SUN/57.9

Starlink Project
Starlink User Note 57.9

David Terrett & Nicholas Eaton

12 July 1995

GNS

Graphics Workstation Name Service 1.4 User & Programmer's Manual

---

Abstract

This note contains information for three categories of GNS consumer:

The user of a graphics application. This consumer is interested in the syntax of graphics workstation names and need only read section 2.

The system manager. This consumer is interested in the contents of the database and how to set up and modify it. System managers will need to be familiar with the first part (section 3.1), which deals with workstation names. The remainder (section 3.2) will be of interest to anyone adding support for a new GKS or IDI device.

The application programmer. This consumer is interested in the programmer interface (Appendix B), the mechanism for reporting errors (section 4) and the means of compiling and linking the programs (section 5).

Contents

1 Introduction **1**

2 Workstation Names **1**
2.1 GKS Names . . . 1
2.2 IDI Names . . . 2
2.3 GWM windows . . . 2
2.4 Demonstration Program . . . 3

3 Database Management **3**
3.1 Workstation Names . . . 3
3.1.1 GKS Names . . . 3
3.1.2 IDI Names . . . 4
3.1.3 Sequence numbers . . . 5
3.2 Workstation Description Table . . . 5
3.2.1 GKS Descriptions . . . 5
3.2.2 IDI Descriptions . . . 7

4 Error Messages **7**

5 Compiling and Linking Programs with GNS **7**

A Workstation Description File Keywords **9**

B Programmer Interface **10**
B.1 Summary of GNS calls . . . 10

C Subroutine Specifications **11**
GNS_GTN . . . 12
GNS_GWNG . . . 13
GNS_GWNI . . . 14
GNS_IANG . . . 15
GNS_IANI . . . 16
GNS_IDNG . . . 17
GNS_IETG . . . 18
GNS_IGAG . . . 19
GNS_IIAI . . . 20
GNS_IONG . . . 21
GNS_ITWCG . . . 22
GNS_IWCG . . . 23
GNS_IWSG . . . 24
GNS_START . . . 25
GNS_STOP . . . 26
GNS_TNDG . . . 27
GNS_TNG . . . 28
GNS_TNI . . . 29

1 Introduction

Almost any program that does any graphics requires the user to identify which graphics device is to be used. When the graphics package used by the program is GKS (SUN/83)—and for the majority of Starlink software it is, at least indirectly—graphics devices are identified by two integers, a 'workstation type' and a 'connection identifier'. ('Workstation' is GKS terminology for a graphics device of any description and this is the sense in which it is used throughout this note.) No one can be expected to remember the workstation types of all the devices supported by Starlink (nearly 50 at present) so a library has been provided that translates 'friendly' and easy to remember names into their GKS equivalents.

Most high level graphics packages, such as SGS SUN/85) and PGPLOT SUN/15), call GNS to perform the necessary name translation when a workstation is opened, so unless you are writing programs that open GKS workstations directly (by calling `GOPWK`), or need to make specialised device inquiries, you will not need to call any GNS routines yourself.

The graphics workstation name service library provides three sorts of service to the users and writers of graphics subroutine libraries and applications programs:

- The translation of workstation names to their GKS equivalents.
- Generating a list of the names and types of the graphics devices available on a system.
- Answering a variety of enquiries about the properties of a particular graphics device; for example, its category (pen plotter, image display etc.) or its device name.

GNS also provides support for the Image Display Interface (IDI—SUN/65) and the Applications Graphics Interface (AGI—SUN/48), but in both cases the use of GNS is internal to the librariy and so its presence is not normally apparent to the programmer.

2 Workstation Names

2.1 GKS Names

A GKS workstation name is a character string that represents a graphics device that may be opened and manipulated using GKS. The simplest way to get a list of valid workstation names is to run the demonstration program (section 2.4). When you supply a workstation name to a program that uses GNS (either directly or indirectly via a high level graphics package) you can abbreviate the name provided that the abbreviation is not ambiguous. You can also supply a environment variable that translates to a workstation name; this can be used to create "nicknames" for devices that you use frequently.

Each workstation name identifies a device type and may also identify a particular device (or file). For example, the workstation name `cifer_t5` might translate to `801` (the GKS workstation type for a Cifer T5 terminal), while `ps_l` might translate to `2700` and `gks74.ps` (the GKS workstation type and the usual file name of a landscape orientation PostScript printer file).

If a name translates to just a workstation type then a "default" device is used. All device types have a default device name associated with them, and this is most important for terminals where the default is the terminal from which the program is being run. In this case the default device name depends on which terminal is being used.

The device name implied by a workstation name can be overridden by appending a semicolon followed by a new device or file name to the workstation name. For example, the name `ps_l;plot.ps` would use `plot.ps` instead of `gks74.ps`.

If an explicit device name is supplied there are no checks made to ensure that the device is the right type; attempting to use the wrong type will either cause the attempt to open the GKS workstation to fail or garbage to be sent to the device.

An application program can also override the device name *implied* by a workstation name; if it does, the program's documentation should make this clear. An application cannot override a device name which is supplied *explicitly* as part of a workstation name.

It is also possible to specify an explicit GKS device type and connection identifier as a pair of integers separated by a comma, space or underscore. If the connection identifier is omitted, it defaults to zero.

Finally, a workstation name can be an environment variable that translates to a valid workstation name (which can be another environment variable). Any explicit device name is removed before the translation is attempted and the first device name encountered is the one that is used; any subsequent device names are ignored. Environment variable translation continues recursively until either no translation is found or the resulting name begins with "`GKS_`".

2.2 IDI Names

An IDI workstation (or device) name is simply a character string (for example, `xwindows`) which identifies an image display and its type. The default device name can be overridden by appending a semicolon followed by the new device name after the workstation name. Environment variables can be set up and translation proceeds recursively until a name is found that does not translate to another name.

2.3 GWM windows

There are some special features of GNS that apply to GWM window devices; these are X window display devices under the control of the Graphics Window Manager (SUN/130). All GWM windows have a name (usually displayed in the title bar) which need not be the same as the workstation name supplied at a prompt or as an argument to a routine such as GNS_TNG. This window name is the one that is supplied as an argument to the xmake or xdestroy commands. When a GWM window is accessed through GNS (for both GKS and IDI) the supplied workstation name is translated into a GWM window with a default name. For instance the workstation name "xwindows" usually translates into a window named "xwindows". Note that the GKS workstation name "xoverlay" also translates into a window named "xwindows" because the overlay device is part of the same GWM window.

Both GKS and IDI will create a new GWM window with the default name if one does not already exist. The default window name can be overridden by using the device name field of the workstation name. The device name field is separated from the workstation name by a

semi-colon; thus the string 'xw;bert' is parsed into the workstation name 'xw' and the device name 'bert'. The device name supersedes the default and so a window named 'bert' is used rather than one named 'xwindows'

By default a GWM window created by a GKS or IDI graphics application will not have an overlay plane unless the overlay option is specified in the X defaults file (see SUN/130) . The user can request that a window is created with an overlay plane by appending a '+' to the name when prompted for it. Thus the workstation name 'xwindows+' or 'xw+' will result in a GWM window with an overlay. Note that when using GKS specifying 'xw+' will create a window with an overlay, but the drawing will be done in the base plane (specifying 'xoverlay' automatically results in a window with an overlay).

2.4 Demonstration Program

A complete list of all the GKS workstation names defined on your current node can be obtained by running the program `$STARLINK_DIR/share/examples/gnsrun_gks`. After listing all the GKS names along with a short description of each workstation, the program prompts for a workstation name. If one of the names in the list (or any other valid workstation name) is entered, a simple demonstration picture is plotted on the device selected.

A similar program called `gnsrun_idi` will also be available if the IDI library has been installed on the system.

If running either program produces any unexpected error messages you should report them to your system manager; however, you should not necessarily expect all workstations on the lists to be useable. A device could, for example, be in use by someone else.

3 Database Management

3.1 Workstation Names

It is the responsibility of the system manager to set up and maintain the database of workstation name translations. This section describes the structure of the database and the modifications that the system manager typically has to make when installing the system.

3.1.1 GKS Names

The GKS workstation names are stored in a text file which is found by looking in turn for:

- The translation of the environment variable `GNS_GNSNAMES`.
- A file called `gns_gksnames` in a directory found by translating the environment variable `GNS_DIR`.
- A file called `../etc/gns_gksnames` relative to each directory on the PATH.

Each line in the file describes one workstation name and contains the name, the GKS workstation type that it translates to, the device name of the workstation, a short description of the workstation and, optionally, the network node from which the device is accessible and a sequence number (see section 3.1.3). The fields are separated by the slash (/) character and a complete record might look like:

```
IKON_A/3200/XAA0:/Ikon Image display A/NODEA/ 1/
```

However, the device name field is usually blank which means that the default device associated with the workstation type is used, the node field will usually be blank as it is only used on VMS systems and the sequence number will normally be absent; so most records in the file will be more like:

```
tek4010/201/ /Tektronix 4010 terminal/
```

An explicit device name is only required when you need different names for two or more devices of the same type.

For X window devices the device name is used as the GWM window name, and the sequence number should be left blank.

```
xwindows/3800/xwindows/X Windows/
```

The system is distributed with a file containing a name for every workstation type supported by RAL-GKS and so will work without any modification. However, programs that list the available workstations will present users with a long list of workstation names, many of which are probably not available, so the first task is to delete from the file the definitions for devices that are not available on your system.

As well as eliminating extraneous names from listings of available workstations this will also make abbreviations more useful. For example, some of the names are somewhat unwieldy because they have to be able to distinguish between different devices made by the same manufacturer such as `cifer_t5` for the Cifer T5 terminal and `cifer_2634` for the Cifer 2634. If, on a system with only one sort of Cifer terminal the unwanted name is deleted, then `cifer` becomes an acceptable abbreviation.

Having edited `gns_gksnames` you should test the system by running the demonstration program described in section 2.4.

3.1.2 IDI Names

The IDI workstation names are stored in a text file which is opened in the same way as the GKS names files but with GKS replaced by IDI. Each line in the file describes one workstation name and contains the name, the IDI workstation type that it translates to (a two or three character code), the device name of the workstation, a short description of the workstation and, optionally, the network node from which the device is accessible and a sequence number (see section 3.1.3). For example:

```
xwindows/XW/xwindows/X Windows/ / /
x2windows/XW2/xwindows2/X Windows/ / /
```

The use of the node name field is the same as for GKS names (see section 3.1.1). For X window devices the first two characters of the workstation type must be 'XW', the device name is a string (up to 10 characters) which is used as the GWM window name, and the sequence number should be left blank.

3.1.3 Sequence numbers

The sequence number can be thought of as a kind of serial number which uniquely identifies a device. The main purpose of the sequence number is to identify multiple names that refer to the same physical device (or workstation window). Each device has to have its own sequence number as in the following GKS example:

```
xwindows/   3800/xwindows/X window 1/ / 1/
xoverlay/ 3805/xwindows/X window 1/ / 1/
xwindows2/   3801/xwindows2/X window 2/ / 2/
xoverlay2/ 3806/xwindows2/X window 2/ / 2/
```

The sequence number refers to the physical device, so that devices that have multiple names should have the same sequence number. This also applies to devices that have multiple GKS workstation types, such as an overlay plane, or image display with VT terminal interface.

If a sequence number has been set in the `gns_gksnames` file and the same device has an entry in the `gns_idinames` file then it should be given the same sequence number (and vice versa). That is if a device supports both GKS and IDI then the sequence number should be the same in both files. Continuing the previous example the `gns_idinames` file would have the following entries:

```
xwindows/   XW/xwindows/X window 1/ / 1/
xwindows2/  XW2/xwindows2/X window 2/ / 2/
```

A sequence number is an integer in the range 1 to 99. If the sequence number is not present in the database then a default value of 0 is used.

3.2 Workstation Description Table

3.2.1 GKS Descriptions

The workstation description file only needs to be modified if a new device type is supported by GKS so this section will only be of interest to someone adding a new workstation handler.

The workstation description table is stored in a binary file `gns_gksdevices` (located in a similar way to the names files). This binary file is built from a text version of the description table by running the program `gksbuild`. (`gksbuild` is not normally installed and must be created by re-building the gns libary.)

The description table distributed with the system contains an entry for every workstation type supported by GKS-UK and is built from the text file `gksdevices.txt` which can be used as a template for any changes or additions.

The text file looks something like:

```
WORKSTATION = 10
  CLASS = METAFILE_INPUT
  OUTPUT =  FILE
WORKSTATION = 50
  CLASS = METAFILE_OUTPUT
  OUTPUT = FILE
WORKSTATION = 101
  CLASS = TERMINAL
  SCALE = 3.3833E-4
  OUTPUT = DIRECT
  CLEAR = SELECTIVE
```

Each workstation description starts with `WORKSTATION` = *type* and contains a list of statements of the form *keyword* = *value*.

The value field takes one of the following forms:

Integer A decimal integer.

Real A real number constructed according to the usual FORTRAN rules but without any imbedded spaces.

Keyword A character keyword.

Character string A character string delimited by the first non-space character following the equals sign and the next occurrence of that same character. When character strings are being interpreted the ^ character is used as an "escape" character for inserting control characters (e.g. control Z (ASCII 26) is represented by ^Z). A literal ^ character is represented by ^^.

The only mandatory keyword is `CLASS` which indicates to which category of device the workstation belongs and must have one of the following keywords as its value [1]:

```
GRAPHICS_OVERLAY
IMAGE_DISPLAY
IMAGE_OVERLAY
MATRIX_PRINTER
METAFILE_INPUT
METAFILE_OUTPUT
PEN_PLOTTER
TERMINAL
WINDOW
WINDOW_OVERLAY
```

For all workstation types other than those of class `TERMINAL`, `METAFILE_INPUT` and `METAFILE_OUTPUT` the keyword `DEFAULT_NAME` is also mandatory. The value must be the name of the device used when the workstation is opened with a connection identifier of 0. For devices of class `TERMINAL` the logged-on terminal is used in these circumstances.

Other keywords describe such things as the approximate size of "device units" and a character string that can be used to clear the text screen of a terminal.

A complete list of all the keywords that can appear in the description file can be found in appendix A.

[1] an image overlay is a graphics overlay that is also capable of plotting cell arrays

3.2.2 IDI Descriptions

There is an equivalent description file for IDI stored in a binary file `gns_ididevices` (located in a similar way to the names files). This binary file is built from a text version of the description table by running `idibuild`. At present the description table only contains one entry which is the `AGITYPE`; this is mandatory for an IDI workstation. The text file looks something like:

```
WORKSTATION = /XW/
  AGITYPE = 3800
WORKSTATION = /XW2/
  AGITYPE = 3801
```

4 Error Messages

All GNS routines use the so called *inherited status strategy* (see SUN/104), which means that if the `STATUS` argument is not set to the value `SAI__OK` (equals zero) on entry the routine will exit without performing any action. If an error occurs during the execution of a routine, `STATUS` will be set to one of the values defined in the FORTRAN include file `GNS_ERR`.

5 Compiling and Linking Programs with GNS

To include either of the GNS include files, create links in the directory containing the program source code with the command:

```
% gns_dev}
```

and use and include statement such as:

```
INCLUDE 'GNS_ERR'
INCLUDE 'GNS_PAR'
```

The link procedures for most high level graphics packages (e.g. IDI, SGS etc.) already contain a reference to GNS. Programs that do not link with such packages can be linked to GNS by using the shell script gns_link or gns_link_adam. For example to compile and link an ADAM task called 'task.f' the following is used

```
% alink task.f `gns_link_adam`
```

(note the use of the backward quotes)

To compile and link a standalone application called 'prog.f' the following is used

```
% f77 prog.f -o prog `gns_link`
```

A Workstation Description File Keywords

CLASS	keyword	The workstation class; the following classes are defined: GRAPHICS_OVERLAY IMAGE_DISPLAY IMAGE_OVERLAY MATRIX_PRINTER METAFILE_INPUT METAFILE_OUTPUT PEN_PLOTTER TERMINAL WINDOW WINDOW_OVERLAY
ERASE_TEXT[2]	character	A character string that can be sent to the terminal to clear the text screen. Control characters are represented by ^. (For devices of class TERMINAL only.)
DEFAULT_NAME[3]	character	The name used by the workstation handler to open the device when a connection identifier of 0 is used. (Not for devices of class TERMINAL.)
SCALE	real	The approximate size in metres of the workstation's device units. This item is ignored if the GKS 'device units' for the workstation are metres.
OUTPUT	keyword	Either DIRECT if the workstation handler sends instructions directly to the device or FILE if they are written to a file.
CLEAR	keyword	SELECTIVE if areas of the display surface can be erased by writing with colour index zero.
OPEN	keyword	NORESET if the device can be opened without resetting the device.
AGITYPE	integer	This is used internally to construct the AGI name. It is mandatory for every IDI workstation. It is also required in GKS for those devices where the AGI type does not correspond directly to the GKS type (e.g. for devices which are image overlays).

B Programmer Interface

When the first GNS routine is called, the GNS database files are opened. It is therefore not necessary to call GNS_START unless you want to verify that the database can be opened before it is accessed. GNS_STOP will close the database but can be omitted in stand alone programs where image exit will close the database.

Most character output arguments are accompanied by an integer length argument; when the requested information is not available, the length argument is returned as zero. This length must therefore always be checked before being used to access a sub-string of the character argument. The symbolic constants representing the maximum lengths of the various character arguments are defined in a FORTRAN include file that can be referenced by the name `GNS_PAR`.

All routines that are specific to GKS have names ending in the letter "G" and routines specific to IDI have names ending in the letter "I".

B.1 Summary of GNS calls

GNS_GTN (NAME, LNAME, STATUS)
: Get terminal name.

GNS_GWNG (FILTER, ICNTX, NAME, DESCR, LD, STATUS)
: Get next GKS workstation name.

GNS_GWNI (FILTER, ICNTX, NAME, DESCR, LD, STATUS)
: Get next IDI workstation name.

GNS_IANG (NAME, AGINAM, STATUS)
: Inquire AGI name of GKS workstation.

GNS_IANI (NAME, AGINAM, STATUS)
: Inquire AGI name of IDI workstation.

GNS_IDNG (IWKID, NAME, LNAME, STATUS)
: Inquire device name of GKS workstation.

GNS_IETG (IWKID, ERTXT, LTXT, STATUS)
: Inquire string to erase text screen.

GNS_IGAG (AGINAM, NAME, STATUS)
: Inquire GKS workstation name from AGI name.

GNS_IIAI (AGINAM, NAME, STATUS)
: Inquire IDI workstation name from AGI name.

GNS_IONG (IWKID, NAME, LNAME, STATUS)
: Inquire overlay device name of GKS workstation

GNS_ITWCG (IWKTYP, CHAR, VALUE, STATUS)
: Inquire a workstation characteristic from its type.

GNS_IWCG (IWKID, CHAR, VALUE, STATUS)
: Inquire a workstation characteristic.

GNS_IWSG (IWKID, SCALE, STATUS)
Inquire workstation scale.

GNS_START (PKG, STATUS)
Start the GNS system for the specified package.

GNS_STOP (PKG, STATUS)
Stop the GNS system for the specified package.

GNS_TNDG (NAME, DEVICE, IWKTYP, ICONID, STATUS)
Translate a name and device to a GKS device specification.

GNS_TNG (NAME, IWKTYP, ICONID, STATUS)
Translate a name to a GKS device specification.

GNS_TNI (NAME, TYPE, DEVICE, STATUS)
Translate a name to a IDI device specification.

C Subroutine Specifications

GNS_GTN
Get terminal name

Description:

The physical device name of the terminal attached to the current process (or its parents) is returned. If there is no terminal available (for example in batch, network or detached processes) the name is set to blanks and the length set to zero.

If the name is longer than the supplied character variable the name is truncated but the length returned is the length of the actual name.

Invocation:

```
CALL GNS_GTN( NAME, LNAME, STATUS )
```

Arguments:

NAME = CHARACTER∗(GNS__SZTER) (Returned)
The device name of the terminal attached to the process

LNAME = INTEGER (Returned)
The number of characters in the terminal name

STATUS = INTEGER (Given and Returned)
The global status

GNS_GWNG
Get next GKS workstation name

Description:

The name and description of the "next" GKS workstation from the list of defined workstation names is returned. If the context argument is set to zero the first name in the list will be returned. The context argument is incremented each time a new name is returned until there are no more names in the list when it will be set to zero.

FILTER is the name of a logical function that is called for each workstation name in the list and can be used to select or reject workstations on criteria such as workstation class. It should return the value .TRUE. if the name is to be included in the list and .FALSE. if it should not. FILTER has one integer argument; the GKS workstation type.

The GNS library contains a suitable function called GNS_FILTG which rejects any workstations that are not supported by the copy of GKS being run (if GKS is open; if it is not all workstation types are selected).

Invocation:

```
CALL GNS_GWNG( FILTER, ICNTX, NAME, DESCR, LD, STATUS )
```

Arguments:

FILTER = LOGICAL FUNCTION (Given)
The name of the filter routine (which must be declared as external in the calling routine)

ICNTX = INTEGER (Given and Returned)
Search context. An input value of zero starts at the beginning of the list; returned as zero when there are no more names in the list.

NAME = CHARACTER*(GNS__SZNAM) (Returned)
Workstation name

DESCR = CHARACTER*(GNS__SZDES) (Returned)
Text description of the workstation

LD = INTEGER (Returned)
Length of description

STATUS = INTEGER (Given and Returned)
The global status

GNS_GWNI
Get next IDI workstation name

Description:

The name and description of the "next" IDI workstation from the list of defined workstation names is returned. If the context argument is set to zero the first name in the list will be returned. The context argument is incremented each time a new name is returned until there are no more names in the list when it will be set to zero.

FILTER is the name of a logical function that is called for each workstation name in the list and can be used to select or reject workstations on criteria such as the workstation class. It should return the value .TRUE. if the name is to be included in the list and .FALSE. if it should not. FILTER has one character argument; the IDI workstation type.

The GNS library contains a suitable function called GNS_FILTI which selects all workstations.

Invocation:

```
CALL GNS_GWNI( FILTER, ICNTX, NAME, DESCR, LD, STATUS )
```

Arguments:

FILTER = LOGICAL FUNCTION (Given)
The name of the filter routine (which must be declared as external in the calling routine).

ICNTX = INTEGER (Given and Returned)
Search context. An input value of zero starts at the beginning of the list; returned as zero when there are no more names in the list.

NAME = CHARACTER*(SZNAM) (Returned)
Workstation name

DESCR = CHARACTER*(SZDES) (Returned)
Text description of the workstation

LD = INTEGER (Returned)
Length of description

STATUS = INTEGER (Given and Returned)
The global status

GNS_IANG
Inquire AGI name of GKS workstation

Description:

The AGI name corresponding to the specified GKS workstation is returned. If the name is longer then the supplied character variable the name is truncated.

Invocation:

```
CALL GNS_IANG( NAME, AGINAM, STATUS )
```

Arguments:

NAME = CHARACTER∗(∗) (Given)
GKS workstation name

AGINAM = CHARACTER∗(GNS__SZAGI) (Returned)
AGI name

STATUS = INTEGER (Given and Returned)
The global status

Notes:

The AGI name is constructed from 'AGI_<wktype>_<seqno>' using the GKS workstation type and a sequence number. The GKS workstation type from the GKSNAMES file is used unless there is an AGITYPE keyword for that device. A sequence number of zero is used unless there is an explicit sequence number in the GKSNAMES file.

This routine is used by AGI and will not normally be called by an applications program.

GNS_IANI
Inquire AGI name of IDI workstation

Description:

The AGI name corresponding to the specified IDI workstation is returned.

If the name is longer than the supplied character variable the name is truncated.

Invocation:

```
CALL GNS_IANI( NAME, AGINAM, STATUS )
```

Arguments:

NAME = CHARACTER∗(∗) (Given)
IDI workstation name

AGINAM = CHARACTER∗(GNS__SZAGI) (Returned)
AGI name

STATUS = INTEGER (Given and Returned)
The global status

Notes:

The AGI name is constructed from 'AGI_<wktype>_<seqno>' using the GKS workstation type and a sequence number. The GKS workstation type is obtained from the AGITYPE attribute. A sequence number of zero is used unless there is an explicit sequence number in the IDINAMES file.

This routine is used by AGI and will not normally be called by an applications program.

GNS_IDNG
Inquire device name of GKS workstation

Description:

The physical device name or file name of the specified GKS workstation is returned. The name may be a logical name that translates to the device or file name rather than the name itself.

If the name is longer than the supplied character variable the name is truncated but the length returned is the actual length of the name.

Invocation:

```
CALL GNS_IDNG( IWKID, NAME, LNAME, STATUS )
```

Arguments:

IWKID = INTEGER (Given)
GKS workstation identifier

NAME = CHARACTER∗(GNS__SZDEV) (Returned)
Device or file name

LNAME = INTEGER (Returned)
Length of name

STATUS = INTEGER (Given and Returned)
The global status

GNS_IETG
Inquire string to erase text screen

Description:

A character string is returned that will clear the text screen if written to the specified device (normally a terminal). The string may contain control characters.

If no string is available a length of zero is returned.

Invocation:

```
CALL GNS_IETG( IWKID, TXT, LTXT, STATUS )
```

Arguments:

IWKID = INTEGER (Given)
GKS workstation identifier

TXT = CHARACTER∗(GNS__SZTXT) (Returned)
Text string

LTXT = INTEGER (Returned)
Length of text string

STATUS = INTEGER (Given and Returned)
The global status

GNS_IGAG
Inquire GKS workstation name from AGI name

Description:

A GKS workstation name that corresponds to the specified AGI name is returned. If the name is longer than the supplied character variable the name is truncated.

Invocation:

```
CALL GNS_IGAG( AGINAM, NAME, STATUS )
```

Arguments:

AGINAM = CHARACTER∗(∗) (Given)
AGI name

NAME = CHARACTER∗(GNS__SZDEV) (Returned)
Device name

STATUS = INTEGER (Given and Returned)
The global status

Notes:

This routine is used by AGI and will not normally be called by an applications program.

GNS_IIAI
Inquire IDI workstation name from AGI name

Description:

An IDI workstation name that corresponds to the supplied AGI name is returned. If the name is longer than the supplied character variable the name is truncated.

Invocation:

```
CALL GNS_IIAI( AGINAM, NAME, STATUS )
```

Arguments:

AGINAM = CHARACTER∗(∗) (Given)
AGI name

NAME = CHARACTER∗(GNS__SZDEV) (Returned)
Device name

STATUS = INTEGER (Given and Returned)
The global status

Notes:

This routine is used by AGI and will not normally be called by an applications program.

GNS_IONG
Inquire overlay device name of GKS workstation

Description:

The name of the overlay device for the GKS workstation specified by the workstation identifier is returned. This can then be used to open the device with GKS. If the device does not have an overlay then an error is returned.

If the name is longer than the supplied character variable the name is truncated but the length returned is the actual length of the name.

Invocation:

```
CALL GNS_IONG( IWKID, NAME, LNAME, STATUS )
```

Arguments:

IWKID = INTEGER (Given)
GKS workstation identifier

NAME = CHARACTER∗(∗) (Returned)
Overlay device name

LNAME = INTEGER (Returned)
Length of name

STATUS = INTEGER (Given and Returned)
The global status

GNS_ITWCG
Inquire workstation characteristic from its type

Description:

The specified characteristic is returned as a character string, blank filled or truncated as necessary. If the characteristic does not exist then a blank string is returned.

Any characteristic with a keyword value (see appendix A) can be

Invocation:

```
CALL GNS_ITWCG( IWKTYP, CHAR, VALUE, STATUS )
```

Arguments:

IWKTYP = INTEGER (Given)
GKS workstation type

CHAR = CHARACTER∗(∗) (Given)
Characteristic name

VALUE = CHARACTER∗(GNS__SZKEY) (Returned)
The value of the requested characteristic

STATUS = INTEGER (Given and Returned)
The global status

Notes:

This performs the same function as GNS_IWCG except that the device is specified by its GKS type and therefore the characteristics can be queried before the device is opened by GKS.

GNS_IWCG
Inquire workstation characteristic

Description:

The specified characteristic is returned as a character string, blank filled or truncated as necessary. If the characteristic does not exist then a blank string is returned.

Any characteristic with a keyword value (see appendix A) can be inquired with this routine.

Invocation:

```
CALL GNS_IWCG( IWKID, CHAR, VALUE, STATUS )
```

Arguments:

IWKID = INTEGER (Given)
GKS workstation identifier

CHAR = CHARACTER∗(∗) (Given)
Characteristic name

VALUE = CHARACTER∗(GNS__SZKEY) (Returned)
The value of the specified characteristic

STATUS = INTEGER (Given and Returned)
The global status

GNS_IWSG
Inquire workstation scale

Description:

The size in metres of 'device units' for the specified GKS workstation is returned.

Note that device units as defined by GKS are not necessarily the same as the device resolution. On devices with a well defined actual size such as plotters and printers device units are metres and a value of 1.0 will be returned.

Invocation:

```
CALL GNS_IWSG( IWKID, WSSCA, STATUS )
```

Arguments:

IWKID = INTEGER (Given)
GKS workstation identifier

WSSCA = REAL (Returned)
Size in metres of device units

STATUS = INTEGER (Given and Returned)
The global status

GNS_START
Start the GNS system

Description:

The GNS databases for the specified package are opened. This routine is called automatically by any other routine that accesses the databases.

Invocation:

```
CALL GNS_START( PKG, STATUS )
```

Arguments:

PKG = CHARACTER∗(∗) (Given)

The package name. The only packages currently supported are GKS and IDI.

STATUS = INTEGER (Given and Returned)

The global status

GNS_STOP
Stop the GNS system

Description:

The GNS databases for the specified package are closed.

Invocation:

```
CALL GNS_STOP( PKG, STATUS )
```

Arguments:

PKG = CHARACTER*(*) (Given)

The package name. The only packages currently supported are GKS and IDI.

STATUS = INTEGER (Given and Returned)

The global status

GNS_TNDG
Translate name and device to GKS specification

Description:

The workstation name and physical device specification are translated to a GKS workstation type and connection identifier and, if necessary, a logical name created to map the connection identifier onto the specified physical device or file.

Invocation:

```
CALL GNS_TNDG( NAME, DEVICE, IWKTYP, ICONID, STATUS )
```

Arguments:

NAME = CHARACTER∗(∗) (Given)
Workstation name

DEVICE = CHARACTER∗(∗) (Given)
Physical device name

IWKTYP = INTEGER (Returned)
GKS workstation type

ICONID = INTEGER (Returned)
Connection identifier

STATUS = INTEGER (Given and Returned)
The global status

Side Effects :

An environment variable GCON or of the form GKS_nnnn may be created.

A GWM window with an overlay may be created.

GNS_TNG
Translate name to a GKS device specification

Description:

The workstation name is translated to a GKS workstation type and connection identifier and, if necessary, a logical name created to map the connection identifier onto the device implied by the workstation name.

This routine is the same as GNS_TNDG but without an explicit physical device name argument.

Invocation:

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

Arguments:

NAME = CHARACTER∗(∗) (Given)
Workstation name

IWKTYP = INTEGER (Returned)
GKS workstation type

ICONID = INTEGER (Returned)
Connection identifier

STATUS = INTEGER (Given and Returned)
The global status

GNS_TNI
Translate name to an IDI device specification

Description:

The workstation name is translated to an IDI workstation type, and a physical device name.

Invocation:

```
CALL GNS_TNI( NAME, TYPE, DEVICE, STATUS )
```

Arguments:

NAME = CHARACTER∗(∗) (Given)
Workstation name

TYPE = CHARACTER∗(∗) (Returned)
IDI workstation type

DEVICE = CHARACTER∗(∗) (Returned)
Device name

STATUS = INTEGER (Given and Returned)
The global status

Notes:

IIDOPN requires a GNS workstation name as its name argument which is translated by IIDOPN calling GNS_TNI; GNS_TNI is therefore not normally called by applications programs.

Side Effects :

A GWM window with an overlay may be created.