PatternTool 17

Introduction

PatternTool displays the textual representation of a test pattern as if you were viewing it with a text editor, allowing you to display, manipulate, and create the `Pattern` objects. Also, refer to Features.

This tool is used with Waveform Tool so the alias characters you define in the patterns by using PatternTool are also displayed in the waveforms shown in WaveTool.

Besides using PatternTool to create, modify, and debug the `Pattern`, `SignalHeader`, `PatternGroup`, and `PatternMode` objects used in the enVision tools and by other objects, PatternTool includes two unique editors:

- Signal Header Editor—creates or edits the `Signal Headers`, add signals to the Signal Header, or create signal buses; refer to
- Pattern Map Editor—creates or edits `Pattern Maps` that are the lists of patterns used by a test program.

For more information about `Pattern` objects, refer to the *Pattern, Waveform, and Timing* chapter of the *enVision Digital Programming* manual.

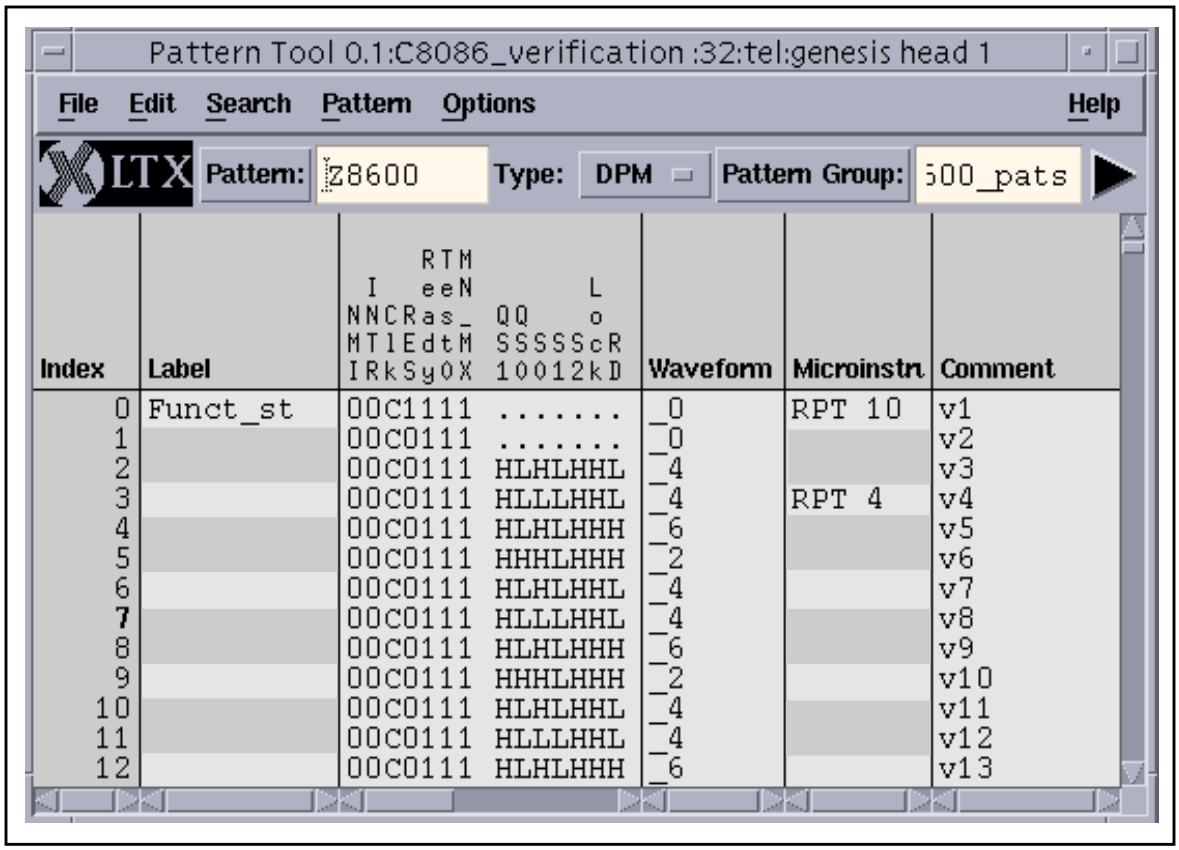


Figure 17.1: PatternTool Display

Features

- Develop patterns online; refer to PatternTool Tasks.
- Modify patterns online when debugging a test pattern; refer to Signal Header Editor. Regions can be selected and cut, copied, and pasted across other regions of the patterns. Signals may be edited to occur earlier or later in time relative to the other signals on the device.
- Create and modify `SignalHeaders`; refer to Signal Header Editor.
- Create Pattern Groups; refer to Create Pattern Group Dialog.
- Create and edit Pattern Maps; refer to Creating a Pattern Map.
- Display the vector failing information, highlighting the failing pins at the failing vectors when used with Datalog. Display fail information directly to the pattern, by means of different colors.

PatternTool is unlike some of the other enVision tools:

- Unlike other enVision tools, as you edit a pattern with PatternTool, the working memory and tester memory are not updated immediately as you move from field to field in the tool. PatternTool works this way so you can perform PatternTool operations that would otherwise render your pattern temporarily un-parseable, such as pasting differently formatted vector data. As a result, PatternTool internally buffers data until you update the workspace. By updating the workspace, this current information is loaded in the appropriate hardware. Also, refer to Performance Notes.

- When the displayed pattern differs from the pattern seen by the tester and the rest of enVision, the LTX icon in the top left corner of the tool is displayed in reverse video.

- What you type directly in to the pattern fields, rather than through a separate text-entry field. Because of this, it has somewhat different rules for selection, cursor-movement, and data entry in general; refer to PatternTool Main Window, Menus, and Dialogs.

Sample Pattern Files

For examples of creating pattern objects, refer to *Getting Started* chapter in the *Getting Started with enVision++* manual. This manual includes a tutorial on how to build a simple `Adapterboard` and `Pattern` object.

A similar example, but more on the design aspects of the object, is presented in the *Preparing the Pattern Files* section of the *Digital Programming Techniques* chapter in the *enVision Digital Programming* manual. In this example includes the resulting .evo file.

PatternTool Main Window, Menus, and Dialogs

The main window of PatternTool contains menus and fields for viewing and specifying the name and type of the `Pattern` object, `PatternGroup` membership, and other characteristics of a pattern; refer to Table 17.1 and Figure 17.1.

Certain PatternTool menu items use dialogs.

Table 17.1: PatternTool Panes and Text Fields, Main Window

Name	Meaning
Label	Display labels, which are valid enVision names used as symbolic address references by any objects, including this or other `Pattern` objects. Single or multiple labels (separated by whitespace) are allowed only in the first vector of a VectorGroup. This restriction is required to support partial `MuxMode` patterns; the other label fields are grayed out. Labels can be edited, deleted, or inserted.
Vector Data	Displays pin names on header section and associated data (in aligned columns with the pin names), for each displayed vectors. Data is presented as alias characters that represent the binary data and the waveshape (defined by `WaveformTable`). Alias characters must be defined in the default `WaveformTable` for the `PatternGroup` before they can appear in the pattern; refer to Create a Default WaveformTable for Pattern Object. Characters are shown in the color defined in the default `WaveformTable`, or in bold red if they do not agree with pre-defined alias characters or appear outside of the allowed columns defined in the header. Also, to support partial `MuxMode` patterns, PatternTool classifies any non-`MuxMode` pins of odd vectors as read only. Read-only pins have the same background color of the vector. The enVision `.Xdefault` for these read-only areas is `readOnlyBackground`; also refer to Setting PatternTool .Xdefaults.
Waveform	Displays the `WaveformTable` Reference for each vector of the displayed vectors. This reference selects the `WaveformTable` via the Zipper Table; refer to the *Pattern Sequence Tool* chapter in this manual.
Microinstruction	Displays the `Pattern` micro-instructions only for first vector of the VectorGroup. This restriction is so enVision can support partial `MuxMode` patterns; the other label fields are grayed out. Using micro-instructions compacts the pattern data in a more efficient, algorithmic manner. The field is bold red if its current contents can not be parsed successfully. Micro-instructions are listed in the *Pattern, Waveform, and Timing* chapter of the *enVision Digital Programming* manual.

Table 17.1: PatternTool Panes and Text Fields, Main Window (Continued)

Name	Meaning
Comment	Displays the user comments for each vector of the displayed vectors. Comments may also be entered across the full width of the pattern display by simultaneously pressing the Control and Return keys or by selecting Add Comment from the right menu button menu.

PatternTool Menus

Pattern Tool main menu in the toolbar consists of these items:

- File—Table 17.2
- Edit—Table 17.3
- Search—Table 17.4
- Pattern—Table 17.5
- Options—Table 17.6
- Help—Table 17.7

PatternTool has context-sensitive pop-up menus that are opened by using the mouse button M3; refer to Context-Sensitive Menus in PatternTool.

Table 17.2: PatternTool—File Menu

Selection	Action
Find	Same as pressing the Pattern: button on the main window. Opens a dialog showing all available `Pattern` objects; refer to Create New Pattern Dialog.
Create	Defines new pattern name. After entering this selection, dialog appears; refer to Create New Pattern Dialog. Most users attach the patterns generated by their CAD systems.
Clone	Copies current pattern. Unlike other enVision objects, patterns have associated files separate from the test program .eva file. When you copy a pattern, new pattern files text (.evo) and cache (.flex) are generated with specified name in directory from which .eva file was loaded.
Attach Ascii (.evo) Pattern Files	Compiles and loads one or more pattern files, and generates a cache (`.flex`) file for each in the directory from which the test program was loaded (not directory where pattern file is located, if different).

Table 17.2: PatternTool—File Menu (Continued)

Selection	Action
Detach	Disconnects the pattern from your test program, deleting the cache (.flex) file, but leaving the .evo file. The pattern must first be loaded into the PatternTool before it can be detached.
Update Workspace	Updates objects in memory, with current text changes in the tool, verifying syntax, and re-compiling new and changed vectors. Does not update the saved .flex or .evo files.
Print	Prints the window contents; refer to the *User Interface Tools* chapter in this manual for options and more information.
Close	Closes the window. Closing this window does not affect the Pattern object, only saving the test program does. If you exit enVision before updating the working memory, any changes will be lost.

Table 17.3: PatternTool—Edit Menu

Selection	Action
Undo	Restores the last edit. Repeating this command removes further operations, subject to memory limitations.
Redo	Restores last edit reversed by Undo.
Cut	Copies selected text to clipboard, and then deletes it.
Copy	Copies selected text to clipboard.
Paste	Either replaces selected text with contents of clipboard or inserts contents of clipboard at text cursor.
Delete	Deletes selected characters or vectors. Cannot delete vectors of MuxMode or MultiState patterns.
Select All	Selects all vectors in pattern.
Insert Vector	Same as pressing Return key: inserts a vector after current vector containing the cursor. Cannot insert vector for MuxMode or MultiState patterns.
Insert Vector Before	Inserts a vector before the vector containing the cursor. Cannot insert vector for MuxMode or MultiState patterns.
Insert Comment	Inserts a full-width comment after the vector containing the cursor.
Insert Comment Before	Inserts a full-width comment before the vector containing the cursor.

Table 17.3: PatternTool—Edit Menu (Continued)

Selection	Action
Change Header	Opens an object dialog for choosing new header for either selected vectors, or if no vectors are selected, for vector containing the cursor.
Change Waveform Ref	Opens an enVision text dialog for specifying a new Waveform Reference for either the selected vectors, or if no vectors are selected, for the vector containing the cursor.
Object Comments	Enter or view comments in the comment field of `Pattern` object.

Table 17.4: PatternTool—Search Menu

Selection	Action
Find	Searches selected field for a given string. When selected, the dialog, shown in Figure 17.4, provides a set of search rules.
Replace	Searches selected field for a given string and replaces it with another given string. When selected, the dialog, shown in Figure 17.4, provides a set of search and replace rules.
First Header Pin	Selecting this item causes a text box to open; see Figure 17.6. In this box enter a pin name or part of the name and click Find to search for it. If found, the cursor moves to the searched pattern column.
Next Error	Moves down to next pattern syntax error or waveform datalog failure in pattern. If viewing failures in waveform datalog, you cannot use this control to move forward through the log. To review pattern failures in their order of occurrence, use buttons in Waveform Datalog dialog.
Previous Error	Moves up to previous pattern syntax error or waveform datalog failure in pattern. If viewing failures in waveform datalog, you cannot use this control to move forward through the log. To review pattern failures in their order of occurrence, use buttons in Waveform Datalog dialog.
Go To Vector	Selects and scrolls to a particular index in the pattern.

Table 17.5: PatternTool—Pattern Menu

Selection	Action
Edit Pattern Map	Opens the Pattern Map Editor.
Edit Signal Header	Opens `SignalHeader` Editor to edit or view selected `SignalHeader`; Signal Header Editor. If no header is selected, the header of vector containing the cursor is shown.
Set Default Header	Changes default `SignalHeader` of pattern. Default header has no affect on pattern in memory or in PatternTool, except as a header for the default vector. In reading and writing pattern (.evo) files, default header allows header name to be omitted on vectors using a default header.
Set Default Vector	Sets default vector for pattern. Default vector allows pattern to include headers and corresponding vector data that do not supply data for all pins used in pattern. Enter the vector exactly as it appears in the pattern default header. Because the dialog does not display header for reference, enter the default vector in the pattern, and then paste it to the dialog.
Set Default Waveform Ref	Sets the default Waveform Reference. Select it when you do not enter a string in the Waveform column of PatternTool. Allows waveform reference to be omitted on vectors in the .evo file by using default string.
Default Waveform Table	Opens the WaveformTable Editor to display the default `WaveformTable` for the `PatternGroup` of the pattern. Default `WaveformTable` defines alias characters in a pattern.
Waveform Data Log	Opens a dialog for stepping through the entries in waveform datalog. Pressing **+** and **-** buttons or entering an index number in the dialog steps through the log entries in the order of collection. If different pins of a given vector have multiple failures, current log index controls which failures are shown.
Memory Descriptor	Attach only one `Memory` object to this `Pattern` object. A finder provides the list of the existing Descriptors and allows you to select one; refer to the *Memory Tool* chapter in this manual for details.
Hide Signals	Hides selected signals. `SignalHeader` signals are selected by dragging mouse across them in the `SignalHeader` display; refer to Selecting Pattern Characters or Vectors.

Table 17.5: PatternTool—Pattern Menu (Continued)

Selection	Action
Show Signals	Shows selected signal. Hidden signals are represented as small black squares in the header area. Select them, like visible signals, by dragging mouse across them in header display.
Show All Signals	Shows all hidden signals in selected `SignalHeader`, or in the `SignalHeader` for vector containing the cursor if no `SignalHeader` is selected.
Bus Radix	Sets display radix for selected bus. Option As Name shows individual pin names in header, and displays alias characters, rather than values. Header signals are selected by clicking or dragging mouse across them; refer to Selecting Pattern Characters or Vectors.

Table 17.6: PatternTool—Options Menu

Selection	Action
Overstrike General	Sets insert or overstrike mode for all text other than vector data. Insert-mode is default; use it for entering text.
Overstrike Vector	Sets insert or overstrike mode for vector data. Overstrike mode is default for entering vector data, because it maintains alignment of data with the header. Use insert mode only to shift signals or to correct alignment problems.
Constrain to Header	Allows typing only in columns defined in vector's `SignalHeader`. Default state. With this selection, if text already appears, you can position the cursor in areas not defined in the header (separator fields or beyond the end) and edit and delete these characters. Once these characters have been deleted, character positions are again constrained in the header.
In-line Headers	If vector uses a `SignalHeader` different from previous vector, new header is displayed between two vectors in scrolling portion of display.
Vertical Cursor Motion	Same as clicking on direction arrow at top-right of tool. It controls both field-tabbing direction and character cursor motion within vector data field. In horizontal mode (default), Tab key advances cursor horizontally from field to field, and typing in vector data field is left to right. In vertical mode, Tab moves cursor vertically within a single field, and typing moves vertically along a single pin.
Split Window	Adds viewing pane to window, so two separate parts of pattern may be viewed simultaneously.

Table 17.6: PatternTool—Options Menu (Continued)

Selection	Action
Close Pane	Closes viewing pane containing cursor.
Xdefaults	Changes fonts and colors; refer to Setting PatternTool .Xdefaults. Note `cellColor1` through `cellColor10`, which define color of alias characters, are based on the color set in tool. By default, these colors should match. If you change fonts, the `patternFont` and `headerSignal` fonts have matching bold versions. They must match in both height and width, or bolding will not function properly.
Save Geometry	Saves screen position of a tool, and its current size in a file in the user's home directory: `$HOME/app-defaults/PatternTool`. This file is read each time the tool starts.

Table 17.7: PatternTool—Help Menu

Selection	Action
On PatternTool	Opens a PDF file of this chapter, which is in the manual listed in the top right-hand corner of the page; refer to Index option under Help.
Index	Opens an PDF file that lists all customer documentation for this tester. Open the manual by clicking on it.
On Version	Opens the Tool Version pop-up that lists information about the enVision software release.

PatternTool Dialogs

PatternTool includes the following dialogs:

- Create Pattern Group Dialog
- Create New Pattern Dialog
- Pattern Object Finder Dialog
- replace Dialog
- Search Header Dialog

Create Pattern Group Dialog

Use this dialog to create a `PatternGroup` or select an existing one; see Figure 17.2. The dialog items are listed in Table 17.8. Also, refer to the procedure Create or Use an Existing PatternGroup Object.

Each PatternGroup requires a default WaveformTable, which defines the legal alias characters for the patterns in the group, and an initial SignalHeader to define the set of pins used by all patterns; refer to the procedure Create a Default WaveformTable for Pattern Object.

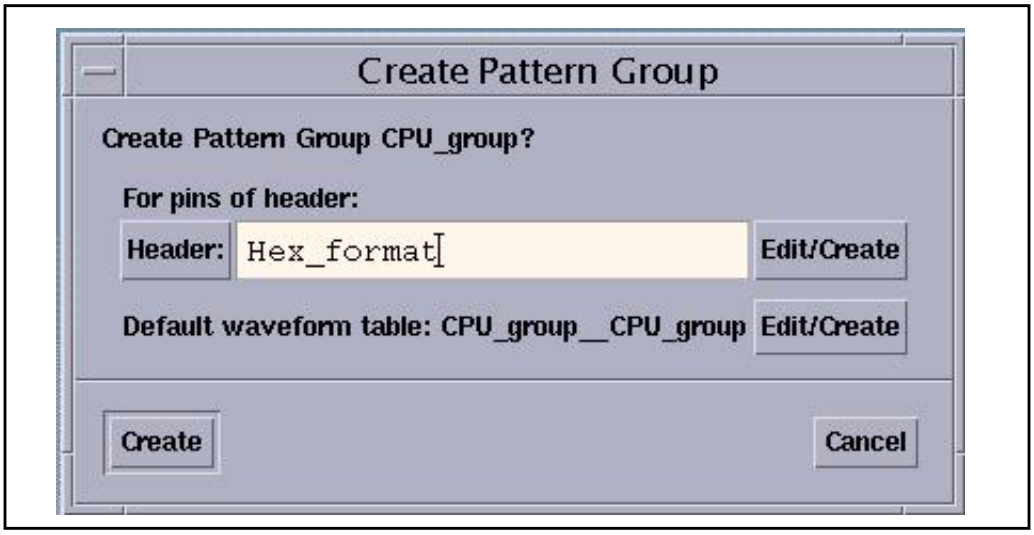


Figure 17.2: Create Pattern Group Dialog

Table 17.8: PatternTool—Create Pattern Group Dialog

Menu	Action
Header	Enter the header name in the field or click this button to select from a pop-up, like the one shown in Figure 17.4, a SignalHeader for the PatternGroup being created.
Edit/Create (header)	Edit or create a SignalHeader, refer to Signal Header Editor.
Default waveform table: Edit / Create	Edit or create a default PatternGroup WaveformTable, refer to Create a Default WaveformTable for Pattern Object.
Cancel	Dismisses this dialog.

replace Dialog

Use this dialog for searching and replacing text; see Figure 17.3. The dialog items are listed in Table 17.9.

Table 17.9: PatternTool—replace Dialog

Menu	Action
Search in field(s)	Selects which pattern field to search in: a. All fields—search across all fields in the `Pattern` object. b. Vector, Waveform Reference, Micro-instruction, or Comment—searches only in one of these fields.
Search For	Enter the string to search for, which may contain wild cards or be case sensitive.
Replace With	Enter string to replace the searched one for, if desired.
Case Sensitive	Toggle button specifies a case-sensitive search. Replacement string is always case sensitive.
Regular Expression	Toggle button to search for either given string or regular expression string with wildcards.
Search Vertically	Searches by columns. Used for finding specific bit patterns on multiple vectors for a single pin.

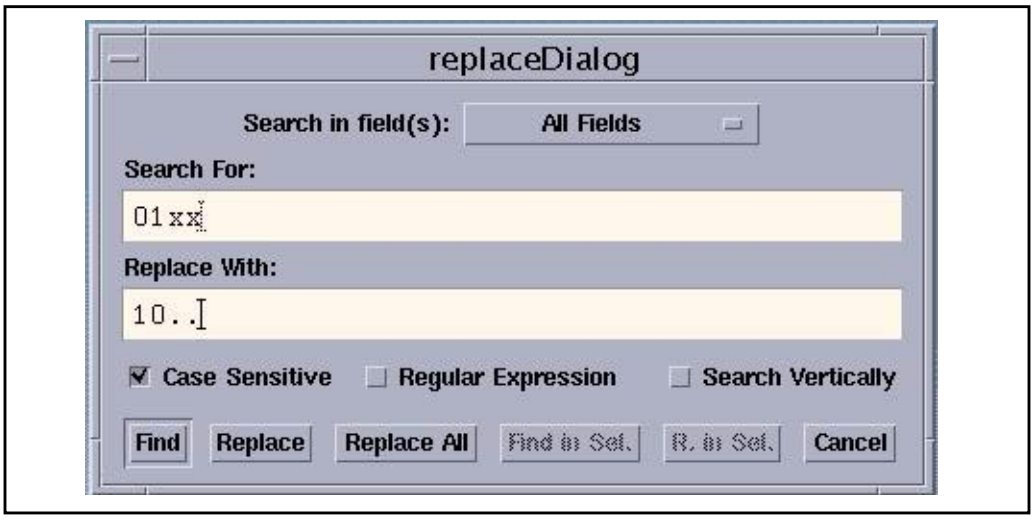


Figure 17.3: replace Dialog

Pattern Object Finder Dialog

Selecting Find opens a dialog for choosing a `Pattern` object from a list of `Pattern` objects already created. It is the same as pressing the Pattern button next to the LTX logo; see Figure 17.4. To load an object in PatternTool for viewing or editing, double click on it or type in a new name for this object in the Selection field and click OK.

NOTE Typing in the Pattern object name in the Selected field is not the proper way to select an existing Pattern object. You must press the Pattern button next to the LTX logo to select an existing one. When you type the name in the entry area, you are changing the existing object name. Using either naming method is valid only for an un-named object. By naming this unnamed Pattern object, you have appeared to have created a Pattern object.

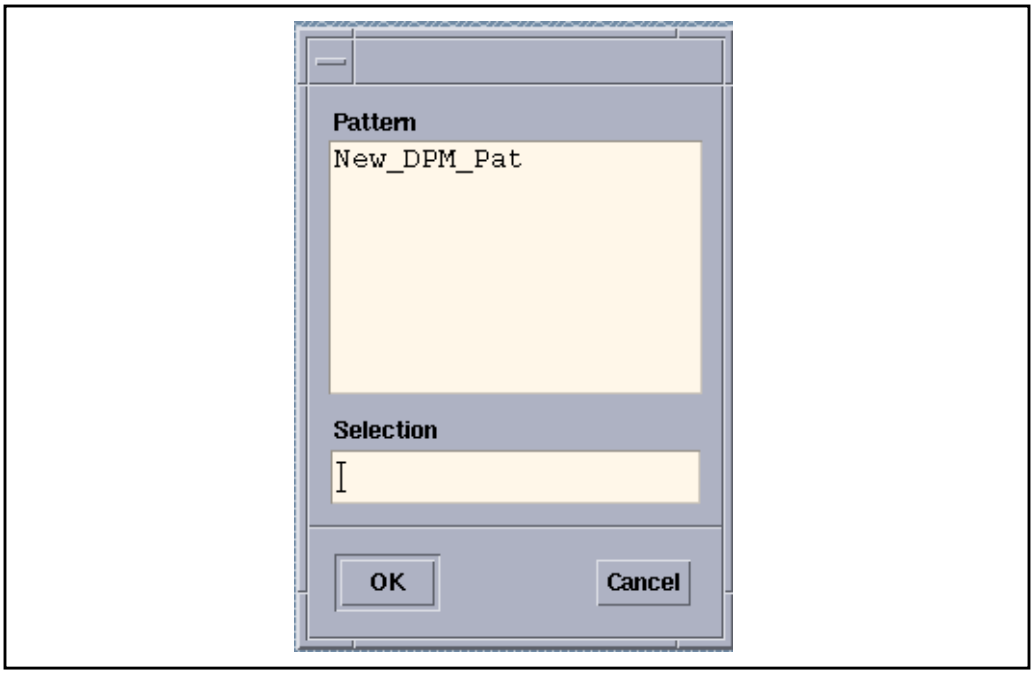


Figure 17.4: Pattern Object Finder

Create New Pattern Dialog

Most users create a pattern by attaching the patterns generated by their CAD systems. You can also create a pattern by entering a new pattern name in the text field next to the Pattern button or clicking on the Pattern button and entering a new pattern in the Selection field of the Pattern object finder dialog that appears; refer to Pattern Object Finder Dialog. After you enter a new name, a Create New Pattern dialog appears, as shown in Figure 17.5. enVision creates a Pattern object with the entered name and no other data. Table 17.10 lists the text fields and buttons for this dialog. Also, refer to Creating a Pattern—Pattern Type, Pattern Name, and Vectors.

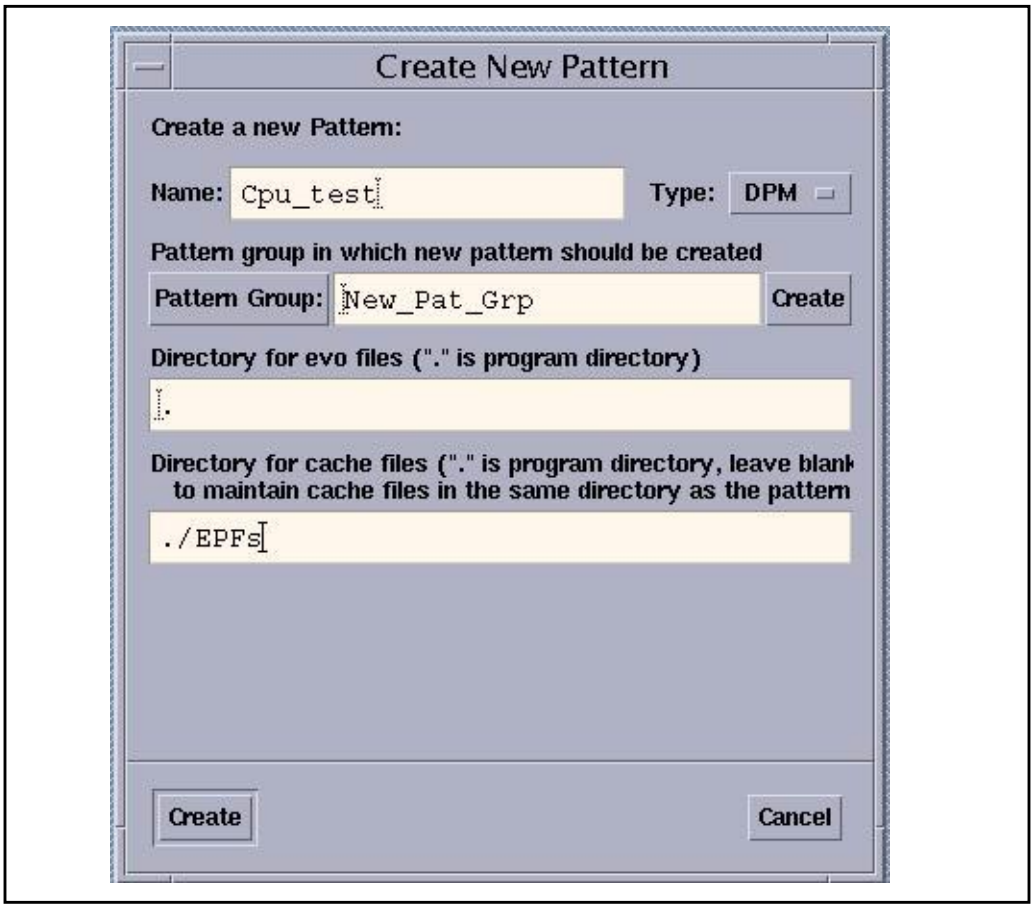


Figure 17.5: Create New Pattern Dialog

Table 17.10: PatternTool—Create New Pattern Dialog

Menu	Action
Name	Name of `Pattern` object. It is the name entered after the keyword *Pattern* in the pattern file. Do not confuse this name with the pattern filename.
Type	Select the type of `Pattern` object. Pattern types are described in *Pattern, Waveform, and Timing* chapter of the *enVision Digital Programming* manual.
Pattern Group	Name of `PatternGroup`. Enter a new or existing Pattern Group name in the text field and click the Create button next to this field; refer to Create Pattern Group Dialog.
Create (Pattern Group)	Click to open a dialog for creating a `PatternGroup` with the name given in the Pattern Group field; refer to Create Pattern Group Dialog.

Table 17.10: PatternTool—Create New Pattern Dialog (Continued)

Menu	Action
Directory for .evo files	Specify the path where pattern source files will be stored, absolute or relative path.
Directory for cache files	Specify the path where the pattern binary (cache) files will be stored, absolute or relative path.
Create (bottom)	Creates the `PatternGroup` with the information specified in the text fields in this dialog.

Search Header Dialog

This dialog finds all or part of a pin name in the Signal Header. Enter the desired text in the field and then click Find; see Figure 17.6.

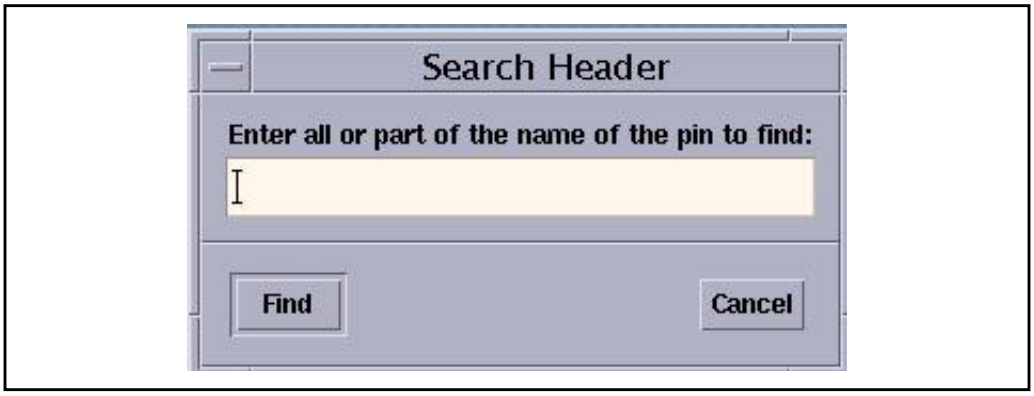


Figure 17.6: Search Header Dialog

PatternTool Tasks

You can edit or modify the pattern elements or manage and operate PatternTool by:

- Editing Vectors
- Using the Mouse for PatternTool Operations
- Using Keyboard Shortcuts for PatternTool Operations
- Setting PatternTool .Xdefaults
- Constructing a New Pattern

- Attaching Pattern Files to an enVision Test Program
- Overlaying Pattern Information to a Test Program

Editing Vectors

Pattern Tool is like a standard text editor: the usual rules for entering text in fields also applies to editing text fields for pattern vectors; refer to Selecting Pattern Characters or Vectors and Performance Notes.

Selecting Pattern Characters or Vectors

Most PatternTool commands operate on the selected text, which is text highlighted by dragging the cursor across it while holding down mouse button M1, or by holding the Shift key while moving the text cursor with the keyboard arrow keys. To select more or less of the highlighted text, press and hold the Shift key while clicking M1. Once you have selected text, commands that otherwise would insert text, instead replace the selected text.

Dragging the mouse over the pattern data highlights one of the two selected types:

- Character selections are contained entirely with a single field.
- Vector selections span all vector fields, may span multiple vectors.

Many commands operate on both character selections and vector selections, but their meaning is different. Vector selections cannot be replaced by character data from the clipboard or visa-versa. Select a vector by dragging the mouse outside of the bounds of a field.

You may select columns of data by enclosing them with a rectangle: either press and hold the Control key while dragging the mouse or press and hold the Alt key while using the arrow keys. A rectangular selection may cross vector boundaries, but it remains in a single field.

In addition to selecting text and vector data, you may select signals in the headers in the PatternTool window, or those displayed in-line with the pattern data. To select a header, drag the mouse within the `SignalHeader`, like selecting text. Header selections specify the signals operated on by the Hide, Show, and Change Radix commands.

Performance Notes

Editing large numbers of vectors can be slow when reading vectors from and writing vectors to the workspace. Once a vector is modified, however, a copy is stored in PatternTool's internal buffers, where access is nearly an order of magnitude faster. Consequently, if you are changing many vectors, LTX recommends editing all vectors together and then executing an Update Workspace command.

Using the Mouse for PatternTool Operations

You can use the mouse buttons combined with other keys to perform certain functions; refer to Table 17.11:

- Left mouse button (M1)—positions cursor and selects.
- Middle mouse button (M2)—moves and copies the selections and drags the field boundaries.
- Right mouse button (M3)—displays context-sensitive menus; refer to Context-Sensitive Menus in PatternTool.

Mouse actions can be modified by modifier keys, which by themselves, do nothing. However, by pressing and holding them in combination with other keys or mouse buttons, the actions of the keys are modified; refer to Modifier Keys.

Table 17.11: Mouse Functions in PatternTool

Button and Context	Function
M1 in Vectors and Comments	Click—moves cursor. Double-click—selects whole word. Triple-click—selects whole line or vector. Shift and click—extends or shrinks selection, or if no existing selection, begins new selection between cursor and mouse. Control and Shift and click—extends or shrinks selection rectangularly. Drag—selects text between where mouse was pressed and where it was released. Control and drag—selects rectangle between where mouse was pressed and where it was released.

Table 17.11: Mouse Functions in PatternTool (Continued)

Button and Context	Function
M1 in Headers	Click—selects a signal. Double click—selects separated set of signals. Triple click—selects whole header. Shift and click—extends or shrinks selection. Drag—selects text between where mouse was pressed and where it was released.
M2	Click—copies primary selection to clicked position. Shift and click—moves selection to clicked position, deleting it from its original position. Drag on field boundary or top-header boundary—extends or shrinks position of boundary.
M3	Displays context-sensitive menu of actions appropriate for vector or header. If multiple vectors are selected, all selected vectors are effected.

Table 17.12: Modifier Keys for Mouse Functions

Key	Modification
Shift key	Extends selection to mouse pointer.
Control key	Makes a selection rectangular.

Context-Sensitive Menus in PatternTool

Clicking the right mouse button on pattern data opens separate context-sensitive menus for selecting vector data and headers. They display selected items from the top-level menus for quicker and convenient access. These menus contains essentially the same choices as the Edit menu, with the addition of column specific items: in the Pattern Group column, the Pattern Group command opens a list of existing `PatternGroups`, and in the remove column, the items Yes and are shortcuts for entering True and False values. Cut, Copy, and Paste from the background menu act on the whole cell, as opposed to selected text as they do in the Edit menu.

If the cursor is not clicked on an existing selection, choosing a command from the context-sensitive menu, rather than from the Edit menu, will act upon or insert at the clicked position.

The difference between using the M3 pop-ups and the menu bar menus or keyboard accelerators is that if a selection does not exist, M3 acts upon the button click position, rather then the cursor position.

Using Keyboard Shortcuts for PatternTool Operations

Keyboard shortcuts are listed on the right hand sides of the pull-down menus. Besides these keyboard shortcuts, you can use modifier keys to enable more shortcuts; refer to Modifier Keys and Action Keys.

Modifier Keys

Pressing modifier keys by themselves has no effect, but pressing and holding them in combination with other keys, modifies the actions of those keys; refer to Table 17.13.

Table 17.13: Modifier Keys

Keys	Modification
Control	Extends scope of action key. For example, Home normally moves cursor to start of a line, while pressing Control and Home moves the cursor to beginning of pattern. Back Space deletes one character, while Control and Back Space deletes one word.
Shift	Extends selection to cursor position. If nothing is already selected, begins selection point between the old and new cursor positions.
Alt	When modifying a selection, makes the selection rectangular.

Action Keys

Action keys and their functions are listed in Table 17.14.

Table 17.14: Action Keys

Keys	Function
Escape	Cancels current operation: menu selection, drag, or selection. Equivalent to Cancel button in dialogs.
Back Space	Deletes character before cursor.
Control and Back Space	Deletes word before cursor.
Left Arrow	Moves cursor to left one character.
Control and Left Arrow	Moves cursor backward one word.
Right Arrow	Moves cursor to the right one character.

Table 17.14: Action Keys (Continued)

Keys	Function
Control and Right Arrow	Moves cursor forward one word.
Up Arrow	Moves cursor up one line.
Down Arrow	Moves cursor down one line.
Return	Inserts a vector.
Shift and Return	Inserts a vector above vector containing cursor.
Control and Return	Inserts a comment after cursor vector.
Control and Shift and Return	Inserts a comment before cursor vector.
Tab	Moves to next field.
Shift and Tab	Moves to previous field.
Control and /	Selects everything. Same as Select All menu item or ^A.
Control and \	Unselects.
Control and U	Deletes to start of line.
Control and Insert	Copies primary selection to clipboard., Same as Copy menu item or ^C.
Shift and Control and Insert	Copies primary selection to cursor location.
Del	Deletes character after cursor (or under cursor in overstrike mode).
Control and Del	Deletes to end of line.
Shift and Del	Removes currently selected text and places it in the clipboard. Same as Cut menu item or ^X.
Home	Moves cursor to beginning of line.
Control and Home	Moves cursor to the first vector of pattern.
End	Moves cursor to the end of line.
Control and End	Moves cursor to last vector of pattern.
Page Up	Scrolls up one page.
Page Down	Scrolls down one page
F10	Activates menu bar for keyboard input (Arrow Keys, Return, Escape, and Space Bar).

Setting PatternTool .Xdefaults

The following list are the .Xdefaults for this tool, with the default values:

```
PatternTool*xdefFont:                    variable
PatternTool*scanWrapMargin:              100
PatternTool*patternBackground:           #e5e5e5
PatternTool*patternForeground:           black
PatternTool*selectBackground:            #bbbbbb
PatternTool*headerBackground:            #cccccc
PatternTool*failColor:                   red
PatternTool*headerEditForeground:        black
PatternTool*readOnlyBackground:          #F4E5D6
PatternTool*headerEditBackground:        #e5e5e5
PatternTool*headerEditTitleBackground: #cccccc
PatternTool*cellColor1:                  black
PatternTool*cellColor2:                  #006400
PatternTool*cellColor3:                  #FFF68F
PatternTool*cellColor4:                  #2368BD
PatternTool*cellColor5:                  #A60049
PatternTool*cellColor6:                  #FFC500
PatternTool*cellColor7:                  #616470
PatternTool*cellColor8:                  #1E90FF
PatternTool*cellColor9:                  #E9967A
PatternTool*cellColor10:                 #00CD00
PatternTool*patternFont:  -adobe-courier-medium-r-
normal-*-14-*-*-*-*-*-*-*

PatternTool*boldPatternFont:-adobe-courier-bold-r-
normal-*-14-*-*-*-*-*-*-*

PatternTool*headerTitleFont:-adobe-helvetica-bold-r-
normal-*-12-*-*-*-*-*-*-*

PatternTool*headerEditFont:  -adobe-courier-medium-
r-normal-*-14-*-*-*-*-*-*-*

PatternTool*headerSignalBoldFont:-misc-fixed-bold-r-
semicondensed--13-*-*-*-*-*-*

PatternTool*headerSignalFont:-misc-fixed-medium-r-
semicondensed--13-*-*-*-*-*-*-*
```

```
PatternTool*headerEditTitleFont:-adobe-helvetica-
bold-r-normal-*-12-*-*-*-*-*-*-*
```

Also, refer to *Xdefaults* section in the *User Interface Tools* chapter of this manual.

Constructing a New Pattern

To construct a new `Pattern` object, complete the following steps:

NOTE Correctly specifying the `PatternGroup` name and pattern type before creating the pattern is preferable since they are more difficult to change after the pattern is created.

- Create a new `PatternGroup` object or use an existing one; refer to Create or Use an Existing PatternGroup Object.
- Create a new `SignalHeader` or use an existing one; refer to Create a Signal Header.
- Create the default `WaveformTable` to define all alias symbols used in the new `Pattern` object; refer to Create a Default WaveformTable for Pattern Object.
- Create the `Pattern` object by selecting the pattern type and defining the labels, vectors, waveform table references, including optional microinstructions and comments; refer to Creating a Pattern—Pattern Type, Pattern Name, and Vectors.

Create or Use an Existing PatternGroup Object

1. If PatternTool is not already started, start it by selecting Operator Tool -> Tools -> Development -> Pattern.
2. In PatternTool:
   a. Click mouse button M1 in the text field next to the Pattern Group button.
   b. To create a Pattern Group, go to step 4. To use an existing Pattern Group, go to step 3.
3. To select an existing Pattern Group:
   a. Click the Pattern Group button. Pattern Group dialog appears.

b. Click an existing name in the Pattern Group field of this pop-up. Its name will appear in the Selection field.

c. Click OK to select the Pattern Group. Go to Create a Default WaveformTable for Pattern Object.

4. Enter the new Pattern Group name and press Return; see Figure 17.7. The Create Pattern Group pop-up appears; refer to Create a Signal Header.

Create a Signal Header

1. In the Create Pattern Group pop-up:

   a. Click M1 in the Name field next to the Header button.

   b. Enter the new Signal Header name.

   c. Click M1 on the Edit/Create button. A Create Header window appears; refer to Figure 17.7.

NOTE If you would have pressed Return after entering the name in the Header field, a pop-up would appear. It would inform you that the Signal Header you entered does not exist and request that you click on the Edit/Create button. If this pop-up appears, click M1 on its Dismiss button.

2. In the Create Header window, click M1 on the Create button. The Signal Header Editor opens.

3. In the Signal Header Editor:

   a. Click M1 on the Edit pull-down menu and choose Add Pin to add a pin name (Figure 17.8) or choose Add Pin Group Pins (Figure 17.9) to add a pin group to the Signal Header. Repeat this step for each pin of the DUT.

   b. Use M2 to re-arrange the signal names, if required; see Figure 17.10.

   c. Use mouse button M3 to change the Sep field from No to Yes if a column of space is needed for readability; see Figure 17.11.

   d. A completed Signal Header is shown in Figure 17.12.

4. After you have completed the Signal Header, choose Close from the PatternTool's File menu to dismiss the window.

5. Create a default WaveformTable; refer to Create a Default WaveformTable for Pattern Object.

Create a Default WaveformTable for Pattern Object

A default `WaveformTable` is necessary for defining the alias symbols used for the pins in the vectors in the new `Pattern` object. These alias symbols are defined with the data direction (Output, Don't Care, Bidirectional or Input) and the data relationship and state. To create a default WaveformTable:

1. In the Create Pattern Group pop-up, click M1 on Edit/Create button next to the field named Default waveform table. WaveformTool appears; see Figure 17.13.

2. In Figure 17.13, WaveformTool shows two default rows. You may keep them, delete a row, or change them to define the alias symbols for the your pins. To delete a row, go to step 4. To change the Pingroup definition; refer to step 3. To add a row, go to step 5.

3. To change the default Pingroup definitions in WaveformTool:

   a. Click M3 in the Pingroup field for the pin group you wish to change and select Find. The Pin Finder dialog appears with the pin highlighted; see Figure 17.14. Note that the highlighted pin group has a plus (+) sign before its name.

   b. Click M1 on the highlighted pin. Note a minus (-) sign appears before the highlighted name in the Pingroup field in WaveformTool and in the Selected Pin Expression field of the Pin Finder dialog. You have re-defined the Pingroup to not include the highlighted.

   c. In the Pin Finder dialog, click OK to confirm the changes. Go to step 6.

4. To delete a row in WaveformTool:

   a. Click M3 in the Row you wish to delete and select Delete Row from the context-sensitive menu.

   b. In the confirm Delete Row dialog, click OK. Go to step 6.

5. To add a new row in WaveformTool:

   a. Position the cursor where you wish to add a row and click M3. The Add Row-WaveformCell pop-up opens.

b. Select the pins and define the row name. Go to step 6.

6. In WaveformTool, select Exit from the File menu. WaveformTool closes.

7. In the Create Pattern Group pop-up, click M1 on the Create button to complete the Pattern Group creation.

8. Create a pattern by naming it and by selecting the pattern type; refer to Creating a Pattern—Pattern Type, Pattern Name, and Vectors.

Creating a Pattern—Pattern Type, Pattern Name, and Vectors

Before naming the new `Pattern` object, you must define the pattern type, which specifies where the pattern data will be loaded in the tester. Next, the vectors are defined by using the alias symbols already entered in the default WaveformTable. Also, you can now specify any labels, waveform table references, and microinstructions.

1. Specify the pattern type by selecting the required type from the Type field in PatternTool; see Figure 17.15.

2. Name the `Pattern` object by using PatternTool:

   a. Ensure the name of the proper Pattern Group appears in the Pattern Group field.

   b. Click M1 in the field next to the Pattern button. Enter the pattern name and press Return. A Create New Pattern window appears.

NOTE Even though you can change the name, type, or group of an existing pattern, changing one may affect the other two. For example, changing the name of an existing pattern requires changing the name of its associated source and cache files. Also, changing the type or group is a drastic change, requiring PatternTool to erase and regenerate the pattern. Also, when you change the type of an existing pattern, incompatible micro-instructions are dropped.

   c. In the Create New Pattern window, click M1 on Create. Note the LTX logo is now inverse video. Also, the Signal Header appears, with the cursor is placed below the first pin in index 0, waiting for the first vector to be entered.

3. Create the test pattern vectors by entering the alias symbols for the pins, necessary labels and waveform table references, and optional microinstructions. You may cut, copy, and paste to create multiple vectors.

4. Update the workspace by choosing Update Workspace from the File menu. Note the LTX logo changes from reverse video to normal video after the workspace is updated.

5. Save the test program by choosing Save from the File menu in the OperatorTool. By saving the test program the `Pattern` object is saved into an ASCII (.evo) and binary (.flex) files and also into the common pattern group file.

Attaching Pattern Files to an enVision Test Program

One method to add patterns to an enVision test program is by introducing patterns from an external pattern file into the existing enVision test program with the Attach Ascii (.evo) Pattern File(s) command. You can either use existing `PatternGroup` and `SignalHeader` objects or create them. This method assumes you have already created a `PatternGroup` and `SignalHeader` objects.

1. In PatternTool, acess the pattern file to be attached by choosing Attach Ascii (.evo) Pattern Files from the File menu. The Attach Pattern Files dialog appears; see Figure 17.16.

2. In the Attach Pattern Files dialog,

   a. In the Pattern Group field, enter a new or existing Pattern Group in the Pattern Group field; refer to Create or Use an Existing PatternGroup Object. If you are creating a Pattern Group, also refer to Create a Signal Header and Create a Default WaveformTable for Pattern Object.

   b. In the Directory for cache files field, enter the the directory where the cache files of the pattern file will be stored. If you enter a period (.), the cache files are placed in the same directory as the test program. Leaving the text field blank will place the cache files in the same directory as the ASCII pattern files.

   c. Click Add File button. A file browser dialog appears.

3. In the file browser dialog, navigate to the pattern file to load. Click M1 on the file to load. Its name appears in the selection field. Click OK to add the file to the Attach Pattern dialog; see Figure 17.16.

4. In the Attach Pattern Files dialog, click Attach to start loading the pattern file.

5. After the pattern starts loading, ErrorTool will display a message. Click Yes to continue loading the pattern.

6. In PatternTool:

   a. Click the Pattern button. The Pattern finder dialog appears. It lists the attached pattern. Select the attached pattern and click OK to load it.

   b. Save the pattern file by selecting Save As from the File menu and naming the pattern file.

Overlaying Pattern Information to a Test Program

You can overlay a complete pattern file into an empty test program. A sample .eva pattern file is shown in Figure 17.17. An empty test program does not contain any pattern information; see sample .evo file in Figure 17.18. Once the pattern file is overlaid, the `PatternGroups`, `Signal Headers`, default WaveformTable, and external references to the `Pattern` object can be inserted into a test program. The pattern files must meet the requirements listed in Table 17.15. You must save the test program to retain the overlaid information.

Table 17.15: Requirements for Overlaying Pattern Files

File	Object	Notes
.eva	`SignalHeader`	Pin names in SignalHeader must be match those in overlay pattern file.
	`PatternGroup`	Defined for all pattern under it.
	External reference	One per pattern. Specifies pathname to each .evo pattern file and its associated .flex file
	Default `WaveformTable`	Used by `PatternGroup` to determine the data states and signal direction of all alias characters within all patterns in a `PatternGroup`. Must be related to `PatternGroup`.
.evo	---	Source patterns formatted in enVision pattern syntax.

1. In PatternTool, select Load from the File menu. The Select Test Program to Load dialog appears; see Figure 17.19.

2. In The Select Test Program to Load dialog, select the test program file to be overlayed. It is an ASCII file with an .eva extension.

3. In OperatorTool, select Overlay File from the File menu. The Overlay File dialog opens.

4. In the Overlay File dialog, select the overlay pattern information file and click OK. It is an ASCII file with an .eva extension. You can view it to confirm it has the `SignalHeader`, `PatternGroup`, and default `WaveformTable` objects and external references, including the external reference to the .evo `Pattern` object file.

5. Depending on your enVision software release, a status window may appear, showing the loading progress.

6. The Error Message Log appears in ErrorTool. Click Yes to compile the pattern and continue loading the pattern.

7. In PatternTool, click the Pattern button. A Pattern object finder appears; refer to Pattern Object Finder Dialog.

8. In the Pattern object finder, select the desired `Pattern` object to use in your test program and click OK. The number of `Pattern` object appearing in this dialog depends on the number of `Pattern` objects in the overlay file. The selected Pattern object appears in the main window of PatternTool.

9. In OperatorTool, select Save As from the File menu. The Save Test Program As dialog opens.

10. In the Save Test Program As dialog, navigate to the proper directory and enter the filename of the test program to be saved in the Test Program Name field. Select .eva extension. Click OK to save the test program.

11. Reload the test program to have the overlaid pattern take effect.

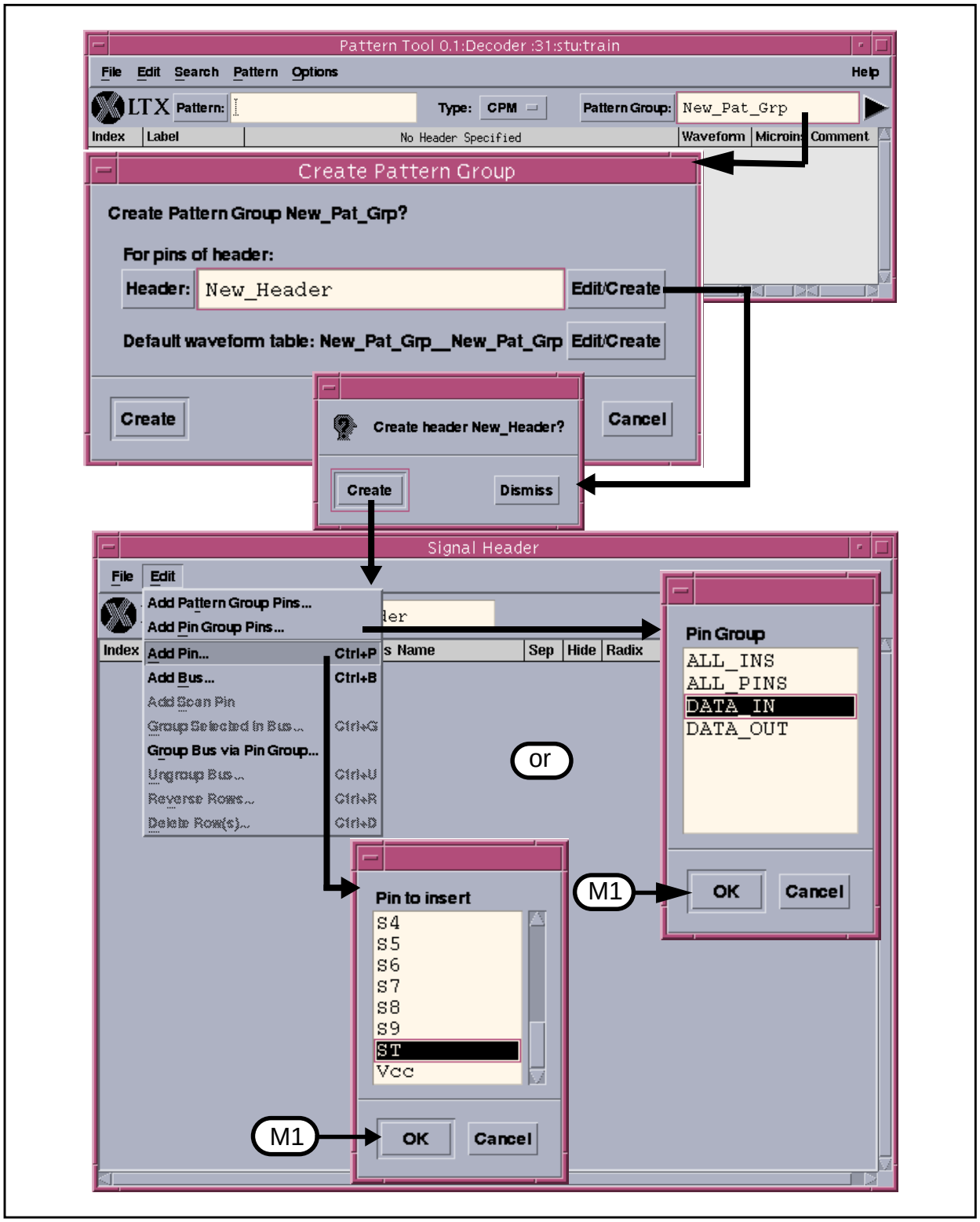


Figure 17.7: Creating Pattern Group and Signal Header

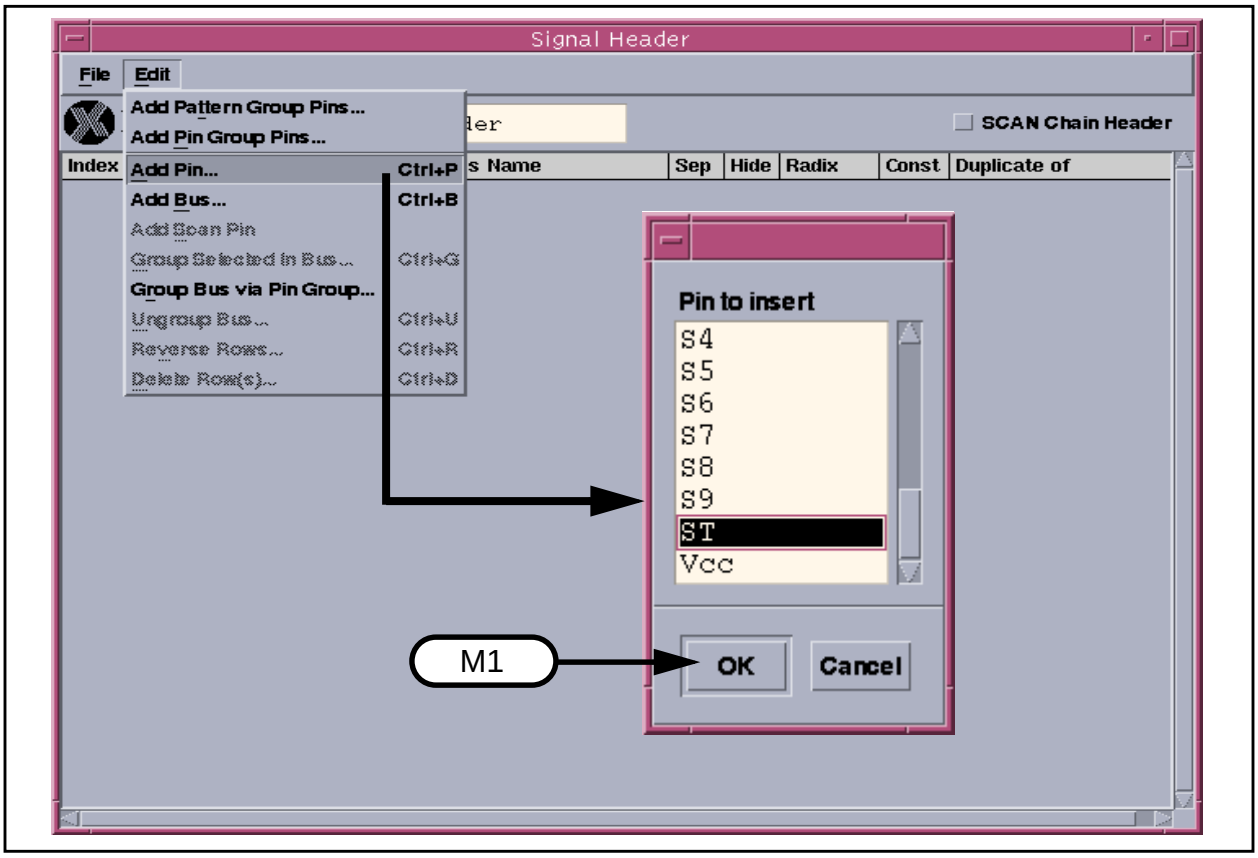


Figure 17.8: Adding a Pin to the Signal Header

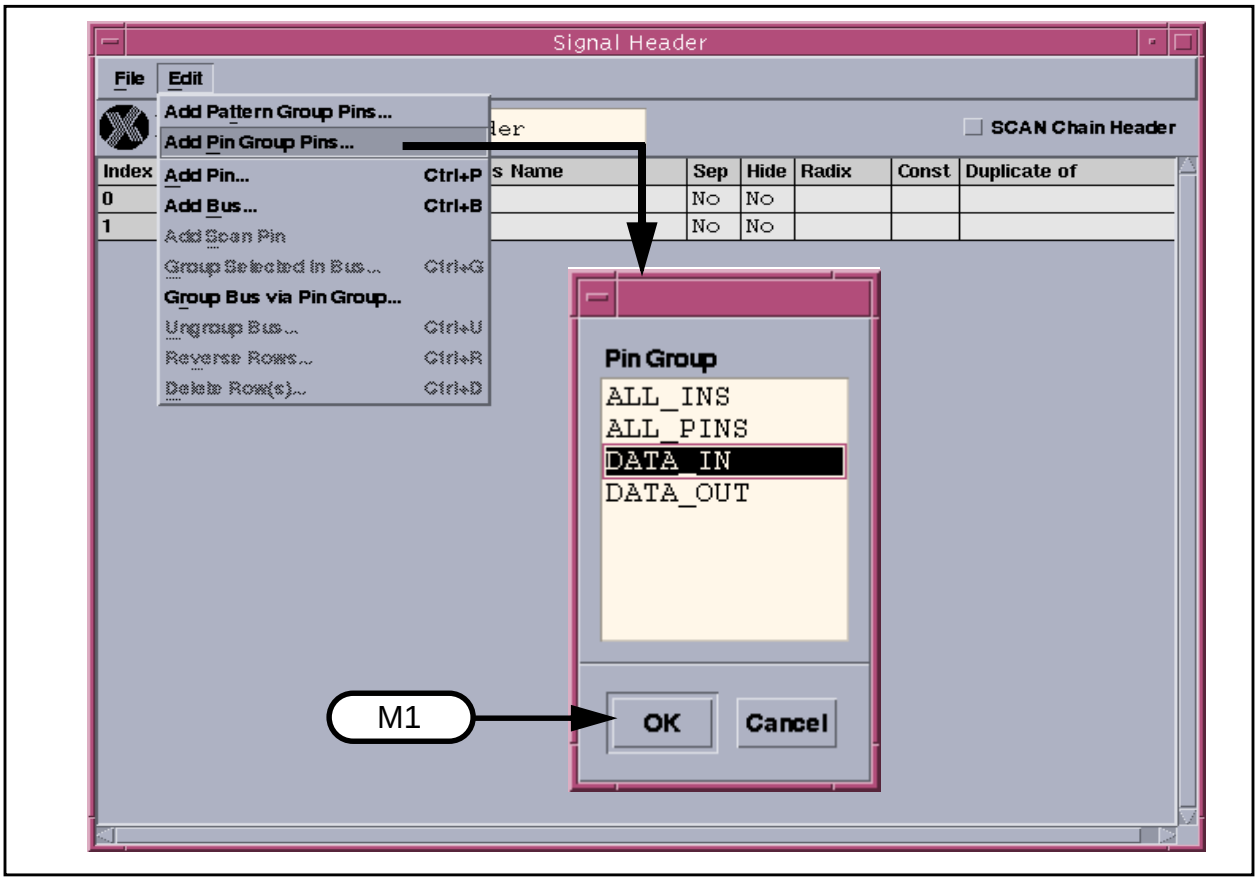


Figure 17.9: Adding a Pin Group to the Signal Header

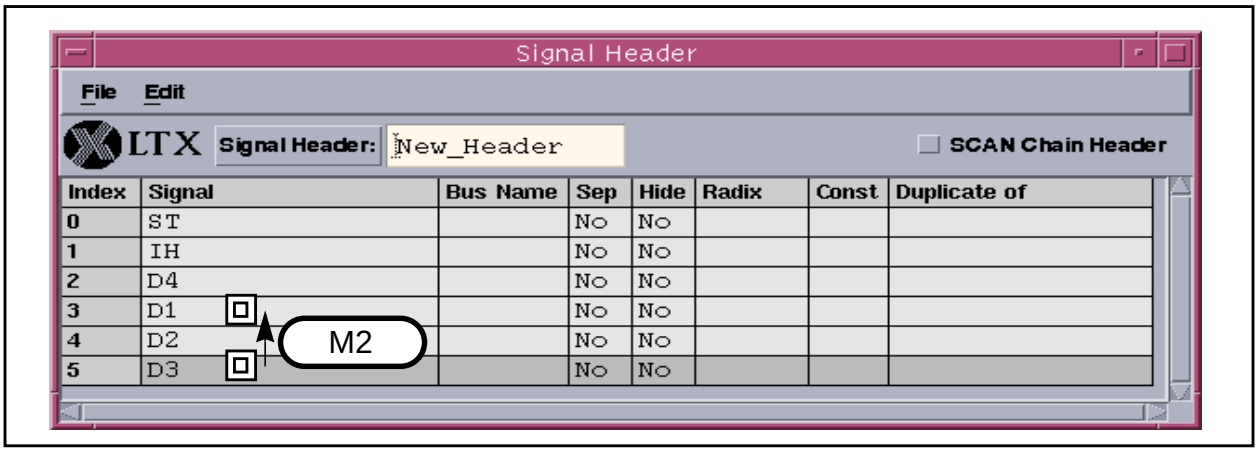


Figure 17.10: Using M2 to Reposition the Pin Names

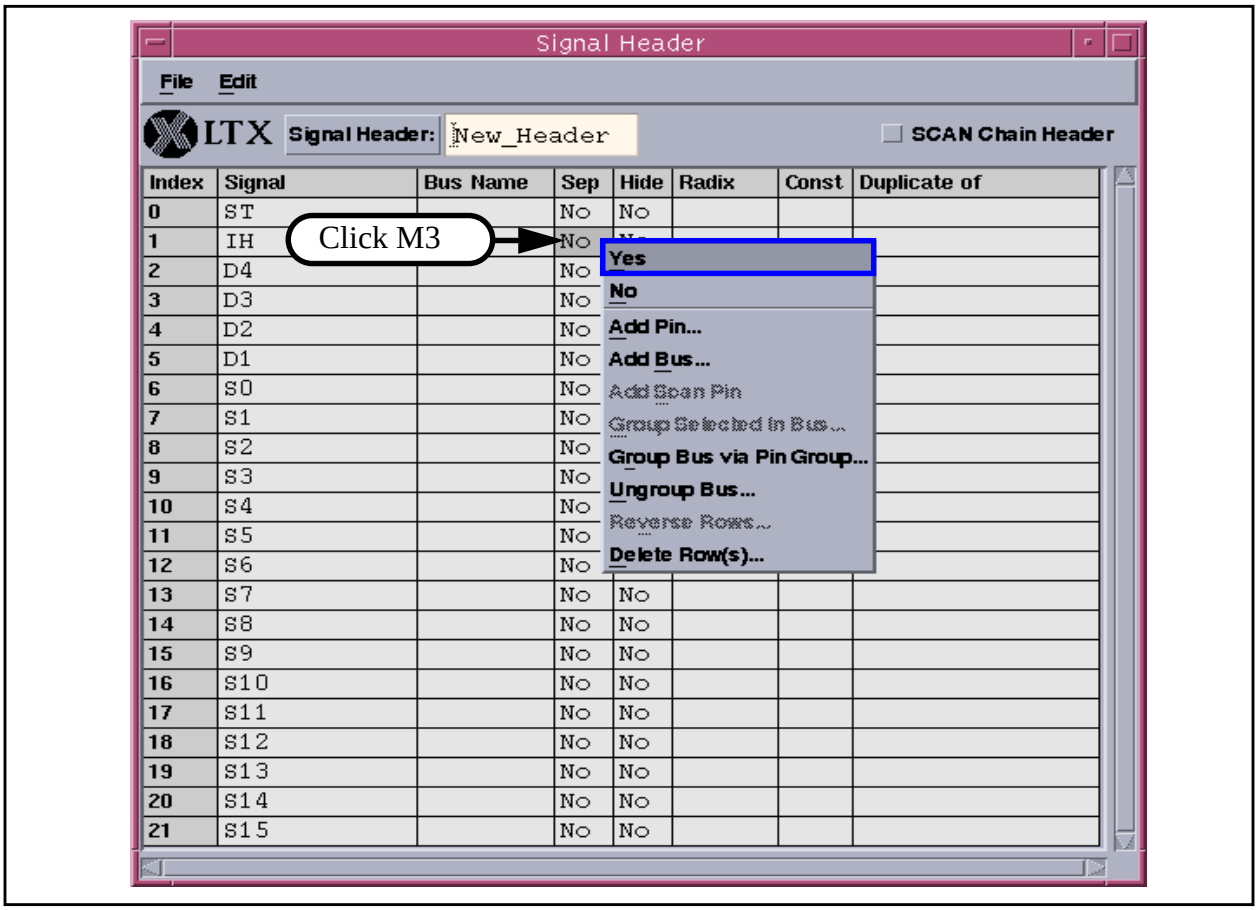
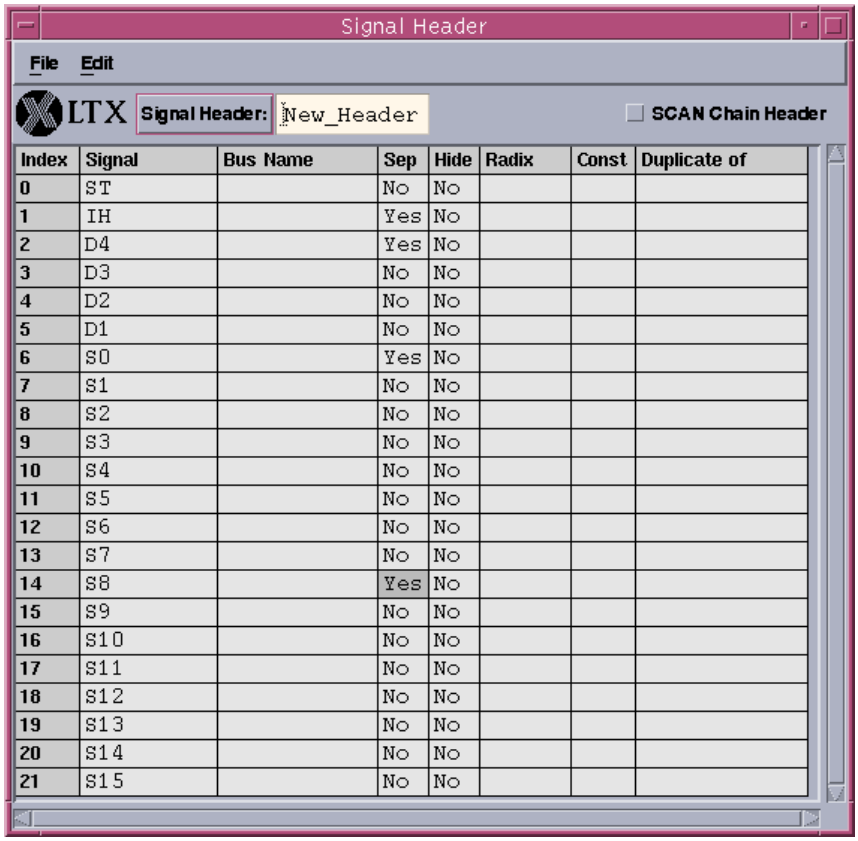


Figure 17.11: Changing Sep Field from No to Yes



Index	Signal	Bus Name	Sep	Hide	Radix	Const	Duplicate of
0	ST		No	No			
1	IH		Yes	No			
2	D4		Yes	No			
3	D3		No	No			
4	D2		No	No			
5	D1		No	No			
6	S0		Yes	No			
7	S1		No	No			
8	S2		No	No			
9	S3		No	No			
10	S4		No	No			
11	S5		No	No			
12	S6		No	No			
13	S7		No	No			
14	S8		Yes	No			
15	S9		No	No			
16	S10		No	No			
17	S11		No	No			
18	S12		No	No			
19	S13		No	No			
20	S14		No	No			
21	S15		No	No			

Figure 17.12: Completed Signal Header

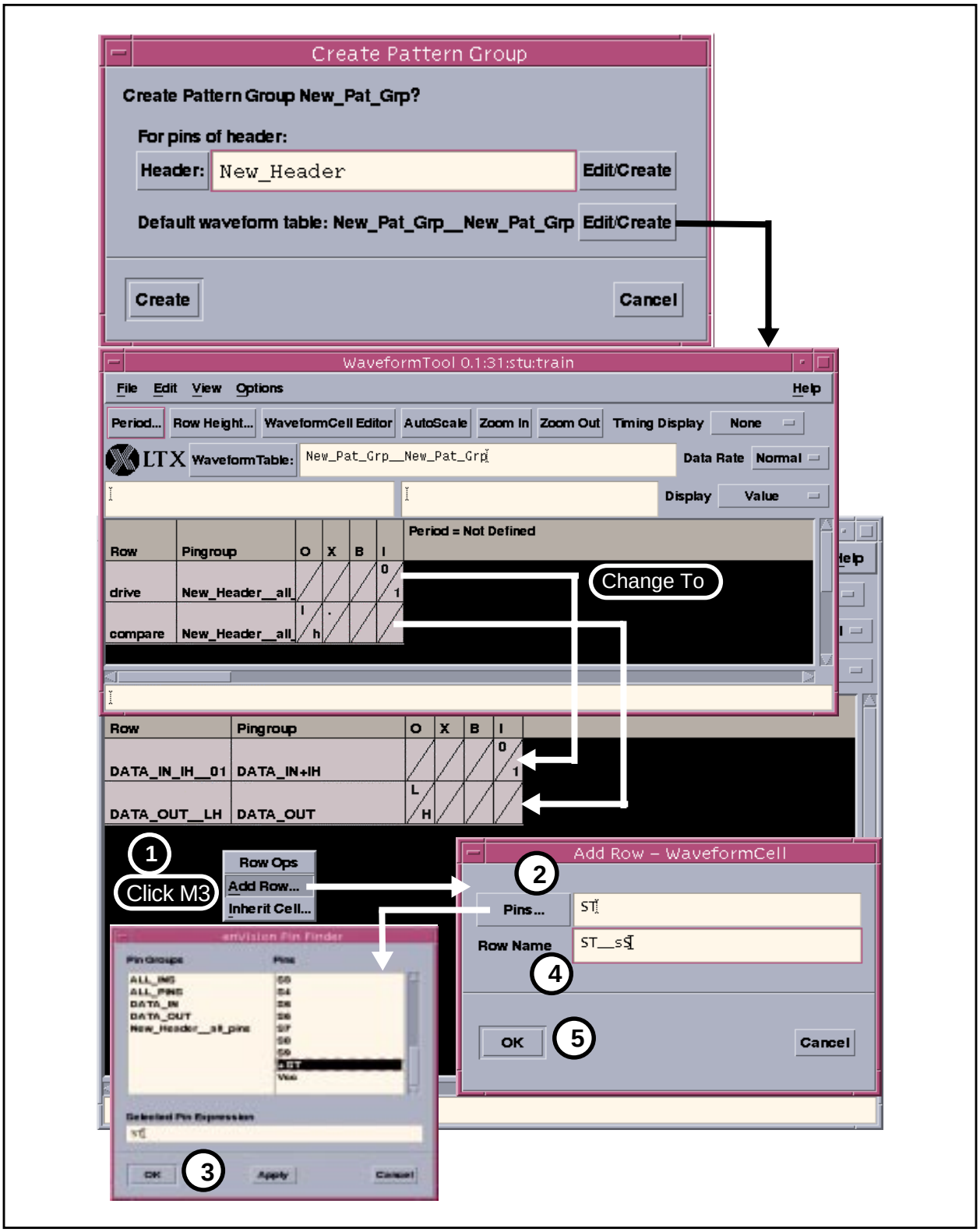


Figure 17.13: Creating Default WaveformTable

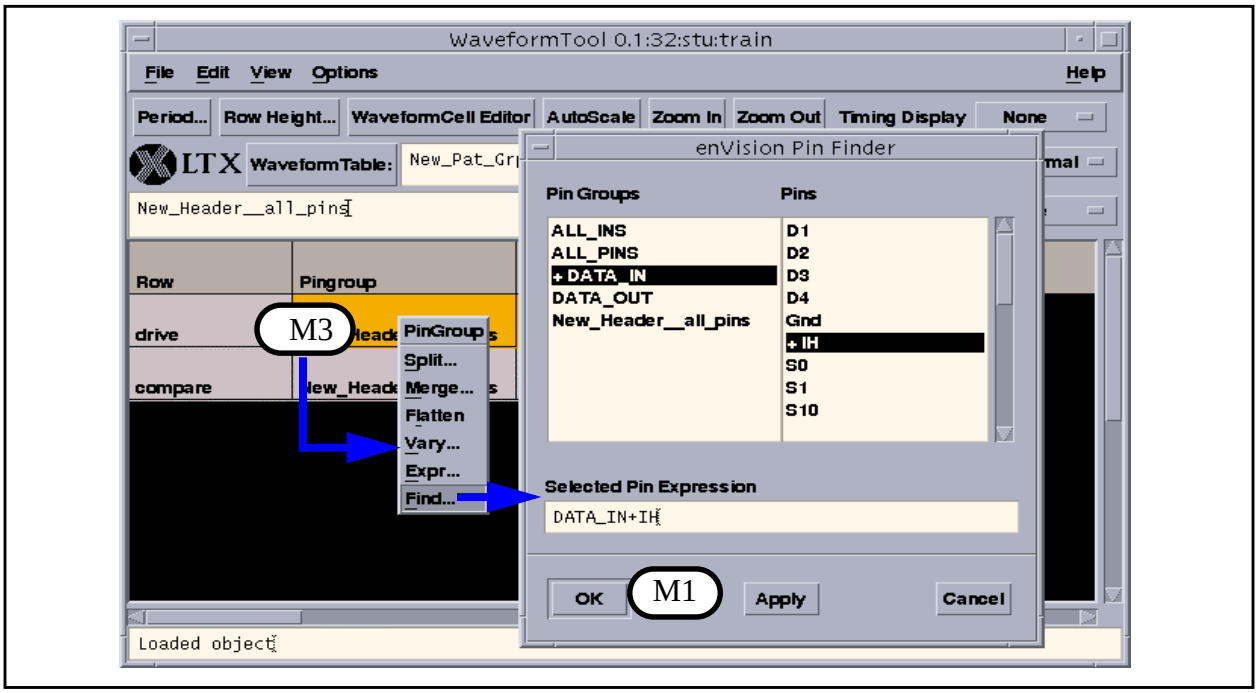


Figure 17.14: Pin Finder Dialog and WaveformTool

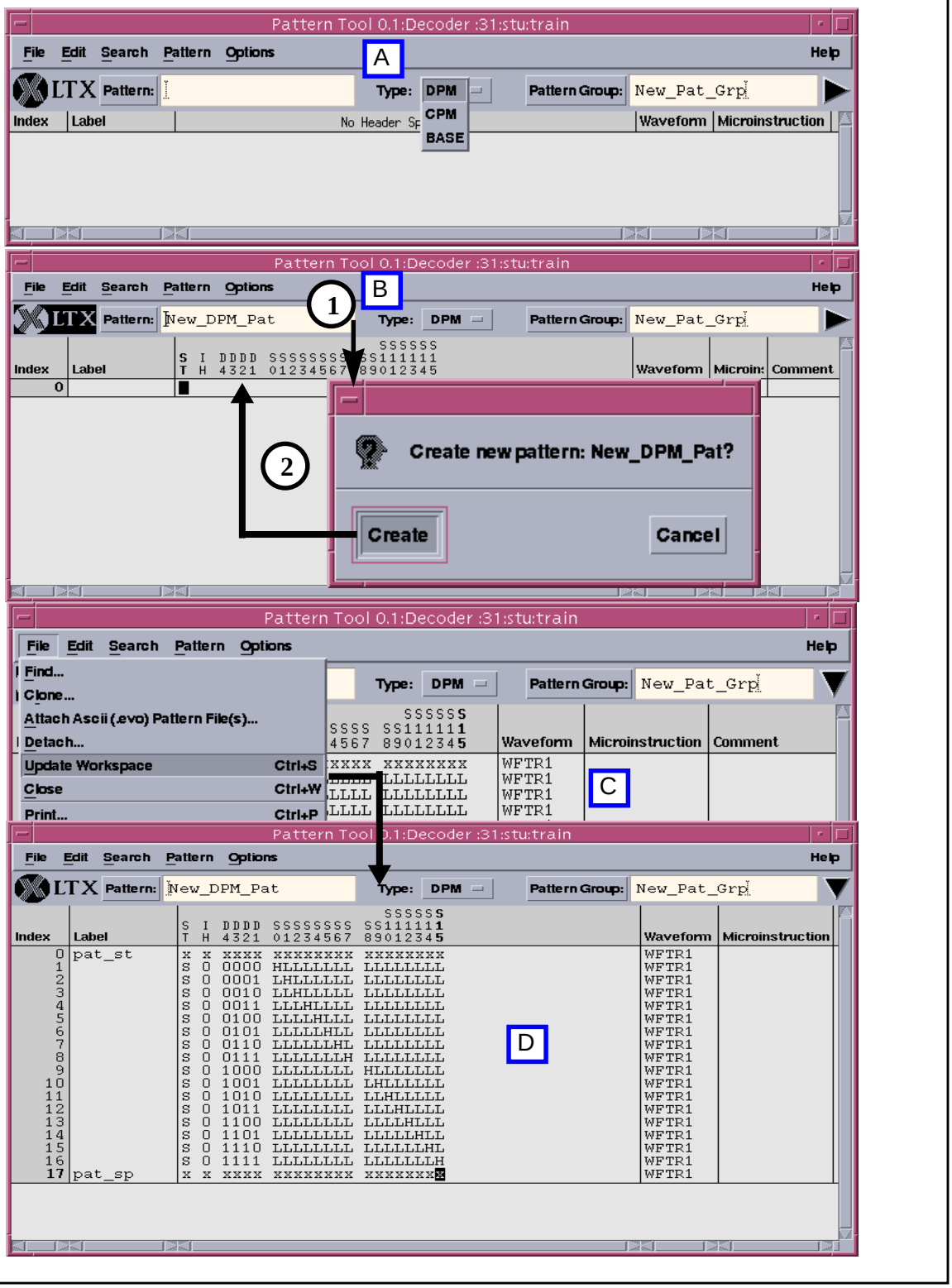


Figure 17.15: Creating the Pattern Object

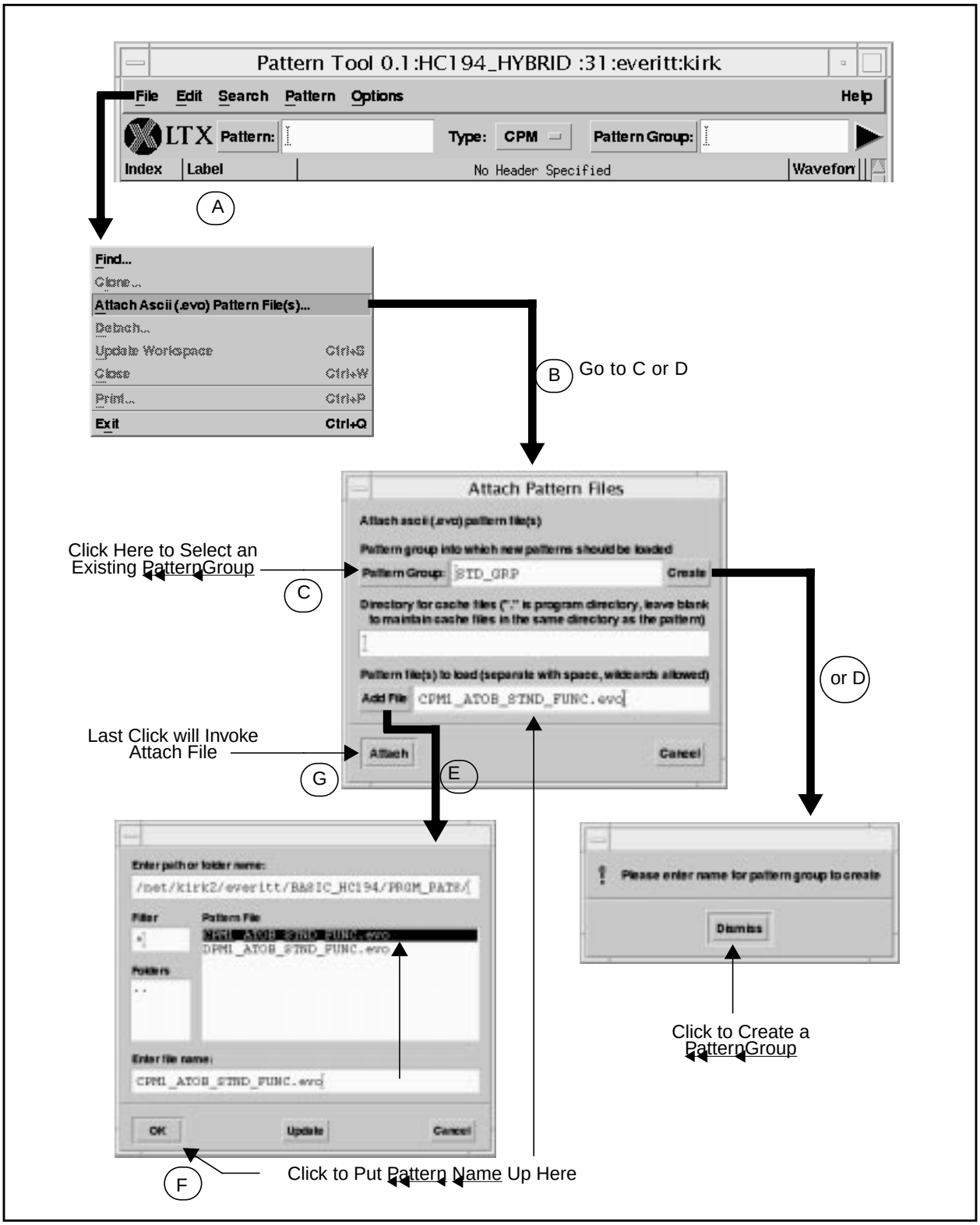


Figure 17.16: Attaching Pattern Files

Text Editor – ONLY_PAT_OBJS.eva

File Edit Format Options Help

```
enVision:"bl8:R10.3.0.SJ0916:S3.0"; /* 5.7 */

SignalHeader STD_HDR {
    Signals {
        RSTL %CLKH %SEL4 SEL3 SEL2 SEL1 %SIDT SIDF %AIO4 AIO3 AIO2 AIO1
        %BIO4 BIO3 BIO2 BIO1 %SODT SODF
    }
}
PatternGroup STD_GRP {
    Cache;
    CachePath ".";
    SignalHeader STD_HDR;
}
WaveformTable STD_GRP STD_GRP {
    Cell "ALL_DUT" 0/1 DEFAULT_01 {
        Data 6/7;
    }
    Cell "ALL_DUT" c/C DEFAULT_02 {
        Data 6/7;
    }
    Cell "ALL_DUT" L/H DEFAULT_03 {
        Data 0/1;
    }
    Cell "ALL_DUT" z/Z DEFAULT_04 {
        Data 0/1;
    }
    Cell "ALL_DUT" X DEFAULT_07 {
        Data 2;
    }
}
ExternalRef {
        Type = Pattern;
        Path = "PRGM_PATS/";
        File = "CPM1_ATOB_STND_FUNC.evo";
        CachePath = "./PRGM_PATS";
        Cache = CPM1_ATOB_STND_FUNC;
        ObjectInfo = "STD_GRP";
}
ExternalRef {
        Type = Pattern;
        Path = "PRGM_PATS/";
        File = "DPM1_ATOB_STND_FUNC.evo";
        CachePath = "./PRGM_PATS";
        Cache = DPM1_ATOB_STND_FUNC;
        ObjectInfo = "STD_GRP";
}
```

SignalHeader Definition

PatternGroup Declaration

PatternGroup Default WaveformTable

External Object Type

Directory for .evo File

Name of .evo File

Directory Holding .flex File

Name of Pattern

PatternGroup Owning This Pattern

Note: This .eva File will not Run as a Stand-Alone Program

Figure 17.17: Sample .eva Pattern File with Only Pattern-Related Objects

Text Editor – CPM1_ATOB_STND_FUNC.evo

File Edit Format Options Help

```
enVisionObject:"bl8:R6.0.2.P"; /* 5.7 */

Pattern CPM1_ATOB_STND_FUNC {
Type Cpm;
Default SignalHeader STD_HDR;

$STRT_ATOB
*0 X 0001 XX XXXX LLLL XX* SEL_OPS;
*1 C 0001 XX 0000 LLLL XX* STD_FNL;
*1 C 0001 XX 0001 LLLH XX* STD_FNL;
*1 C 0001 XX 0010 LLHL XX* STD_FNL;
*1 C 0001 XX 0011 LLHH XX* STD_FNL;
*1 C 0001 XX 0100 LHLL XX* STD_FNL;
*1 C 0001 XX 0101 LHLH XX* STD_FNL;
*1 C 0001 XX 0110 LHHL XX* STD_FNL;
*1 C 0001 XX 0111 LHHH XX* STD_FNL;
*1 C 0001 XX 1000 HLLL XX* STD_FNL;
*1 C 0001 XX 1001 HLLH XX* STD_FNL;
*1 C 0001 XX 1010 HLHL XX* STD_FNL;
*1 C 0001 XX 1011 HLHH XX* STD_FNL;
*1 C 0001 XX 1100 HHLL XX* STD_FNL;
*1 C 0001 XX 1101 HHLH XX* STD_FNL;
*1 C 0001 XX 1110 HHHL XX* STD_FNL;
*1 C 0001 XX 1111 HHHH XX* STD_FNL;
$STOP_ATOB
*1 X 0000 XX ZZZZ ZZZZ ZZ* SEL_NOP;
}
```

User-Defined Pattern Name
Define This Pattern as Cpm Type
Default Header for this Pattern
Typical Pattern Label
Pattern Vector with its Waveform Table Reference Pointer (Sel_ops)

Text Editor – DPM1_ATOB_STND_FUNC.evo

File Edit Format Options Help

```
enVisionObject:"bl8:R6.0.2.P"; /* 5.7 */

Pattern DPM1_ATOB_STND_FUNC {
Type Dpm;
Default SignalHeader STD_HDR;

$STRT_ATOB
*0 X 0001 XX XXXX LLLL XX* SEL_OPS;
*1 C 0001 XX 0000 LLLL XX* STD_FNL;
*1 C 0001 XX 0001 LLLH XX* STD_FNL;
*1 C 0001 XX 0010 LLHL XX* STD_FNL;
*1 C 0001 XX 0011 LLHH XX* STD_FNL;
*1 C 0001 XX 0100 LHLL XX* STD_FNL;
*1 C 0001 XX 0101 LHLH XX* STD_FNL;
*1 C 0001 XX 0110 LHHL XX* STD_FNL;
*1 C 0001 XX 0111 LHHH XX* STD_FNL;
*1 C 0001 XX 1000 HLLL XX* STD_FNL;
*1 C 0001 XX 1001 HLLH XX* STD_FNL;
*1 C 0001 XX 1010 HLHL XX* STD_FNL;
*1 C 0001 XX 1011 HLHH XX* STD_FNL;
*1 C 0001 XX 1100 HHLL XX* STD_FNL;
*1 C 0001 XX 1101 HHLH XX* STD_FNL;
*1 C 0001 XX 1110 HHHL XX* STD_FNL;
*1 C 0001 XX 1111 HHHH XX* STD_FNL;
$STOP_ATOB
*1 X 0000 XX ZZZZ ZZZZ ZZ* SEL_NOP;
}
```

Define This Pattern as Dpm Type

Figure 17.18: Sample .evo Pattern File

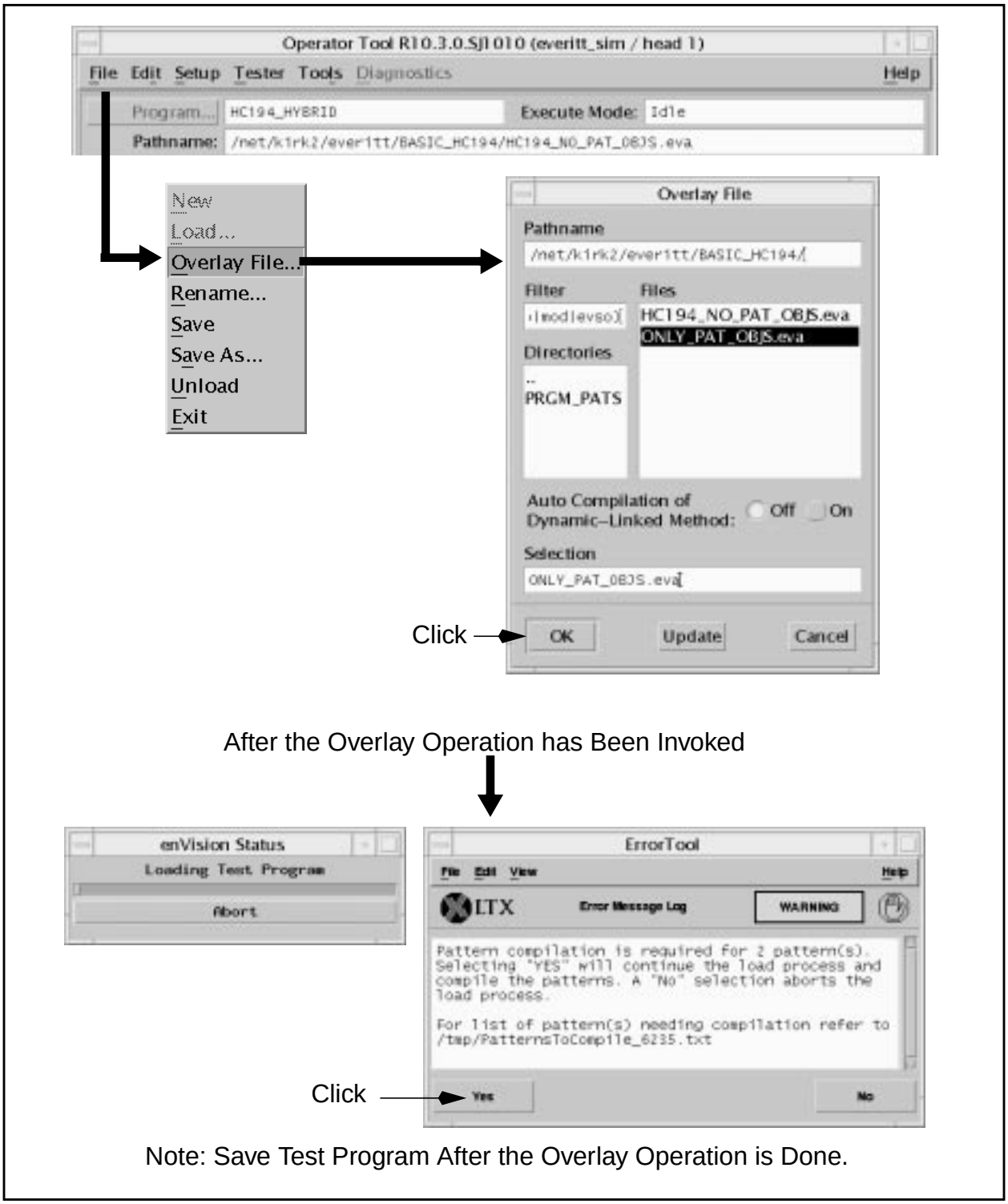
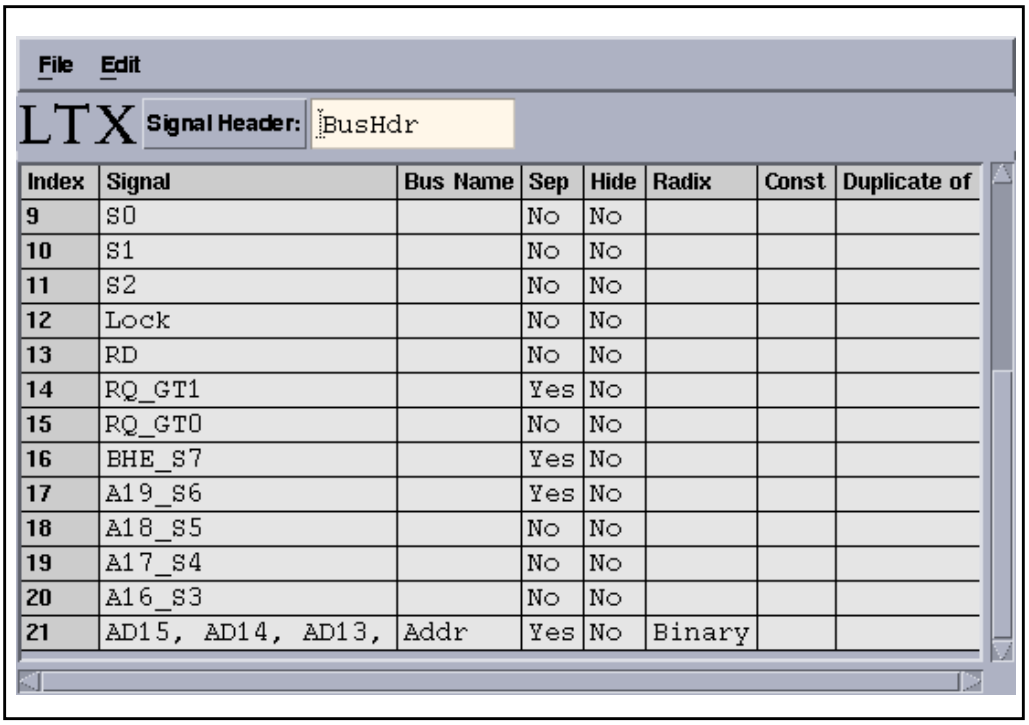


Figure 17.19: Overlaying the Skeleton .eva Pattern File

Signal Header Editor

The SignalHeader Editor is invoked by selecting Edit Signal Header from the Pattern menu of the Pattern Tool. It is used to create, edit, and view SignalHeader objects, which define the order and display characteristics of the pins in patterns; refer to Figure 17.20

You can freely edit the SignalHeaders only until they are referenced in a pattern or PatternGroup; refer to Signal Header Editor Fields. Once used, they become static and can be changed only by copying (use Clone command in File menu) and then re-referencing in the patterns; refer to Signal Header Editor Menus.



Index	Signal	Bus Name	Sep	Hide	Radix	Const	Duplicate of
9	S0		No	No			
10	S1		No	No			
11	S2		No	No			
12	Lock		No	No			
13	RD		No	No			
14	RQ_GT1		Yes	No			
15	RQ_GT0		No	No			
16	BHE_S7		Yes	No			
17	A19_S6		Yes	No			
18	A18_S5		No	No			
19	A17_S4		No	No			
20	A16_S3		No	No			
21	AD15, AD14, AD13,	Addr	Yes	No	Binary		

Figure 17.20: Signal Header Editor Display

Signal Header Editor Fields

You can change the contents of the displayed fields by clicking mouse button M3 over the field, causing a context sensitive menu to appear. In this menu the items for changing the field contents appear first. Some fields contain values that can be changed by double clicking on them with mouse button M1 (yes / no); refer to Table 17.16.

Table 17.16: SignalHeader Editor Fields

Field	Meaning
Signal	Name of signal (or signals in a bus) to be displayed.
Bus Name	For buses, name to be displayed in PatternTool; otherwise, it is empty. Buses are required to group a number of pins, so you can view or enter data in a condensed form for all those pins.
Sep	Separate signal from its previous signal by a space.
Hide	Do not display this signal in PatternTool. Duplicate and constant pins are always hidden.
Radix	On buses, sets the radix in which the pattern data will be read, and default radix to display data in Pattern Tool. Radix modes: Per Name, Binary, Octal, or Hex; refer to Example: Radix Mode in Pattern File and Valid Alias Characters.
Constant	Sets pin to a constant value. Constant pins are not assigned a column in pattern file, and are not displayed in PatternTool. Buses may not be constant.
Duplicate of	Sets pin to same value as another pin. Duplicate pins are not assigned a column in pattern file, and are not displayed in PatternTool. Buses may not be duplicated.

Signal Header Editor Menus

The Signal Header Editor has the following menu items:

- File—Table 17.17
- Edit—Table 18

This editor has context-sensitive pop-up menus that are opened by using mouse button M3; refer to Context-Sensitive Menus in Signal Header Editor.

Context-Sensitive Menus in Signal Header Editor

Clicking mouse button M3 in the header display area opens a context-sensitive menu for the selected row and column. Choices include items for changing the selected field and selected items from the menu bar menus. If a selection does not exist, choosing a command from the context sensitive menu, rather than from the Edit menu, will act upon or insert data at the clicked position. Table 17.19 lists the items in the context-sensitive pop-up for this editor.

Table 17.17: Signal Header Editor—File Menu

Selection	Action
Find	Same as pressing Signal Header button at top-left of Signal Header panel. Object finder dialog opens for selecting header for display, or for entering a name to create a `SignalHeader`; refer to
Delete	Removes `SignalHeader` object. Headers already referenced in patterns may not be deleted.
Clone	Copies currently-displayed `SignalHeader` object.
Close	Closes Signal Header Editor window.

Table 17.18: Signal Header Editor—Edit Menu

Selection	Action
Add Pattern Group Pins	Opens object-finder dialog for choosing a `PatternGroup`, and adds all pins from this group not already part of the header to the header; refer to Create New Pattern Dialog.
Add Pin Group Pins	Opens object-finder dialog for choosing a pin group, and adds all pins from the group not already part of header to header; refer to Create New Pattern Dialog.
Add Pin	Adds single pin to header.
Add Bus	Adds group of pins to header as a bus.
Add Scan Pin	Adds single scan pin to header.
Group Selected in Bus	Groups signals currently selected into a single bus. Order of bus is order of signals as they appear in header.
Group Bus via Pin Group	Prompts user for a pin group and then groups the header pins belonging to that pin group into a single bus. If some pins are not already in header, a dialog prompts you to add these pins to header.
Ungroup Bus	Returns a bus to its constituent pins.
Reverse Rows	Reverses the order of selected group of signals.
Delete Row(s)	Deletes one or more signals from header.

Table 17.19: Signal Header Editor—Context Sensitive Menu

Selection	Action
Yes, No, As Name, Binary, Octal, or Hex	Sets field value to selected item name.
Set Const Value	Converts pin to a constant pin and prompts for a new constant value. Constant value must be alias character valid for that signal.
Remove Const Value	Removes constant attribute from pin, converting it to a normal signal.
Set to Duplicate of Pin	Duplicates a pin from another pin, prompts for pin to duplicate.
Clear Duplicate Pin	Removes duplicate attribute from pin, converting it to a normal signal.

Signal Header Editor Tasks

You can edit or modify the `SignalHeader` by:

- Creating a Header
- Selecting Signals and Fields
- Rearranging Pin Order in Header

Creating a Header

You create a `SignalHeader` by using the adding pins to a `PatternGroup`; refer to the procedures to Create or Use an Existing PatternGroup Object and to Create a Signal Header. LTX recommends adding signals to a new header from an existing pin group or `PatternGroup`. Once the signals are entered, you can collect them into buses or leave them as individual signals. Grouping the pins into a bus lets you view and enter data in Pattern Tool in a radix rather than binary; refer to Rearranging Pin Order in Header.

Selecting Signals and Fields

Most Signal Header Editor commands, particularly those for groups of signals, affect the selected field or selected set of signals. To select a field, click on the field.

To select a group of signals, drag the left mouse button across all desired rows, or simultaneously press both Shift and Click to select additional signals.

Rearranging Pin Order in Header

To rearrange the pin order in a header, select the set of pins to move by dragging mouse button M1 over them. Then, drag the selected group of signals to the final location with mouse button M2. The cursor will be transformed into a box while you are dragging it.

Example: Radix Mode in Pattern File

In the following example, pins A0 to A7 (8 pins) are grouped to form a bus, which has the following representation:

```
Per Name  :                H L H L L H H L
Binary    :  <alias char> 1 0 1 0 0 1 1 0
Octal     :  <alias char> 2 4 6
Hex       :  <alias char> A 6
```

The alias character depends on its definition in the `PatternGroup` WaveformTable; refer to Valid Alias Characters.

Valid Alias Characters

Table 17.20 list the alias characters for the Radix mode pattern files:

Table 17.20: Allowed Radix Mode Alias Characters

Character	Representation
a to z	a. . . z
A to Z	A . . . Z
0 to 9	0 . . . 9
underscore	_
comma	,
left brace	{
right brace	}
percent sign	%
semicolon	;
colon	:
period	.

Table 17.20: Allowed Radix Mode Alias Characters (Continued)

Character	Representation
pound sign	#
equal sign	=
less than sign	<
greater than sign	>
dash	–

Pattern Map Editor

The Pattern Map Editor is used to create, edit, and view the `PatternMap` objects that associate the pattern files and `PatternGroups` with the `Pattern` objects; see Figure 17.21. Invoke this editor by selecting Edit Pattern Map from the Pattern menu of Pattern Tool. The `Pattern`, `PatternGroup`, and `PatternMap` objects are described in the *Pattern, Waveform, and Timing* chapter in the *enVision Digital Programming* manual.

A typical enVision program has only one `PatternMap` object, designated the *active* Pattern Map, which is set either in the Pattern Map Editor window, or in the Active Object menu of Operator Tool, under Setup; refer to Pattern Map Editor—Main Window, Pattern Map Editor Fields, and Pattern Map Editor Menus.

Even though most test programs have one Pattern Map, you can use multiple Pattern Maps for revision control or for managing a larger pattern file by dividing it into smaller files; refer to Adding Patterns. Multiple Pattern Maps are attached to the active Pattern Map through inheritance; refer to Using Inheritance.

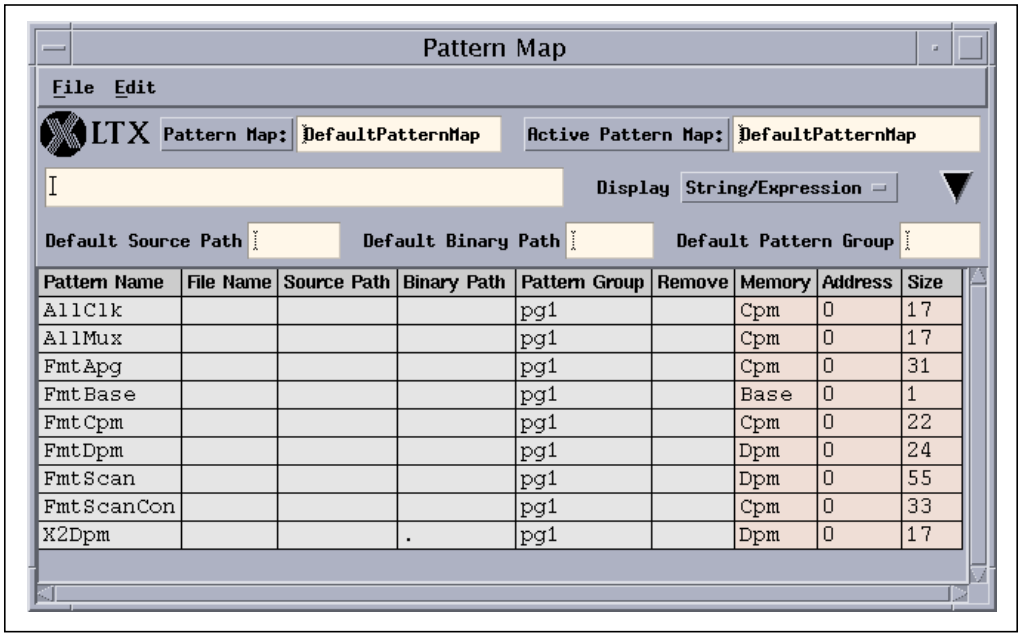


Figure 17.21: Pattern Map Editor

Pattern Map Editor—Main Window

The top row of controls below the menu bar has fields for selecting the displayed Pattern Map, and for setting and displaying the current active `PatternMap` object. The labeled buttons to the left of these text fields, Pattern Map and Active Pattern Map, enable you to select the desired objects from a pop-up like the one shown in Figure 17.4, rather than entering the name in the text field. Entering a name in the Pattern Map field of an existing Pattern Map and pressing Return will change the name of the Pattern Map.

The second row of controls includes the spreadsheet entry field, display mode, and move direction toggles. These controls are like those in other enVision tools; refer to Entering Expressions.

Typing text in the spreadsheet entry field enters it in the selected cell. If the move direction is set to left or down, the selected cell advances to the next cell in that direction. When moving horizontally, the selected cell wraps at the last column. When moving vertically, additional rows are added to the spreadsheet when the selected cell reaches the bottom row.

The third row of controls sets the default values for blank source path, binary path, and pattern group fields.

NOTE When making an entry in these fields, be sure to press Return when finished; otherwise, the text will remain, but the value does not get entered.

Pattern Map Editor Fields

Table 17.21 list the Pattern Map Editor fields.

Table 17.21: Pattern Map Editor Fields

Field	Meaning
Pattern Name	Name of `Pattern` object.
File Name	Name of pattern file (both source and binary) without path or extension. If left blank, defaults to pattern name. May be entered as an expression; refer to Entering Expressions.
Source Path	Path to pattern source file, without filename, relative to test program directory. If left blank, defaults to default source path of Pattern Map. May be entered as an expression; refer to Entering Expressions.
Binary Path	Path to pattern binary file, without filename, relative to test program directory. If left blank, defaults to default binary path of Pattern Map. May be entered as an expression; refer to Entering Expressions.
Pattern Group	Pattern Group for pattern. May be entered as an expression; refer to Entering Expressions.
Remove	If `True`, this pattern is not loaded into tester. If left blank, defaults to `False`. May be entered as an expression; refer to Entering Expressions.
Memory	Read-only field displays which tester memory the pattern is intended for or loaded into, or both.
Address	Read-only field for pattern address in tester memory.
Size	Read-only field for number of vectors in pattern.

Pattern Map Editor Menus

The Pattern Map Editor has the following menu items:

- File—Table 17.22
- Edit—Table 17.23

This editor has context-sensitive pop-up menus that are opened by using mouse button M3; refer to Context Sensitive Menus in Pattern Map Editor.

Table 17.22: Pattern Map Editor—File Menu

Menu	Action
Find...	Same as pressing Pattern Map: button at the top-left of Pattern Map panel. Opens an object finder dialog, like the one shown in Figure 17.4, to select a Pattern Map for display or to enter a name for creating a Pattern Map; refer to Create New Pattern Dialog.
Delete...	Removes displayed `PatternMap` object. To delete active Pattern Map, must first make a different Pattern Map active.
Clone...	Copies Pattern Map currently displayed.
Inherit From...	Attaches additional Pattern Maps to supply entries not included in displayed map.
Apply Mapping Now...	Triggers Pattern Map processing: examines active Pattern Map for new patterns or patterns now bound to different files and Pattern Groups, and create or modify those patterns to agree with the Pattern Map.
Close	Closes Pattern Map Editor window.

Table 17.23: Pattern Map Editor—Edit Menu

Menu	Action
Cut	Copies selected text to clipboard and deletes it from text area. In M3 pop-up, Cut acts on whole cell, even if text is selected in entry text field.
Copy	Copies selected text to clipboard. In M3 pop-up, Copy acts on whole cell, even if text is selected in the entry text field.
Paste	Enters contents of clipboard at cursor position into the focused text widget. In M3 pop-up, Paste replaces whole contents of cell, even if text is selected in the entry text field.
Add Files	Add groups of files via wild card filenames.
Insert Row	Insert a blank row before selected row.
Insert Row After	Insert a blank row after selected row.

Table 17.23: Pattern Map Editor—Edit Menu (Continued)

Menu	Action
Reverse Rows	Reverse order of selected group of rows.
Delete Row(s)	Delete one or more patterns from the Pattern Map.
Copy to this Map	Copies inherited entries to displayed Pattern Map, obscuring the corresponding entries in the inherited Pattern Map; refer to Using Inheritance.

Context Sensitive Menus in Pattern Map Editor

Clicking mouse button M3 in the header display area opens a context-sensitive menu for the selected row and column. Choices include items for changing the selected field and selected items from the menu bar menus. If a selection does not exist, choosing a command from the context sensitive menu, rather than from the Edit menu, will act upon or insert data at the clicked position.

The items in the context-sensitive pop-up for this editor are the same as for the Signal Header Editor; refer to Table 17.19.

Pattern Map Editor Tasks

You can edit or modify the `PatternMap` by:

- Creating a Pattern Map
- Adding Patterns
- Entering Expressions
- Using Inheritance
- Evaluating Pattern Maps

Creating a Pattern Map

To create a `PatternMap` object, select Find from the File menu of the Pattern Map window and enter the new name in the Selection field of the Find dialog and click OK.

Adding Patterns

If you are creating a test program or adding a large number of patterns, use the Add Files dialog from the Edit menu in the Pattern Map window. By adding a wildcard to the filenames, you can add an entire directory of patterns to an enVision program with a single command. Another way to reduce unnecessary data entry is to use Pattern Map default values. As with other enVision tools, the move arrow moves the cursor either horizontally or vertically after each field entry. Blank rows are ignored and not saved.

Entering Expressions

For most applications, strings are entered in the Pattern Map fields; however, you can enter enVision expressions in the pattern fields. For example, a filename can depend on an operator variable. To enter an expression, make sure the entered text is enclosed by parenthesis. Since expressions in the enVision language have no string operators, you should add parenthesis only when the field will be defined by a single variable. As with other enVision tools, you can view the field data as either an expression or expression value by selecting the String, Expression, or Value in the Display option menu. Changes in dependent expression variables (or any other type of change to a Pattern Map) do not affect the Pattern Map until enVision evaluates it, either at the beginning of the next test run or by manually selecting the Apply Mapping Now command; refer to Evaluating Pattern Maps.

Evaluating Pattern Maps

Changes to Pattern Maps do not immediately rebind the pattern files or Pattern Groups. Pattern Maps are evaluated at the end of the .eva loading, after the on-load flow, and before each test run. This evaluation can also be triggered manually by using the Apply Mapping Now command in the File menu of the Pattern Map window.

Using Inheritance

To inherit entries from another Pattern Map, enter the name of the map in the inheritance list, under Inherit From in the File menu. Inherited maps can themselves inherit from other Pattern Maps, which form an inheritance tree descended from the active Pattern Map.

Most programs will not require an elaborate structure, but the sample inheritance tree in Figure 17.22 is a multi-level tree that shows how inherited maps are processed by depth (deepest first) in their order in the inheritance list.

At each level in the tree, the inheriting map can override the entries in the maps from which it inherits by including a row with the same pattern name. When searching for a pattern, each map looks first at its own entries, and then at the entries of each inherited maps in the order they appear in the inheritance list, resulting in a depth-first search of the inheritance tree.

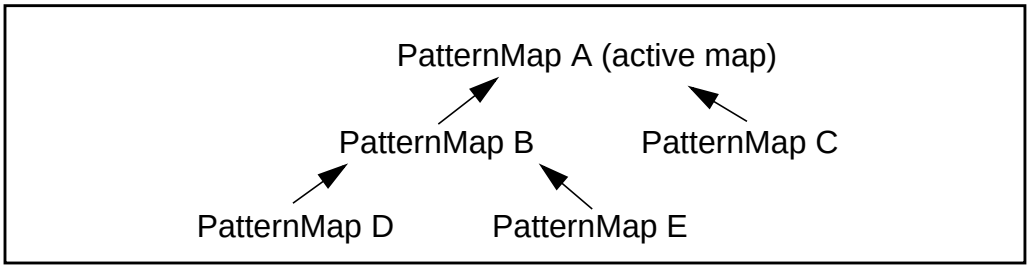


Figure 17.22: Sample Inheritance Tree

Pattern rows inherited from other maps appear under the heading Inherited Entries in the Pattern Map window. While inherited rows are displayed in the tool, they must be edited in the maps to which they belong. Inherited entries can be overridden by importing them into the displayed Pattern Map with the Copy to this Map command in either the Edit or context sensitive menu. Deleting a row that is overriding inherited entries will remove it from the local entries, but it will also reappear among the inherited entries. The only way to delete a pattern from an inherited map is to include it in the local map with the remove field set to True.

For more information about inheritance, refer to the *Pattern, Waveform, and Timing* chapter of the *enVision Digital Programming* manual.