

***Actel® Libero™ Integrated Design Environment v2.3
Mixed-Mode Design Tutorial***

Table of Contents

Design Flow in Libero™ IDE v2.3

Step 1 - Design Creation	3
Step 2 - Design Verification	3
Step 3 - Design Synthesis/EDIF Generation	3
Step 4 - Design Implementation	3
Step 5 – Timing Simulation	3
Step 6 – Device Programming	3

Mixed Mode Design Flow in Libero™ IDE v2.3

Mixed-Flow example	4
Step 1 - Creating a project in Libero IDE	5
Step 2 - Creating ACTgen components	5
Step 3 - Entering the Control block HDL Description.	7
Step 4 - Creating Schematic Symbols	8
Step 5 - Creating the 32 bit output register schematic	9
Step 6 - Creating the top-level Schematic	13
Step 7 - Creating the testbench	14
Step 8 - Pre-Synthesis simulation of top	16
Step 9 - Synthesizing with Synplicity	18
Step 10 - Post-Synthesis simulation of top	19
Step 11 - Design Implementation - Place and Route	20
Step 12 - Timing simulation with back-annotated timing	21

1

Design Flow in Libero™ IDE v2.3

This chapter describes the basic design flow for creating designs using the Libero IDE software.

Step 1 - Design Creation

Your first step is to plan your design and enter it as either HDL (VHDL or Verilog), structural schematic, or mixed-mode (schematic and RTL).

Step 2 - Design Verification

After you have defined your design, you must verify that it functions the way you intended. After creating a test bench using WaveFormer Lite use the ModelSim for Actel VHDL or Verilog simulator to perform functional simulation on your schematic or HDL design.

Step 3 - Design Synthesis/EDIF Generation

A design must be synthesized if the design was created using VHDL or Verilog. Use Synplify or Synplify Lite from Synplicity to generate your EDIF netlist. You can re-verify your design "post-synthesis" using the VHDL or Verilog ModelSim for Actel simulator used in step 2. While all RTL code must be synthesized, pure schematic designs are automatically "netlisted" out via the Libero IDE tools to create a structural VHDL or structural Verilog netlist.

Step 4 - Design Implementation

After you have functionally verified that your design works, the next step is to implement the design using the Actel Designer software. The Designer software automatically places and routes the design and returns timing information. Use the tools that come with Designer to further optimize your design. Use Timer to perform static timing analysis on your design, ChipEdit to customize your I/O macro placement, PinEdit for I/O customization, SmartPower for power analysis, and Netlist Viewer to view your netlist.

Step 5 - Timing Simulation

After you are done with design Implementation, you can verify that your design meets timing specifications. After creating a test bench using WaveFormer Lite, use the ModelSim for Actel VHDL or Verilog simulator to perform timing simulation.

Step 6 - Device Programming

Once you have completed your design, and you are satisfied with the timing simulation, create your programming file. Depending upon your device family, you need to generate a Fuse, Bitstream, or STAPL programming file.

2

Mixed Mode Design Flow in Libero™ IDE v2.3

The example below demonstrates a mixed (schematic / HDL) design flow in Libero IDE. You should review each of the steps to become familiar with the mixed flows and create the project in the Libero IDE design software.

Design Description

The design in Figure 1 consists of a 32 bit serial-in parallel-out (SIPO) shift register, a control block and an output register. The SIPO was generated using ACTgen, the control block was described in VHDL (or Verilog) and the register is a hierarchical schematic using components created by ACTgen. The control block enables the output register after 32 bits of data have been shifted in.

Using Actel's Libero IDE tool suite, you will complete the following:

- Create the 32 bit shift register and 8 bit register with ACTgen
- Use the HDL editor to describe the control block
- Use Libero's schematic editor to enter the 32 bit register schematic using the 8 bit register generated by ACTgen
- Create a top level schematic to tie the blocks together
- Functionally simulate using the ModelSim for Actel HDL simulator
- Synthesize the design with Synplicity
- Simulate the post-synthesis netlist

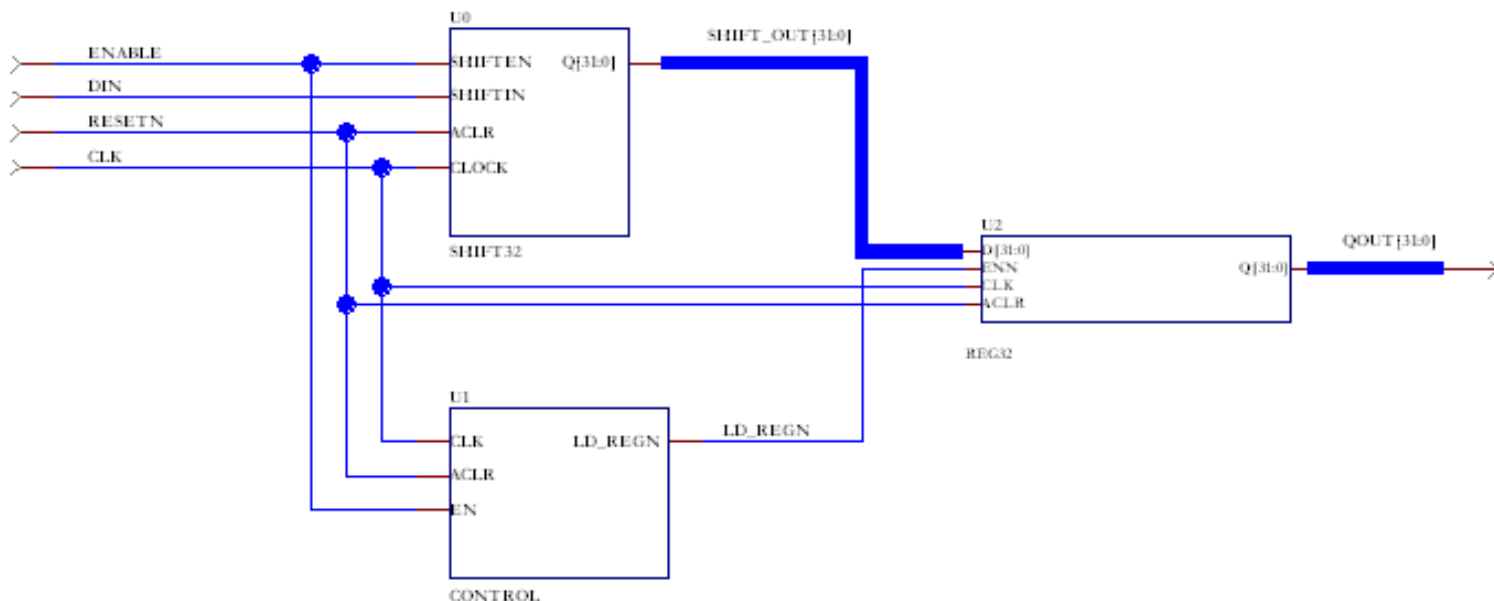


Figure 1 - Libero IDE mixed flow schematic

Step 1 - Creating a Project in Libero IDE

To create the shifter project:

1. Start Libero IDE by double-clicking on the Actel Libero IDE icon on the desktop of your PC.
2. From the File menu click *New*. The New Project dialog box appears, as shown in Figure 2. Enter the following in the New Project dialog box:
 - **Family:** Select 54SXA from the Family drop-down list box
 - **HDL:** Select the HDL (VHDL or Verilog)
 - **Location:** Specify a location in the Location field, or select a location by clicking Browse and browsing the project directory
 - **Project Name:** Enter shifter in the Project Name field
3. Click **OK**. The project “shifter” is created and opened

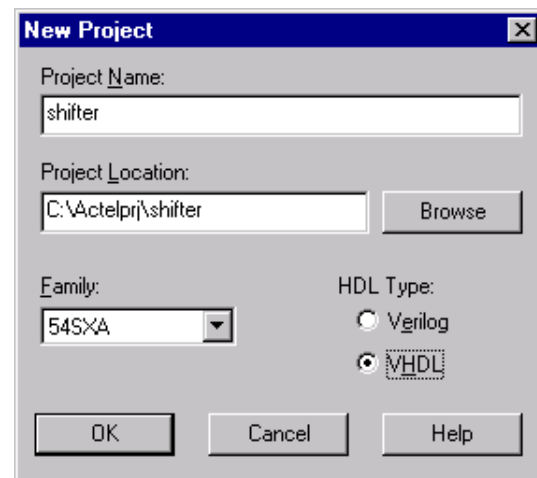


Figure 2 - Libero's New Project dialog box

Step 2 - Creating ACTgen components

In this step you will create a 32 bit shift register and an 8 bit storage register with ACTgen.

To create the shift register with ACTgen:

1. From within Libero IDE, double-click on *ACTgen* in the process window. The New File dialog box is displayed (Figure 3).
2. Select ACTgen macro under File Type.
3. Specify *shift32* as the name. Click **OK**. ACTgen opens in a separate window.
4. From the left side of the ACTgen GUI select **Register**, then choose the Shift Register tab, and specify the following options (Figure 4):
 - Variations: Serial-In/Parallel-Out
 - Width: 32
 - Sequential Type: Default
 - Async Clear: Active Low
 - Clock: Rising
 - Shift Enable: Active High

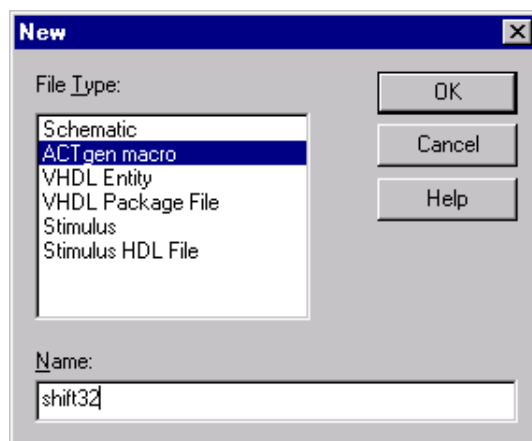


Figure 3 - Libero's New File dialog box

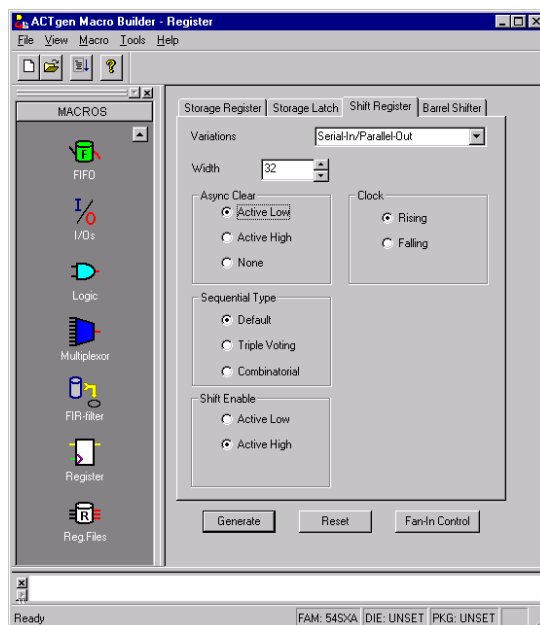


Figure 4 - ACTgen GUI

5. Select **Generate** from the File Menu and set the following options in the Generate Macro dialog box (Figure 5):

- Netlist / CAE Formats: VHDL (or Verilog)
- File Name: Accept default setting (shift32.gen)
- Other Options: (Leave as default)

6. Click **SAVE** to generate the 32 bit shift register.

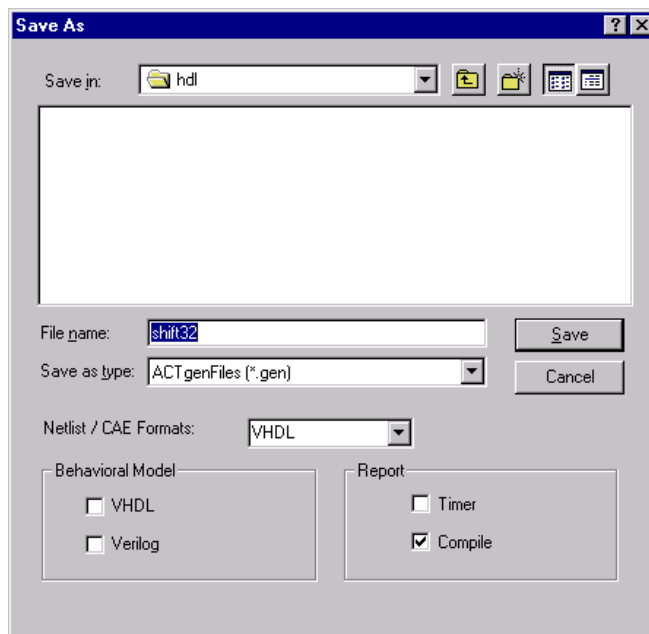


Figure 5 - ACTgen Generate Macro dialog box

ACTgen will generate a 32 bit shift register described in structural VHDL (or structural Verilog) named shift32.vhd (or shift32.v).

To create the 8 bit storage register with ACTgen:

1. From the ACTgen GUI, create another register named reg8 by selecting the Storage Register tab and specifying the following options (Figure 6):

- Width: 8
- Sequential Type: Default
- Async Clear: Active Low
- Load Enable: Active Low
- Clock: Rising

2. Click **Generate** from the File Menu and then **SAVE** the VHDL or Verilog file “reg8” with options from Steps 5 and 6 above. ACTgen will generate an 8 bit storage register described in structural VHDL or Verilog named reg8.vhd or reg8.v, respectively.

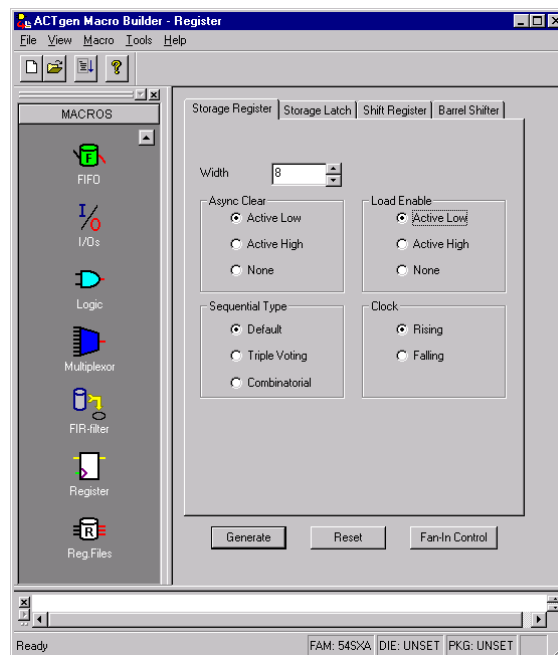


Figure 6 - Creating an 8 bit storage register with ACTgen

Close ACTgen by clicking **Exit** in the File menu.

The files generated by ACTgen, reg8.vhd (or reg8.v) and shift32.vhd (or shift32.v) will be visible on the Design Hierarchy tab and on the File Manager tab in the Libero IDE (Figure 7).

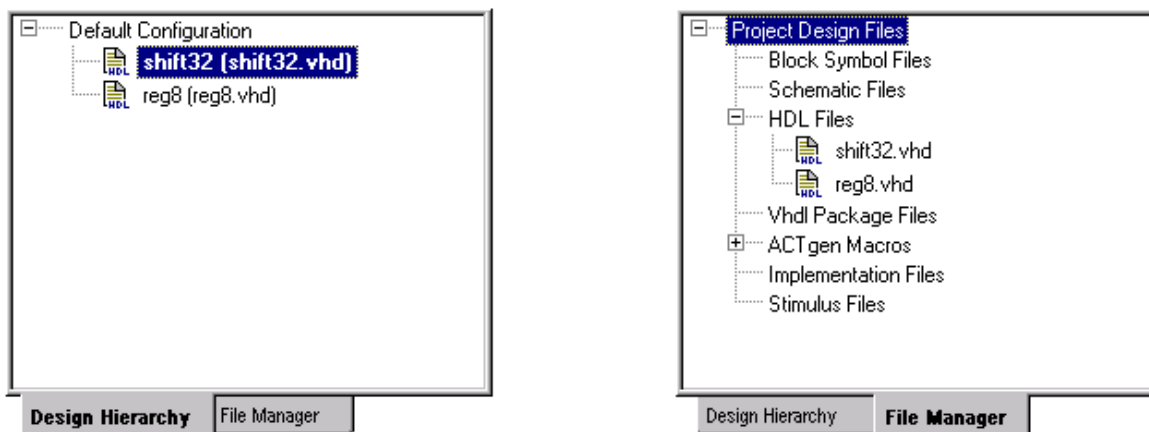


Figure 7 - ACTgen macros in Libero IDE

Step 3 - Entering the Control block HDL Description

Use the Libero HDL Editor enter the VHDL or Verilog description of the control block.

To create the control block HDL description:

1. From the File menu in the Libero IDE, click **New**. Select VHDL Entity (or Verilog Module) under File Type in the New dialog box and specify the name as **control** (Figure 8).
2. Click **OK**. The HDL Editor will open.
3. Type in the VHDL or Verilog file for the control block shown on the next page (Figure 9) or just cut and paste it from this document, if you have it opened in an electronic form.
4. Save the file by clicking **Save** from the File menu. The control block will appear on the Design Hierarchy and File Manager tabs .
5. Check the HDL file for errors by selecting “CheckHDL” from the File Manager tab.

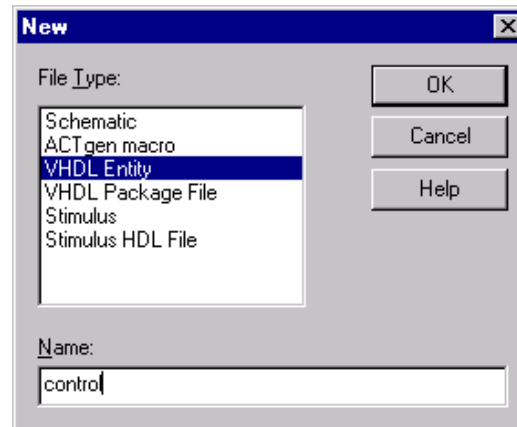


Figure 8 - Opening the HDL editor

```

-- control
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity control is
port (clk, aclr, en: in std_logic;
      LD_REGn: out std_logic);
end control;

architecture rtl of control is

    signal count: std_logic_vector(4 downto 0);

begin

    process (aclr, clk)
    begin
        if (aclr = '0') then
            count <= (others => '0');
            LD_REGn <= '1';
        elsif (clk 'event and clk = '1') then
            if (en = '1') then
                count <= count + 1;
            end if;
            if (count = "11111") then
                LD_REGn <= '0';
            else
                LD_REGn <= '1';
            end if;
        end if;

    end process;
end rtl;

```

```

/* control block for mixed mode design
   in Verilog
*/

module control (clk, aclr, en, LD_REGn);

    input clk, aclr, en ;
    output LD_REGn ;

    reg [4:0] count ;
    reg LD_REGn ;

    always@(posedge clk or negedge aclr)
    begin
        if (aclr == 1'b0)
        begin
            count <= 5'b0 ;
            LD_REGn <= 1'b1 ;
        end
        else
        begin
            if (en == 1'b1)
                count <= count + 1 ;
            if (count == 5'b11111)
                LD_REGn <= 1'b0 ;
            else
                LD_REGn <= 1'b1 ;
            end
        end
    end

endmodule

```

Figure 9– HDL descriptions for control block (VHDL and Verilog)

Step 4 - Creating Schematic Symbols

Before creating the schematic for the 32 bit register or the top level schematic, you must create ViewDraw symbols for the components you created in the previous steps.

To create schematic symbols:

1. In the Design Hierarchy tab, select *shift32*, then right mouse click and select **Create Symbol** (Figure 10). A ViewDraw symbol will be created for the shift register.

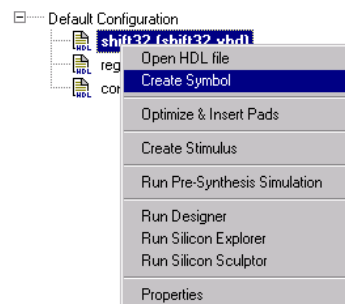


Figure 10 Creating ViewDraw symbols

2. Repeat the procedure above to create symbols for the *reg8* and *control* blocks.
3. The ViewDraw symbol will be visible under Block Symbol Files on the File Manager tab (Figure 11).

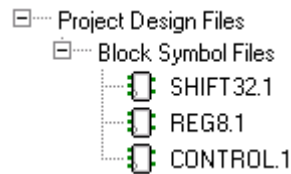


Figure 11- ViewDraw symbols on Libero IDE File Manager tab

Step 5 - Creating the 32 bit output register schematic

The 32 bit register will be crafted as a hierarchical schematic using the reg8 macro that was created using ACTgen. The register has an active low asynchronous reset and an active low synchronous enable. The completed schematic is shown in Figure 12.

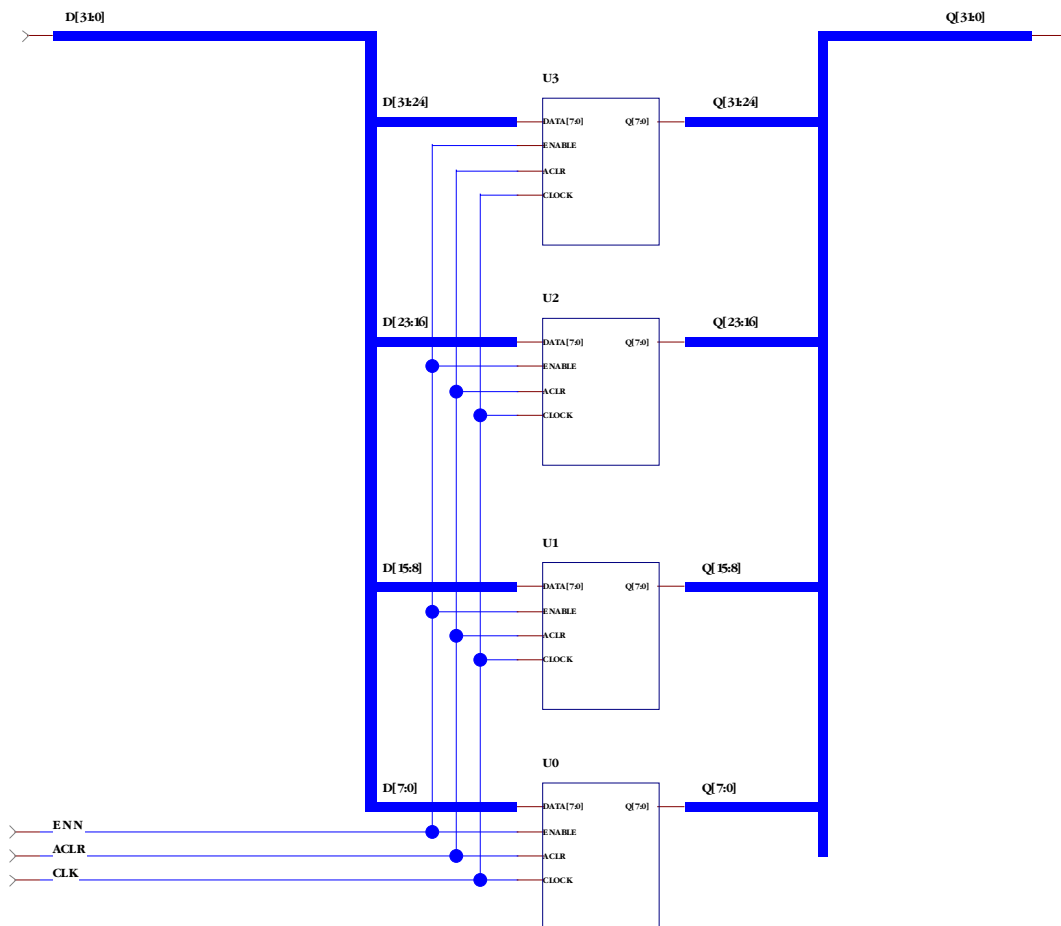


Figure 12 - Completed reg32 schematic

To create the schematic for the 32 bit register:

1. From the File menu, click *New* again or double-click on *ViewDraw* in the process window. The New File dialog box is displayed (Figure 13).
2. Select Schematic in the New dialog box.
3. Specify *reg32* as the name. Click **OK**. ViewDraw opens in a separate window.

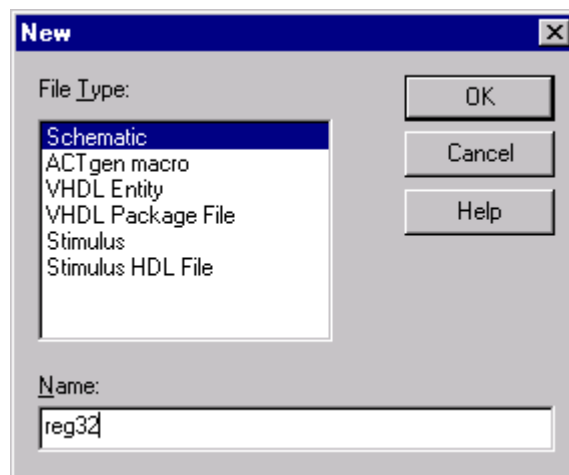


Figure 13 - Libero's New File dialog box

4. From ViewDraw, choose *Add* from the Component menu. The Add Component dialog box appears (Figure 14).

In the Add Component dialog box, highlight *C:\Actelprj\shifter\viewdraw* in the Directory window and *reg8.1* in the Symbol window (Figure 14).

Drag the reg8 symbol from the Symbol preview window to the schematic. Add 3 more copies of reg8 to the schematic then close the Add Component window.

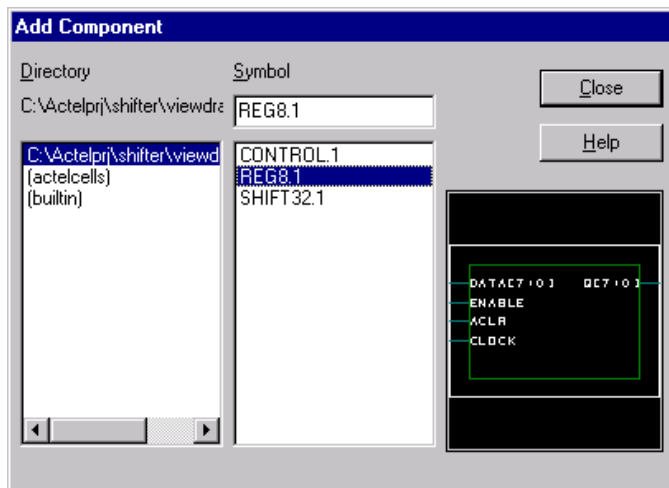


Figure 14 - ViewDraw Add Component dialog box

5. Make the necessary wire and bus connections to complete the schematic as shown in Figure 12.

To add a bus to the schematic:



- a. Click **Bus** (**Add** menu) or click the Bus toolbar button.
- b. Position the cursor at the originating point for the bus and press the left mouse button.
- c. With the left mouse button depressed, drag the mouse to draw the bus. Add a bend in the bus by clicking the right mouse button.
- d. Release the left mouse button at the endpoint. Add additional busses as needed.

To add a wire to the schematic:



- Click **Net** (Add menu) or click the Net toolbar button.
- Position the cursor at the originating point for the net and press the left mouse button.
- With the left mouse button depressed, drag the mouse to draw the net. Add a bend in the net by clicking the right mouse button (with the left button depressed).
- Release the left mouse button at the endpoint. Add additional nets as needed.

Label the nets and busses as indicated in the schematic handout by double clicking on the net or bus and entering the name on the Name tab in the Net Properties dialog box (Figure 15).

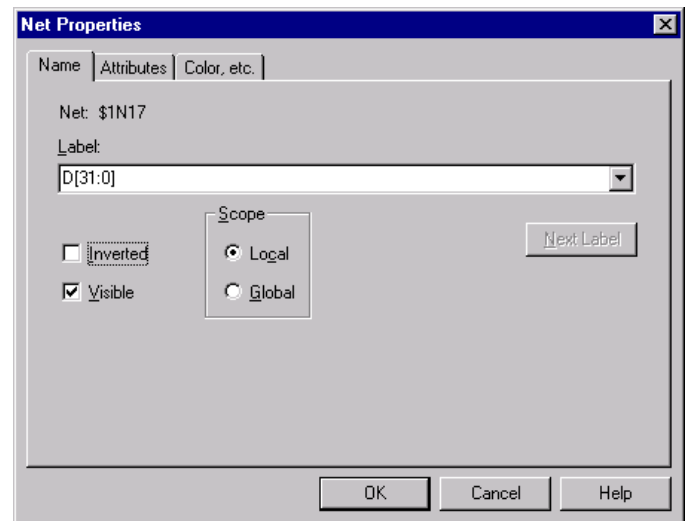


Figure 15 - Adding a bus name

6. Add the missing hierarchical connectors as follows:

- Click Component (Add menu) or click the Component icon on the ViewDraw toolbar.
- Select builtin from the library list and select in.1 from the symbol list (Figure 16).
- Drag the Hierarchical input connector from the Symbol Preview window into the schematic and position as indicated in the lab handout. Make sure the connector connects to the bus or wire.

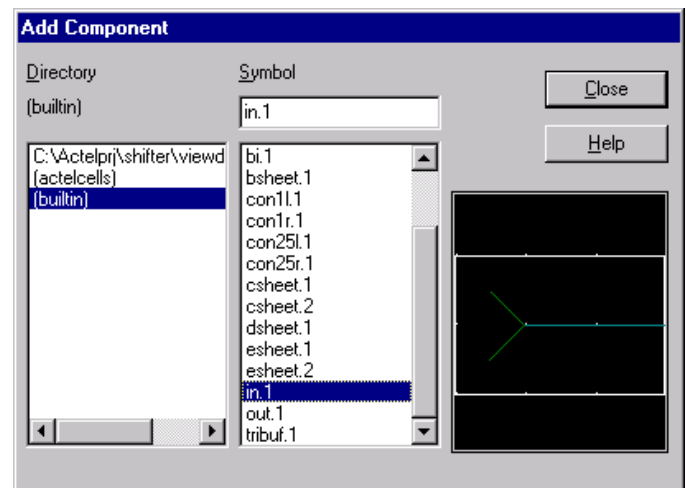


Figure 16 - adding hierarchical connectors

7. Repeat step 6 to add the Hierarchical Output connector. Select **out.1** from the Symbol list to add the output connectors and position to connect to the output bus as shown in Figure 12.
8. Label the components as indicated on the lab handout by double clicking on the component and adding the name on the **Name** tab in the **Component Properties** dialog box (Figure 17).

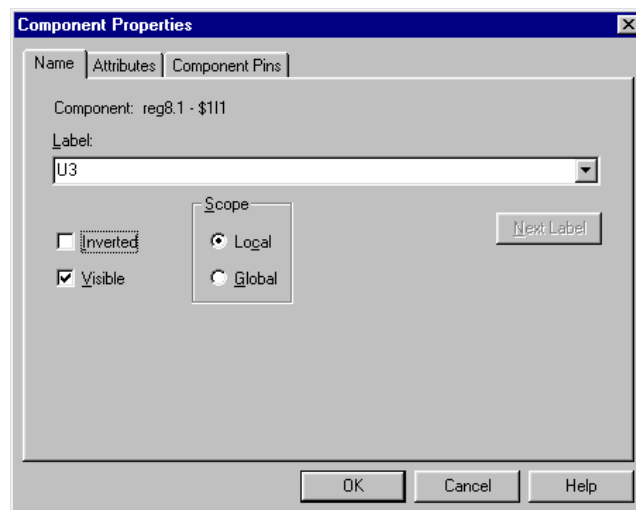


Figure 17 - labeling components



9. After completing the reg32 schematic, save the file and check for errors by clicking **Save + Check** (**File** menu) or by clicking the Save + Check icon on the ViewDraw toolbar. Correct any errors that are reported. Close the ViewDraw schematic editor by clicking **Exit** (**File** menu).
10. When completed, the hierarchy of the reg32 design will appear as shown in Figure 18 on the Libero IDE Design Hierarchy tab.

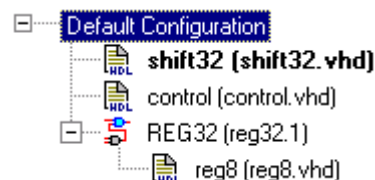


Figure 18 - reg32 design hierarchy in Libero IDE

11. Optional Step: Run *Check Schematic* on the REG32 block in order to perform some additional schematic checks not provided in ViewDraw (see Figure 19)

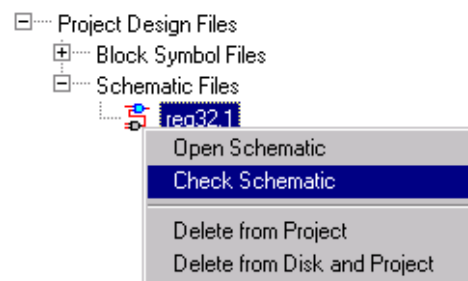


Figure 19— Perform “Check Schematic”

Step 6 - Creating the top-level schematic

The final step is to tie all the blocks (control, shift32 and reg32) together with a top level schematic. Before creating the schematic, you need to create a block symbol for the 32 bit register (reg32). The completed top level schematic is shown below (Figure 20).

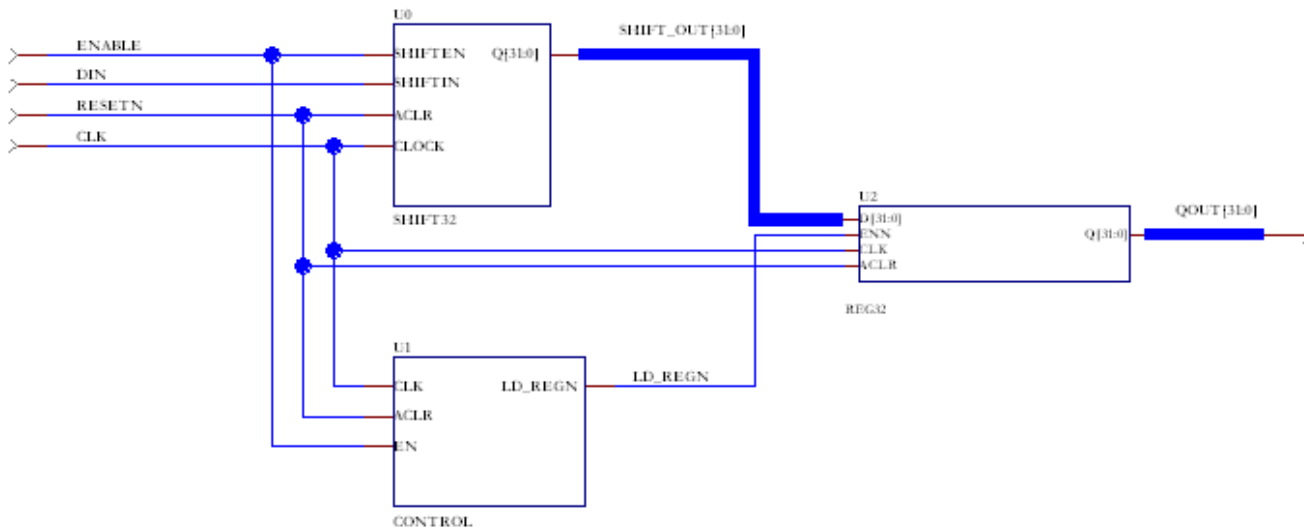


Figure 20 - completed top level schematic

To create the reg32 schematic symbol:

1. In the Libero IDE Design Hierarchy tab, select *REG32*, then right mouse click and select **Create Symbol** (Figure 21). A ViewDraw symbol will be created for the storage register.

The ViewDraw symbol will be visible under Block Symbol Files on the Libero IDE File Manager tab.

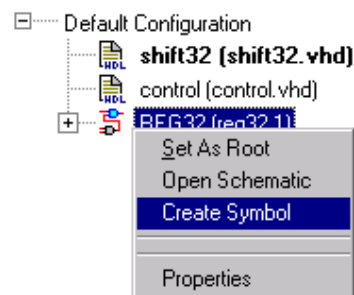


Figure 21 - Creating a ViewDraw symbol for REG32

To create the top-level schematic:

1. From the Libero IDE File menu, click *New* again or double-click on *ViewDraw* in the process window. The New File dialog box is displayed (Figure 22).
2. Select Schematic in the New dialog box.
3. Specify *top* as the name. Click **OK**. ViewDraw opens in a separate window.

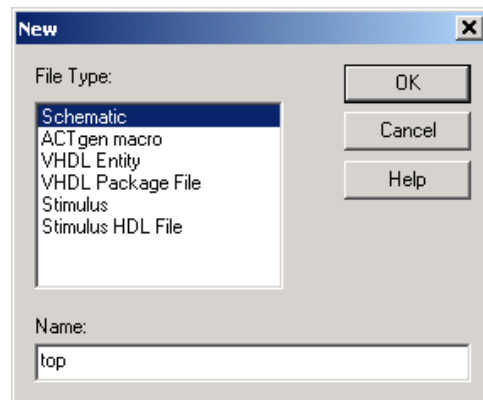


Figure 22 - Creating the top level schematic

- Follow the instructions in Step 5 to complete the top level schematic. When finished your schematic should look like Figure 20.



- After completing the top level schematic, save the file and check for errors by clicking **Save + Check** (**File** menu) or by clicking the Save + Check icon on the ViewDraw toolbar. Correct any errors that are reported. Close the ViewDraw schematic editor by clicking **Exit** (**File** menu).
- When completed, the hierarchy of the shifter design will appear as shown in Figure 23 on the Libero IDE Design Hierarchy tab. If TOP is not in bold font, set it as root by highlighting, right mouse clicking and selecting **Set as Root**.
- Optional Step: Run *Check Schematic* on the TOP block in order to perform some additional checks not provided by ViewDraw.

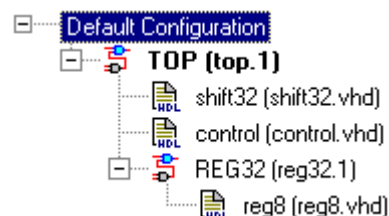


Figure 23 - Design hierarchy for TOP

Step 7 - Creating a testbench

You can simulate the shifter design (top) prior to synthesis. First, create a testbench using WaveFormer Lite. (Please refer to the Appendix for a WaveFormer Lite tutorial.)

Generating an HDL testbench for top:

In this step, you will create a Design Stimulus File and testbench for top using SynaptiCAD's WaveFormer Lite. Following the instructions in the Appendix, define the values for the clock signal (CLK), the reset signal (RESETN), the enable signal (ENABLE) and the serial input data (DIN).

- Create the clock by right clicking CLK and then selecting *Signal(s) <-> Clock(s)* to convert from signal to clock format. Double-click CLK again, and select the *Clock Properties* sub-menu. Define a clock frequency of 40 MHz. Maintain the default 50% duty cycle.

Click **OK** in the Clock Parameters window to generate clk with a 40 MHz frequency.

- Following the instructions on the previous pages, create waveforms for resetn, load and enable as described below:
 - RESETN low 0 ns - 50 ns
 high 50 ns - 1 us
 - DIN high 0 ns to 1 us
 - ENABLE low 0 ns to 100 ns
 high 100 ns to 500 ns
 low 500 ns to 550 ns
 high 550 ns to 1 us

Your waveforms should appear as shown below (Figure 24).

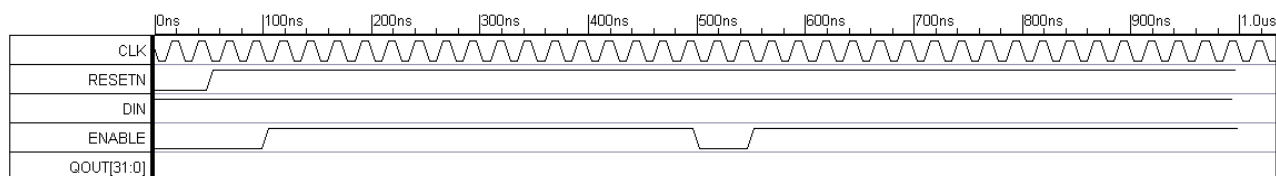


Figure 24 - Timing diagram for top

- After creating the all waveforms successfully, select Save As (timing diagram) from the File menu to save the waveforms. In the Save As dialog box, enter *top.btim* as the file name and click **Save**.
- After saving the timing diagram file, click the Export in the Wave Former Lite menu. From the Export menu, click *Export Timing Diagram As*. Save the stimulus file as either VHDL or Verilog (according to the netlist format you exported from Designer). For top, export a testbench as follows:

VHDL users will select "VHDL Wait with Top Level Testbench" in files of type and enter *top_tb.vhd* for the file name. Verilog users will select "Verilog with Top Level Testbench" and name the file, *top_tb.v*. Click **Save** to generate the testbench.

- The VHDL or Verilog testbench can be viewed in the WaveFormer Lite Report window. The top level of design hierarchy is instantiated inside the code.
- Exit WaveFormer Lite (**File > Exit**).
- The stimulus files are visible on the Libero IDE File Manager tab (Figure 25).

Your design is ready to simulate!

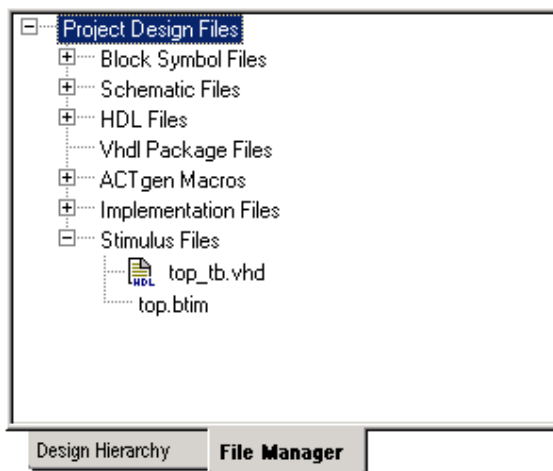



Figure 25 - Stimulus files for top in Libero IDE

Step 8 - Pre-Synthesis simulation of top

Follow the steps below to perform a pre-synthesis simulation of *top*.

1. Double click the **ModelSim Simulation**  button in the Libero IDE Process window, or right mouse click on *top* on the Libero IDE Design Hierarchy tab in the Design Explorer Window and select **Run Pre-Synthesis Simulation**. Libero IDE will prompt you to associate a stimulus. Select the testbench you just created, *top_tb.vhd* or *top_tb.v*.
2. The ModelSim for Actel Simulator will open and compile the source files (Figure 26).

Note: If the message below appears while the VHDL files are being compiled, select **Compile Options** from the **Compile** menu from the ModelSim for Actel main window and select "Use Explicit Declarations only" (Figure 27). Close ModelSim for Actel, return to Libero IDE and run the pre-synthesis simulation again. (Error occurs because source files reference multiple VHDL packages.)

"ERROR: D:/Actelprj/VHDL_labs/Solutions/FifoDemo/hdl/testfifo.vhd(217): Subprogram "=" is ambiguous. Suitable definitions exist in package 'std_logic_1164' and 'std_logic_unsigned'.

ERROR: D:/Actelprj/VHDL_labs/Solutions/FifoDemo/hdl/testfifo.vhd(217): (Use the '-explicit' option to disable the previous error check)"

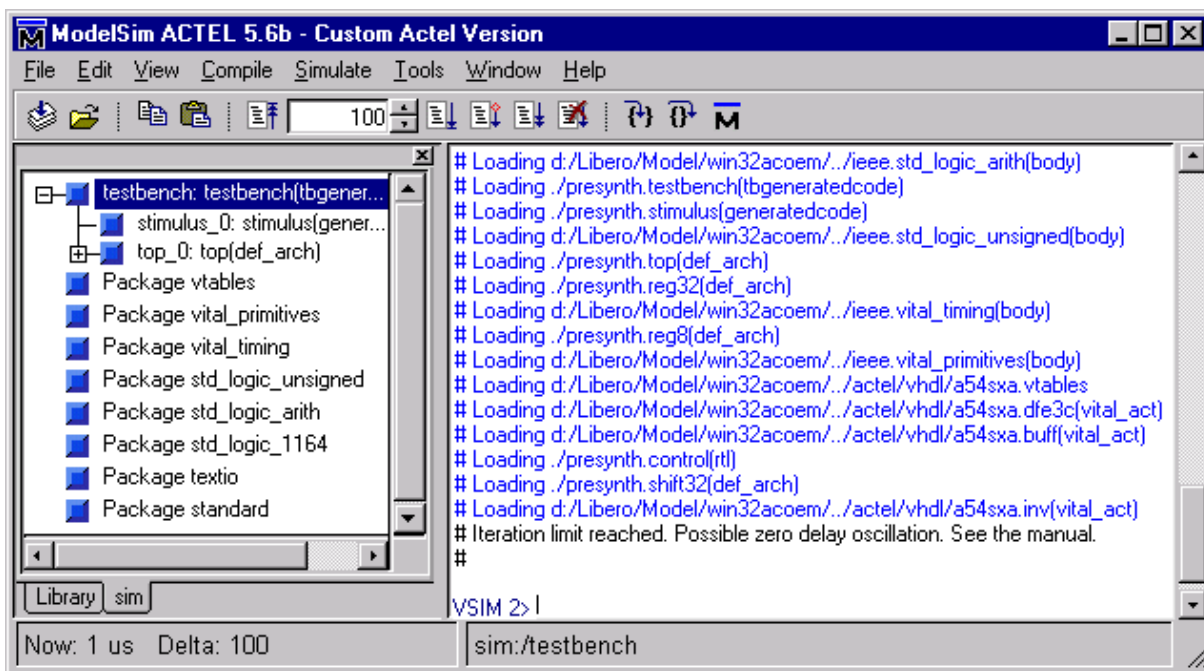


Figure 26 - ModelSim for Actel main window after compiling top

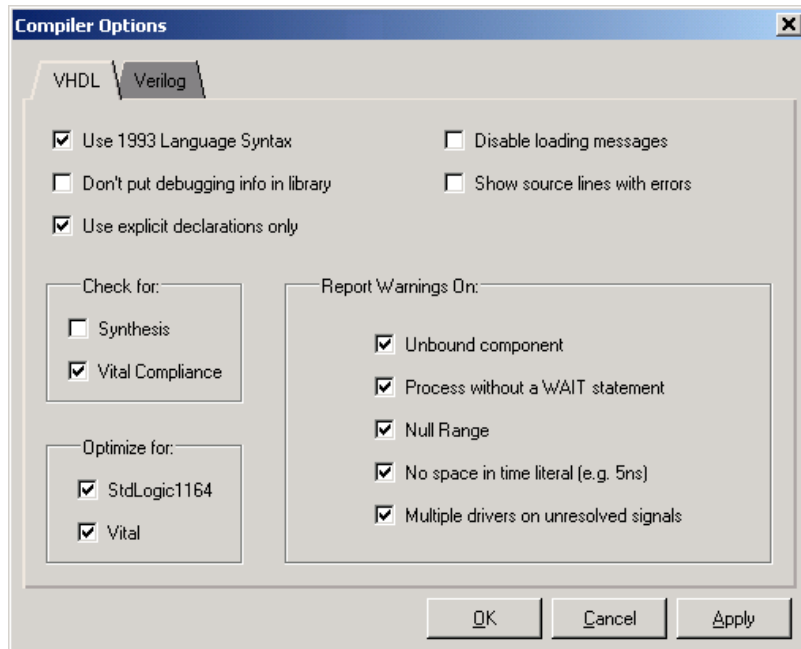


Figure 27 - Changing VHDL compile options in ModelSim for Actel

3. When the compilation completes, the simulator will run for 1 us and a Wave window will open to display the simulation results (Figure 28). Scroll in the wave window to verify the counter functions properly.

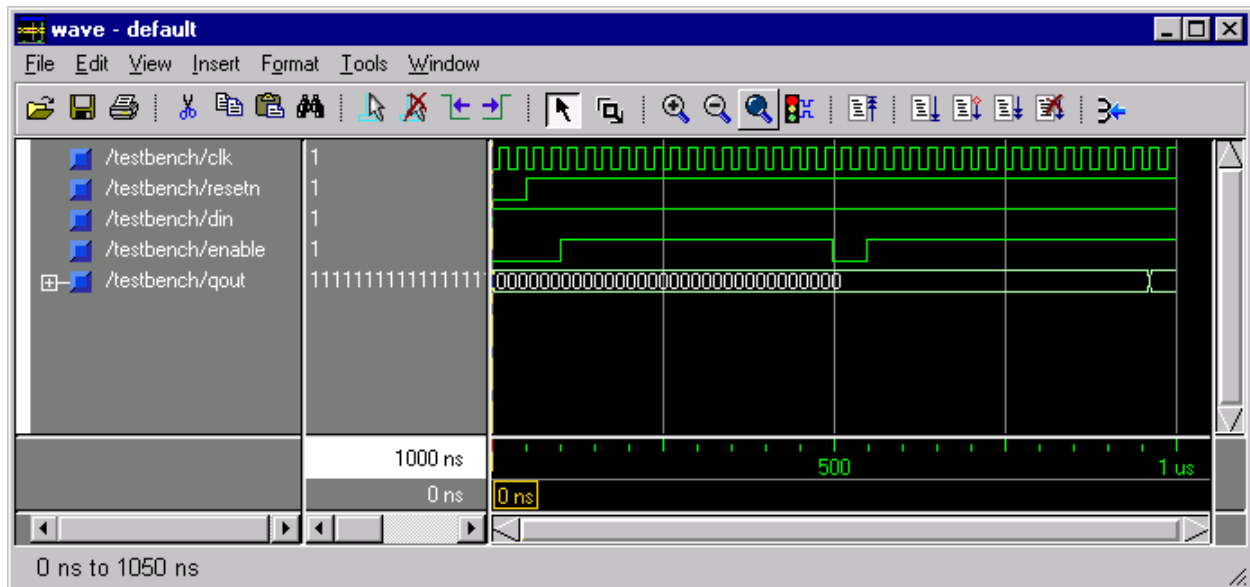


Figure 28 – ModelSim for Actel Wave window for top

Step 9 - Synthesizing with Synplicity

Synthesize the design in Synplify to generate an EDIF netlist. For HDL or mixed-mode designs, Libero IDE will launch and load Synplicity's Synplify with the appropriate design files. In this example the design files for top will be loaded along with the Actel SXA VHDL or Verilog library for synthesis.

Follow the instructions below to synthesize top using Synplify:

1. In Libero IDE, right click on the *top* on the Design Hierarchy tab and select **Synthesize** (Figure 29).

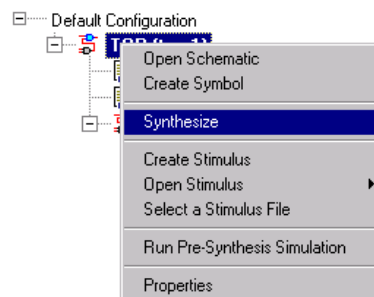


Figure 29 - Launching Synplicity from Libero IDE

2. Synplicity's Synplify application will launch and be loaded with the appropriate design files and set a few default values. From the Target menu, click Change Options. The Options for Implementation dialog box is displayed. Set the following in the dialog box and click OK:
 - **Technology:** Actel 54SXA (set by Libero IDE)
 - **Part:** 54SX08A (default)
 - **Speed Grade:** Std (default)
 - **Fanout Guide:** 10 (default)
 - **Hard Limit:** off (default) This refers to the Fanout limit.
 - **Disable I/O Insertion:** off (default) This is always off for Libero's Synplify Lite.
3. Click the *RUN* button. Synplify will now compile and synthesize the *top* design and output an EDIF netlist named *top.edn*. When the **Done!** appears in the Synplicity menu, the design has been successfully synthesized.

The resultant EDIF file, *top.edn*, will appear under the Implementation Files on File Manager tab in Libero IDE. Libero IDE will also create a structural VHDL or Verilog netlist (*top.vhd* or *top.v*) from the EDIF netlist.

4. From the File menu, click *Exit* to close Synplify. Click on Yes to Save changes to top.prj in the Synplify dialog box that comes up to save any settings that you have made while in Synplify.

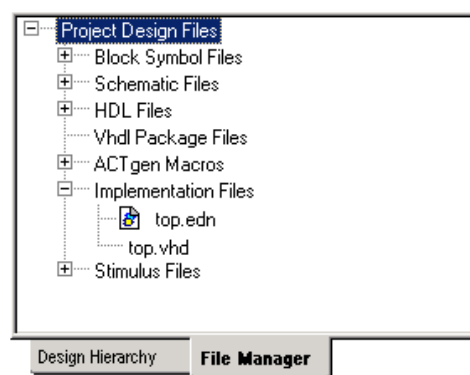


Figure 30

Note: Should any errors appear after you have pushed the Run button, you can edit the file using the Synplify editor. Double click the file name in the Synplify window to open the editor. Any changes made here will be saved to your original design file in the Libero IDE, so close the file in the Libero HDL editor before you do this.

Step 10 Post-Synthesis Simulation of top

Post-Synthesis simulation with the ModelSim for Actel simulator:

1. Invoke the ModelSim for Actel simulator by Double clicking the **ModelSim Simulation** button in the Libero IDE Process window or by selecting *top* in the Libero IDE Design Hierarchy tab then right clicking and selecting **Run Post-Synthesis Simulation**.
2. The ModelSim for Actel Simulator will open and compile the source files (Figure 31).

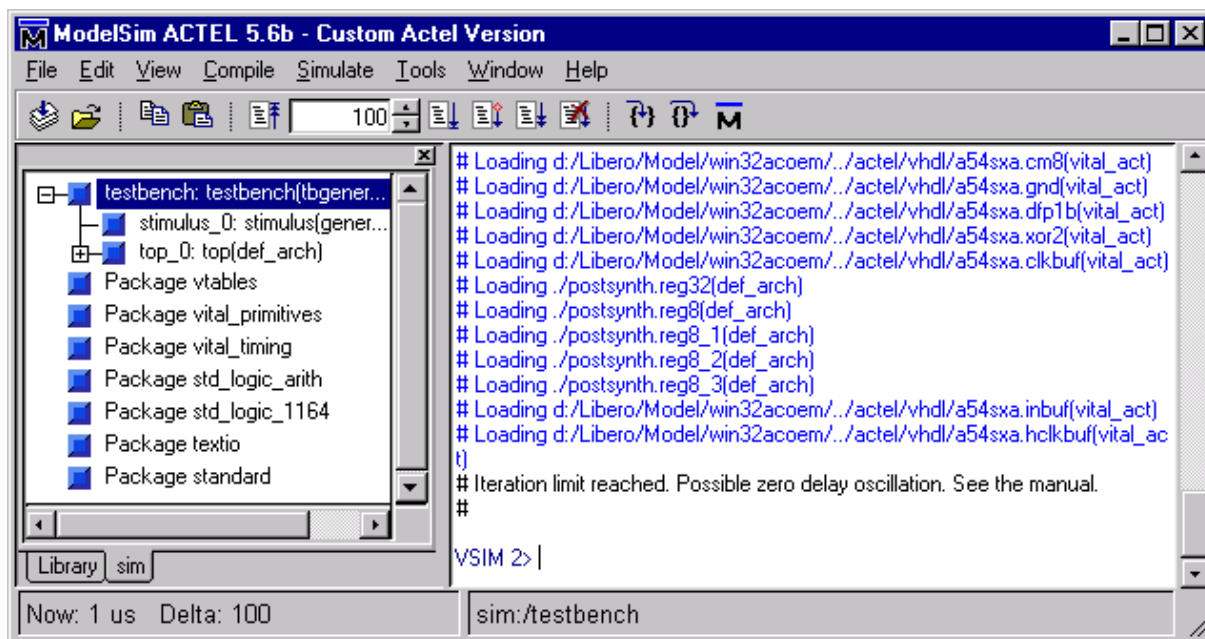


Figure 31 - ModelSim for Actel main window for top post-synthesis simulation

2. When the compilation completes, the simulator will run for 1 us and a Wave window will open to display the simulation results (Figure 32). Scroll in the wave window to verify the design functions properly.

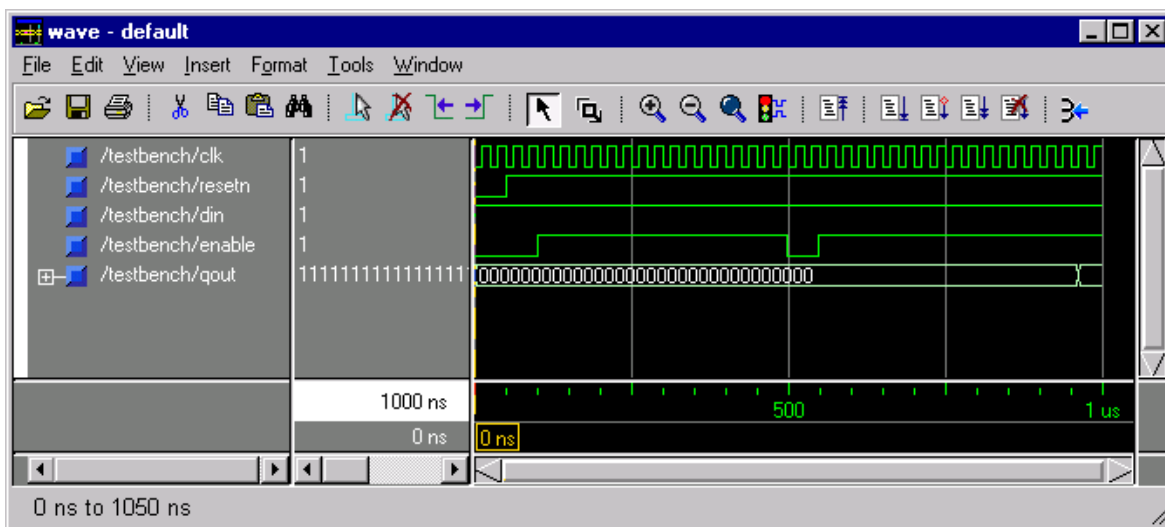


Figure 32 - ModelSim for Actel Wave window for top post-synthesis simulation

Step 11 - Design Implementation - Place and Route

Implement the design using Actel's Designer software

1. **Run Designer.** Select the Design Hierarchy tab in Libero IDE. Right-click on *top* and select **Run Designer**. Actel's Designer application will open and your design file will be read in.
2. **Compile your design.** From the Tools menu, click **Compile** (or click the **Compile** button on the Designer GUI). A series of menus will query you on device type, device package, speed grade, voltage, and operating conditions. Select **A54SX08A** in the Die window and **TQ100** in the Package window. Click on **Next** and **Finish** to complete the remaining steps. Once you have finished the setup, Designer will compile your design and show you the utilization of the selected device. Also note that the Compile box in Designer will turn green indicating that compile has successfully completed.
3. **(Optional) Designer User Tools.** Once you have successfully compiled your design, you can assign pins with PinEdit, view pre-layout static timing analysis with Timer, set timing constraints in Timer, and use ChipEdit to assign modules. You can use each of these functions by left clicking on the flow tree. For more information on these functions please refer to the Designer User's Guide and online help. For this example, we will make no changes to the design in this step.
4. **Lay out your design.** From Designer, click on **Layout**. In the Layout dialog box click **OK** to accept standard layout. The Layout box in Designer will turn green indicating that layout has successfully completed.
5. **Back-Annotate your design.** From Designer, click on **Back-Annotate**, or select *Back-Annotate* from the Tools menu. Choose SDF as CAE type and ensure that you have **Export Netlist** and **VHDL** (or Verilog) selected as the appropriate simulation language. Click **OK**. Designer will export the timing information in SDF format as *top_ba.sdf*.
6. **Click on Fuse in the Designer flow tree if you wish to create a programming file for your design.** This step can be performed later after you are satisfied with the back-annotated timing simulation.
7. **Save and Close Designer.** From the File menu, click **Exit**. Select Yes to save the design before closing Designer. Designer saves all of the design information in a *.adb file. The file *top.adb* appears under the Implementation Files on the File Manager tab in Libero IDE (Figure 33). You can reopen this file by right clicking and selecting **Open in Designer** to launch the Designer application.

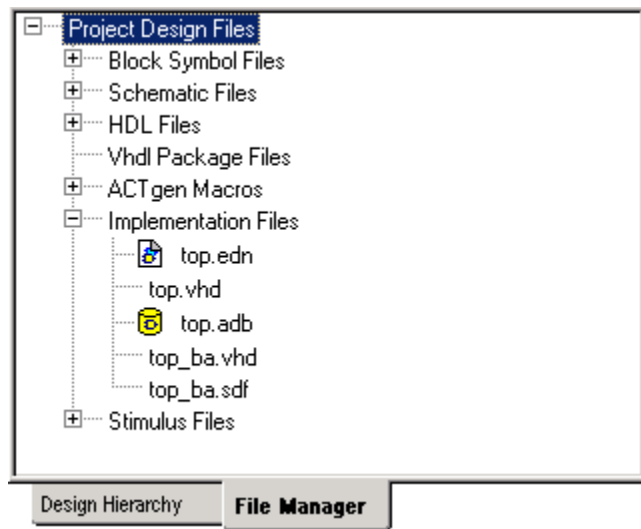


Figure 33 - Top Designer files in Libero IDE File Manager

Step 12 – Timing simulation with back-annotated timing

To perform timing simulation with back annotated timing in the ModelSim for Actel HDL simulator:

1. Invoke the ModelSim for Actel simulator by Double clicking the **ModelSim Simulation** button in the Libero IDE Process window or by selecting *top* in the Libero IDE Design Hierarchy tab then right clicking and selecting **Run Post-Layout Simulation**.
2. The ModelSim for Actel Simulator will open and compile the source files.
3. When the compilation completes, the simulator will run for 1 us and a Wave window will open to display the simulation results (Figure 34).

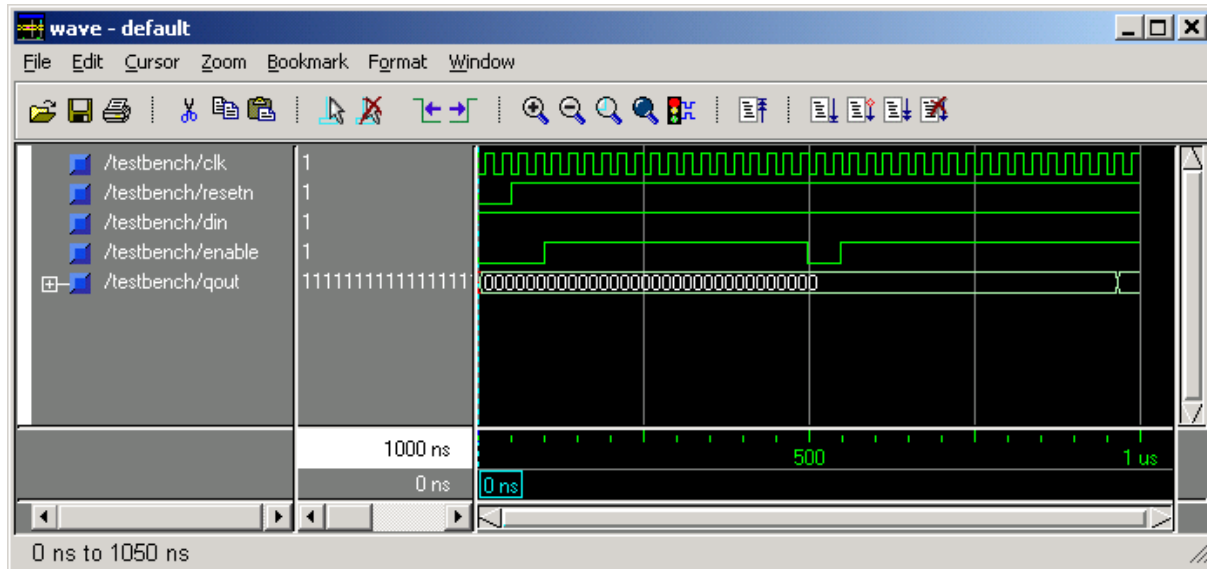


Figure 34 - Post-Layout simulation for top

4. Scroll in the wave window to verify that the design works correctly. Use the cursor and zoom buttons to zoom in and out and measure timing delays.
5. Exit the simulator by selecting **Quit** from the File menu in the ModelSim for Actel main window.
6. Close the Libero IDE by selecting **File > Exit**.

End of tutorial

APPENDIX - WaveFormer Lite Tutorial

Creating stimulus with WaveFormer Lite:

Follow the steps below to create stimulus for using WaveFormer Lite.

1. From the Libero IDE Design Hierarchy tab, double click the left mouse button on the WaveFormer Lite Stimulus icon.
2. The Graphical Test Bench Generator will display the signals you assigned to your design in the schematic.

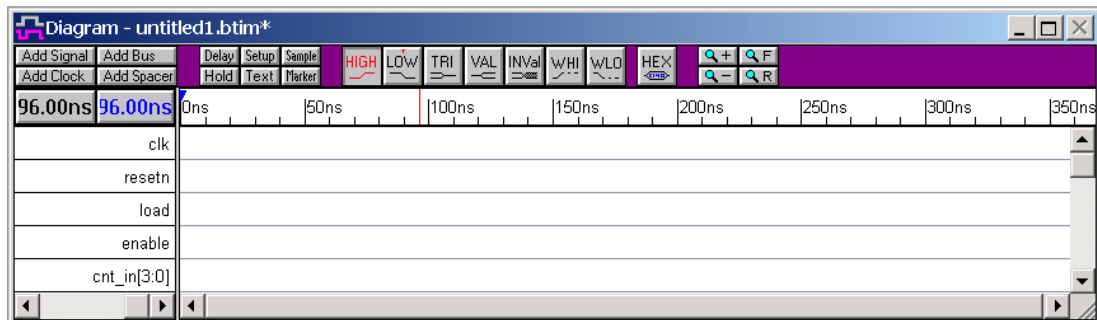


Figure A1 - WaveFormer Lite Diagram window for example2

To draw a waveform in the Timing Diagram Editor for a Clock signal:

1. Create a clock by right clicking CLK and then selecting *Signal(s) <-> Clock(s)* to convert CLK from signal to clock (ie. periodic) format. (Figure A2).
2. Double-click CLK again to bring up the Signal Properties menu, and then select the *Clock Properties* sub-menu. Define the clock frequency.
3. Specify the Frequency or Period and a Duty Cycle to generate clock signals. Click **OK** to create the clock waveform.

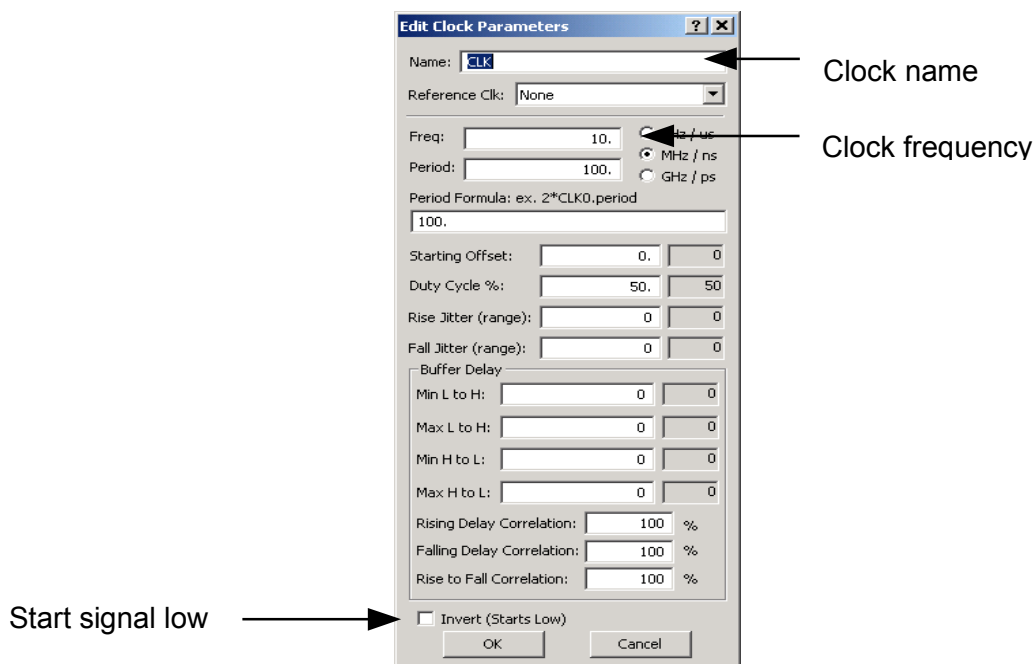


Figure A2 - Edit Clock Parameters window

To draw a waveform in the Timing Diagram Editor for a bus signal:

1. Select the state (value) of the signal you wish to draw by clicking on one of the state values in the toolbar. For example, to draw a valid bus value, select the VAL button. The selected state button turns red (see Figure A3).

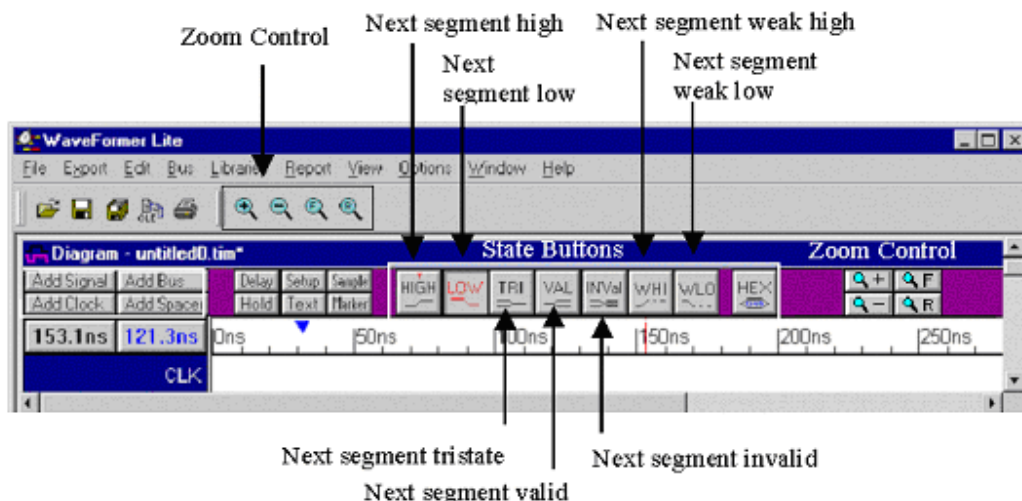


Figure A3 - WaveFormer Lite Diagram editor controls

2. Place the mouse cursor inside the diagram window in the same row as the bus name and position the mouse cursor at the time value where you want the selected state to end. Left click the mouse to draw a waveform from the end of the previously existing signal (if there is any) to the mouse cursor position.
3. Move the mouse to the right and click again to draw another segment. Click on a bus state button (TRI, VAL or INVal) to activate it. The state buttons automatically toggle between the two most recently activated states. The waveform to be drawn next will be determined by the button that was selected (highlighted red). Double-click on a button to prevent it from toggling.

When you draw signals using the mouse, the signal edges are automatically aligned to the closest edge grid time. To fine tune and make minor adjustments to signal transition times, position the cursor over the transition until it changes to a bolded bi-directional arrow. Left click and drag the transition edge either left or right in time.

To draw a waveform in the Timing Diagram Editor for a regular signal:

4. Select the state of the signal you wish to draw by clicking on one of the state values in the toolbar. To draw a '1' value for a signal, select the HIGH button. Place the mouse cursor inside the diagram window in the same row as the signal name. Then, position the mouse cursor at the time value where you want the selected state to end. Left click the mouse to draw a waveform from the end of the previously existing signal (if there is any) to the mouse cursor position.
5. Move the mouse to the right and left click again to draw another segment. Click on a state button to activate it. The state buttons automatically toggle between the two most recently activated states. This allows you to draw alternating High and Low values easily without having to select the toolbar state each time.

Copying Waveforms

You can copy sections of waveforms and paste those sections either onto (overwrite) or into (insert) any signal in the diagram. To copy and paste waveform sections:

1. Select the names of the signals that you want to copy. If no signals are selected, the Block Copy command will select all the signals in the diagram.
2. Choose the Edit > Block Copy Waveforms menu option. This opens the Block Copy Waveforms dialog box (Figure A4), with the selected signals displayed in the Change Waveform Destination list box.
3. In the dialog, enter the values that define the copy and paste:
 - Choose either Time or Clock cycle for the base units of the dialog. If you are copying just signals (no clocks) then time is the default base unit of the dialog. If you are copying part of a clock then it is best to choose a clock cycles base unit and choose the copied clock as the reference clock. If you select time when copying clocks, the (end_time - start_time) must equal an integral number of clock periods, and the place_at time must be at the same clock period offset as the start_time.
 - Start and End define the times of the block copy.
 - Place At is the time that at which the block will be pasted.
 - The Insert and Overwrite radio buttons determine whether the paste block will be inserted into the existing waveforms or overwrite those waveforms.
 - The list box at the bottom of the dialog determines which signal the copied waveforms will be pasted into. To change this mapping:
 - i) Select a line in the list box. This places the destination signal into the drop-down list box on top of the list box.
 - ii) Choose another signal from the drop-down list box. Each destination signal can be used only once per copy.
 - iii) Click OK to complete the copy and paste operation.

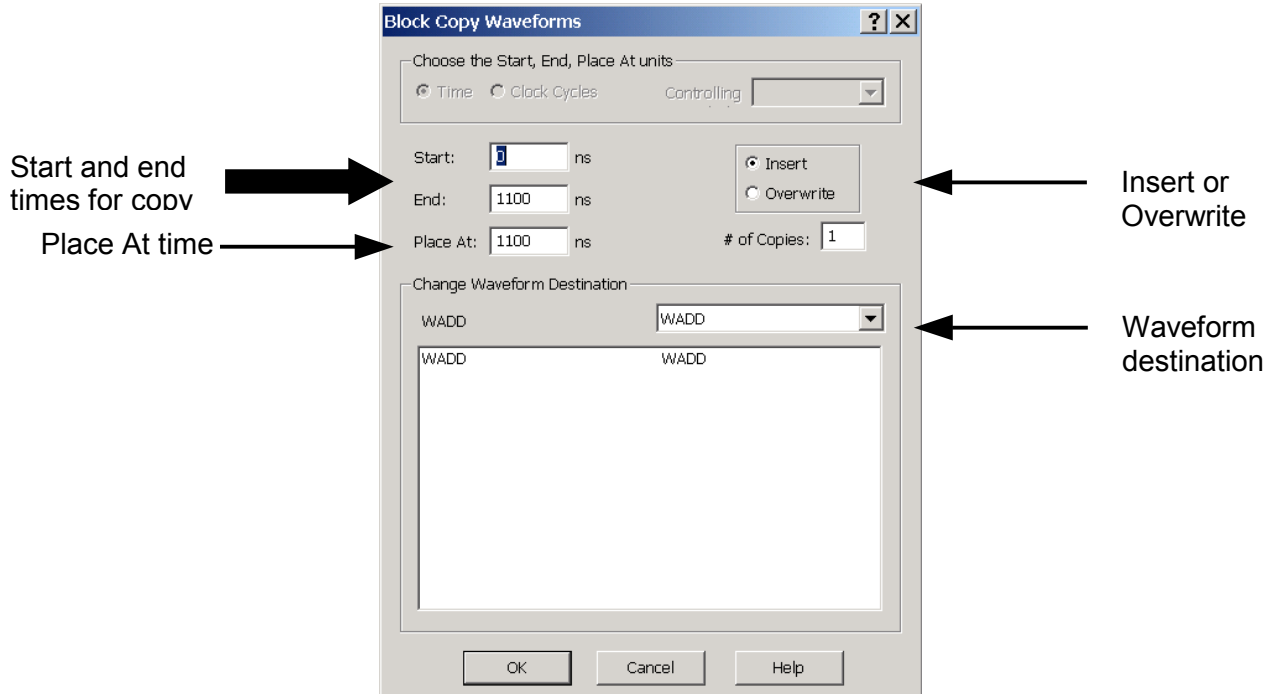


Figure A4 - Block Copy Waveforms dialog box