# GFLASH--A Graphics Instruction Manipulation Package

**Author:** Fred Clare

**Technical Writer/Editor:** Brian Bevirt

---

## Table of Contents



---

### Introduction

GFLASH is a package of Fortran subroutines that provides a limited picture segmentation capability, that is, various parts of a picture may be stored independently in segments and reinserted into any subsequent picture. GFLASH duplicates the capabilities of the FLASH package in the previous NCAR System Plot Package (NSPP), but in a Graphical Kernel System (GKS) environment.

GKS is a set of basic functions for computer graphics programming. The GKS standard has been adopted by both the American National Standards Institute (ANSI) and the International Standards Organization (ISO). GKS can be implemented in one of twelve levels, depending on the graphical input and output capabilities. The GKS standard also specifies how the defined graphics functionality must be implemented in Fortran. NCAR Graphics contains a full FORTRAN 77 implementation of GKS at Level 0A. GKS at Level 0A has roughly the same functionality as the System Plot Package in previous NCAR graphics packages.

The NCAR GKS package has been enhanced to support the Level 2 GKS functions necessary to support GFLASH. GFLASH can be used with any Level 2A GKS package, but its primary use will be in support of users of the NCAR pre-GKS Graphics Package who are converting to the NCAR GKS-based Graphics Package. Fully implemented Level 2A GKS packages provide more complete picture segmentation capabilities than those offered by GFLASH. For example, you can rotate, scale, and translate segments as well as rename them, set display priorities, and inquire segment attributes. Users of such packages will thus most likely want to use those capabilities rather than using GFLASH.

# Functional Capabilities

GFLASH captures an arbitrary set of graphics instructions in a dataset and inserts these instructions into any subsequent picture, making it unnecessary to regenerate the instructions. A classic use of this capability is in making movies. For example, a background that will be used for many pictures can be saved and inserted into all desired pictures in the movie without having to regenerate the background for every picture.

## Using GFLASH with the NCAR GKS Package

GFLASH is a part of the SPPS (System Plot Package Simulator) package. It supports essentially identical functionality to that in the NSPP FLASH package (entries FLASH1, FLASH2, FLASH3, and FLASH4.) The entries in GFLASH are simpler to use than the old FLASH entries, but there is an additional requirement that the package must be opened before use and closed after use. You can open usage of GFLASH when using the NCAR GKS package with the following call:

```
CALL GOPWK(ID,IC,3)
```

where ID can be any nonnegative integer and IC is a valid Fortran logical unit number used to write the plotting instructions to disk. A valid unit number is one not currently being used and not units 5 or 6. Recall that the NCAR GKS package uses unit 1 for its output. ID is a workstation identifier which is required by the close workstation call (see below). The NCAR GKS package opens all files itself, and no file preconnections are required. The third argument to GOPWK is a workstation type, and "3" indicates Workstation Independent Segment Storage (WISS) in the NCAR GKS package. WISS is implemented on disk storage in the NCAR package. See below for the appropriate GOPWK call when using a non-NCAR GKS package. An error will be reported if the above call is not made before using any of the entries described below. It is also assumed that the underlying GKS package is open and the metafile workstation has been opened and made active (see the man page for OPNGKS).

Note that ID must not be the same value used to open the metafile. If you are using OPNGKS to open the NCAR GKS package, this means that ID must not equal "1" when you call GOPWK as described above.

GFLASH must be closed before deactivating and closing the metafile output workstation (i.e., before calling CLSGKS if you are using that SPPS entry). To close GFLASH, issue the call

```
CALL GCLWK(ID)
```

where ID is the same identifier as was used in the call to open GFLASH.

## Description of the GFLASH Entries

The GFLASH analogues of FLASH1, FLASH2, FLASH3, and FLASH4 are GFLAS1, GFLAS2, GFLAS3, and GFLAS4.

```
SUBROUTINE GFLAS1(IB)
```

A call to GFLAS1 initiates storage of plotting instructions into a disk file. GFLASH uses disk writes instead of writing to memory as the NSPP FLASH package did for reasons of portability and consistency with the I/O done by the NCAR GKS package. Instructions subsequent to GFLAS1, but prior to a GFLAS2 call, described below, will not be inserted into the output metafile, but rather will be stored on disk. The argument IB can be any integer between 0 and 99 inclusive, thus making it possible to define 100 different storage buffers in a single job step. To use GFLAS4, described below, you may need to know the name of the disk file where the plotting instructions are stored. GFLAS1 automatically assigns the name GNFB*xx* to the file that will receive subsequent plotting instructions, where *xx* is the integer value of IB. For example, if GFLAS1 is called with an argument of 9, then subsequent plotting instructions will be stored in file GNFB09. The GNFB*xx* (GKS New Flash Buffer) files are all created using Fortran logical unit IC as specified in the GOPWK call.

```
SUBROUTINE GFLAS2
```

A call to GFLAS2 will terminate putting plotting instructions to disk and resume putting plotting instructions to the metafile output. A call to GFLAS2 can only be made after a previous call to GFLAS1. GFLAS2 has no arguments.

```
SUBROUTINE GFLAS3(IB)
```

A call to GFLAS3 inserts the instructions saved on disk with a previous GFLAS1 identifier IB into the output metafile. GFLAS3 can be called only after a previous GFLAS1 and GFLAS2 sequence or after a call to GFLAS4. GFLAS3 also uses Fortran logical unit IC for its reads.

```
SUBROUTINE GFLAS4(IB,FNAME)
```

A call to GFLAS4 allows you to access a disk file of plotting instructions generated with a GFLAS1 and GFLAS2 sequence in a previous job for use in a GFLAS3 call. The first argument to GFLAS4 is the identifier to be used for subsequent GFLAS3 calls. This identifier must be between 0 and 99 just as the argument to GFLAS1. The second argument to GFLAS4 is the name of the file in which the plotting instructions are stored; FNAME is a Fortran CHARACTER variable specifying the file name of the disk file.

No plotting results from a GFLAS4 call. GFLAS4 simply does the necessary bookkeeping to associate the plotting instructions stored in file FNAME with the identifier IB. A call to GFLAS3 will be required to actually put the instructions in FNAME into the current output metafile. There is no restriction on the filename except that it be a legal name on the current host.

The disk file specified by FNAME has all the same attributes as a standard NCAR metafile, and it may be sent across ethernet or via File Transfer Protocol just like an NCAR metafile.

## Using GFLASH with a non-NCAR GKS Package

With the exception of GFLAS4, GFLASH routines can be used with any GKS package that is level 2A or higher. The GFLASH entries call the appropriate GKS segmentation functions to provide the desired functionality. GFLAS4 uses a specially defined GKS ESCAPE function in the NCAR GKS package that is not defined in any other GKS package; consequently, if you try to use GFLAS4 with a non-NCAR GKS package, you will get an error message to the effect that no ESCAPE function with function id of

-1392 exists.

GFLASH requires that WISS be opened before use and closed after use. You can open usage of GFLASH when using a non-NCAR GKS package with the following call:

```
CALL GOPWK(ID,IC,IW)
```

where ID can be any nonnegative integer identifier, IC is a valid Fortran logical unit number, and IW is the workstation type for WISS. IW is implementation-dependent; you can obtain the correct value for it from the documentation for the GKS package being used.

To complete usage of GFLASH, WISS must be closed; to do this, issue the call

```
CALL GCLWK(ID)
```

where ID is the same identifier used in the call to open WISS.

To avoid possible errors, it is important to know how GFLASH works in a generic GKS environment. A call to GFLAS1 deactivates all active workstations except WISS, and it activates WISS. The consequence of this is that all subsequent plotting instructions are saved in WISS, but they are not sent to any other output workstation. A call to GFLAS2 will deactivate WISS and reactivate all workstations that were active when GFLAS1 was called. A call to GFLAS3 will send all instructions saved by the associated GFLAS1 and GFLAS2 sequence to all currently active workstations.

## General Usage Notes

- It is not legal to call FRAME (or to clear workstations) between a call to GFLAS1 and GFLAS2.
- The plotting instructions captured by a GFLAS1 and GFLAS2 sequence reflect the exact environment in effect when the instructions were created, including the clipping rectangle and all attribute settings. A call to GFLAS3 will insert the requested instructions just as they were created, without regard to attribute settings in effect at the time that GFLAS3 was called. For example, if you call GFLAS1 and a pink line is drawn and then call GFLAS2, the instructions to draw a pink line are saved. If you later call GFLAS3 at a time when the current line color is blue, the pink line will still be drawn as it was when created in the GFLAS1 and GFLAS2 sequence. However, a line drawn after the GFLAS3 call has finished executing would be blue.
- Instructions created by a GFLAS1 and GFLAS2 sequence are scaled and clipped in accordance with the window and viewport settings at the time the instructions were created. When the instructions are inserted into the output stream with a GFLAS3 call, they will appear as they were created and will not be effected by new window or viewport settings. GFLAS3 will have no affect on the current window and viewport setting, nor will it be affected by the current window and viewport setting. The current attribute settings at any given point in a job are those that would apply if all GFLAS*n* calls were removed.
- As described above, when you use the NCAR GKS package a file name of the form GNFB*xx* is created for each separate GFLAS1 and GFLAS2 sequence, and the appropriate plotting instructions are saved in those files. What is done with these files is implementation dependent. On the NCAR Cray computers, these files are local datasets and will disappear at job termination unless saved as permanent datasets. On most other hosts, however, the GNFB*xx* files will remain after job termination, and you will have to purposely delete them. The GNFB*xx* files have the same file attributes as the default metafile GMETA. However, the GNFB*xx* files are not complete

metafiles and cannot be plotted independently without error.

- On the NCAR CRAY X-MP/48, the default maximum size for a disk file is 40,000 512-word blocks. This can be increased to 500,000 blocks by setting LM equal to the desired number of blocks on an ASSIGN statement.
- FLASH buffers can be reused within the same executing program; that is, the same index can be used more than once in a GFLAS1, GFLAS2 sequence. One common use for this feature is to duplicate frames for a movie sequence - frame number n can be put into FLASH buffer 1 and put to the output file two or more times using GFLAS3; then frame number n+1 can be put into FLASH buffer 1 and put to the output file two or more times, and so forth.

## Example

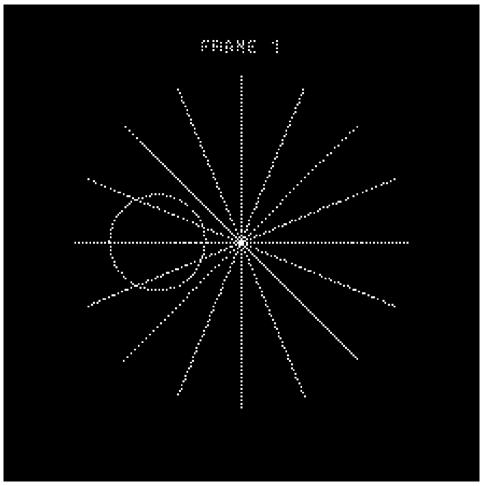
The code for an example begins on the next page, and the output is on the pages following the code.

```
      PROGRAM GFXMPL
C
C  Data for the graphical objects.
C
      DATA    TWOPI/6.283185/
      DATA    X0C, Y0C, RC, NPTC/ 0.325, 0.5, 0.10 , 210/
      DATA    X0P, Y0P, RP, NPTP/ 0.500, 0.5, 0.35 ,  16/
      DATA    X0S, Y0S, RS      / 0.675, 0.5, 0.10      /
C
C Use the SPPS entry to open GKS.
C
      CALL OPNGKS
C
C Establish the viewport and window.
C
      CALL SET(0.,1.,0.,1.,0.,1.,0.,1.,1)
C
C  Initialize the GFLASH package.  If using a non-NCAR GKS package
C  the final argument in the following call should be replaced with
C  the workstation type for WISS.
C
      CALL GOPWK(9,1,3)
C
C  Put a circle in a buffer with identifier 1.
C
      CALL GFLAS1(1)
      DTHETA = TWOPI/NPTC
      CALL FRSTPT(X0C+RC,Y0C)
      DO 20 I=1,NPTC
      ANG = DTHETA*REAL(I)
      XC = RC*COS(ANG)
      YC = RC*SIN(ANG)
      CALL VECTOR(X0C+XC,Y0C+YC)
   20 CONTINUE
      CALL GFLAS2
C
C  Put a polygonal fan in a buffer with identifier 2.
C
      CALL GFLAS1(2)
      DTHETA = TWOPI/NPTP
      DO 10 I=1,NPTP
      ANG = DTHETA*REAL(I)
      XC = RP*COS(ANG)
```

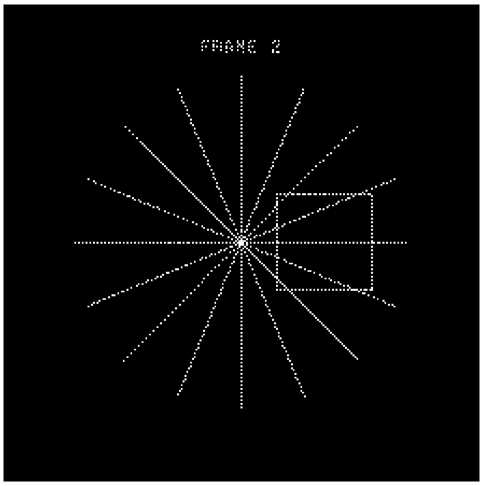
```
      YC = RP*SIN(ANG)
      CALL LINE(X0P,Y0P,X0P+XC,Y0P+YC)
   10 CONTINUE
      CALL GFLAS2
C
C  Put a square in a buffer with identifier 3.
C
      CALL GFLAS1(3)
      CALL FRSTPT(X0S+RS,Y0S+RS)
      CALL VECTOR(X0S-RS,Y0S+RS)
      CALL VECTOR(X0S-RS,Y0S-RS)
      CALL VECTOR(X0S+RS,Y0S-RS)
      CALL VECTOR(X0S+RS,Y0S+RS)
      CALL GFLAS2
C
C  Put a background perimeter in a buffer with identifier 4.
C
      CALL GFLAS1(4)
      CALL FRSTPT( 0.0, 0.0)
      CALL VECTOR( 1.0, 0.0)
      CALL VECTOR( 1.0, 1.0)
      CALL VECTOR( 0.0, 1.0)
      CALL VECTOR( 0.0, 0.0)
      CALL GFLAS2
C
C  Create plots.
C
C  Frame 1 -- title, circle, and fan.
C
      CALL WTSTR(0.5,0.91,'FRAME 1',25,0,0)
      CALL GFLAS3(1)
      CALL GFLAS3(2)
      CALL GFLAS3(4)
      CALL FRAME
C
C  Frame 2 -- fan, title, and square.
C
      CALL GFLAS3(2)
      CALL WTSTR(0.5,0.91,'FRAME 2',25,0,0)
      CALL GFLAS3(3)
      CALL GFLAS3(4)
      CALL FRAME
C
C  Frame 3 -- circle, square, fan, and title (note that the change
C             in window affects the WTSTR call, but not the elements
C             in the buffers -- this illustrates the independent
C             nature of the FLASH buffers).
C
      CALL SET (0.,1.,0.,1.,0.,10.,0.,10.,1)
      CALL GFLAS3(1)
      CALL GFLAS3(3)
      CALL GFLAS3(2)
      CALL WTSTR(5.0,9.1,'FRAME 3',25,0,0)
      CALL GFLAS3(4)
      CALL FRAME
C
C  Close the GFLASH package, and GKS.
C
      CALL GCLWK(9)
      CALL CLSGKS
C
```

```
      END
```

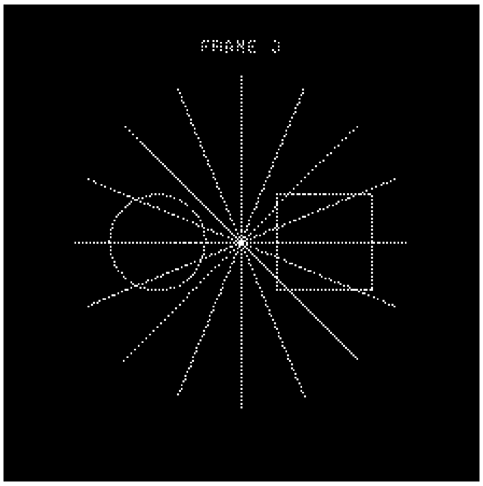
**Frame 1** (click on plot to see a larger and clearer picture)




**Frame 2** (click on plot to see a larger and clearer picture)




**Frame 3** (click on plot to see a larger and clearer picture)


FRAME J