XAPP350 (v1.0) December 20, 2000

Implementing HDL with WebPACK ECS Schematic Editor

Summary

This application note provides an introduction to the capabilities and functionality of the WebPACK™ ECS Schematic Editor for implementing Hardware Description Language (HDL) CPLD designs in WebPACK Project Navigator.

Introduction

The Engineering Schematic Capture (ECS) Schematic Editor is the WebPACK Project Navigator primary graphical design entry tool. ECS is used to capture digital circuit designs and can be used as the starting point in the development process. The ECS tool can be used solely for documentation or as a source for translating a design into a format that is compatible with HDL simulators. The ECS tool allows schematics to consist of symbols (either local or from the ECS library), connect wires between components, and/or assigned attributes. A design in ECS can contain block elements that represent HDL source code or other schematics. This application note provides a guide for embedding HDL source code as a symbol into a ECS schematic for use in WebPACK Project Navigator.

Several help guides exist for using ECS Schematic Editor within the WebPACK Project Navigator help index. The help files are updated with each release of software and provide a reliable source of information. The WebPACK help menu can be located on a PC from the **Start | Programs | Xilinx CPLD WebPACK | Help and Technical Support** menu. The ECS help can be found in the tab **Contents** under **CPLD WebPACK ISE | Tools | Design Entry | Schematic Editor (ECS)**.

ECS Overview

Available design entry using the ECS schematic tool is shown in Table 1.

Table 1: **ECS Supported Design Entry**

Description	CPLD Support	FPGA Support
Schematic design as described with library components	Yes	No
Schematic with ABEL source code	Yes	Yes
Schematic with VHDL source code	Yes	Yes
Schematic with Verilog source code	Yes	Yes

Note that using the ECS Schematic Editor tool allows only one design source code flow. The WebPACK Project Navigator tool does not allow a top-level schematic design with embedded VHDL and ABEL, or VHDL and Verilog, or Verilog and ABEL source code.

Note that WebPACK FPGA support does not include the ECS library components. The ECS Schematic Editor for use with FPGA designs is strictly for use as an HDL block editor.

Downloading ECS

The ECS Schematic Editor is available to WebPACK users with installation of the main design entry tool, for both CPLD and FPGA designs regardless of design type (FPGA or CPLD). The WebPACK design entry tools are available for download at http://www.xilinx.com/products/software/webpowered.htm. For CPLD designs that include

logic level drawings, the available library schematics for use in ECS can be downloaded if selected, as shown in Figure 1.

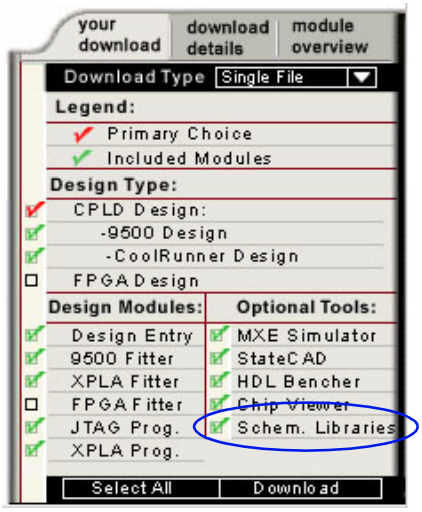

Figure 1: **Selecting Module Download**

The logic functions available in the ECS schematic library include: arithmetic operations, buffers and inverters, comparators, counters, encoders and decoders, latches, shifters, multiplexers, input and output functions, and logic functions. More information on the available library components can be found in the help menu.

Using ECS

ECS Schematic Editor can be used for creating top-level schematic designs with embedded VHDL, Verilog, or ABEL source code. This application note explains the steps involved in and capabilities of embedding design source code with a schematic design. The design example shown in this application note can be found in **XAPP348: CoolRunner XPLA3 Serial Peripheral Interface Master**. The CoolRunner SPI design includes a microprocessor interface and the necessary logic to create the popular serial interface.

Invoking ECS

To use the ECS Schematic Editor, open a new or existing project in WebPACK Project Navigator. If opening an existing project, ensure the project contains either ABEL, Verilog, or VHDL design source code files. If opening a new project, add or create the new design source code files to be embedded into the schematic design.

To create a new schematic in WebPACK Project Navigator, select **Project | New Source...** from the Project Navigator window. The window as shown in Figure 2 should appear and *Schematic*

should be selected from the list of design types, followed by specification of the *File Name* and *Location*.

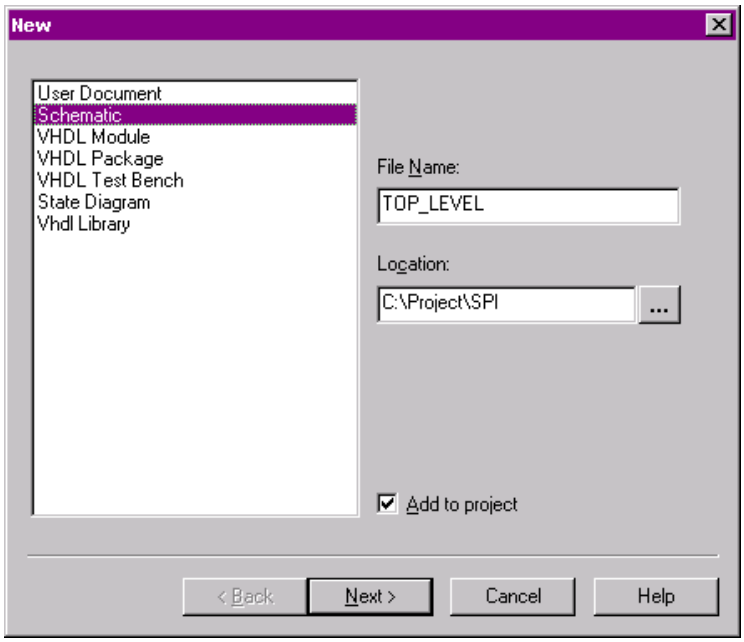

Figure 2: **Specifying New Schematic**

Once the file is specified, the ECS Schematic Editor will be invoked. To embed a design source code file, a symbol of the design file must be created.

Creating an ECS Symbol

An ECS symbol can be created for any design source code file including VHDL, Verilog, and ABEL. The compiled symbol is a block diagram representation of the I/O for use in the schematic editor. The symbol is also linked to the design source code from the schematic drawing. The compiled symbols reside as local symbols in the project directory (for more information see Inserting a Local Symbol).

To create a symbol for an existing design source file, ensure the selected file is highlighted under the *Sources in Project* window as shown in Figure 3. Under the *Design Entry Utilities* in the *Processes for Current Source* window, the option to *Create Schematic Symbol* should appear. By double-clicking on the *Create Schematic Symbol* icon, a symbol will be created

under the design source code file name with a *.sym* extension. For the example shown in Figure 3, the output file created is *uc_interface.sym*.

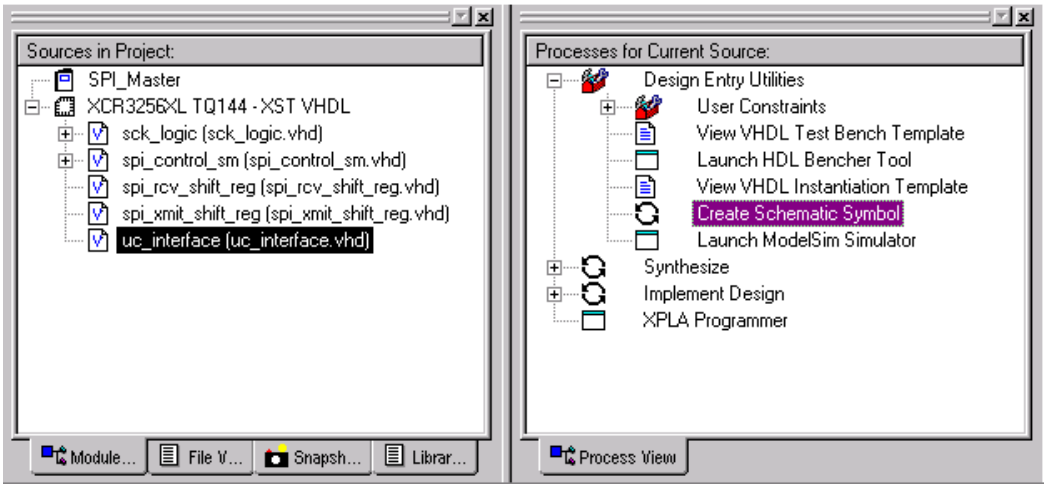

Figure 3: **Creating a Symbol**

These steps must be repeated for creating symbols from multiple source files. A symbol must be created for every design file that is to be inserted into an ECS schematic drawing.

Creating a Schematic Module

In the ECS Schematic Editor, it is possible to create a hierarchy structure of schematics. For this design example a low-level schematic is created, which can be instantiated in a top level schematic drawing. The low-level schematic that is constructed from this design is the SPI interface. The schematic includes the following modules: *spi_control_sm*, *sck_logic*, *spi_rcv_shift_reg*, and *spi_xmit_shift_reg*. After creating a new schematic, the necessary modules can be added to the schematic diagram.

Inserting a Local Symbol

Local symbols can be inserted in a schematic drawing under the **Add | Symbol...** menu in the ECS tool. The *Symbol Libraries* window should appear as shown in Figure 4. After selecting *Local Symbols* from the top window under *Libraries/Directories*, the list of available symbols

should appear in the lower window under *Symbols*. Selecting a specific symbol allows that symbol to be placed in the schematic.

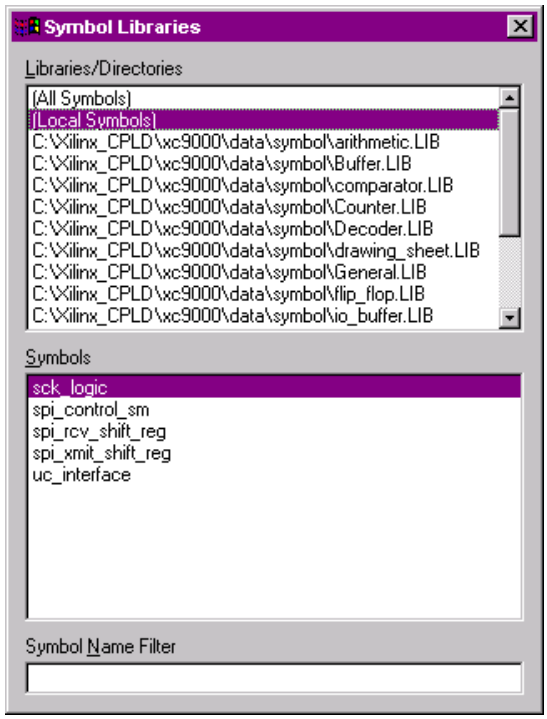

Figure 4: **Inserting Local Symbols**

After adding the local symbols that comprise the SPI interface logic to the schematic drawing, the connections and labels can be made as necessary as shown in Figure 5.

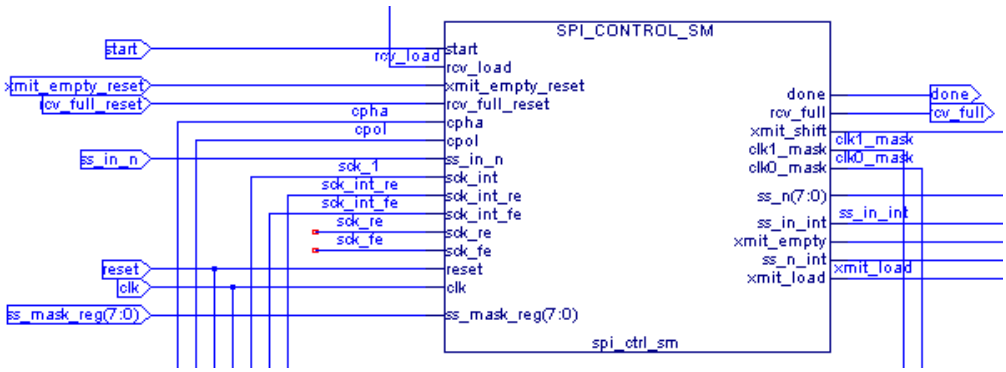

Figure 5: **SPI_CONTROL_SM Symbol Connections**

Figure 6 illustrates the complete SPI interface logic schematic.

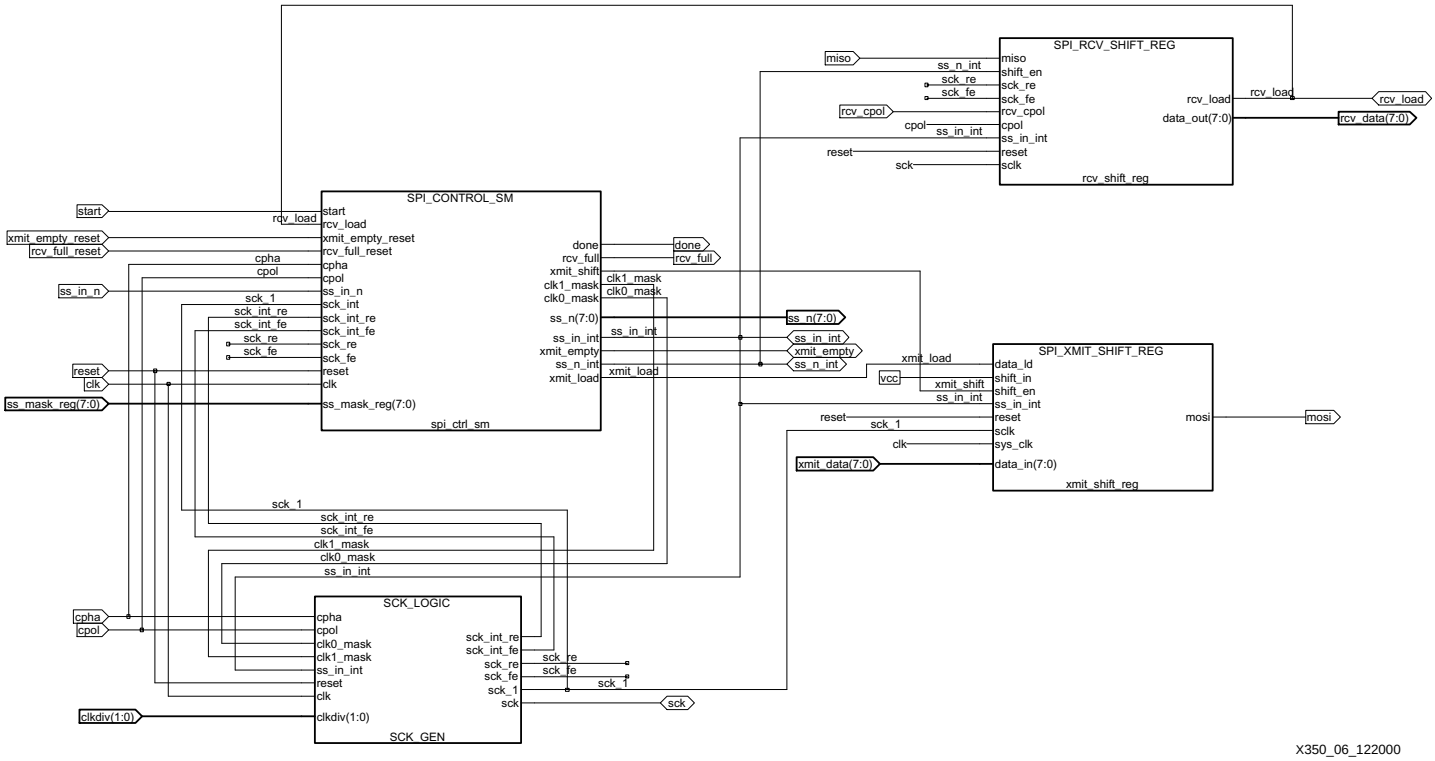

Figure 6: **SPI Interface Schematic**

Once a schematic has been constructed as described, it is possible to create a symbol that describes the current schematic. The created symbol can then be used in another schematic drawing to create a hierarchy structure. One method of creating a symbol from an existing schematic is similar to creating a symbol from a design file. To create a schematic symbol, ensure the schematic file is highlighted (*.sch* extension) in the *Sources in Project* window as shown in Figure 7. A symbol is then created by double-clicking on *Create Schematic Symbol* under the *Design Entry Utilities* as shown in Figure 7.

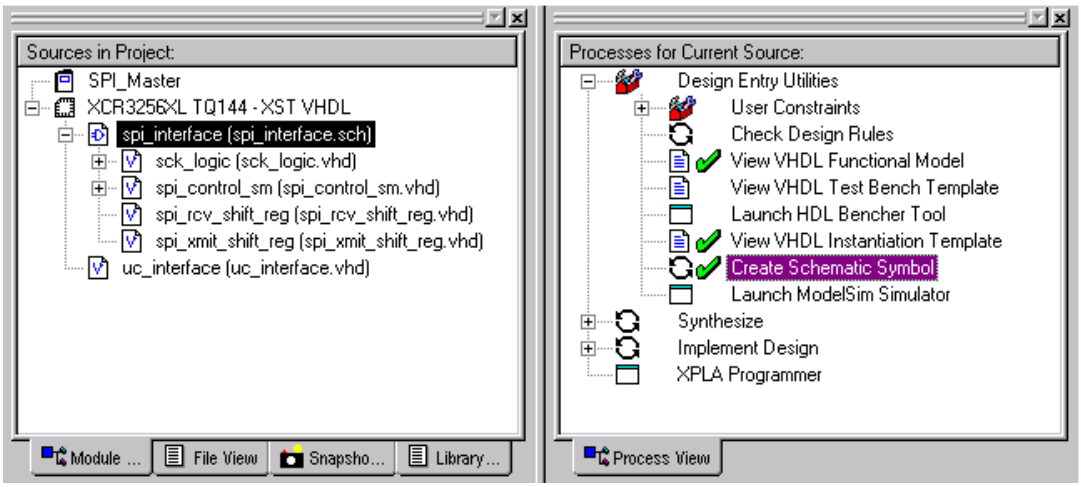

Figure 7: **Creating a Schematic Symbol**

Another way to create a symbol from an existing schematic is done in the ECS tool. From the **Template | Block Symbol...** menu the window shown in Figure 8 should appear. Clicking on the *Use Data From This Block* button in Figure 8 will define the symbol from the data in the schematic.

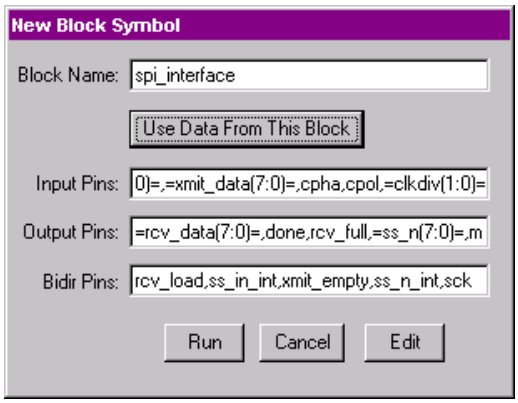

Figure 8: **ECS Block Symbol Window**

Creating A Hierarchical Schematic

For the SPI design example described in this document, it is beneficial to create a hierarchical schematic. This top level schematic provides a visual overview of the main components in the design. The high-level SPI design, *spi_master*, contains two main modules: *spi_interface* and *uc_interface*. The *spi_master* ECS schematic representation is constructed by adding local symbols in the project to the schematic drawing as shown in Figure 9.

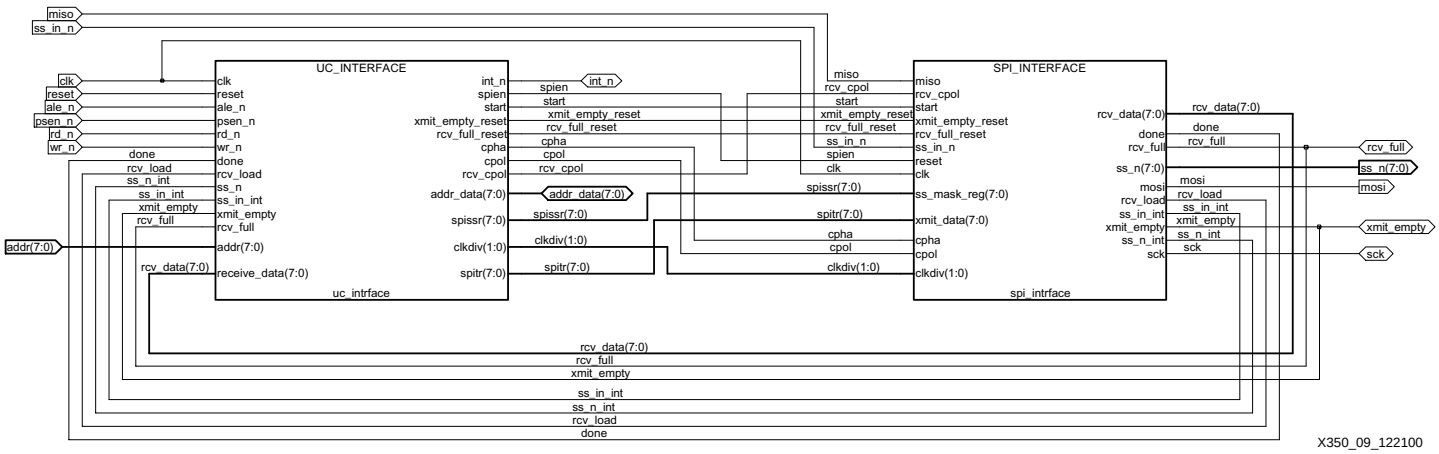

Figure 9: **SPI_MASTER Schematic Drawing**

Push and pop commands exist within the ECS Schematic Editor that allow a designer to view the hierarchical structure of the schematic. In the SPI example, from the *spi_master* schematic shown in Figure 9, a designer can push into the *spi_interface* symbol and view the low-level schematic.

Useful Tools

For designs that incorporate HDL with the ECS schematic tool, several tools exist that bring ease of use to designers.

HDL Functional Model

For designs using schematic entry, it is possible to view the HDL model for the respective schematic drawings. For example, since the SPI design was created in VHDL, it is possible to view a VHDL functional model. To view the functional HDL model, ensure the selected schematic design is highlighted under the *Sources in Project* window in Project Navigator as shown in Figure 10. By double-clicking on *View VHDL Functional Model* under the *Design Entry Utilities*, a window should appear displaying the functional model. The designer can then save this file for use in another project or for future reference.

For the example SPI design shown in Figure 10, highlighting *spi_master.sch* will display the VHDL model created from the schematic, *spi_master.sch*.

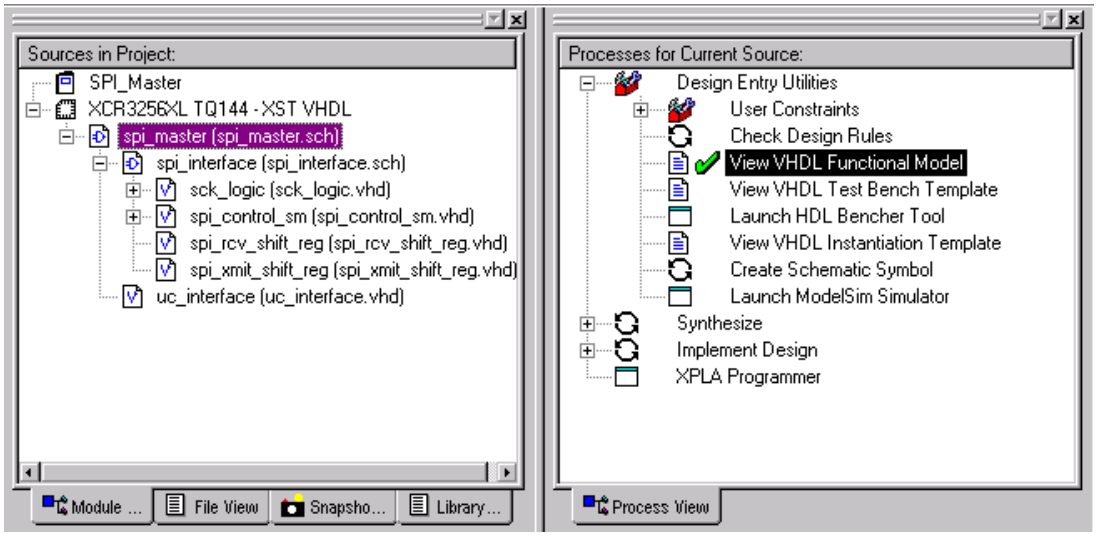

Figure 10: **HDL Functional Model**

The functional VHDL model that is created from the schematic design is saved with a *.vhf* file extension. For the design example in Figure 10, the functional VHDL model of the *spi_master.sch* file is saved as *spi_master.vhf*.

HDL Instantiation Template

Another useful tool in Project Navigator for designers using schematics is the ability to view the HDL instantiation template. This template can be used for designers that wish to use an HDL schematic in a VHDL source file. The HDL Instantiation Template creates a model of the schematic, in terms of inputs and outputs. For VHDL designs, a component declaration and a blank component instantiation (port map) is created. Designers can simply cut and paste from this model into a VHDL design.

By double-clicking on the *View HDL Instantiation Template* under the *Design Entry Utilities* window as shown in Figure 11, a window should appear that contains the instantiation template.

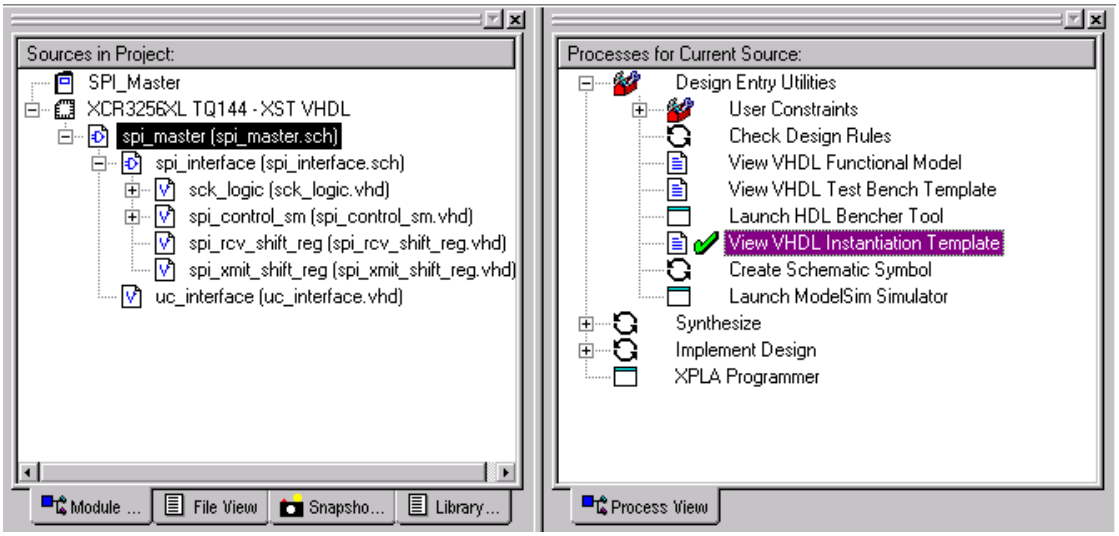

Figure 11: **HDL Instantiation Template**

When the VHDL instantiation template is invoked in Project Navigator a file with a *.vhi* extension is created for the design. For the design example shown in Figure 11, the *spi_master.sch* instantiation template is saved as *spi_master.vhi*.

HDL Test Bench Template

WebPACK Project Navigator software includes the capability to do functional and post-route timing simulations. Simulations are enabled through the free WebPACK Xilinx Edition Starter ModelTech ModelSim software package that is available for download. More information on using ModelSim with WebPACK Project Navigator can be found in **XAPP338: Using Xilinx WebPACK and ModelTech ModelSim Xilinx Edition (MXE)**.

ModelSim software can be used to simulate designs that incorporate embedded HDL in a schematic drawing. The first step for creating a simulation environment is to create a test bench for the design. Project Navigator has the functionality to create a template test bench for an existing design file. Since the SPI example design has been created with embedded VHDL source files in the schematic, a VHDL test bench template will be created. If a top level design schematic exists, ensure this file is highlighted as shown in Figure 12 in the *Sources in Project* window. By double-clicking on the *View VHDL Test Bench Template* under the *Design Entry Utilities* as shown in Figure 12, a window should appear that contains the test bench template. The VHDL test bench template is created from the functional VHDL model and is saved as a vht

file. For the design shown in Figure 12, the *spi_master.sch* test bench template is saved as *spi_master.vht*.

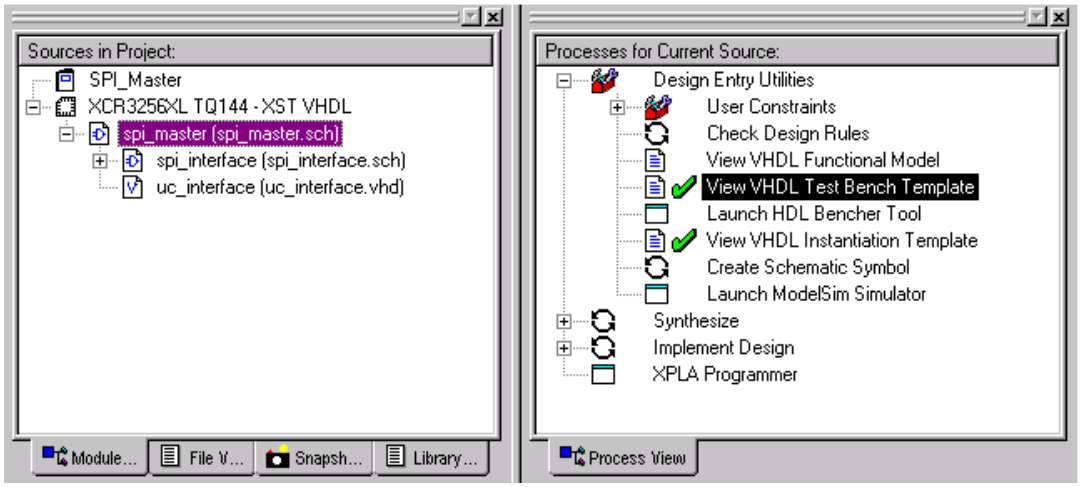

Figure 12: **Create Test Bench Template**

The VHDL test bench template contains the component(s) module instantiation according to the schematic drawing and allows the user to specify the design simulation. The revised test bench can then be saved and brought into the Project Navigator project. When adding a test bench source file to the project in Project Navigator, ensure the *VHDL Test Bench* characteristic is highlighted as shown in Figure 13.

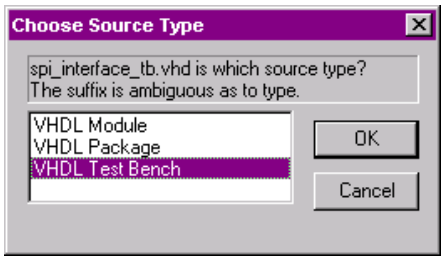

Figure 13: **Add VHDL Test Bench Source File**

Once the VHDL test bench file has been specified in Project Navigator, ModelSim can be invoked to run either the functional or post-route timing simulation. Note that during the compilation process in ModelSim, a design that contains VHDL in schematic drawings will compile both the VHDL source files and the functional VHDL models (*.vhf* file name extension) of the schematic drawings.

Conclusion

WebPACK Project Navigator and the ECS Schematic Editor is easy to use and simplifies HDL designs. Separate modules in a HDL design can be easily integrated and instantiated visually in a schematic editor. For further assistance with the WebPACK Project Navigator software or using the ECS Schematic Editor, search for solutions online at **http://support.xilinx.com** or contact Xilinx Technical Support.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
12/20/00	1.0	Initial Xilinx release.