM has been changed to not clear the screen after each plot — this is more appropriate for an interactive environment.

Because of a name conflict with a Fortran run-time library routine on UNIX, the NCAR routine FLUSH has been removed from the UNIX release of the NCAR library. In order to get the same behaviour as a call to FLUSH when using NCAR, the call

```
CALL PLOTIT( 0, 0, 2 )
```

may be used (see the SPPS description in the NCAR manual).

2 Documentation

A manual containing extensive descriptions of all the NCAR routines is available as a Starlink Miscelaneous User Document (MUD) and can be obtained from your Starlink site manager.

3 Using the Utilities

Before calling any of the utilities, GKS must be open and at least one workstation open and active. This can be done with the following sequence of GKS calls:

```
CALL GOPKS( LUERR, -1 )
CALL GOPWK( IWKID, ICONID, IWTYPE )
CALL GACWK( IWKID )
```

where LUERR is a Fortran logical unit to which messages resulting from GKS internal errors may be written (unit 6 is recommended on all operating systems used by Starlink), IWKID is

[1] Note: because of portability problems this utility is not available on UNIX.

a workstation identifier (any integer you care to choose) and IWTYPE and ICONID are the workstation type and connection identifier respectively.

A more friendly user interface can be provided if, instead of asking the user for a GKS workstation type and connection identifier, a GNS workstation name is used (*i.e.* as used by SGS). A GNS name can be translated to its GKS equivalent by the GNS (SUN/57) call

```
CALL GNS_TNG( NAME, IWKTYP, ICONID, STATUS )
```

where STATUS is the inherited status (returned SAI__OK if the translation is successful). More information on workstation names can be found in SUN/57.

If you wish to do any plotting with SGS routines, SGS_OPEN must be used to open GKS and the workstation. SGS makes the entire display surface of a workstation available for plotting, which in general means that part of the normalized device co-ordinate unit square will not be visible. Since the default behaviour of the NCAR utilities is to use the whole of the NDC unit square, using them becomes a little more complicated. This is discussed further in the section on co-ordinate systems (*i.e.* §4).

When plotting is complete, GKS must be closed down with either:

```
CALL GDAWK( IWKID )
CALL GCLWK( IWKID )
CALL GCLKS
```

or

```
CALL SGS_CLOSE
```

as appropriate.

4 Co-ordinate Systems

When two graphics packages are used from the same program, one package may interfere with the correct operation of the other. When mixing the NCAR utilities and SGS, both packages manipulate the GKS transformations and so precautions must be taken to avoid interference.

The NCAR documentation refers to “fractional co-ordinates” which in GKS terms are normalized device co-ordinates (NDC). To plot in NCAR fractional co-ordinates with GKS calls, all that is necessary is to select the GKS normalization transformation 0 using the routine GSELNT. To do locator or stroke input in fractional co-ordinates, the GKS normalization transformation 0 must be made the highest priority using the routine GSVPIP.

With the exception of the utilities that plot three-dimensional objects, a two-dimensional “user co-ordinate” system (which may be logarithmic in one or both dimensions) is defined. A set of functions is provided by NCAR to convert points between user co-ordinates and fractional co-ordinates. These functions are:

CFUX(X) — Returns the X user co-ordinate corresponding to the fractional co-ordinate X.

CFUY(Y) — Returns the Y user co-ordinate corresponding to the fractional co-ordinate Y.

CUFX(X) — Returns the X fractional co-ordinate corresponding to the user co-ordinate X.

CUFY(Y) — Returns the Y fractional co-ordinate corresponding to the user co-ordinate Y.

The use of these functions is illustrated in the example program in Appendix A. These functions are only valid while the GKS transformations set up by the NCAR utilities are still current. When mixing NCAR calls with SGS or GKS calls it may be necessary to save the normalization transformation immediately after the NCAR call (using the GKS routine GQNT) and restore it before using NCAR again (using the GKS routines GSWN and GSVP). Note that calling any SGS zone selection routine will always correctly restore the state of SGS.

If you use SGS_OPEN to open the GKS workstation, the entire NDC unit square will not usually be mapped onto the display surface. This is an inescapable consequence of making the whole display surface available for plotting and means that the area of NDC (or fractional co-ordinates) used by the utilities must be changed from the default. The method described here demonstrates how to make the utilities plot in the current zone. When the current zone is the base zone, this is equivalent to plotting on the whole workstation.

The first step is to inquire the NDC limits of the current zone (SGS always uses normalization transformation number 1):

```
REAL VIEWP( 4 )
REAL WIND( 4 )

CALL GQNT( 1, IERR, WIND, VIEWP )
```

The zone limits in NDC are now stored in the array VIEWP. The next step depends on the utility being used:

AUTOGRAPH —

```
CALL AGSETP( 'GRAPH.', VIEWP, 4 )
```

CONRAN,CONRAS,CONRAQ —

```
CALL CONOP1( 'SCA=PRI' )
```

EZMAP —

```
CALL MAPPOS( VIEWP( 1 ), VIEWP( 2 ), VIEWP( 3 ), VIEWP( 4 ) )
```

HSTGRM —

```
CALL HSTOPR( 'WIN', VIEWP, 4 )
```

THREED —

```
CALL SET3( VIEWP( 1 ), VIEWP( 2 ), VIEWP( 3 ), VIEWP( 4 ), ...
```

The plotting area used by HAFTON, STRMLN and the CONREC family is controlled by variables in common blocks (see the individual routine documentation for details). ISOSRF, ISOSRFHR, SRFACE and VELVCT always use the entire unit square.

5 Linking

All the routines in the NCAR library have standard Fortran six character names, so you must beware of your own routines having the same names as these library routines. If this happens, the linker will not report any error and the subsequent aberrant behaviour of your program will not point unambiguously to the source of the problem. The names of all the routines in the library are listed in Appendix B.

To link a non-ADAM program with NCAR the command line is:

```
% f77 program.o -L/star/lib `ncar_link` -o program.out
```

By default, the DASHCHAR dashed line drawing package is used. If one of the other line drawing packages (DASHLINE, DASHSMTH or DASHSUPR which all reside in the directory `/star/lib`) is required, it must be included in the link command explicitly, *e.g.*:

```
% f77 program.o /star/lib/dashsmth.o -L/star/lib `ncar_link` \
       -o program.out
```

will use the DASHSMTH package. Similarly, the default version of the CONREC family of contour routines is CONREC which can be replaced by CONRECQCK or CONRECSUPR in the same way.

To link an ADAM application with NCAR the command line is:

```
% alink application.o `ncar_link_adam`
```

SUN/144 gives further details of compiling and linking ADAM applications with the UNIX operating system.

6 Demonstration Programs

Built version of the example programs may reside in the directory `/star/bin/examples/ncar`. The example module names on UNIX are:

```
exampl  tautog  tcnqck  tcnsmt  tcnsup  tconan
tconaq  tconas  tconre  tdashc  tdashl  tdashp
tdashs  tezmap  tgrida  thafto  thstgr  tisohr
tisosr  tpwrtx  tpwry   tpwrzi  tpwrzs  tpwrzt
tsrfac  tthree  tvelvc
```

If the binaries have been removed to conserve disk space, the sources may be found in directory `/star/sources/ncar-examples`. This directory includes a `makefile` and `mk` script which can be used to build the example programs. Copy the entire contents to a scratch directory and `cd` to it, then with the `SYSTEM` environment variable set appropriately, the individual examples may be built *e.g.*:

```
% setenv SYSTEM alpha_OSF1
% ./mk exampl
```

7 References

SUN/15 PGPLOT — Graphics Subroutine Library.
SUN/57 GNS — Graphics Workstation Name Service.
SUN/83 GKS — Graphical Kernel Syatem (7.2).
SUN/85 SGS — Simple Graphics System.
SUN/90 SNX — Starlink Extensions to the NCAR Graphics Utilities.
SUN/144 ADAM — UNIX Version.

A Example Program

```
      PROGRAM EXAMPL
*+
*   Simple program to illustrate the use of AUTOGRAPH with SGS:
*      a set of X,Y points is plotted with AUTOGRAPH and then the mean
*      Y value marked with a horizontal line drawn with SGS.
*+

      INTEGER N
      PARAMETER( N = 10 )

      CHARACTER * 20 WS
      INTEGER I, IBASE, ISTAT
      REAL X( N ), Y( N ), WIND( 4 ), VIEWP( 4 )
      REAL TOTAL, AMEAN, ANDC, XST, XEN

      DATA X / 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 /
      DATA Y / 1, 2, 4, 3, 5, 6, 4, 5, 6, 10 /

*  Get the workstation name.
      WRITE ( *, '( 1X, A )', ERR=999 ) 'Workstation'
      READ ( *, '( A )', ERR=999 ) WS

*  Open SGS.
      CALL SGS_OPEN( WS, IBASE, ISTAT )
      IF ( ISTAT .NE. 0 ) GO TO 999

*  Obtain the viewport limits and set the limits for AUTOGRAPH.
      CALL GQNT( 1, IERR, WIND, VIEWP )
      CALL AGSETP( 'GRAPH.', VIEWP, 4 )

*  Draw the graph.
      CALL EZXY( X, Y, N, 'Autograph example_$' )

*  Find the mean Y value.
      TOTAL = 0.0

      DO 10 I = 1, N
         TOTAL = TOTAL + Y( I )
 10   CONTINUE

      AMEAN = TOTAL / REAL( N )

*  Convert the end points of thhe line to fractional co-ordinates (i.e. NDC)
*  before restoring SGS state.
      ANDC = CUFY( AMEAN )
      XST = CUFX( 1.0 )
      XEN = CUFX( 10.0 )

*  Re-establish the SGS zone
      CALL SGS_SELZ( IBASE, ISTAT )
```

```
*  Set the world co-ordinates to match the NDC (i.e. fractional co-ordinates).
      CALL SGS_SW( VIEWP( 1 ), VIEWP( 2 ), VIEWP( 3 ), VIEWP( 4 ),
     :             ISTAT )

*  Draw the line and close SGS.
      CALL SGS_LINE( XST, ANDC, XEN, ANDC )
      CALL SGS_CLOSE

 999  CONTINUE

      END
```

B NCAR Subroutine Names

AGAXIS	AGBACK	AGBNCH	AGCHAX	AGCHCU	AGCHIL
AGCHNL	AGCTCS	AGCTKO	AGCURV	AGDASH	AGDFLT
AGDLCH	AGDSHN	AGEXAX	AGEXUS	AGEZSU	AGFPBN
AGFTOL	AGGETC	AGGETF	AGGETI	AGGETP	AGGTCH
AGINIT	AGKURV	AGLBLS	AGMAXI	AGMINI	AGNUMB
AGPPID	AGPWRT	AGQURV	AGRPCH	AGRSTR	AGSAVE
AGSCAN	AGSETC	AGSETF	AGSETI	AGSETP	AGSRCH
AGSTCH	AGSTUP	AGUTOL	ANOTAT	BOUND	CALCNT
CCHECK	CFUX	CFUY	CFVLD	CHKCYC	CHSTR
CLGEN	CLSET	CLSGKS	CMFX	CMFY	CMUX
CMUY	CONBD	CONBDN	CONCAL	CONCLD	CONCLS
CONCOM	CONDET	CONDRW	CONDSD	CONECD	CONGEN
CONINT	CONLCM	CONLIN	CONLOC	CONLOD	CONOP1
CONOP2	CONOP3	CONOP4	CONOT2	CONOUT	CONPDV
CONPMM	CONPMS	CONRAN	CONRAQ	CONRAS	CONREC
CONREO	CONSLD	CONSSD	CONSTP	CONTLK	CONTNG
CONTOR	CONXCH	CPFX	CPFY	CPUX	CPUY
CTCELL	CUFX	CUFY	CURVE	CURVE3	CURVED
DANDR	DASHBD	DASHDB	DASHDC	DCHECK	DISPLA
DRAWI	DRAWPV	DRAWS	DRAWT	DRCNTR	DRLINE
DRWSTR	DRWVEC	E9RIN	ENCD	ENTSR	EPRIN
ERROF	EXPAND	EZCNTR	EZHFTN	EZISOS	EZMXY
EZMY	EZSRFC	EZSTRM	EZVEC	EZXY	EZY
FDUM	FENCE3	FILLIN	FL2INT	FLUSH	FRAME
FRST3	FRSTC	FRSTD	FRSTPT	FRSTS	GBYTES
GETHOL	GETSET	GETSI	GETUSV	GNEWPT	GRAY
GRID	GRIDAL	GRIDL	GRIDT	GTDGTS	GTNUM
GTNUMB	GTSIGN	HAFTON	HALFAX	HFINIT	HST1
HSTBKD	HSTEXP	HSTGRM	HSTLST	HSTMED	HSTOPC
HSTOPL	HSTOPR	HSTSTR	HTABLE	I1MACH	I8SAV
IDICTL	IDIOT	INIT3D	INITZI	INITZS	INITZT
INTZI	INTZS	INTZT	ISHIFT	ISOSRB	ISOSRF
KFMX	KFMY	KFPX	KFPY	KMPX	KMPY
KPMX	KPMY	KUMX	KUMY	KUPX	KUPY
KURV1S	KURV2S	LABMOD	LASTD	LINE	LINE3
LINE3W	LINED	MAPBD	MAPCEM	MAPCHI	MAPDRW
MAPEOS	MAPFST	MAPGRD	MAPGTC	MAPGTI	MAPGTL
MAPGTR	MAPINT	MAPIO	MAPIQ	MAPIT	MAPLBL
MAPLMB	MAPLOT	MAPPOS	MAPROJ	MAPRS	MAPRST
MAPSAV	MAPSET	MAPSTC	MAPSTI	MAPSTL	MAPSTR
MAPTRE	MAPTRN	MAPTRP	MAPUSR	MAPVEC	MAPVP
MINMAX	MKMSK	MMASK	MXMY	NERRO	OPNGKS
PERIM	PERIM3	PERIML	PLOTIT	POINT	POINT3
POINTS	PSYM3	PWRIT	PWRITX	PWRITY	PWRX
PWRXBD	PWRY	PWRYBD	PWRYGT	PWRYSO	PWRZ

C Portability

C.1 Overview

This section discusses the portability of NCAR, including the coding standard adopted and a list of those Starlink packages which need to be ported to the target machine before a port of NCAR can proceed.

C.2 Coding and porting prerequisites

The standard of Fortran used for the coding of NCAR is totally compliant with ANSI Fortran 77. A small number of operating system-specific routines have also been written in ANSI C. To use NCAR on any computer system the operating system specific routines must be modified to reflect the integer and floating point arithmetic used.

NCAR requires either GKS Vn. 7.2 or GKS Vn. 7.4 to be available.

C.3 Operating system specific routines

Several NCAR subroutines make use of operating system features which are specific to the machine upon which they are implemented. The names of these routines and their purpose are as follows:

I1MACH (I) : Define INTEGER machine dependent constants.

R1MACH (I) : Define REAL machine dependent constants.

ncar.h : C language versions of the following routines are also available and may be used by a particular implementation: IAND, IOR, ISHIFT. [2] These routines assume a Fortran/C language interface which is defined within the C header file `ncar.h`. `ncar.h` will require modification for use on new operating systems.

[2]The routines IAND and IOR perform logical AND and OR operations between two given INTEGERS. They are often provided by implementations of Fortran 77 as intrinsic functions, but are not featured in the ANSI Standard. C language versions are provided for use with Fortran implementations without IAND and IOR intrinsic functions.