GPSS/H System Guide

www.wolverinesoftware.com

1. INTRODUCTION (Be sure to read this.)

What Is In The System Guide?

The System Guide contains three kinds of information:

- Brief tutorial-style sections on how to install and run GPSS/H Professional, Personal GPSS/H and Student GPSS/H.
- A reference-style table describing options that you can specify when you run GPSS/H, in order to change the way your model will be compiled and/or executed.
- Reference-style sections with information that can be helpful if your GPSS/H model reads or writes files, or if you encounter memory shortages.

How Much Do I Really Need To Read?

Only the following is essential:

- Check Section 2 to make sure that your computer meets the minimum requirements for using GPSS/H.
- Read Sections 3 through 6; then install and try out the software.
- Glance at Table 1 just enough to get an idea of what it contains. You can come back to it later, as needed.

Read other sections only if they apply to you:

- If your GPSS/H model will read or write data files while running, or interact with its user, read Section 7.

Do I Need to Read Anything Else?

Any “late-breaking news” not covered in this document will be described in a file named "readme.txt" file on the GPSS/H CD. If this file does not exist on the CD, you may assume that these instructions are up-to-date.

If you have questions about the GPSS/H language, consult the GPSS/H Reference Manual.

2. SYSTEM HARDWARE AND SOFTWARE REQUIREMENTS

- IBM-compatible computer with an 80486DX or higher CPU
- Windows 95/98, Windows ME, Windows NT4, Windows 2000, or Windows XP.
- Disk space: 3-20MB for GPSS/H and other software; 1-50 MB for user models.
- Parallel port or USB port (for plugging in a security key – not required for Student GPSS/H)

3. INSTALLING AND REMOVING GPSS/H

3.1 Installation

GPSS/H is distributed on a CD-ROM. To install GPSS/H, simply insert the CD-ROM into a CD-ROM drive on your computer. Unless you have disabled the automatic recognition of CD insertion, Windows should automatically begin executing the software installation procedure on the CD. If the CD is not automatically recognized, you can manually run setup.exe, contained on the CD.

The installation procedure for GPSS/H requires Microsoft's Windows Installer, which is built into Windows 2000, Windows ME, and Windows XP, but not Windows 95/98 or Windows NT. On systems that do not have the Windows Installer, setup.exe will copy an appropriate version of the installer from the CD to the system's hard drive.

The installation procedure will lead you through a sequence of dialogs, allowing you to specify which software components you'd like to install and where you'd like to place them on your computer. By default, files are placed into the folder named Program Files\Wolverine and into Wolverine-defined subfolders of that folder. If you wish to choose a different base folder, you may do so; however, you should not move, remove, or rename any subfolders created by the installation procedure. The folder architecture we have defined provides a convenient way of installing a variety of products from Wolverine Software.

3.1.1 Security Keys

GPSS/H Professional and Personal GPSS/H require the use of a security key (dongle) that plugs into a parallel or USB port on your computer. Student GPSS/H does not require a security key. As part of the installation process for the key-protected versions of GPSS/H, software drivers are installed to enable GPSS/H to access the key.

To install GPSS/H Professional or Personal GPSS/H under Windows NT4, Windows 2000 or Windows XP, you need to be logged on with administrator privileges. If you are installing GPSS/H Professional or Personal GPSS/H under either of these operating systems, and you do not have administrator privileges, you'll have to get assistance from an authorized system administrator.

3.1.2 USB Port Security Keys

Wolverine provides two types of keys, one that plugs into your computer's parallel port, and one the plugs into your computer's USB (Universal Serial Bus) port. USB keys are available only for Run-Time GPSS/H. In case you've never used a USB port, a brief description follows. USB devices use a rectangular plug-and-socket arrangement, about the size of the end of your little finger. Our USB security keys have only a male plug and no female socket. With no provision for "pass-through," a USB key must be plugged into its own USB socket. If all the USB sockets on your computer are in use, and you wish to use a USB key, you'll have to buy a USB hub. A USB hub allows you to connect multiple USB devices to a single USB socket on your computer.

Do not plug in a USB security key until after the GPSS/H installation procedure has completed.

If you plug a USB key in before the installation is completed, Windows will recognize that a new USB device has been connected, but it will have difficulty figuring out how to handle the new device. If you plug the key in after the software installation is complete, Windows will automatically recognize the security key and setup USB driver software for accessing it.

3.2 Modifying, Repairing, and Removing GPSS/H

The Windows Installer provides three actions that can be performed on installed software: Modify, Repair, and Remove.

You can use the "Modify" action to add or delete software components. For example, we provide a demo of our Proof Animation software along with GPSS/H. If you elected not to install this demo when you installed GPSS/H, but later decided you'd like to take a look at it, you could use the "Modify" option.

The "Repair" action causes Windows to replace any files that were originally installed but have become corrupted.

The “Remove” action removes all GPSS/H files that were installed.

To modify, repair, or remove GPSS/H, click Start, Settings, Control Panel, Add/Remove Programs, and select GPSS/H from the list of programs that is displayed. Once you select GPSS/H, a dialog will appear, giving you the three options described above.

4. WINDOWS SESSIONS

4.1 Introduction

GPSS/H operates as a Windows console-mode application and must be launched from a command line. The installation procedure creates icons on your desktop for initiating command prompt sessions for the version(s) of GPSS/H you install, and it places a shortcut in the Programs\Wolverine folder. When you click on the icon or invoke the shortcut in the Programs menu, a command prompt session is initiated. To invoke GPSS/H, you must issue commands (described in Section 5.3).

The shortcuts created by the installation procedure initiate windowed GPSS/H command prompt sessions. Under Windows 2000 and Windows XP, you can modify the width and height of the command prompt window from its default 80 x 25 size. We strongly recommend that you do so, since a larger window will reduce the need to scroll debugger/output windows.

If you change the size of the command prompt window, set the scrolling region to be the same size as the window itself. Otherwise, confusing “double scrolling” will result. (GPSS/H provides its own scrolling via function keys, described in Table 2. If you make the scrolling region smaller than the window’s size, Windows will provide scroll bars.)

4.2 Memory Usage

GPSS/H automatically uses *virtual* memory, which uses a swapfile on disk to simulate additional physical ("real") memory, ***if needed,*** at some potential cost in execution speed. This means that the maximum memory available to GPSS/H depends on both the amount of physical memory installed ***and*** the size of the Windows swapfile, and can exceed the amount of physical memory installed on your computer. Approximately 700KB of virtual memory is needed to hold GPSS/H itself; memory required by your GPSS/H model is in addition to that amount.

Under Windows, the amount of memory available to GPSS/H can sometimes differ dramatically from run to run when other software is executing at the same time. This is inherent in the nature of the operating system itself, and is not specific to GPSS/H.

GPSS/H’s* memstats *option can provide helpful information on memory usage.

If you need very large amounts of memory to run your models, try to meet as much of the need as possible by installing more physical memory (to maintain performance). You may also need to make your Windows swapfile(s) larger. Information on how to set up and modify swapfiles can be found in your Windows documentation.

5. HOW TO RUN A GPSS/H MODEL— A QUICK TUTORIAL

5.1 Managing Multiple Projects

To manage a multiplicity of simulation projects, you should create separate folders for each project. You can create these folders as subfolders of Wolverine/GPSSH or place them anywhere you wish. GPSS/H models, data files, and so on for each project can be stored in the project folder. To work on a project, simply execute one or more "`cd`" commands, as necessary, to make the project directory your current directory, and then run GPSS/H by issuing commands, as described in Section 5.3.

5.2 Overview

GPSS/H compiles and executes user-created GPSS/H models. GPSS/H models are ASCII text files that are created with a text-editor or word-processor. The files are made up of GPSS/H Control Statements, Block Statements, and comments. When you run GPSS/H, you will supply the name of the file containing the model that you want GPSS/H to compile and execute. You may also specify the names of options that change the way in which your model is compiled and/or executed.

5.3 Creating GPSS/H Model Files and Compiling & Executing Them

In order to use GPSS/H for your models, you must do the following:

A. ***Enter the GPSS/H model into a plain (ASCII) text file*** using any convenient editor. (The Windows NotePad or WordPad applets will work fine for this purpose with small- to medium-sized files.) If you do not have an editor on your computer, but you do have a word-processing program, you can use the word-processing program as an editor as long as it can save files in "text only" format.

Be sure to save a word-processed model as "text only." **Beware that some editors may insist on appending a ".txt" file extension to all files saved as text files. For example, the first time you save a file under WordPad, if you try to save it as "myfile.gps", WordPad will save it as "myfile.gps.txt". If this happens, you can rename the file to the desired "myfile.gps" and reopen it with WordPad. WordPad does *not* append ".txt" extensions to pre-existing files, so the "double extension" problem occurs only on the first attempt to save a new file.**

Earlier versions of GPSS/H required that models be entered in UPPER CASE LETTERS. The latest versions of GPSS/H no longer impose this restriction; however, you must be consistent in your use of user-defined names. "Joe", "joe", and "JOE" are different names.

GPSS/H, like other implementations of GPSS/H, assumes that 132 columns of screen and printer width are available. When GPSS/H is running in "test" (interactive) mode, left-right scrolling is implemented using the "<–" and "–>" keys, so that you can examine all 132 columns of output. We recommend using an editor that supports 132 columns (either through left-right scrolling or through extended video modes) for viewing listing ("`.lis`") files, and using a printer that supports 132 columns for printing them.

B. Compile and execute the model under GPSS/H Professional by using the *hpro* command. The format of the *hpro* command is as follows:

hpro *filename* [*options*...]

In the above, *filename* is the name of the file containing the GPSS/H model. If a file extension is specified as part of *filename* (e.g., `myprog.mdl`), it is used by GPSS/H; if no file extension is specified, GPSS/H will ***assume*** a file extension of "`.gps`."

[*options*...] represents a sequence of zero or more keywords (option names) that change the way GPSS/H compiles and/or executes a model. ***A complete list of all options available under GPSS/H is given in Table 1.*** If any options are specified, they are processed one at a time in left-to-right sequence.

If no options are specified, all "standard" model output is placed in a listing file with the same name as the model file, but with a file extension of ".lis". The file is created ***in the directory that contains the model source file.*** If the `type` option of the GPSS/H command is used, "standard" model output is directed to the screen and no listing file is created; `type` and the other options of the *hpro* command are described in detail in Table 1.

To compile and execute a model under Personal GPSS/H, type *hpers32*, rather than *hpro.* Other than the command name, the format of the *hpers32* command is identical to that of the *hpro* command, described above

To compile and execute a model under Student GPSS/H, type *gpssh*, rather than *hpro.* Other than the command name, the format of the *gpssh* command is identical to that of the *hpro* command, described above

GPSS/H contains a very powerful and easy-to-use ***interactive debugger.*** You will want to use this debugger as you develop and test your models. You can learn it, as needed and at your own pace, by consulting Chapter 11 of the GPSS/H Reference Manual.

5.4 Command Line Examples

The following examples illustrate how to run a model by using the *hpro* command in combination with several common options. Remember, if you're using Personal GPSS/H, type *hpers32*, rather than *hpro*, and if you're using Student GPSS/H, type *gpssh*, rather than *hpro*.

```
hpro myprog
```

(Source, dictionary, and cross-reference listings and model output are placed in a file named `myprog.lis.`)

```
hpro myprog type
```

(Source, dictionary, and cross-reference listings are suppressed. Model output appears directly on the screen. No listing file is created.)

```
hpro myprog tvtnw
```

("`tvtnw`" is an abbreviation for "`tv type nowarn`". This combination of three options is handy for making quick debugging or model-checkout runs. Source, dictionary, and cross-reference listings are suppressed. Compile-time warning messages are suppressed. Model output appears directly on the screen. No listing file is created. At the point at which execution ordinarily would begin, the interactive debugging system assumes control, and a `READY` message is displayed on the screen. The debugger will operate in "windowed mode", with model source code displayed.)

```
hpro myprog nos nodict noxref nowarn
```

(Source, dictionary, and cross-reference listings are suppressed. Model output is placed in a file named `myprog.lis`. If any compilation warning messages are issued, they are placed ***only*** in the file `myprog.lis` and are not echoed on the screen.)

```
hpro myprog test type source dict xref
```

(Source, dictionary, and cross-reference listings and all model output appear directly on the screen. The `source`, `dict`, and `xref` options are needed because the `type` option, which has already been processed, causes suppression of these listings. At the point at which execution ordinarily would begin, the interactive debugging system assumes control, and a `READY` message is displayed on the screen.)

6. INTERRUPTING MODEL EXECUTION AND OUTPUT

When GPSS/H is running, model execution can be interrupted at any point by typing CTRL-BREAK (holding down the CTRL key while simultaneously pressing the BREAK key). GPSS/H will acknowledge the CTRL-BREAK interrupt by displaying an "`ATTENTION!`" message on your screen, followed by a ": " prompt. The "`ATTENTION!`" message will be preceded by an "`Unplanned entry to DEBUG`" message if neither the `test` nor the `tv` option was specified when GPSS/H was started. In either case, you can proceed with model execution under the GPSS/H interactive debugger, as described in Chapter 11 of the GPSS/H Reference Manual.

CTRL-BREAK can also be used to terminate most forms of GPSS/H output. For example, if the interactive debugging command "`display cec`" (Current Events Chain) produces output on more Transactions than needed, the output can be terminated by typing CTRL-BREAK. GPSS/H will respond with an "`OUTPUT SUPPRESSED`" message, followed by an "`ATTENTION!`" message. At that point, you can get the information on the specific Transactions of interest using other debugging commands.

The SCROLL LOCK key can sometimes be used as an alternative to CTRL-BREAK within the debugger. By hitting the SCROLL LOCK key, you can freeze the scrolling of output on the screen. Once you have looked at the section of output on the screen, you can resume scrolling by hitting the SCROLL LOCK key again.

7. USER INTERACTION AND FILE ACCESS IN GPSS/H MODELS

7.1 User Interaction

GPSS/H has built-in facilities that allow a running model to write text to the screen, or read from the keyboard:

1. The GPSS/H language provides for routing model output to the screen by using an extended I/O statement with `FILE=SCREEN`, or with no `FILE=` operand (`SCREEN` is the default). Before GPSS/H Release 3, the special name `SERCOM` was used in place of `SCREEN`.

2. The GPSS/H language provides for a model to accept input from the keyboard by using an extended I/O statement with `FILE=KEYBOARD`, or with no `FILE=` operand (`KEYBOARD` is the default). Before GPSS/H Release 3, the special name `GUSER` was used in place of `KEYBOARD`.

7.2 File Access

If you want a GPSS/H model to read or write data files while running, you may need to take special steps so that GPSS/H can access or create the appropriate file(s):

1. The GPSS/H language provides for routing model output to the listing ("`.lis`") file by using an extended I/O statement with `FILE=LISTING`. Note that if a model using such a statement is run with the `type` option of the command used to invoke GPSS/H, the output that would normally go into the listing file will be written to the screen instead. Before GPSS/H release 3, the special name `SYSPRINT` was used in place of `LISTING`.

2. A model can also perform input from, or output to, a user-defined disk file. This is accomplished by using a GPSS/H extended I/O statement in which the `FILE=` operand uses a GPSS/H filename other than `LISTING`, `SCREEN`, or `KEYBOARD` (`SYSPRINT`, `SERCOM` or `GUSER` before GPSS/H Release 3). No further action is needed if GPSS/H can use the `FILE=` operand ***directly*** as the name of a disk file in the user's ***current*** directory. This will work if the name of the file has no extension (does not contain a period), ***and*** either exists, or will be created, in the directory in which the user is positioned when GPSS/H is invoked. For example, if an extended I/O statement specified "`FILE=MYFILE`", GPSS/H would look for or create a file named `MYFILE` in the directory from which GPSS/H is started.

   If the file has a name that includes an extension, ***or*** requires specification of a pathname, then a GPSS/H `FILEDEF` Control Statement (or an equivalent `BFILEDEF` Block) must be used. These allow you to equate a GPSS/H internal file name, which cannot contain special characters such as periods, to the ***actual text string*** that should be passed to the operating system when GPSS/H attempts to open or create the file. Please see Table 5.2 (BFILEDEF Block) and Section 6.15 (FILEDEF Control Statement) of the GPSS/H Reference Manual for more information.

3. Although the GPSS/H language has no internal limits on how many files can be in use at the same time in a model, such limits ***may*** be imposed by the operating system

4. GPSS/H keeps to a minimum the number of “system” files open when models are running. The only GPSS/H system files that can be open are:

The error message file	(always open)
The listing ("`.lis`") file	(open unless the `type` option was used)
The debugger checkpoint file	(opened on the first `checkpoint` command; deleted when GPSS/H exits)
The `CHECKPOINT` statement checkpoint file	(opened when the first `CHECKPOINT` statement executes; deleted when GPSS/H exits)

Table 1: GPSS/H Command Line Options

Option	Default	Interpretation
DICT NODICT	DICT (NODICT if TYPE used)	The DICT option causes printing of a dictionary of all GPSS/H entities used in a model. The dictionary is grouped by entity class and sorted by entity number within each class. It is useful for distinguishing which entity names and/or numbers have been used in a model and which are still available for use. The NODICT option can be used to suppress printing of the entity dictionary. DICT is the default, unless the TYPE option is used, in which case NODICT becomes the default.
ERR NOERR	ERR	The ERR option causes compilation error messages to be typed on your screen when the model compilation listing is directed to a destination other than the screen. The NOERR option can be used to suppress the typing of compilation error messages on the screen under these circumstances.
MAKEOBH	(off)	The MAKEOBH option causes GPSS/H Professional to compile a model and write out an "object" file (primarily for use with Runtime GPSS/H). The filename of the "object" file is the same as the source file, except that the file type is "`.obh`". When this option is specified, the model is **not** executed. If you want to run the model with GPSS/H Professional, you must use a command of the form: `hpro fname.obh` [*options*...] where the name of the file to be run explicitly contains the "`.obh`" extension. NOTE: If you want to run an "`.obh`" file using TV-mode debugging with source code displayed, you must specify the TV (or TVTNW) option along with the MAKEOBH option when the model is compiled. Otherwise, the Block source code will not be included in the "`.obh`" file.
MEMSTATS	(off)	The MEMSTATS option causes information on memory available to GPSS/H, memory used by GPSS/H, and model size to be included in standard output.
NOSIM	(off)	The NOSIM option produces the same effect as commenting-out a model's SIMULATE statement. This causes GPSS/H to compile a model and prepare it for execution, then stop without actually executing it. The NOSIM option is a convenient way to produce a GPSS/H compiler listing of a model, or to ensure that your computer has enough memory to satisfy a model's requirements without having to halt the running model and terminate the run.

QUIET NOQUIET	NOQUIET (But QUIET for Runtime GPSS/H)	This option is provided so that a "canned" model can be run, under the control of a batch file or user-provided "front-end" program, without the person running the model being aware that GPSS/H is running. The QUIET option causes GPSS/H not to write any of its *own* output to the screen during execution, unless a severe error occurs. All output is written to the listing (".`lis`") file instead. Any user-specified output to FILE=SCREEN *will* be written to the screen, but using DOS-style standard output instead of direct writes to video memory. This means that user-specified output can be redirected to a file, if desired, using standard DOS redirection techniques.

SIZE=A
SIZE=B
SIZE=C
NOTE: Although supported, the SIZE= option is obsolete.
You should use a REALLOCATE statement if you need to override the SIZE=A default.
	SIZE=A	The SIZE option establishes a default value for the number of members of each GPSS/H entity class that will be available when a model is run. The compiler may alter these values, based on what it "sees" as it compiles the model. For example, if it sees that no entities of a given class are referenced, it will not provide any entities of that class to the model, thus saving memory. The compiler can also provide more than the default number of entities for a given class. For example, if the compiler encounters a reference to Facility number 350, it will guarantee that at least 350 Facilities are provided for model execution. For a complete explanation, see Section 3.5 of Chapter 3 in the GPSS/H Reference Manual.
		The number of members of each entity class implied by SIZE=A, SIZE=B, and SIZE=C is summarized in Table 6.1 of the GPSS/H Reference Manual.
		In many cases, the compiler cannot tell how many members of a given entity class are required. For example, if a model contains a "SEIZE PH(NUM)" Block, the number of Facilities required cannot be determined at compile time, because the values of PH(NUM) are not known. For entity classes so referenced, the compiler has no alternative but to use the number of members implied by the SIZE option. If the number of members implied by the SIZE option is inadequate, you must supply one or more REALLOCATE control statements, to override the default number. ***Use of the REALLOCATE statement is strongly recommended in place of the obsolete "SIZE=" option.***
		The REALLOCATE statement is described in Section 6.39 of Chapter 6 of the GPSS/H Reference Manual.

SOURCE NOS	SOURCE (NOSOURCE if TYPE is used)	If the SOURCE option is used, a source listing of the model is included in model output. The SOURCE option is assumed unless the TYPE option is used to provide direct screen output. NOS can be used to override a (default) SOURCE option.

TEST	(off)	The TEST option is used to invoke the GPSS/H interactive debugging system. When the TEST option is used, compilation and loading of a model proceed in the usual fashion; however, at the point at which model execution would normally begin, control is passed to the GPSS/H interactive debugging processor, which displays a `READY` message on your screen. At that point, you assume control of model execution. You can then issue commands to selectively display model output, to stop model execution at arbitrarily specified Blocks (breakpoints), and, if necessary, to step through a model, one or more Blocks at a time. See Chapter 11 of the GPSS/H Reference Manual for details of how to use the interactive debugger.
TTNW	(off)	TTNW is a convenient abbreviation for TEST TYPE NOWARN.
TV	(off)	TV is an acronym for Test Video. This is the three-window debugging mode with source-code display. See Chapter 11 of the GPSS/H Reference Manual for information on specific features of TV mode. **See Table 2 for TV mode function key usage.**
TVTNW	(off)	TVTNW is a convenient abbreviation for TV TYPE NOWARN. It is especially useful for quickly making repeated short debugging runs of a model: no time is consumed creating a listing file, compile-time warning messages are suppressed, source-code is available on screen, and GPSS/H Error and Warning messages will be forced to the screen (where they're harder to miss).
TYPE	(off)	The TYPE option causes all standard model output to be displayed directly on your screen. No listing ("`.lis`") file is created. When the TYPE option is used, the NOS, NODICT, and NOXREF options are assumed. These assumptions can be overridden by placing the SOURCE, DICT, and XREF options, respectively, to the right of the TYPE option.
WARN NOWARN	WARN	The WARN option causes compilation warning messages to be displayed on your screen when the model compilation listing is directed to a destination other than the screen. The NOWARN option can be used to suppress the display of compilation warning messages on the screen under these circumstances.
XREF NOXREF	XREF (NOXREF if TYPE used)	The XREF option causes a cross-reference listing of all entity symbols and constants to be included in model output. The NOXREF option can be used to suppress cross-reference output. XREF is the default, except when the TYPE option is used.
XSORT=A	(off)	By default, a cross-reference listing is grouped into entity classes and sorted by entity name/number within each class. Use of the XSORT=A option suppresses the grouping of a cross-reference listing into entity classes; i.e., the entire collection of entity names/numbers is sorted and printed without regard to entity class. ***This is very useful for finding unintentional name duplications.***

XWF	(off)	The XWF option causes conditions which would normally result merely in the issuance of an execution warning message to be treated as fatal errors. For example, an attempt to store a value greater than 32767 into a halfword location ordinarily results in issuance of a warning message, and execution is continued by storing a truncated value. In a very large model, this action could result in a lengthy run producing meaningless results. Use of the XWF option prevents such problems.

Table 2: GPSS/H TV Mode Predefined Function Keys

Home Key	Scroll output window to the top.
End Key	Scroll output window to the bottom.
Page ↑	Scroll output window up ½page.
Page ↓	Scroll output window down ½page.
↑	Scroll output window up one line.
↓	Scroll output window down one line.
→	Scroll output window right 20 columns.
←	Scroll output window left 20 columns.
F3	Enter full-screen output mode.
F4	Enter windowed (debugger) output mode
F5	Scroll source window up.
F6	Scroll source window down.
F7	Scroll source window left.
F8	Scroll source window right.
F10	“Step 1” command