USER'S GUIDE

WaveFormer Lite Manual (version 8.6c) copyright 1994-2002 SynaptiCAD

For latest product information and updates contact SynaptiCAD at:

web site: http://www.syncad.com

email: sales@syncad.com

phone: (540)953-3390

Contents

WaveFormer Lite Introduction

WaveFormer Lite is a special version of WaveFormer Pro that can generate VHDL and Verilog stimulus-based test benches for the Actel design software. WaveFormer Lite fits seamlessly into the Actel's design environment, automatically extracting signal information from your HDL design files, and producing HDL test bench code that can be used with any standard VHDL or Verilog simulator.

WaveFormer Lite Design Flow has instructions for generating VHDL and Verilog with the Actel design environment.

WaveFormer Lite Upgrade Options describes how to upgrade to the professional version of WaveFormer Pro. The professional version has features that help you create timing diagrams faster, analyze circuit timing, produce Data Book quality images, and translate waveform information from over 35 formats including most popular logic analyzers, pattern generators, and simulators.

WaveFormer Lite Design Flow

WaveFormer Lite generates VHDL and Verilog test benches from drawn waveforms. There are three basic steps for creating test benches using WaveFormer Lite and the Actel design software:

1. **Automatically Import Signal Information** directly from Actel following the instructions in the Actel software. Signal information includes signal names, type, direction, and size. If you later change the ports of the Model Under Test, you can force a signal extraction by clicking the **Extract MUT Ports into Diagram** button  located on the program level button bar.

   Note: Signals can also be manually added to the drawing window using the information in *Section 1.1 Adding and Deleting Signals* and *Section 1.2 Editing Signal Properties*.

2. **Draw Waveforms** to describe the test bench. *Section 1.3 Drawing Waveforms* describes how to use the mouse and the state buttons to draw waveforms. The rest of *Chapter 1: Signals and Waveforms* describes different waveform editing techniques. WaveFormer Lite also supports clocks that automatically draw themselves, and group busses that reflect the values of the member signals, these topics are covered in *Chapter 2: Clocks* and *Chapter 3: Virtual Buses.*

3. **(Optional) Add VHDL Libraries and Use Clauses** for VHDL export. These libraries or packages can be included using the *VHDL Libraries and Use Clauses* dialog. Select the **Options > VHDL Libraries and Use Clauses...** menu item to open this dialog. *Section 4.3 VHDL Libraries and Use Clauses* has more information about including libraries and use clauses.

4. **Export VHDL or Verilog Test Bench** by selecting **Export > Export Timing Diagram** the menu option and choosing the type of file to generate. You can generate a test bench with a top-level module that automatically hooks up the model under test to the test bench, or you can generate just a test bench model. Below is a detailed description of the two methods:

   To generate a Top-Level Model and a Test Bench model choose one of the "top-level" scripts from the save as type drop-down list box. The top-level module will instantiate the model under test and hook it up to the test bench. For this script to work the top-level module needs to be defined in the project. For WaveFormer Lite customers, the Actel Software should automatically set this option. For VeriLogger Pro and TestBencher Pro customers, you can set the top-level module by adding the files to the project and compiling it. Below is a list of top-level scripts:

   - VHDL Wait with Top Level TestBench (*.vhd)s

   - VHDL Transport with Top Level TestBench (*.vhd)s

   - Verilog with Top Level TestBench (*.v)s

   To generate a plain test bench model (which does not instantiate your model under test) then choose one of the VHDL or Verilog scripts. To simulate with the test bench model, you will need to write a top-level model that instantiates the test bench model and the model under test. This is the method used by WaveFormer Pro customers. Below is a list of VHDL and Verilog test bench generation scripts:

   - VHDL Wait (*.vhd)s

- VHDL Transport (*.vhd)s

- Verilog

Note: The Report Window automatically displays the generated code so that you can quickly inspect the test bench.

Note: If you added extra signals to the test bench and do not want to export those signals, then double click on the signal's names to open the *Signals Properties* dialog and uncheck the **Export** check box.

WaveFormer Lite ships with the full WaveFormer Pro documentation so that you can see the types of features that are included in the full version. WaveFormer Lite partially supports the features described in WaveFormer Pro's *Chapter 1: Signals and Waveforms, Chapter 2: Clocks, Chapter 3: Group, Simulated and Virtual Buses*, and *Chapter 4: Stimulus Generation and Waveform Import*. To learn to draw waveforms you can do the first 7 sections of Basic Drawing and Timing Analysis tutorial. For complete documentation and tutorials, please see the WaveFormer Pro documentation.

WaveFormer Lite Upgrade Options

WaveFormer Pro is a professional version of WaveFormer Lite that you can purchase separately from SynaptiCAD to use with your Actel design software. WaveFormer Pro has many features that help you create timing diagrams faster, analyze circuit timing, produce Data Book quality images, and translate waveform information from over 35 formats including most popular logic analyzers, pattern generators, and simulators. Here are just a few of the major features included in WaveFormer Pro:

An **Interactive Simulator** lets you describe waveforms using Boolean and registered logic equations, and then generates the waveform so that you do not have to draw it by hand.

Delay, **Setup**, and **Hold** parameters allow you to document the causal relationships between the waveform edges making the timing diagrams more readable.

Logic Analyzer and **Pattern Generator** support lets you move test data back and forth between the simulation and hardware prototyping environment. You can capture an existing hardware interface using a logic analyzer and use WaveFormer Pro to translate the data into a VHDL or Verilog testbench. And WaveFormer Pro can take your simulation output waveforms and convert them into stimulus files for the pattern generator.

Professional Documentation features like vector and bitmap image generation enables you to publish timing diagrams in Data Books, on web sites, or in the most incredible design review document you have ever created.

In addition to WaveFormer Pro, SynaptiCAD also offers VeriLogger Pro and TestBencher Pro product upgrades. VeriLogger Pro is a new type of Verilog simulation environment that combines all the features of a traditional Verilog simulator with the most powerful graphical test vector generator on the planet. Model testing is so fast in VeriLogger Pro that you can perform true bottom-up testing of every model in your design, a critical step often skipped in the race to market.

TestBencher Pro generates VHDL, Verilog, System C, and Open Vera bus-functional model test benches that can be used to test large systems. In comparison, WaveFormer Lite generates simple stimulus based test benches. TestBencher Pro automates the most tedious aspects of test bench development, allowing you to focus on the design and operation of the test bench. This is accomplished by representing each bus transaction graphically and then automatically generating the code for each transaction. TestBencher makes use of the powerful features of the language that is being generated and the engineer does not have to hand-code each transaction. When hand coding, the designer would have to take the time to deal with the specifics of the design (port information, monitoring system response, etc.) as well as common programming errors (race conditions, minor logic errors, and code design problems). This removes a considerable amount of time from the test bench design process because TestBencher manages the low-level details and automatically generates a valid test bench.

Upgrade WaveFormer Lite

After purchasing a license for another SynaptiCAD product, you will need to set up a *Design Preferences* option so that Libero will automatically run that product instead of WaveFormer Lite:

- Install and run your new SynaptiCAD product.

- Select the **Options > Design Preferences** menu option to open the *Design Preferences* dialog.
- In the **Substitute for WaveFormer Lite** dropdown menu, select the product you purchased.
- Click **OK** to close the dialog. This will save a special option in the **.ini** file that will automatically launch your other program instead of WaveFormer Lite when you use Libero.
- Run Libero and double-click on the WaveFormer Lite icon in the project window. This will launch your new SynaptiCAD product.

Chapter 1: Signals and Waveforms

This chapter introduces the basic techniques for manually adding signals and drawing waveforms.

1.1 Adding and Deleting Signals

The Actel software should be automatically importing the relevant information into WaveFormer Lite. However, signals can also be manually added using the techniques in this section and in the next section on Editing Signal Properties.

To add a new signal:

- Click on the **Add Signal** button.

To delete a signal or multiple signals:

- Select the signal names by clicking on them. A selected signal's name is highlighted.
- Then press the delete key on the keyboard.

Note: When a signal is deleted, any parameters or text connected to it are also deleted. If you make a mistake deleting, you may undo the delete by choosing the **Edit > Undo** or **Edit > Undo Delete** menu item.

1.2 Editing Signal Properties

To edit signal properties (name, Boolean equation, export, direction, and type):

- Double-click on the signal name to open the Signal Properties dialog box.

Some properties that are useful for WaveFormer Lite include:

- **Name:** There will be a valid default name already in the edit box. Name must be at least one character long (no spaces allowed). Signal names beginning with "**$$**" are reserved for internal use. Signal names should not begin with a digit or contain mathematical operators. Signal names should not be the same as common Boolean operators.
- The **Export Signal** check box determines whether or not a signal will be exported to stimulus and test bench files.
- **Direction:** Used for exporting VHDL and Verilog code.
- **VHDL** and **Verilog:** Determine the HDL type of the signal (e.g., std_logic, integer, wire,...). The user can also enter a new type into the drop-down edit box. The VHDL and Verilog history lists can be rearranged by editing the **.ini** file (**syncad.ini**) located in the Windows directory. Be sure that WaveFormer Lite is not running when you edit the **.ini** file
- **Bus MSB** and **LSB**: Determine the bit size of the signal. A single bit signal is (0,0). This is used by the Interactive HDL Simulator to determine the result of multi-bit operations. This is also used during the export of VHDL or Verilog stimulus files and test benches to determine how to initialize the signal in the HDL model.
- **Radix**: Determines the base in which signal values are displayed.

1.3 Drawing Waveforms

The timing diagram editor is always in drawing mode.

To draw the waveform of a signal:

- Place the mouse cursor inside the *Diagram* window at the same vertical row as the signal name.
- Click the left mouse button. This draws a waveform from the end of the signal to the mouse cursor. The red state button on the button bar determines the type of waveform drawn (see below for more information on the state buttons). The cursor shape also mirrors the red state button.
- Move the mouse to the right and click again to draw another segment.

The state buttons are the buttons with the waveforms drawn on their face: HIGH, LOW, TRIstate, VALid, INValid, WHI weak high, and WLO weak low. When a state button is activated, it is pushed in and colored red. The active state will be the type of waveform that is drawn next. To activate a state button, click on it.

The state buttons automatically toggle between the two most recently activated states. The state with the small red "T" above the name is the toggle state. The initial activated state is HIGH and the initial toggle state is LOW.

When you draw signals using the mouse, the signal edges are automatically aligned to the closest edge grid time. The edge grid can be controlled from the **Options > Grid Settings** menu item.

1.4 Editing Waveforms with the mouse

There are five mouse-based editing techniques used to modify existing waveforms. The first two techniques act on signal transitions:

1) Drag-and-Drop a signal transition:
   - Click on a signal transition and drag it to the desired location.
   - Note: If you try to drag a transition past a second transition you will end up pushing the second transition unless it is fixed by a delay parameter or it has been locked in the *Edge Properties* dialog box. When several transitions are squeezed together they look much thicker than a normal transition.
2) Drag-and-Drop groups of signal transitions with **<1>** and **<2>**:
   - Hold down the **<1>** or **<2>** key while dragging a transition. This causes all the transitions to the right or left (respectively) to move with the selected edge.
   - Note: Holding down both keys causes all the edges on the signal to move.
   - Select multiple edges by holding down **<Ctrl>** while clicking on the edges. Then release **<Ctrl>** and all the selected edges can be dragged.

The other three techniques all act on signal segments (the waveforms between any two consecutive signal transitions). To select the segment to operate on, click on it. A selected segment will have a highlighted box drawn around it. If you try to select a narrow segment and one of the transitions gets selected, widen the segment by clicking the Zoom In button, which is located on the right hand corner of the button bar.

To insert, change, or delete a segment:

3) Click-and-Drag to insert a segment into a waveform:
   - Click and hold in the middle of a wide segment.
   - Drag left or right and then release the mouse button. A new segment will be added in the middle of the original segment.
4) Change a segment's graphical state by selecting it and then pressing a state button:
   - Click in the middle of the segment to select it.
   - Click on a state button to apply that graphical state to the segment. State buttons are the buttons with the waveforms drawn on them.
   - Note: If you change the level of the segment to the same level as an adjacent section, the transition between them will turn red. This transition can then be deleted if necessary.
5) Delete a segment by selecting it and then pressing the Delete key:

- Click in the middle of the segment to select it.
- Press the **delete** key on the keyboard.

These techniques will only work on signals that are drawn. They will not work on generated signals like clocks and simulated (blue) signals that are covered in later chapters.

Chapter 2: Clocks

Clocks are special repetitive signals that draw themselves based on their attributes: period, frequency, duty cycle, edge jitter, offset, and other parameters. Clocks can be related to other clocks by using the Reference Clock Property or by using formulas that reference another clock's attributes like period, offset, and jitter.

Clocks are a specialized form of a signal, so they are selected, moved, deleted and hidden just like regular signals. However, clock edges are fixed and cannot be pushed by a delay parameter or dragged using the left mouse button.

The following functions cover adding and editing clocks.

2.1 Adding Clocks

To add a new clock:

- Click on the **Add Clock** button to open the *Edit Clock Parameters* dialog.
- Enter the values for the new clock.
- Click the **OK** button to close the dialog

2.2 Clock Parameter Dialog

To edit the properties of an existing clock:

- Double-click on a segment in the clock waveform to open the *Edit Clock Parameters* dialog.

OR

- Double-click on the clock name to open up the *Signal Properties* dialog. Click the **Clock Properties** button, in the center of the dialog, to open the *Edit Clock Parameters* dialog.

Chapter 3: Virtual Buses

WaveFormer Lite supports Virtual Buses, which are normal signals that are drawn with Valid, Invalid, and TriState states and use virtual state information to represent bus values. Virtual buses can be added using the Add Signal button or using the Add Bus button and selecting a bus type of Virtual Bus. The state information is added using the HEX state button. A virtual bus doesn't have member signals.

3.1 Adding and Editing Virtual Buses

To add a virtual bus:

- Click the **Add Signal** button to add a new signal.
- Sketch the bus waveform.
- Buses drawn with valid, invalid, and tristate states look the best. To draw a bus with only consecutive valid states click twice on the Valid state button so it stays active (state buttons will not toggle). The Valid button should be red and have a red T at the top of the button.

To set the width of a virtual bus:

- Double-click on the name of the virtual bus to open the *Signal Properties* dialog.
- Enter the MSB and LSB in the edit boxes at the bottom of the dialog, OR edit the signal name to include the MSB and LSB inside brackets (for example: SIG2[31:0] is a 32-bit bus).

Note: The MSB, LSB, and brackets are not part of the signal name. When using the signal in equations, use only the original signal name (e.g., "SIG1 and SIG2[31:0]" is incorrect and will produce an error).

To edit the segment values of a virtual bus:

- Double-click on a segment to open the *Edit Bus State* dialog.
- Type the bus segment data into the **Virtual** field.
- Use the **<Alt>-N** and **<Alt>-P** keys (or the **Next** and **Prev** buttons) to move between the different segments on the virtual bus.
- Click **OK** to close the dialog.

In the next section you will learn how to use the Virtual field to export user defined type information for VHDL files.

3.2 Virtual State Information

There are seven standard graphical states (high, low, tristate, valid, invalid, weak high, and weak low) which cover most circuit applications. However, some simulation languages require more complex state types such as integers and enumerated data types. To support these types, SynaptiCAD products allow virtual state information to be attached to signal segments. Any arbitrary text string can be used as a virtual state (e.g., A0C, 5 + 3, blue level, and 24 are all valid virtual states). Note that spaces are valid characters in a virtual state string.

Virtual state information is displayed in the center of a signal segment (the text is hidden if the segment is shorter than the text of the virtual state). Since a bus displays the states of its member signals in the same manner, virtual states are used to create virtual buses.

To add virtual state information to an existing signal:

- Click on a segment in the signal. This selects the segment and draws a selection box around it.
- Click on the HEX state button on the button bar. This selects the button and opens the *Edit Bus State* dialog box.
- Type any string into the **Virtual** field to set the virtual state information which will be displayed in the center of the segment.
- Use the **<Alt>-N** and **<Alt>-P** keys (or the **Next** and **Prev** buttons) to move between the different segments on the virtual bus.
- Click **OK** to close the dialog box.

Note: The virtual state information associated with a transition is supplemental to the standard state assigned to a signal transition. The standard state (High, Low, Tristate, etc.) controls the graphical appearance of the segment and the virtual state sets the text in the segment. WaveFormer Pro does not attempt to interpret virtual state information. It only displays and saves the information. Waveperl scripts have access to virtual state information, and it is the script writer's job to define the meaning of virtual state information within a particular script. The VHDL and Verilog scripts use this information.

Chapter 4: Stimulus Generation and Waveform Import

WaveFormer Lite can export waveform data as a stimulus file to many simulators including VHDL, Verilog, Viewlogic, Mentor, Spice, DesignWorks, and more. These programs can also import data from simulators and PC-based test equipment like the Pod-A-Lyzer and HP's logic analyzers.

Stimulus Generation (exporting waveform timing information):

The Perl source code that performs the import/export function is shipped with WaveFormer Pro, VeriLogger Pro and TestBencher Pro and can be modified to perform customized actions. SynaptiCAD's ftp site (ftp://www.syncad.com) maintains a repository for scripts written by SynaptiCAD and users. Two advanced sections deal with these features:

4.1 Export Instructions for Stimulus Generation

To create a stimulus file:

- Select the **Export > Export Timing Diagrams As** menu option to open the *Export* dialog. This is a special version of the *Save As* dialog which remembers the file type of the last file exported.

- Choose the export format using the **Save as Type** list box in the lower left corner of the *Export* dialog box.

- Pick a file name and click the **OK** button. WaveFormer Lite will produce a file with the timing data in that format.

Note: Once WaveFormer successfully creates the file it will display it in the *Report* window so that you can quickly verify that the file is correct.

To exclude a signal from being exported, use one of the following methods:

A) Double-click on the signal name to open the *Signal Properties* dialog. Uncheck the **Export Signal** check box.

B) Select one or more signals. Select the **Export > Exclude Selected** from **Export** menu option. This will uncheck the **Export Signal** check boxes in all of the selected signals.

4.2 Exporting VHDL and Verilog Stimulus

WaveFormer Lite contains several scripts for exporting stimulus vectors to VHDL and Verilog. All the scripts produce a complete entity-architecture or module test bench that can be directly compiled and linked into an external simulator. The differences between the following scripts are:

- VHDL transport: Exports signal transitions as simple VHDL transfer statements.

- VHDL wait: Exports signals using wait statements.

- Verilog: Exports signal transitions as signal assignment statements.

In order for a signal's waveform data to be exported by the VHDL or Verilog scripts, the signal must be declared as an export signal with a direction of out (shared output, output, or persistent output). Also the signal type must be set to match the model which you will be testing. When signals are first added to a project they are defined as export, with a direction of out, and a signal type of std_logic or wire.

To set the export property, direction, and type of a signal:

- Double-click on a signal name to open the *Signal Properties* dialog.

- Check the **Export Signal** check box to indicate that this signal should be included in stimulus generation files.

- Use the **Direction** drop down list box to choose whether the signal is an output or input signal.

- Any of the **out** directions indicates that the signal is an output of the test bench and an input to the circuit under test. Waveform information will be generated for the output signals.

- Any of the **In** directions indicate that the signal is a response expected from the model under test. Only a port statement will be generated for the input signals (no waveform data will be generated).

  Note: The stimulus scripts do not differentiate between the different types of out and in direction types. The direction type is only used internally by TestBencher Pro when generating interactive test benches.

- Use either the **VHDL** or **Verilog** type drop-down list boxes to set the type of the signal. The **VHDL** list box is also an edit box in which you can enter user-defined types.

- Use the **Next** or **Prev** buttons to investigate other signals or click **OK** to close the dialog.

How different information is exported to VHDL and Verilog:

- Clock signals are represented as repeating processes.

- Signals are exported with VHDL and Verilog values that most closely match the graphical state. You can change the state-value mapping by editing the VHDL or Verilog perl script.

- Virtual Buses are exported as vector signals.

- Virtual Buses with Virtual State Information: If a signal is drawn using Valid regions with Virtual state information then during export the Virtual state is written to the export file instead of the valid state value. This only works for valid segments.

How to Add Virtual State Information to a Signal:

The virtual state Information is used to define values for signals whose types do not easily match the graphical state. For example a signal with a type of integer or a VHDL user defined enumerated data type of MYCOLOR cannot be represented with the seven drawing states available in the program.

To add virtual state information:

- Double-click on a valid segment to open the *Edit Bus State* dialog.
- Use the **Virtual** edit box to enter the state of the selected signal.
- If the state begins with **0b** or **0x**, then it will be interpreted as a language independent binary or hexadecimal value. The size of the bus is determined from the *Signal Properties* dialog Bus MSB and LSB.
- If the state does not begin with **0b** or **0x** then the string is used without any text processing.
- Note: The virtual state will only override a valid segment's value. The other graphical states will override any virtual state information
- Use the **Next** or **Prev** buttons to edit other segments or click **OK** to close the dialog.

For more detailed information, please see the WaveFormer Pro documentation.

4.3 VHDL Libraries and Use Clauses

The *VHDL Libraries and Use Clauses to Include* dialog allows you to control the libraries and use clauses used when exporting diagrams to VHDL.

To open the *VHDL Libraries and Use Clauses* dialog:

- Select **Options > VHDL Libraries and Use Clauses...** from the main menu.

Changes made in this dialog will be applied to all diagrams that are exported in VHDL, and saved in the WaveFormer configuration.

The **View** dropdown allows you to change between selecting the VHDL libraries and the use clauses to include in your diagram.

To add a new use clause or library include:

- Select **Use Clauses** or **VHDL Libraries** from the **View** dropdown.
- Double-click on the first empty line in the list window.

Note: If the 'USE' is omitted from a use clause, or the 'LIBRARY' from a library inclusion, WaveFormer will automatically add 'USE' or 'LIBRARY' before the clause in the source file for the diagram. WaveFormer will also automatically add semicolons to the end of library includes and use clauses.

To edit a use clause or library include:

- Double-click on the clause or library in the list window.

To delete a use clause or library include:

- Click on the clause or library in the list window.
- Click the **Delete** button.

The **OK** button saves the current settings and closes the dialog. The **Cancel** button closes the dialog without saving the new settings.

Conclusion:

This manual describes the basic features available in the WaveFormer Lite Product. To learn more about WaveFormer Pro and SynaptiCAD's products, please see SynaptiCAD's website at www.syncad.com.

To purchase WaveFormer Pro or any other SynaptiCAD product, contact SynaptiCAD at:

SynaptiCAD Inc.
520 Prices Fork Rd. #C4
Blacksburg VA 24060
Web Site: www.syncad.com
Email: sales@syncad.com
Phone sales: 1-800-804-7073
Phone tech support: 1-(540)-953-3390
fax: 1-(540)-953-3078