GRAPH/NET Computer Integrated Design System

Introductory User Manual

This manual provides an introduction to the GRAPH/NET Computer Integrated Design System The manual explains general concepts of the GRAPH/NET system; you should be familar with these concepts before using the GRAPH/NET.

Graphic Horizons, Inc.
60 State Street
Boston, Massachusetts 02109
617-396-0075

The information in this document is subject to change without notice and should not be construed as a commitment by Graphic Horizons, Inc. The Company assumes no responsibility for any errors that may appear in this document.

Graphic Horizons, Inc. will make every effort to keep customers apprised of all documentation changes as quickly as possible.

Table of Contents

1. Introduction.

The GRAPH/NET Computer Integrated Design system is based upon the PERQ2 computer from PERQ Systems Corporation. The PERQ provides an integrated computing system for a single user. The GRAPH/NET system comes with the following hardware features:

a) 16-bit micro programmed processor
b) High-resolution graphics using a 1280 X 1024 bit mapped raster display
c) Standard keyboard with special function keys
d) 35-megabyte Winchester technology disk drive
e) Two RS232 & One GPIB I/O interfaces
f) 1 megabyte of main memory
g) bit pad and four button puck

When you receive your GRAPH/NET system, the hard disk contains a bootable copy of the most recent version of the operating system. Successive versions will provide software enhancements as well as corrections to some inevitable bugs. Floppy disks or streaming tape serve as the distribution media for successive operating system versions.

The operating system versions are identified by a letter, to indicate the major level, and a number to indicate the minor level. Both the letter and number are successive. The current software version is R.5.

2. Turning the computer on.

The power switch is located on the front panel of the computer to the right hand side side of the floppy disk drive, and is labeled OFF/ON. Pushing the top of the switch in powers on the GRAPH/NET system. When you switch on the machine on, the cooling fans should start, and the screen will switch on a few seconds later. If this does not happen, check that the machine is plugged into the electrical outlet, and that there is power at that outlet. If there is power at the outlet and you still do not hear the fans then call your GHI service representative.

The computer takes about one minute to boot after it has been turned on, allowing the disk to spin up to speed. At this point the Diagnostic Display (DDS) located on the base of the screen should read 999 and the screen should display a request for you to login.

If the computer has not booted after two or three minutes, try pressing the Boot button on the right hand side of the screen. The machine will attempt to re-boot, which should take no more one or two minutes. If this boot attempt is not successful, call your GHI service representative.

After the computer boots, you will be asked to login. You must supply a user name and password to be allowed access to the machine. After the operating system has been loaded onto a machine, the names defined are "Guest" and " " (just hit the Return key, located to right hand side of the keyboard), neither of which require the use of a password. Your GHI representative will assist you to add other user names if you wish to do so. After you have successfully logged in, the system will be loaded and you will be able to execute commands.

3. How to get some HELP.

To get a list of the commands available to you, type a question mark (?) and press the Return key. The computer will print a list of the commands and a very brief description of their function. To get more information on most commands, type the command name and press the HELP key, located at the top left hand side of the keyboard. So, for example, to get information on the Copy command, type :

```
copy <HELP>
```

which will appear as

```
copy/HELP
```

The computer will respond by typing some more information about the command, if it is available.

To get some more information on the system, simply press the HELP key by itself. This will load a program called Helper, which will give a list of topics for which help is available, and wait for you to type in the name of the topic in which you are interested. You know that Helper is waiting for you to respond because the normal prompt ( ) has been replaced by HELPER To get out of Helper at any stage, just press the RETURN key.

4. Logging Out and Turning the computer off.

Before turning the computer off, you should always log off using the Bye program. If you just power down, some temporary files on the disk will not be cleaned up. Type:

BYE OFF

and press the Return key, to log off and turn the power off for the machine automatically. If this is not possible, you can always turn the machine off by pushing in the bottom of the power switch.

If you wish to leave the computer on for another user, you can log off by simply typing

BYE

and pressing the Return key.

5. Control Characters and Special Keys.

The GRAPH/NET operating system recognizes a number of special control characters. These control characters perform simple input line editing and program control.

The valid control characters are:

BACK SPACE or ^H - erases the last character typed by the user.

^W or ^BACK SPACE - erases the last word typed by the user.

OOPS or ^U - erases the last line typed by the user.

^C - typed once causes a current program to abort the next time that it asks the operating system for input.

^C - typed twice causes the current program to abort immediately. However, this does not cause user command files to exit; see ^SHIFT C.

^S - causes program output to the screen to be suspended.

^Q - allows output to the screen to resume after a ^S.

^SHIFT C - causes a dump of the runtime stack and an immediate return to the Shell. If a command file was being executed, it is aborted and control is returned to the keyboard.

^SHIFT D - causes a dump of the runtime stack to be printed and the preliminary debugger (called Scrounge) can then be entered. If the debugger is not entered, the original program will resume execution. If the debugger is used, the user can request that the program be resumed after investigating the state.

HELP - pressing the HELP key by itself displays a general help message. If you press the key after typing a command, the display contains specific help information for the command.

6. The Command Interpreter, The Shell.

The command interpreter for the GRAPH/NET operating system runs as a separate user program.

The Shell takes commands from the user terminal or from a user command file and executes them. It does not distinguish between upper and lower case letters; you can use whichever you prefer. The commands in a user command file look exactly as if they were typed at the terminal. However, the cursor is a lighter shade so that you can distinguish command file execution from a typed command.

The general form of a Shell command line is a command or program name, followed by any number of optional parameters, followed by any number of optional switches. Some of the switches take parameters. For example:

```
Compile SourceProgram/Symbols=32
```

compiles a Pascal program in file SourceProgram using 32 symbol table blocks. All switches begin with a "/".

An example of a program that takes a number of parameters on the command line is the Copy program. The form is:

```
Copy SourceFile~DestinationFile
```

When you supply a command line to the Shell, it extracts the first symbol on the line and does a unique substring lookup of the symbol against a small set of commonly used commands. You can get a list of these commands by typing "?" or the HELP key. If a match is found, the Shell executes the command. If no match is found, the Shell assumes the symbol is the name of an executable runfile (the output of the Linker), and attempts to execute it. Commands can also be invoked by means of a menu if you have booted from the harddisk. See the section on "PopUp Menus" below.

7. Specifying Commands and Arguments

The Shell uses a default file name as the parameter to certain programs if no parameter is provided. This file name is the last file name typed to one of these programs. The Editor, Pascal Compiler, and Linker use and set the default file name. The TypeFile program uses the default file name but does not set it.

Other programs require that you specify arguments. If a command requires an input and an output file name, you can specify the file names as either

INPUT~OUTPUT

or

INPUT OUTPUT

You can separate the input and output file names with multiple spaces; only the initial space delimiter is relevant.

However, if a command accepts multiple input or output arguments, you must separate like arguments with commas (,) and distinguish input from output with the tilda character (~). For example:

input1,input2,...inputn ~ output1,output2,...outputn

If a command accepts multiple input arguments and no output arguments, you must separate the arguments with a comma (,)

Switches modify the action of the command and therefore must follow the command specification. An exception is the Help switch (either type /HELP or press the HELP key); when specified before the command, the Help switch supplies general information and when specified after a command, the Help switch provides specific information. Switches always start with a slash (/) and are generally optional. If a switch accepts a parameter, specify the parameter after the switch, but preceded by an equal sign (=). For example:

/switch=parameter

The effect of a switch is global; regardless of where the switch appears on the command line, it has the same effect. A switch applies to every argument. If a command accepts multiple input or output arguments, no switch applies to one and not another argument.

For most programs on the PERQ, arguments that have defaults will print the default answer in square brackets ("[]"). This answer can be chosen by simply typing a carriage return (thus providing an

empty argument). To supply a different value, type the value followed by a carriage return. For example, if:

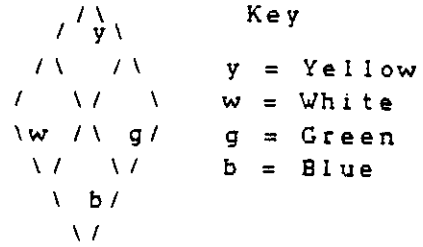
```
Delete FileName.Seg [No]:
```

is the prompt for an argument, a carriage return means no. Of course, you could type "yes" or "no" as the argument. Most programs do not distinguish between upper and lower case for arguments or commands. In addition, you can abbreviate a command to the number of letters unique to other command names. However, this does not work for filenames.

The user's profile file specifies the set of commands recognized by the Shell. You can use a copy of the file DEFAULT.PROFILE initially. Later, you may wish to define commands of your own. Refer to the section on "Profiles" below.

8. The Pointing Device.

GRAPH/NET systems are supplied with a Summagraphics bitpad and four-button puck. The four button puck uses a circle with cross hairs for positioning. The four buttons are arranged in a diamond as shown below.



If you press a button on the puck, a small switch closes. This is called a "press" of the puck or pen. The programs use the puck for pointing and drawing.

The computer reads the puck position when the puck is in contact with the tablet surface and displays a cursor on the screen.

9. PopUp Menus.

The Shell and the FLOPPY and FTP utility programs allow their arguments to be entered by means of menus. The menu holds a list of all legal commands or arguments, and the puck can be used to specify the selection. Using a menu is sometimes more convenient than typing out the name of the command or argument desired. PopUp menus appear when requested. When the selection is made, the menu disappears and restores the screen space that was covered by the menu. Using PopUp menus, therefore, does not require sacrificing any screen area.

For the Shell, you can invoke a menu whenever the prompt (">") is displayed and no characters have been typed. (The prompt is actually a dark grey, right pointing triangle.) To invoke a PopUp menu, simply press the puck. The menu appears at the current cursor position. The cursor can then be moved up or down inside the menu to select the desired command or you can press the HELP key for help on PopUp menus. The selected command is highlighted by reverse video. A press over the selected command causes it to be invoked just as if the user had typed the command on the keyboard.

If the menu is not large enough to hold all the commands, a scrolling mechanism is provided. When scrolling is necessary, a "gauge" is displayed in a black border at the bottom left of the menu. When you move the cursor over this area, the cursor changes to a scroll. A press here allows scrolling. Moving the cursor to the right while pressing causes the menu text to scroll up and moving the cursor to the left while pressing causes the text to scroll down. The further the cursor is moved from the original press position, the faster the text scrolls. A line in the gauge shows how fast the menu is scrolling. Of course, you cannot scroll past either end of the menu (Each end is signified by a row of "~"). Releasing the button causes scrolling to stop.

If you press in an illegal part of the menu, or if you try to invoke a menu and have typed some text, the computer beeps. If a menu is displayed and you press outside of the menu, the menu disappears, causing no side effects. In addition, if you type ^C or ^Shift-C while a menu is displayed, the menu disappears.

10. The "Lights".

The computer does not have a front panel full of lights like many computers. To show the occurrence of certain long operations, the computer simulates these lights by inverting small squares on the top of the screen. If the standard system window is displayed, the lights appear inside the title line.

Currently, the operating system uses three lights. The leftmost is used to show when the computer is doing a disk or floppy recalibration. This is done when the microcode is confused about where the heads are. Opening the floppy door during a disk operation usually causes this to happen. The light appears about 5/8 inches from the left margin.

The next light is used when the file system is attempting to fix up a file at run-time. This is needed occasionally when a file system operation has not completed normally. This light appears about 1-1/2 inches from the left margin.

The last light defined by the operating system is used for swapping. While a swapping operation (swap in or swap out) is in progress, this light is "lit." The swapping light appears about 2-1/4 inches from the left margin.

11. Specifying a Filename.

This is a brief overview, a short introduction to paths and filenames, a description of the various ways to specify a filename, and an explanation of the wildcard convention.

Overview of the File System

The GRAPH/NET file system has a hierarchical structure. This is reflected in the syntax of filenames. Files are stored on disk, which is divided into partitions. Each partition contains a number of files and directories; each directory may contain other directories and files.

Introduction to Path and File Names

The route that is traveled to reach a filename is called a PATH. A full path name specifies device, partition, directories, and filename, in that order. The syntax of a path name is:

```
device:partition>[directory>]filename
```

The brackets surrounding the "directory" part indicate that there may be zero to nine occurences of ">directory". Note that if you specify a directory, the brackets are NOT part of the syntax. To specify the current directory, or one of its subdirectories, you can omit the :partition> and the directory> syntax elements.

1. Device names

Examples of devices include floppy, streaming tape and hard disk. The syntax used to denote a device name is:

```
device:
```

If you omit the device name, the system assumes the name of the device you booted from.

A device is accessable when it is mounted.

2. Partitions are set and named at device initialization time. You can modify partitions using the Partition program, described later.

You must exercise extreme caution in modifying disk partitions, since you can erase files and programs if you are not careful.

Each device is divided into a fixed number of partitions; the recommended size for a partition is 10,080 or fewer 256-word blocks. Files must fit entirely within a single partition as they cannot cross partition boundaries. Generally there are five partitions on a 24-megabyte disk. Examples of partition names are BOOT and DRAFTNET.

3. Directories are easy to create, destroy, rename, and move around. There can be multiple directories in a partition. These are denoted by the symbol > and can take names of up to 25 characters. There can be up to 9 directories listed in a full path name. The symbol:

..

is a convenient way of referring to a parent node in a tree. It goes up one node. An example of its use is:

SYS:BOOT>NEW>..>filename

This will look up filename in SYS:BOOT, rather than in SYS:BOOT>NEW>. The symbol

.

refers to the current directory.

4. Filename is the last thing specified in a file specification. A filename can have up to 25 characters.

The names of all of the directories and the filename cannot exceed 80 characters. You can include most special characters in a filename by preceding the special characters with a single quote ('). For example, you could include the asterisk (*) in a filename by specifying '*. Note that you cannot include the special and control characters described in section 4.

When you boot, the system takes as default device and partition the device and partition that you booted from. For example, the system might come up with a default of:

SYS:BOOT>

Ways to Specify a Filename

1. You can specify a full path name:

device:partition>[directory>]filename

the brackets indicate that you can list up to nine directories. Specifying a full path name permits you to access a file on a different device from the one you're using.

2. You can use the default device that was determined at boot time:

:partition>[directory>]filename

Using this syntax enables you to look for a file in a different partition.

3. You can use the default device and partition that were set at boot time:

>[directory>]filename

The search then starts at the first directory specified.

4. You can just type:

filename

and the search begins at the current directory (which may be set by you, using the PATH command). This can involve any device, any partition, and any directory in that partition.

Setting the default path doesn't affect the default device and partition used in forms 1, 2, and 3.

Wildcard Convention

A number of programs use a wildcard convention when looking up files. The wild cards are as follows:

* matches 0 or more characters.
& matches 1 or more characters.
# matches exactly 1 character.
'0 matches any digit.
'A or 'a matches any alphabetic.
'@ matches any non-alphanumeric.
'* matches *. Other wild cards can be quoted also.

There can be any number of wildcards in file specifications to programs that handle this convention. Examples of wildcard usage are:

```
Dir *.Pas

Dir &Boot*
```

The first command gives a directory of all files with a .pas extension. The second command gives a directory of

all files that have one or more characters followed by the characters "boot", followed by zero or more characters.

12. Default File Extensions.

An extension is a conventional sequence of characters that appears at the end of a filename. In the computer's operating system there is nothing special about extensions. However, there are certain conventions that are used in the system. An extension is found by finding the last "." in the filename and taking the following symbols. Backup files conventionally have a "$" as the last character of their last extension; they take the form Name.Ext$. Following is a list of the standard extensions and how they are used.

.PAS - Pascal source files usually have this extension.

.SEG - The Pascal compiler produces .SEG files when it compiles a .PAS file. The .SEG files are used as input to the Linker and contain the code that will be executed when the program is run.

.RUN - Files produced by the Linker have this extension.

.CMD - A number of programs in the GRAPH/NET system accept commands from a file as well as from the keyboard. Files that contain the commands usually have this extension.

.DR - In the computer's file system directories are files. These files appear in a directory with the extension .DR.

.KST - Character set definitions are kept in files that have the extension .KST.

.MBOOT, .BOOT - When the computer is booted it reads the microcode interpreter and Pascal operating system from files that have the extension .MBOOT and .BOOT.

.INDEX - An index to .HELP files.

.DOC - Formatted documentation has this extension.

.HELP - Files containing help about a system use this extension.

13. Booting the Machine.

The computer can be started up, or booted, in one of two ways. First, when the machine is powered up it will go through the boot sequence. Second, the m achine can be booted by pressing the Boot button on the back of the keyboard. In either case the same sequence of events happens.

The boot sequence for the computer has three steps. As each of these steps progresses, the Diagnostic Display, DDS, increments. The DDS is a three-digit number display under the keyboard. If any step fails it is possible to look at the Diagnostic Display and see where in the boot sequence the failure occurred. The Fault Dictionary provides a list of the DDS values and explanations of their meanings.

In the first part of the boot sequence, microcode is executed out of a small ROM. This ROM covers the lower 2k of standard control store during this part of the boot sequence. This microcode runs a simple diagnostic on the processor and memory systems. If there are any errors, the microcode halts. The value in DDS gives the reason that the machine halted. Once these diagnostics have been passed, the microcode makes a decision about which device is to be used for booting. Currently, there are three possible boot devices.

1. The first alternative is booting from the hard disk. The microcode tries to boot from the hard disk.

2. The second choice of boot devices is a floppy disk. The microcode checks to see if there is a floppy in the floppy drive. If there is a floppy in the drive, PERQ will check to see if the floppy is a boot floppy. If so, the second part of the boot will be done from the floppy.

3. The third possible boot device is another PERQ. The microcode determines if there is a PERQ Link Board plugged into the I/O Option slot of the machine. If the board is plugged in, and there is another PERQ on the other end of the link, the booting PERQ will wait for commands from the link.

If all of these fail, then the DDS will contain an indication of what the error is. See the Fault Dictionary Manual for an explanation of the display number.

After the boot device has been chosen, the second part of the boot sequence can begin. In this part, the computer reads 3k words of microcode from the selected boot device. This microcode is in two sections, a more extensive diagnostic (VFY) and a system boot loader (SYSB). VFY attempts to verify that all of the CPU and Memory systems are working. Any failures that VFY discloses are displayed on the DDS.

If all went well, the microcode determines what set of interpreter microcode and system Pascal code is to be loaded. It does this by checking to see if any key is being held down on the keyboard. If a key is being held down, that key specifies which boot is to be done. If no key is held down, the default boot will be done. The default is the same as holding down "a". Hold the key down until a pattern flashes on the screen. Any of the 26 alphabetic keys can be used to specify a boot. All lower-case characters cause a boot from the hard disk. Upper case characters cause a boot from floppy. If you type "Details/Boots" the Details program will provide you with a list of all of the valid boot characters. Once a boot has been chosen, the microcode interpreter and computer operating system are loaded into the PERQ. Control is then transferred to the third portion of the boot sequence.

In the third portion of the boot sequence, the interpreter microcode does any initialization that is needed and then starts to execute the PERQ operating system. The PERQ operating system also increments the DDS. If there were no errors during the boot sequence, the machine will be running and the DDS will read 999.

14. Run-time Errors.

When the operating system or a user program discovers an error condition, it raises an "exception." If a Pascal exception is not handled, it is given to the preliminary debugger, called Scrounge. First, a dump of the user state is produced and then the user is asked if he wants to debug. If not, the program is aborted and control returns to the Shell and any active command files are terminated. If the user decides to debug, the program can be continued after the point where the exception was raised. It is usually a bad idea to continue from uncaught exceptions. The user can also abort the program and return to the Shell.

15. Profiles.

Profiles can be used to tailor your GRAPH/NET system to its users. The profile is a text file which contains commands that define characteristics of certain utility programs. For example, a profile can direct the Login program to initialize the default path and searchlist. Each user of the system can have his own profile; the UserControl program can assign each user a profile file.

To create a profile file, copy DEFAULT.PROFILE to your own directory. For example:

```
COPY SYS:BOOT>DEFAULT.PROFILE SYS:DRAFTNET>MYPROFILE
```

Now run UserControl and specify SYS:DRAFTNET>MYPROFILE for the profile file. You can then edit the file and establish your specific conventions.

Each entry in the profile file begins with a number sign (#), followed by the name of the subsystem. For example, #LOGIN. The subsequent lines contain switches or data for the subsystem. The general format of a profile file is as follows:

```
#program1 <switches or input for program1>
          <more data for program1>
#program2 <switches for program2>
   . . .
```

For example:

```
#Login /Path=Sys:Draftnet>
       /SetSearch=Sys:Boot>Library
       /CursorFunction=7
#RandomUtility /MaxSize=100
```

The format of each list of entries in the profile is defined by the utility program that uses the profile. You should read the documentation for a particular utility to determine whether it reads the profile, and if so, what entries can be included in the profile.