



---

## 19. Running SATURN Programs Interactively

### Mini-Contents Page

<b>19.</b>	<b>Running SATURN Programs Interactively</b>	<b>19-0</b>
19.1	Introduction	19-1
19.2	Graphics and Text Mode	19-1
19.3	Menus	19-1
19.4	Banner Menus	19-3
19.5	Menu Bars (P1X only)	19-4
19.6	Text Menus	19-5
19.7	Windows List Boxes	19-6
19.8	Setting Variables	19-6
19.9	Windows Screen Edit and Edit Box Inputs	19-7
19.10	Output Text Windows	19-7
19.11	The Help Facility	19-7
19.12	Mouse-based Inputs	19-8
19.13	Version Control	19-9



## INTRODUCTION

### 19.1 Introduction

This section considers how to run interactive programs, primarily **P1X** and **MX**, using the mouse and/or keyboard once they have been started using, e.g., the command line approaches outlined in Section 14. The same programs may also be run in a more batch-like style using **KEY** files as explained in 14.5.

### 19.2 Graphics and Text Mode

At the highest level interactive **SATURN** programs operate under two modes - graphical and text. Under text mode alpha-numeric text display uses the full screen (or full window) and user input is from the keyboard. Under graphics the screen contains a combination of text, graphics, windows, etc and user input is either from the keyboard and/or the mouse. Clearly text is the traditional mode while the latest programs are more graphics-based; nonetheless text continues to exist and has certain advantages.

Programs are not confined to a single mode. **SATDB** and **SATLOOK** (traditional programs) are almost entirely text-based but have some graphics options. **P1X** is basically graphical but when it calls **SATLOOK** or **SATDB** as options it reverts to text.

Note that within **P1X** (only) a graphics window and a text window both exist concurrently as indicated by two minimised tokens at the bottom of the screen with titles "Text P1X ..." and "Graphics P1X ..." but at any one time only one has "focus". In principle the program knows which of the two should currently be in focus so that if, say, the program is in text mode then the text token should be in focus and the program will be expecting keyboard input. Problems may arise if the program gets it wrong and appears to "hang", an indication that the user may need to manually focus on the "other" mode's token.

In either mode user responses may be divided into the following very general categories:

- ◆ choices from a menu;
- ◆ direct- generally numerical - responses to a direct question; e.g., you are asked what node number you want and you key in the desired number;
- ◆ mouse "point and click" input, e.g., on a network plot;
- ◆ screen edit and/or edit box windows; and
- ◆ file opening and other standard Windows screens.

### 19.3 Menus

Menus within **SATURN** appear in four basic forms:

- ◆ "banners" which appear in conjunction with graphical output, by default as a 12-character wide strip on the right of the screen, with user input via either the mouse or the keyboard (19.4);

- ◆ “Menu bars”: menus with pull-down menus which appear along the top of (some) graphics screen and which supplement choices in the banner. (19.5)
- ◆ the “text-based” form whereby the menu occupies the entire screen (or window) and user input is from the keyboard (19.6);
- ◆ Windows list boxes (see 19.7).

In all cases menu choices may be broadly sub-divided into five categories:

- ◆ further sub-menus;
- ◆ executable routines;
- ◆ numerical variables;
- ◆ logical variables; and
- ◆ scrollable variables

Thus a sub-menu choice takes you into a further menu system while an executable routine initiates a particular “do this next” option (although the distinction may be somewhat arbitrary as a do-this next routine may well start with a menu).

Variables come in several forms although all have the basic property of associating a value with a particular parameter, e.g. origin zone = 67. Thus numerical variables can take on a wide range of values as with a parameter associated with zone numbers. By contrast a logical variable can only have two values, effectively true or false, on or off etc. Finally a scrollable variable is basically numerical but one that takes on a relatively small number of values, e.g. a parameter which defines one of three or four allowable options.

If a numerical variable is selected you will next either be prompted for the new value or, possibly, choose from a list of allowable values. A logical variable requires no further action, it simply changes or “toggles” to its logical opposite. A scrollable variable changes to the next value in its list, returning to its “top” value if the bottom of that list is reached.

### 19.3.1 Menu Structure

In general menus in **SATURN** are organized into the hierarchical or tree-like structure where the top or “master” menu can call a number of different sub-menus or executables routines. These in turn may call other sub-menus or routines. We refer to the return to the menu from which the current menu was called as going “back”. Equally we go “forward” to sub-menus.

In certain cases menus and/or routines may appear to be more “sequential”; e.g. if you are defining a new simulation node you may first enter a menu to define banned turns, then one to define lanes etc, etc in a continuing sequence. We refer to this as going to the “next” menu. However, invisible to the user, each next menu is actually generated by a return from the current menu to its upper level menu which then automatically calls the next menu in the sequence.



The distinction between the two has implications for the terminology used to describe “exits” from menus as discussed next.

In addition to the “normal” menu exits of forward, back and next referred to above it is also possible to “quit” a menu abnormally which effectively terminates the current operation and goes back one or more levels in the menu structure. In most situations “quit” has the same effect as “return” or “back”. However when the normal return operation is to the “next” menu quit aborts that step and definitely returns you to an upper level.

### 19.3.2 Terminology: Exit, Return, Continue, Next, Quit, Back, Cancel, etc.

The terminology used to describe the different exit modes may differ between different parts of the programs and some explanation of the terms used may help.

“Return” is used, mostly in text menus, to indicate either “back” or “next” - the context of the menu decides which. Somewhat unfortunately, given what comes next, in graphical banners it is referred to as “Q-return”, the primary reason being that the letter Q is not very often needed to denote other operations whereas R for “Return” could conflict with, say, R for “Repeat”. “Q-continue” is also used to signify continue to the “next” menu.

The abnormal or “quit” exit is also referred to by more than one term or letter - generally Q for Quit in text menus but also as QUIT when asked to set file names from the keyboard or “Cancel” when setting them a Windows dialogue box. Generally speaking abnormal quits are not present within graphical banners.

To exit fully from a program you must successively “return” back up to the master menu where the quit/return option now signifies exit from the program. Alternatively, you may exit at any stage by clicking on the exit cross (upper right) in the current “background” window or by choosing Exit under Files in the menu bar.

## 19.4 Banner Menus

In banners the choices are listed in a strip either to the right or above the main plot. Generally each item occupies a single “slot” of up to 12 characters although “multiple slots” may be used to improve clarity.

The “form” of each menu item - 1 to 5 in 19.3 - is indicated by a symbol on the right hand side of the banner with the following convention:

- 1) Sub-menus: >
- 2) Executable: x
- 3) Numerical Variable: ?
- 4) Logical: a “radio button”
- 5) Scrollable: s

Each item has a single key alphanumeric character associated with it which is displayed in a different colour (equivalent to underlined characters in a standard Windows menu).



## 19.4.1 Keyboard vrs Mouse Control

Selection from a banner menu is based on either “keyboard” or “mouse” control although mouse control is very much the norm these days. In the latter case – “point and click” - the selection under the mouse is indicated by reversing the colour of that item. Under keyboard control type the single highlighted key, either a letter or a number (but without <enter>, see 19.6 below). For letters either upper or lower case is acceptable.

Clicking anywhere but on a highlighted line is ignored. Similarly keying in a character other than those highlighted has no effect. The program only continues with a legitimate response.

The user may toggle between these two, via either the Systems/device menu, (11.3.2) or the banner menu (11.15).

## 19.4.2 Focus

A particular problem encountered within 32-bit Windows **SATURN** is that under keyboard control the input from the keyboard is directed to an “invisible” window which, in effect, only exists as a minimized token on the tool bar at the bottom of the screen with the title “Key Input”. When this token is “highlighted” or “in focus” the program is waiting for a keyboard input to the banner menu; when it is not highlighted it is expecting other forms of input.

Or so it should be. The problem that arises is that when the key input token has been set in focus by the program the user may accidentally transfer focus elsewhere by clicking with the mouse somewhere else on the screen. When this happens the program will “hang” - it wants a keystroke command but the system's attention is elsewhere.

The solution is simple - click on the “key-Input” token in order to highlight it, hit the key you want and off you go.

## 19.5 Menu Bars (P1X only)

These are standard Windows-style bars with pull-down windows which appear at the top of graphical windows and which contain standard options applicable to most (or all) banners. For example, requests to dump the current screen image into a bitmap file (11.3.5) may be accommodated at most times when a choice is to be made from a banner menu. PCX, BMP, JPL and clipboard output options all appear under Files. Options which are not currently available for one reason or another are “greyed out”.

Clicking on an available option - or one from a pull-down menu - from the menu bar is equivalent to making a choice from the options displayed in the banner; both sets are equally available (and indeed in some cases the same choice is available in both). In addition choices may be made from the menu bar even when an “active” banner does not appear but mouse input is required; for example when using the mouse to select a node or link it is possible to use the menu bar to, e.g., change the window.

Selection from the menu bar may either use the mouse or via a “hot key”; e.g. files may be selected by the key combination alt + F.



Note in particular that the menu bar contains a “Back” option which is equivalent to Q-return in the banner proper and both are generally present. Users may find “Back” easier to use for going back over several menus, e.g. to return to the “master menu”, since its position on the screen is fixed.

Alternatively under Back one may also return directly to the Master Menu or to any of the 13 sub-menus choices normally contained there.

Note that the menu bar is NOT active under keyboard control.

## 19.6 Text Menus

In text menus the choices appear as text on the screen with each item given a numerical code 0, 1, 2 ... (plus, occasionally, negative numbers). To make a choice key in that number followed by <enter>. Note, firstly, there is no mouse input and, secondly, that unlike banner keyboard inputs the keystroke must be followed by <enter> (which therefore allows you to correct mistakes).

The “form” of each menu item - 1 to 5 in 19.4 - is indicated by a character on the far right. Note that the text symbol for a logical variable is ‘L’; the others are as in banner menus.

An example of a text-menu with sub-menu choices (2 to 6) and executable choices (0, 1, 2 and 7) is given below.

### CHOICE OF OPTIONS FOR NODE 54

0	RETURN	X
1	PRINT A FULL NODE DESCRIPTION	X
2	CHANGE GLOBAL OR NODE-BASED PARAMETERS	>>
3	CHANGE LINK-BASED PARAMETERS	>>
4	CHANGE TURN-BASED PARAMETERS	>>
5	CHANGE TRAFFIC SIGNAL SETTINGS	>>
6	CHANGE THE TYPE OF NODE	>>
7	GO TO THE SIMULATION AND ANALYSIS PHASE	X

The input value 0 is almost always an acceptable response to a text menu although it may not always appear explicitly in the menu for reasons of brevity. It signifies “return” in the sense of “back” or “next” as appropriate and as explained in 19.3. In text menus the <return> or <enter> key on its own is interpreted as 0, i.e. return/continue.

In most text menus the single letter input value ‘Q’ is also a valid input which terminates the current operation and takes you back to a previous menu. Except for continue menus, as explained in 19.3, this has the same effect as RETURN. In some menus explicit numerical values, generally -1, also allow the user to quit and have explanations such as “Oops - get me out!” or “Do Nowt”. In certain cases Q has no effect, probably due to the fact that you have got so far into an



option that it is no longer feasible to retreat; in desperation control-break or control-alt-delete will probably blow the program!

## 19.7 Windows List Boxes

List boxes are standard Windows screens in which one item must be selected from a list by first selecting that item and then clicking on the OK button, at which point the window is closed. The items in the lists are much as in other **SATURN** menus as listed 1 to 5 in 19.3 and generally include the default selection of return/do nothing/etc.

Note that a selection **must** be made from the list before the program can continue; i.e, you cannot minimise a list box and return later nor close it without making a choice.

“Long” list boxes will have a vertical scroll bar included. Note that occasionally, and for no obvious reason, the last item in even very short lists is left off the bottom of the window and can only be seen by moving the scroll bar. To avoid this problem an option is provided under “Gen Params” in the System/Device menu to add an extra blank line to all list windows in order to avoid the problem.

## 19.8 Setting Variables

If a numerical variable is selected from a menu (where its current value is displayed) the user is prompted for a new value. Under text mode it is input from the keyboard following the prompt. Under graphics it is either (16-bit) input from the keyboard and displayed in a “slot” in the banner or (32-bit) input via a “proper” Windows window. All keyboard inputs are terminated by <enter>; alternatively click OK in a window.

All variables will have default values associated with them, generally their current value. A “null” response, i.e. hitting <enter> directly or the OK window button, returns that value. In 32-bit windows these appear as initial values; in 16-bit the current values are those in the previous menu.

Equally most variables have upper and lower values associated with them so that inputting a value outside these limits will either cause the default (input) value to be retained or, possibly, the nearest limit value to be used. You may even receive a mild rebuke! In any case you will see the “new” value in the menu.

The input value should be in the same “format mode” as the value displayed; i.e. an INTEGER value without a decimal place is required for variables displayed without a decimal. On the other hand the values input for “real” variables should include the decimal place (although it is not mandatory).

Logical variables do not require any further input; they simply “toggle” to the opposite sense. In some cases this may be indicated by a change in the text, in others “on” is replaced by “off”, “Yes” by “No”, etc while in still others T alternates with F.

Equally for a scrollable variable no further input is required; it simply moves to the next permitted value and the text changes as appropriate.



## 19.9 Windows Screen Edit and Edit Box Inputs

Certain data input operations within interactive programs use either a standard Windows-based screen editing or edit box environment.

Thus screen-edit windows display a segment of text which the user may edit using the keyboard and/or mouse. When you are finished “close” the window to return to **SATURN** which then “processes” the text.

Edit boxes allows the values of specific variables (numerical, text, toggles, etc.) to be altered by the user and returned to the calling routine.

Please note that the data in a screen-edit window is “formatted” and that the position of different variables within the window is important since the data must be “read” by the program when the window is closed. Thus if the window is made up of a set of zone numbers plus trip origins with, say, 3 sets of pairs per line the program is expecting to find the first three zones in line 1, the second three in line 2 etc. If, during the editing, you introduce extra lines the program may not be able to interpret the data correctly. You are therefore advised to retain the original line and column structure when editing (although, generally speaking, shifts of data entries along the same line may be correctly interpreted). A bit like being told to write between the lines when you were five years old!

We note that the use of edit boxes and screen editing has certain implications for LOG and KEY files as explained in Section 14.5.9.

## 19.10 Output Text Windows

Windows containing text are created at certain points within interactive programs for a variety of purposes. For example they may:

- 1) List short messages,
- 2) Give more substantial text outputs,
- 3) Provide a more complete explanation of entries in the banner menu.

Thus type 1 for example is used to print error and warning messages which will be deleted by the program automatically when you continue with the next step. Type 2 may be used to give a complete listing of the current properties of a node in node graphics; in this case the window is preserved until deleted by the user. Type 3 windows are automatically deleted when the banner is updated.

These windows differ from those in Section 19.9 above in that they are for display only, they cannot be edited in order to return data to the program.

These windows may differ in terms of whether they have scroll bars, whether they have colour, how they are deleted etc. In most cases it should be possible to “grab” text from the windows using standard cut and paste techniques.

## 19.11 The Help Facility

With all menus it is possible to obtain further information concerning the general form of the menu and/or the choices available by entering the “help mode”. To



enter the help mode simply type 'H' in response to a text menu or click on 'Help' in the banner or menu bar.

Essentially the help mode is a form of on-line editing system whereby the program reads from a pre-prepared ".HLP" file at the relevant spot for the particular menu from which it is called. The user is then able to "browse" through the help file by commands which move the screen up or down in the file, e.g. with the cursor keys. Exit to the original program is as indicated on the screen.

A special feature of all Help files is that a general program description is contained at the very top of the file - this can be accessed by hitting the <home> key so that the "screen" will be shifted to the top of the help file no matter where you are currently positioned.

Unfortunately not all menus have a corresponding pointer into a help file; thus sometimes a request to enter help will only get you a slightly rude message - better luck next time!

A further use of the Help files is to obtain help on the output from an interactive program AFTER the output has been produced - as opposed to obtaining it beforehand whilst viewing the menu. The help in this case describes the interpretation of the previous output.

Since .hlp files are ascii or "text" files it is quite feasible for users to edit them so as to add their own site-specific information. However it must be noted that, from version 9.2 onwards, .hlp files are stored as "direct access" files, which basically means that all lines must be the same length. Standard edit programs may well truncate short added lines such that the edited version is no longer valid under direct access. If this happens, use the MAKEDA77 facility supplied as part of DBOS to recreate the direct access format.

## 19.12 Mouse-based Inputs

Mouse based inputs perform an important range of functions in **P1X**, e.g. to make selections from the banner or to "point and click" to a node. Input is relatively simple; only the left hand button is used and functions are designed to be self-explanatory.

"Point and click" operations, e.g. to select a node from a network plot, are generally accompanied by a set of "buttons" within the banner which allow one to exit from that particular operation or to select further operations, e.g. redefine the window. To select a button, point to it (it will reverse colour) and left click.

In certain situations you need only click the mouse, its position is irrelevant; for example this may be used in a Joy Ride to advance from one node to the next. In keyboard entry terms this is equivalent to "strike any key".



## 19.13 Version Control

JOB NUMBER: 5101396		DOCUMENT REF: Section 19.doc				
Revision	Purpose / Description					
		Originated	Checked	Reviewed	Authorised	Date
1	Re-formatted (Final to DVV)	TF / BG	NS	IW	IW	06/05/06
3	Upgrade to V2 Template	IW				22/06/06
3.2	Web release – Sept 06	DVV	NP	IW	IW	08/09/06
3.3	Web release – Jan 07	DVV	NP	IW	IW	04/01/07
3.4	SATURN v10.7 Release	DVV	NP	IW	IW	12/03/07
3.5	Web release – Jul 07	DVV	NP	IW	IW	19/07/07
3.6	SATURN v10.8 Release	DVV	NP	IW	IW	26/01/08
3.7	Web release – Jul 08	DVV	NP	IW	IW	07/07/08
3.8	Web release – Dec 08	DVV	NP	IW	IW	12/12/08
3.8.21	Web release – Feb 09	DVV	NP	IW	IW	16/02/09
3.8.22	Web release – Jun 09	DVV	NP	IW	IW	16/06/09
10.9.10	SATURN v10.9 Release	DVV	DG	IW	IW	04/09/09
10.9.12	SATURN v10.9 Release (Full)	DVV	DG	IW	IW	22/10/09
10.9.22	Web release – Dec 10	DVV	AG	IW	IW	06/12/10
10.9.24	SATURN v10.9 Release (Full)	DVV	AG	IW	IW	06/05/11
11.1.09	SATURN v11.1 Release (Full)	DVV	AG	IW	IW	31/03/12