Lab Project 3: Introduction to Schematic Capture

Revision: August 31, 2009

DIGILENT®
www.digilentinc.com
215 E Main Suite D | Pullman, WA 99163
(509) 334 6306 Voice and Fax

STUDENT

I am submitting my own work, and I understand penalties will be assessed if I submit work for credit that is not my own.

Print Name _____ ID Number _____

Sign Name _____ Date _____

Estimated Work Hours

1 2 3 4 5 6 7 8 9 10

Overall Weight

1 2 3 4 5 6 7 8 9 10

Point Scale

4: Exemplary
3: Complete
2: Incomplete
1: Minor effort
0: Not submitted

20% will be deducted from scores for each week late
Score = Points awarded (Pts) x Weight (Wt)

LAB ASSISTANT

#	Demonstration	Wt	Pts	Late	Score	Lab Asst Signature	Date
2	Inspect source & sim files	2					
2	Circuit demo	3					
3	Circuit demo	3					

Total In-Lab Score

GRADER

#	Attachments	Wt	Pts	Score
1	Schematic and simulation	3		
2	Schematic	3		
3	Schematic	3		

Weeks late

Total Grading Score

Total score is In-lab score plus grading score

Total Score

Introduction

This Lab project introduces the Xilinx ISE/WebPack schematic capture and simulation tools. A few basic designs are presented as vehicles to illustrate tool use.

Modern CAD tools like ISE/WebPack have a top-level graphical interface called a “framework” from which all individual CAD tools can be launched. This top-level interface, called the Project Navigator, is used to set up new projects, load existing projects, and start programs like the schematic capture, simulation, and synthesizer tools. The Project Navigator shows the status of all individual tools, and keeps track of all work in progress so that no required steps are omitted.

Appendix A provides a detailed tutorial of all steps needed to create, simulate and implement a circuit.

Problem 1. Use the Xilinx schematic capture and simulation tools to enter and simulate the following three individual circuits. Be sure to add I/O markers to both inputs and outputs (and be sure to change the output signals to *output*). Drive the circuit simulation with all possible combinations of inputs. Print and submit a copy of the schematic and simulation waveform files. You do not need to download these circuits to the Digilent board.

1) Y = A·C + B·C' 2) F = A·B' + A'·C + B·C' 3) G = (A+B+C)·(A'+B'+C')

Problem 2. Design and implement a circuit that meets the following requirement. Use the Xilinx CAD tools to capture a schematic and simulate the design, and then implement the circuit on the Digilent board. Print and submit the schematic, and have the lab assistant inspect your work. When your circuit is complete, demonstrate its function to the lab assistant.

Amy, Baker, Cathy, and David are responsible for buying new beans for the "Overhead Coffee Company". Each of them casts a vote to determine if a given lot of beans should be purchased. They sometimes use questionable criteria when deciding to vote, but they have learned through experience that certain combinations of votes yield good results. Design and implement a logic circuit that they can use to indicate whether they should buy new beans. Use slide switches for vote entry (either "buy" or "not buy"), and an LED to indicate when beans should be purchased. A “buy” order is placed if:

David and Baker votes YES,
or Amy votes NO while Cathy vote YES,
or Amy and Baker vote YES and the rest vote NO,
or Cathy and David vote NO,
or they all vote YES.

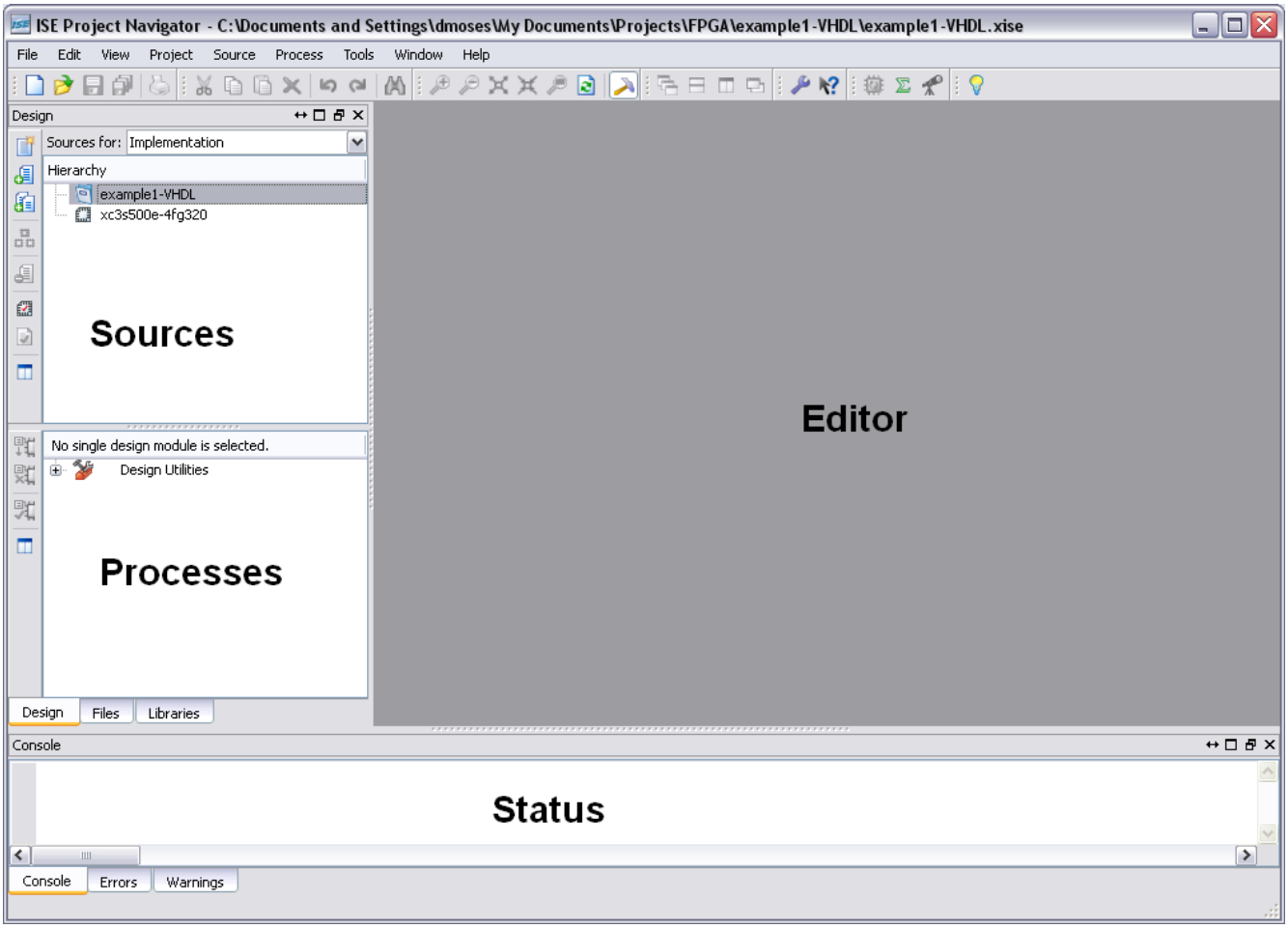
Problem 3. Design and implement a circuit that can illuminate an LED whenever an odd number of the eight slide switches outputs a logic ‘1’. Download this circuit to the Digilent board, and demonstrate your circuit to the lab assistant. Print and submit your schematic.

Appendix A: WebPack schematic design entry tutorial

The Xilinx WebPack CAD software includes schematic capture, simulation, implementation, and device programming tools, all of which can be started from a single “navigator” tool that coordinates the files and processes associated with a given design project. The navigator shows all source files, all CAD tools that can be used with the source files, and any output or status messages and files that result from running a given tool.

Project Navigator

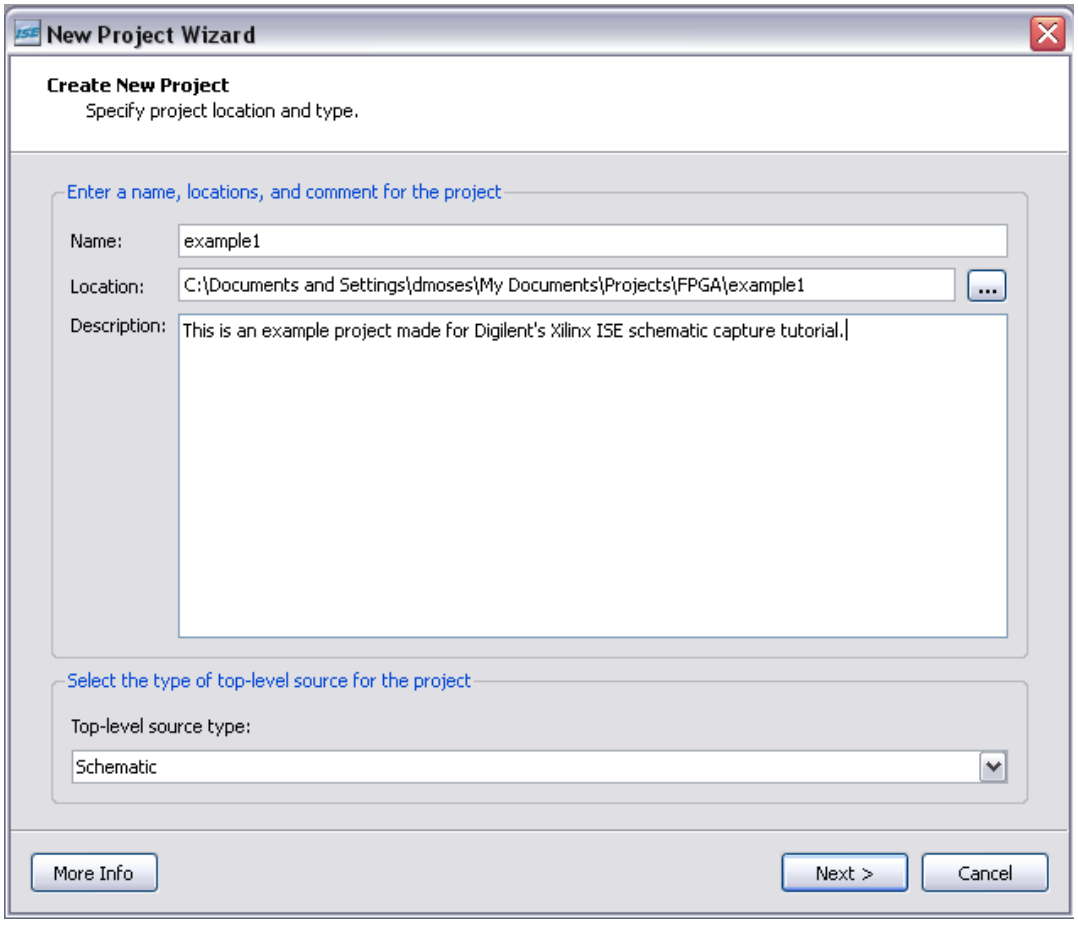
The entry point to the Xilinx ISE or WebPack tool is the **Project Navigator**. The Project Navigator provides a user interface that organizes all files and programs associated with a given design. The main screen is divided into four main panels. The **sources** panel shows all source files associated with a given design. Double-clicking on a file name shown in this panel will open the file in the appropriate CAD tool. The **processes** panel shows all processes that are available for a given source file (different source files have different process options). Double-clicking on any process name will cause that process to run. The **status** panel shows process status, including all warnings and errors that result from running a given process on a given source file. The **editor** panel shows the HDL source code for any selected HDL source file. The project navigator will also open other windows as needed for some applications (for example, the schematic capture tool opens in a separate window). Most designs can be completed without ever leaving the project navigator window.



Starting a new project

New projects can be defined and existing projects can be reopened from within the project navigator window (project navigator can be started from the windows Start menu, or by double-clicking the desktop icon). In general, a new project should be created for each new lab exercise or each new design. The project navigator can be configured to automatically load the last project used, or to not load any project (see the "properties" dialog box).

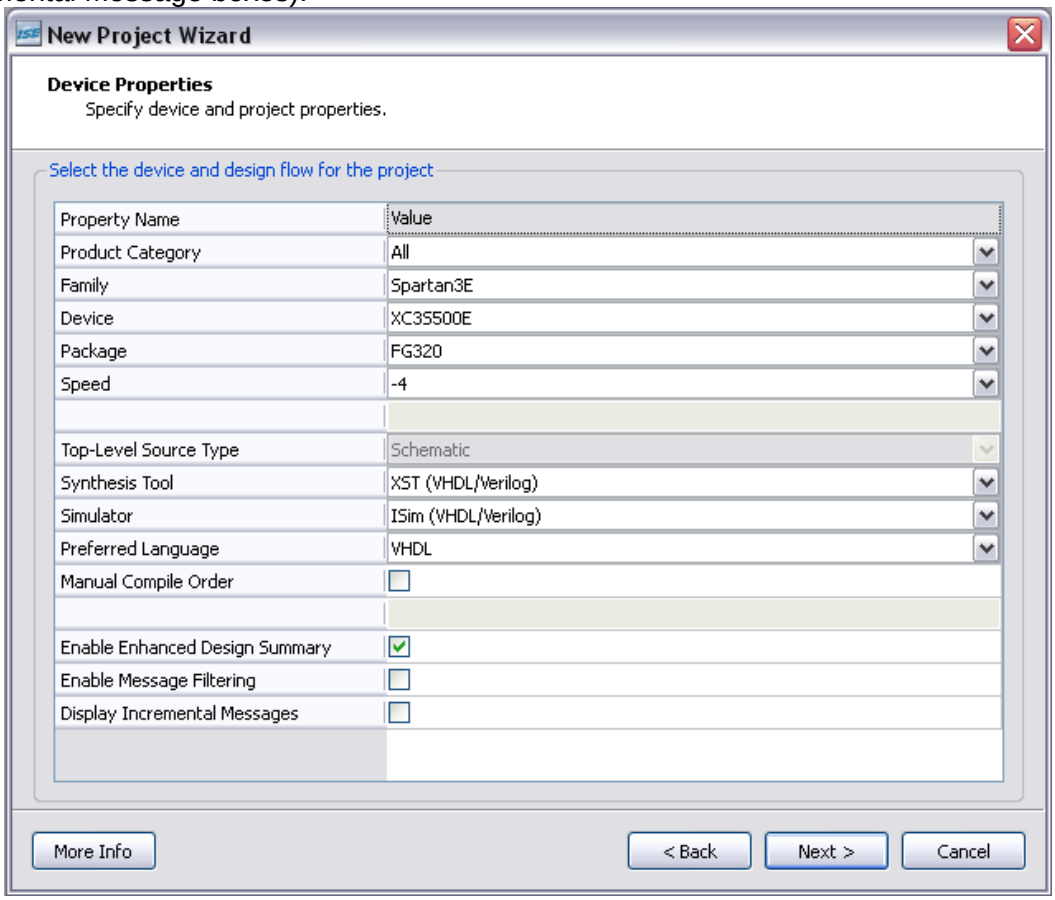
Selecting "new project" from the File pull-down menu will open the **New Project** dialog box, where all information for a new project can be entered. Enter a descriptive name (such as *lab2*) in Project Name box, and choose an appropriate directory in the Location box. This directory will store all design files and all intermediate files, so you will want to choose a directory that is protected, backed-up, and accessible from different locations (if applicable). The Top-Level Source Type box is not critical – here, typically, you will simply choose "Schematic" as shown.



Clicking Next will bring up the Device Properties box. This box can be used to identify several parameters associated with a given design. The **Product Category** field is provided to help organize your projects. The entry in this field is not critical – typically, you will simply choose the default "All". The **Family** and **Device** fields let the CAD tools know what chip you are targeting – this information is required for several of the CAD tools to work properly. For the Basys2 board, choose Spartan3E and XC3S100E, and for the Nexys2, Spartan3E and XC3S500E. For other boards, choose the family and device corresponding to the device loaded on the board (you can typically get this information by

inspecting the chip itself). The **Package** field lets the CAD tools now what chip carrier (or chip package) you are targeting – this information is required so that physical pins can be properly assigned to circuit networks in the chip. The **Speed** field is required so that timing models used by the simulator can accurately model the actual timings in the physical chip itself. This field is only critical if you need very precise simulations of your design.

The **Top-Level Source Type** field can change the user interface display for certain tools. This information is not critical and you will typically choose the default "HDL". For the **Synthesis Tool**, accept the default XST (there are no other choices unless you have loaded other synthesis tools on your PC). For the **Simulator** option, choose ISim simulator. Finally, accept the defaults for the lower three check boxes (check the box for Enhanced Design Summary, and uncheck the Message Filtering and Incremental Message boxes).



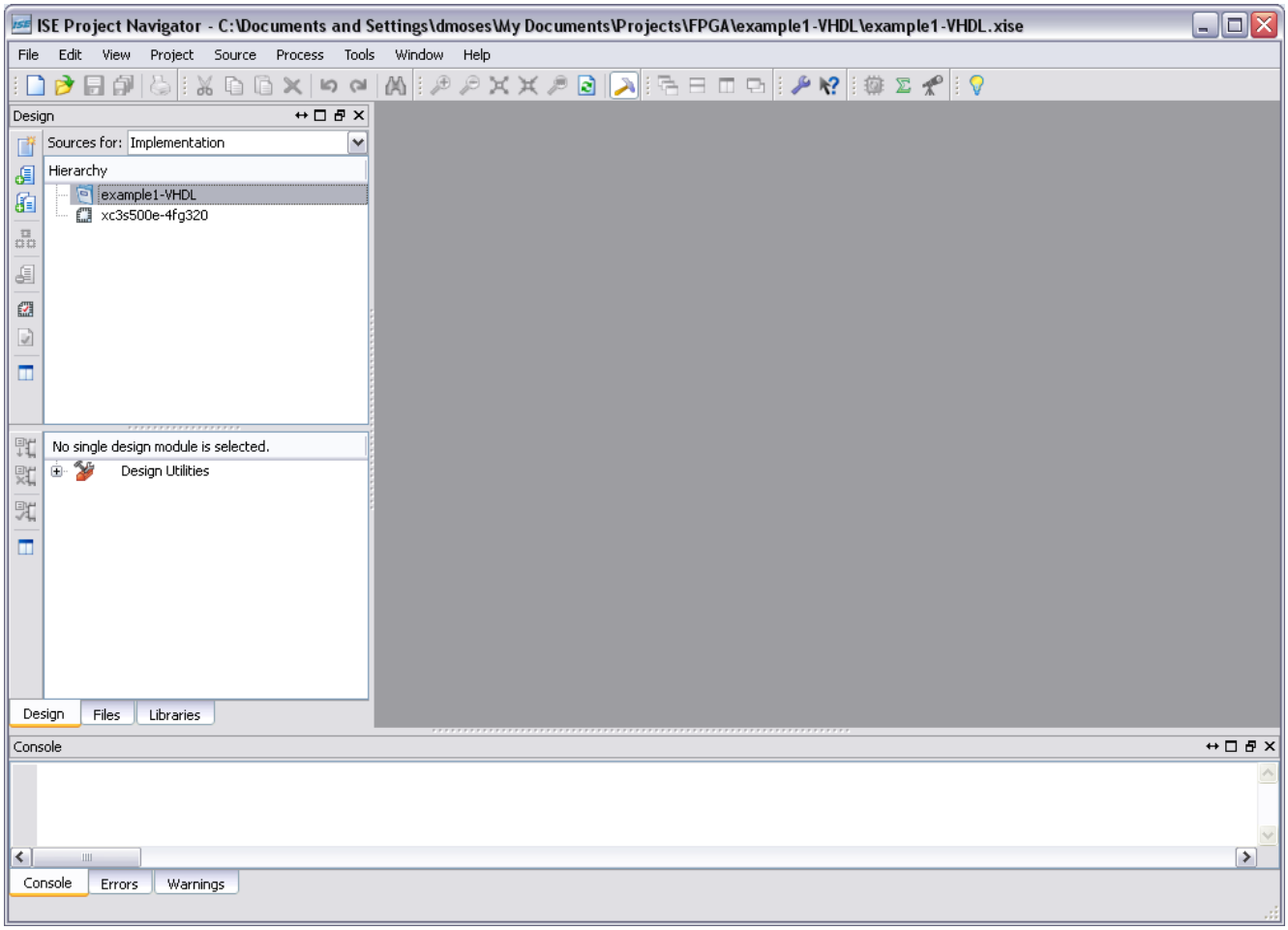
Click Next to bring up the "Create New Source" dialog box. It is easy to create new source files at any point in the project – this box makes it convenient to create new source files right at the start. In some later designs, you may find this convenient. But for now, hit Next without adding any information to this box.

The "Add Existing Sources" dialog box appears allowing you to add existing source files to the project. Again, it is easy to add existing source files at any later stage in the project. This box makes it convenient to define new source files right at the start, and you may wish to do this in later designs. For now, hit Next without adding any information to this box.

This brings up a project summary screen showing all information entered so far. Hit finish to accept the information and launch the new project. Any of the information entered so far can easily be changed later in the project simply by double-clicking on the project name in the Sources window.

The screen shot below shows how the screen should look at this point. You can now define source files that will be used with the project.

In this tutorial, we will start by defining schematic-based projects. Later, we will define VHDL-based projects.

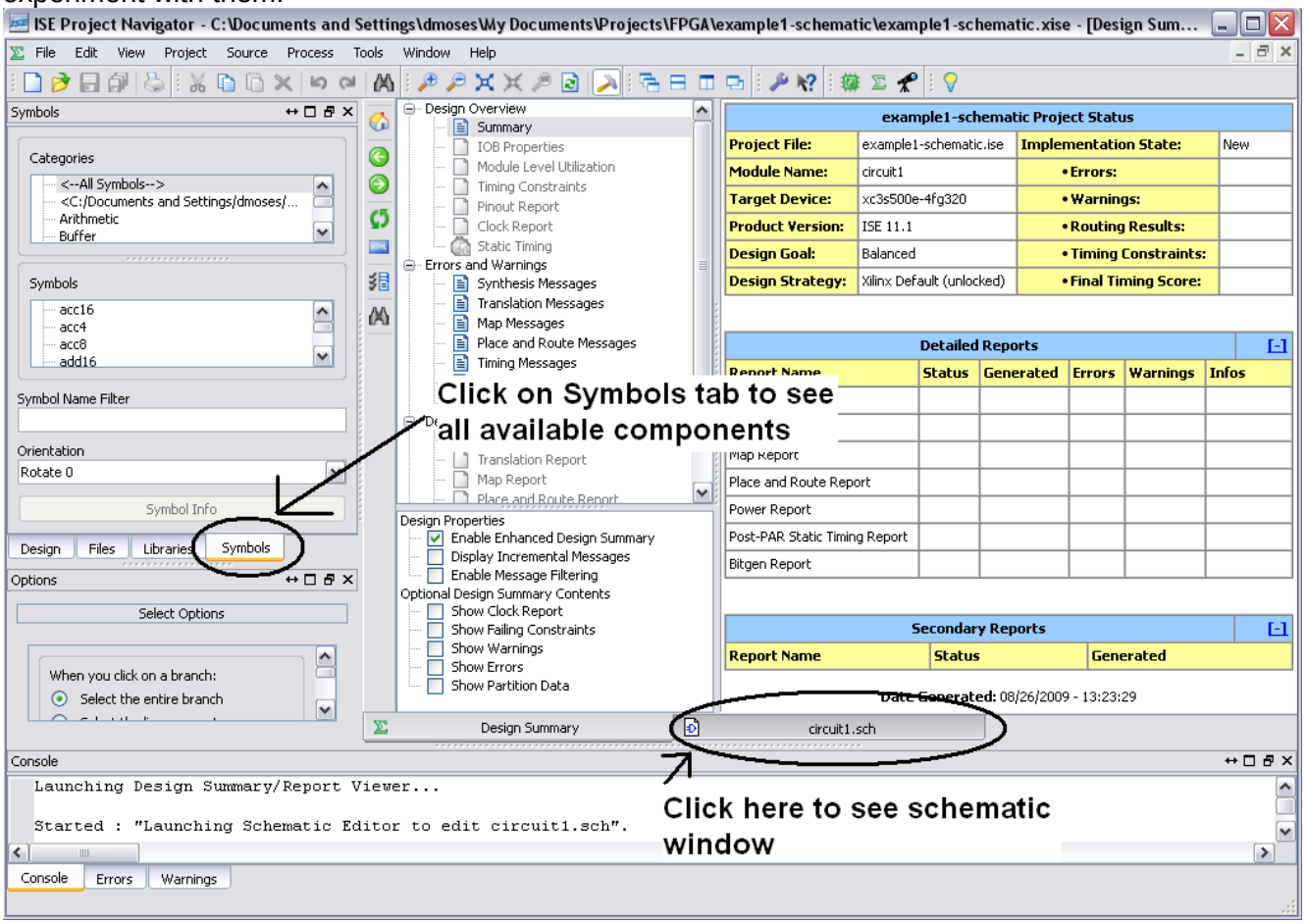


Basic Schematic Capture

To create a new schematic, right-click on the target device and select “New Source” from the drop down menu. This brings up the “Select Source Type” dialog box allowing you to define the new source file. Select “Schematic” from the list of source types, and enter a file name and directory in the provided boxes, check the “add to project” box, and click next to bring up the New Source summary box. Click “Finish” to bring up the schematic editor window (if the Design Summary window opens, click on the *filename.sch* tab at the bottom of the editor window – see figure below).

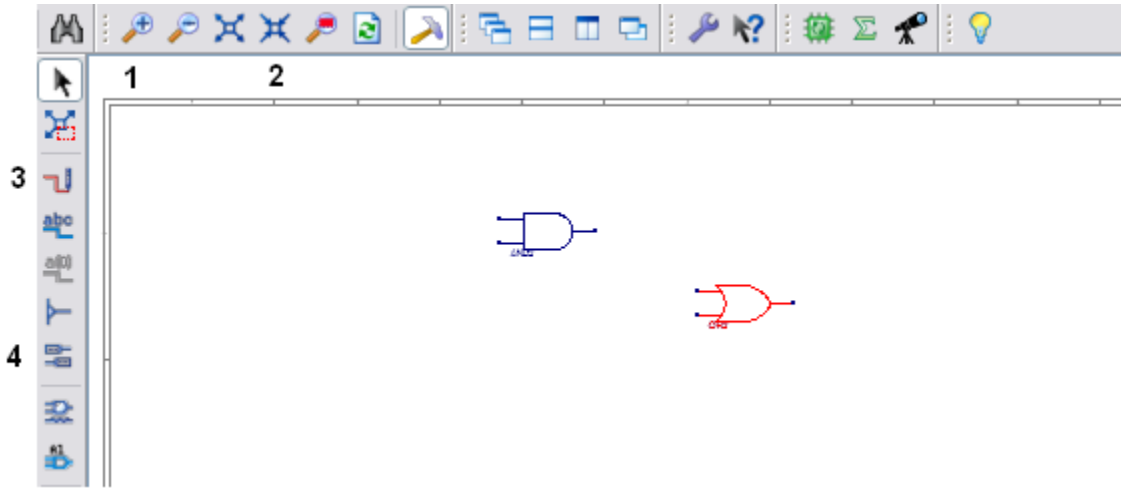
The schematic editor is simply a blank palette to which shapes (representing circuit components) and lines (representing wires) can be added. The schematic tool can be used effectively using tool-bar buttons or pull-down menu choices. In general, the tool-bar buttons and pull-down menus offer the

same functions, but the pull-down menus offer some unique features; you are encouraged to experiment with them.



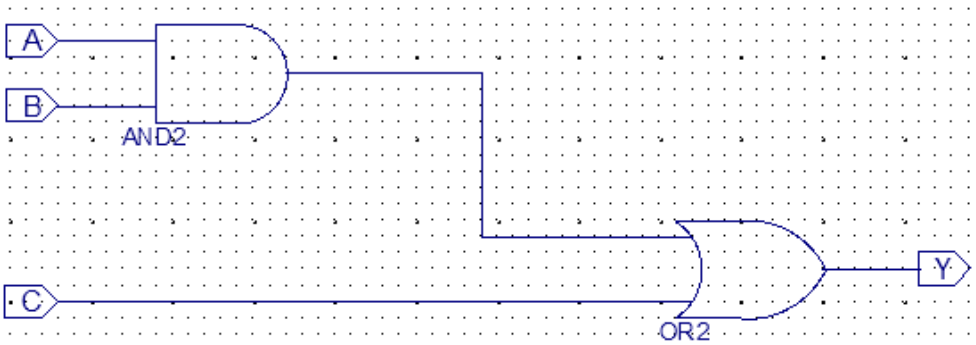
To draw a schematic, components must be added and interconnected with wires. To add components, click the *Add Symbol (or component)* tool-bar button to cause the component library to be displayed in a menu on the left of the schematic entry window. The components shown in the menu depend on which *device family* was selected in the new project setup window – different families use different schematic symbol libraries. Under *Categories*, select "Logic", which restricts the *Symbol* menu to displaying only the more basic logic components like AND and OR gates. To add a particular component, scroll through the menu to locate it, or type its name in the box at the bottom of the menu. Components can be moved after they have been added, so it's generally a good idea to add all needed components first, and then to rearrange them into a neater circuit once they are all present. Selected components can be "dragged and dropped" onto the schematic drawing palette.

The following figure contains descriptions for the basic tools necessary for creating a basic digital circuit using schematic capture:



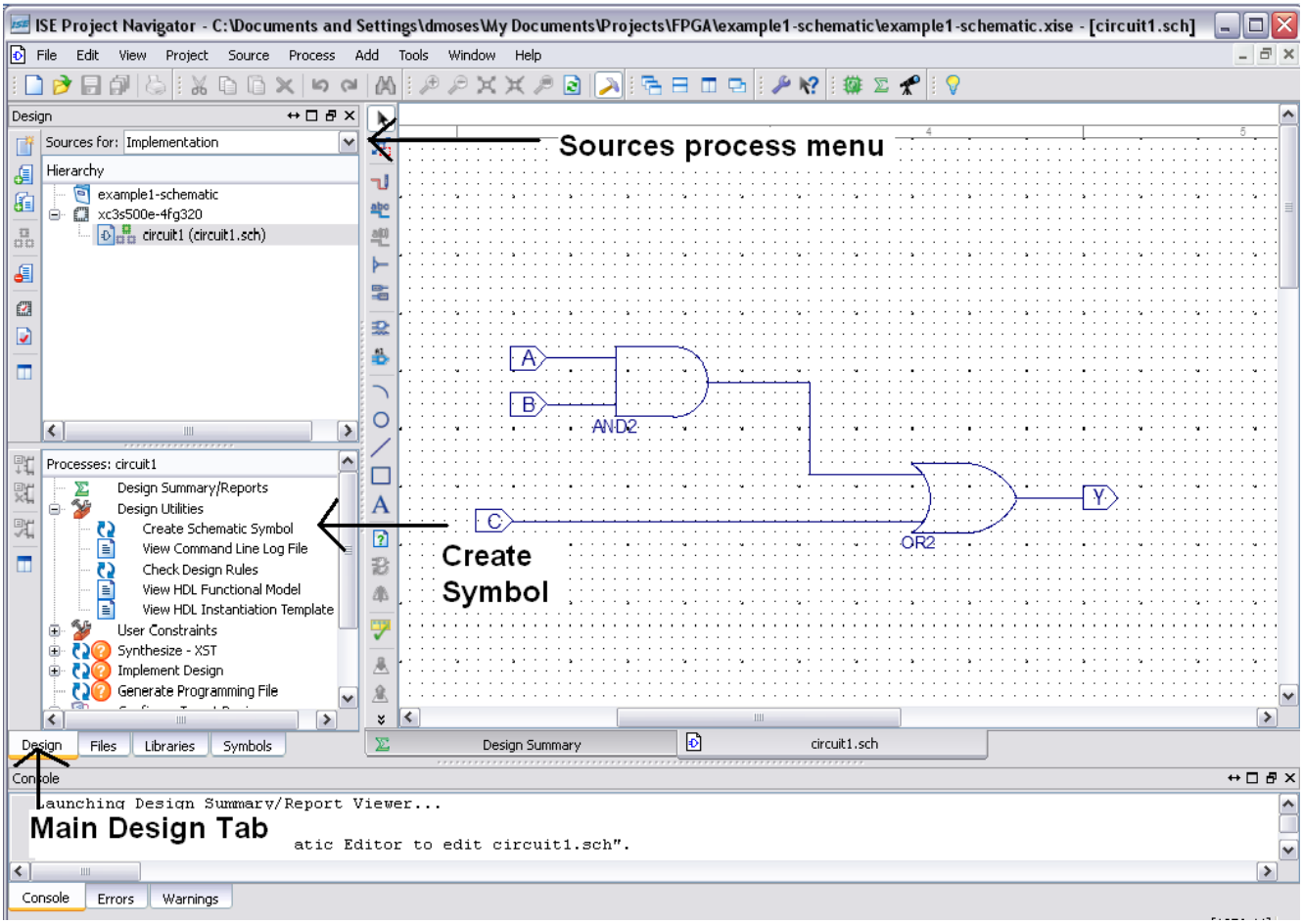
1. The Magnifying Glass icon indiscriminately zooms to the center of the schematic.
2. The Zoom box icon is used to draw a box using the mouse to magnify a specific area of the schematic.
3. The Wire-add tool icon places cursor in wire-add mode.
4. The Add I/O marker tool icon places cursor into add I/O marker mode.

In this example, we'll create the circuit specified by: Y = (A·B) + C. This circuit requires one **and2** gates, and an **or2** gate. These components can be added the schematic by selecting them from the component menu as described, and then dragging-and-dropping them to place them on the schematic palette. Once the needed components are in place, wires can be added by pressing the *add wire* tool button, and then clicking on the source and destination component pins. When connecting components with wires, be sure some amount of wire exists between all component pins. Note that it is difficult to tell whether a wire segment exists between the inverter and the AND gate. In general, enough wire should be used so that it is obvious that the pins are not directly touching. Wires can be ended in "space" by double clicking the screen area where the wire is to be terminated. Labels can be added to wires by selecting the *Add Wire Name* button, and then selecting the wire, or by double-clicking on the wire. Circuit inputs and outputs (as opposed to internal nodes) are identified by selecting the *Add I/O marker* button and clicking on the end of each input or output wire. Unique default names are automatically assigned to I/O markers. To change the default names, click on the *select cursor* toolbar button (or hit escape, which always enters *select* mode) and then double-click on each I/O marker in the schematic. In the window that appears, you can enter a new name in the name field. Save your schematic when it looks like the picture below.

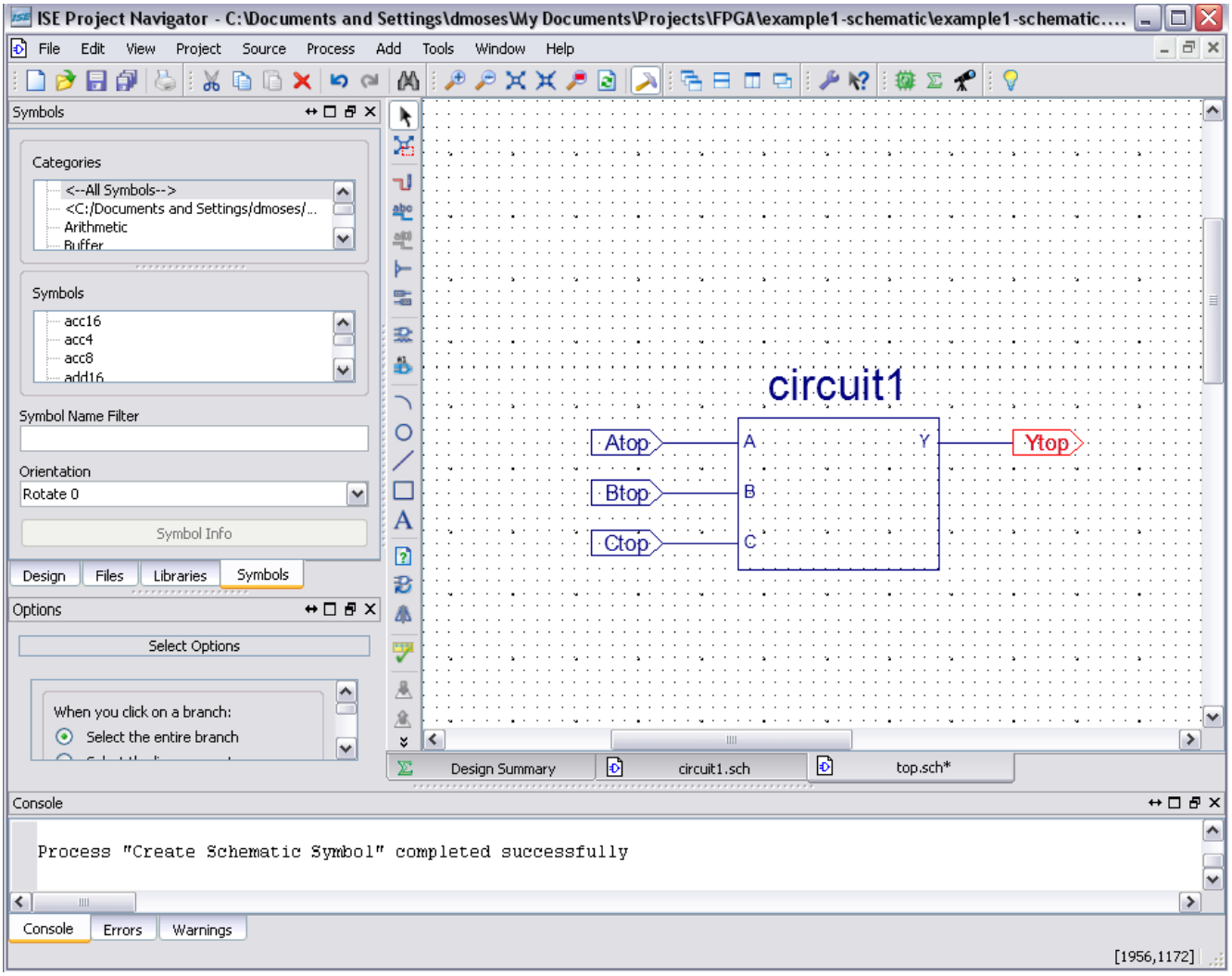

A
B
AND2
C
OR2
Y

Hierarchical design

For all but the simplest circuits, schematics can be made much more readable if certain well-defined parts of the circuit are grouped inside of a "wrapper" called a **macro** or **symbol** (just like in computer programming, where often-used code is placed inside of a subroutine). When creating a macro, it is important to make sure all inputs and outputs have I/O markers, and that all I/O markers are named. These names will appear as pin labels on the macro symbol. A macro can be created from any schematic page, and everything on the schematic page will be placed inside the macro symbol. To create a macro for a given schematic source, select *Synthesis/Implementation* in the Sources Process Menu at the top of the Sources window, and select the *Sources* tab at the bottom of that same window. In the Processes window, select the process tab, and then double-click the "Create Schematic Symbol" process. The screen below shows the key points.



After a macro has been created, it is added to your project and it can be added as a component to any new schematics. To see your symbol, you must create a new schematic and add your symbol to it. To do this, right-click on the target device and select "New Source" to create a new schematic file as before. When the schematic opens, click on the Symbol Tab at the bottom of the Design window. The directory where you stored your project will appear in the Categories pull-down menu. Select your directory, and all schematic symbols you have made will be available in the symbols list. Select the circuit macro name in the symbols pull-down menu and select the Add Symbol hot button. You can now drop your new symbol into the schematic. Add I/O ports and wires to the schematic, and save your work. Using just these basic methods, schematics for circuits of arbitrary complexity can be created.


ISE Project Navigator - C:\Documents and Settings\dmoses\My Documents\Projects\FPGA\example1-schematic\example1-schematic....
File Edit View Project Source Process Add Tools Window Help
Symbols
Categories
<--All Symbols-->
<C:/Documents and Settings/dmoses/...
Arithmetic
Buffer
Symbols
acc16
acc4
acc8
add16
Symbol Name Filter
Orientation
Rotate 0
Symbol Info
Design Files Libraries Symbols
Options
Select Options
When you click on a branch:
Select the entire branch
circuit1
Atop
Btop
Ctop
A
B
C
Y
Ytop
Design Summary
circuit1.sch
top.sch*
Console
Process "Create Schematic Symbol" completed successfully
Console Errors Warnings
[1956,1172]

Basic Logic Simulation

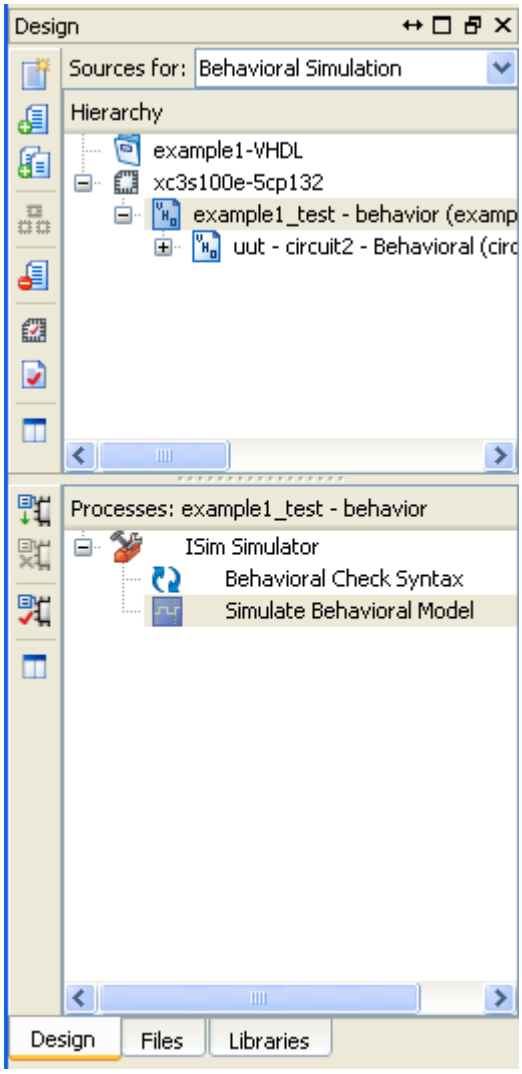
A logic simulator allows a designer to observe circuit outputs in response to all combinations of inputs before the circuit is implemented in hardware. Simulating a circuit is perhaps the best technique an engineer can use to ensure that all required features are present, and that no unintended behaviors are present. For larger designs, simulation is far cheaper and far less error prone than designing and testing a hardware prototype. If errors are observed in the simulator's output, the circuit can easily be corrected and re-simulated as often as necessary.

The simulator requires two kinds of inputs: the circuit description source file, and a set of stimulus values that define all input logic inputs for the duration of the simulation. The circuit description source must be an HDL file; if a schematic source is created, an HDL file is automatically generated whenever the schematic is saved. No matter what type of source file is used to describe a circuit, the designer must define the stimulus inputs.

The simulator functions by dividing the overall simulation into very small time steps (typically 10ps, but this value can be changed by the user). At each time step, the simulator finds all signals that have changed during the preceding time step, and processes those signals as dictated by the circuit's HDL source file. If output signals must change as a result of that processing, then changes to these signals are "scheduled" for a later time step (signal changes are scheduled for a later time step because signals can't change voltage values instantaneously).

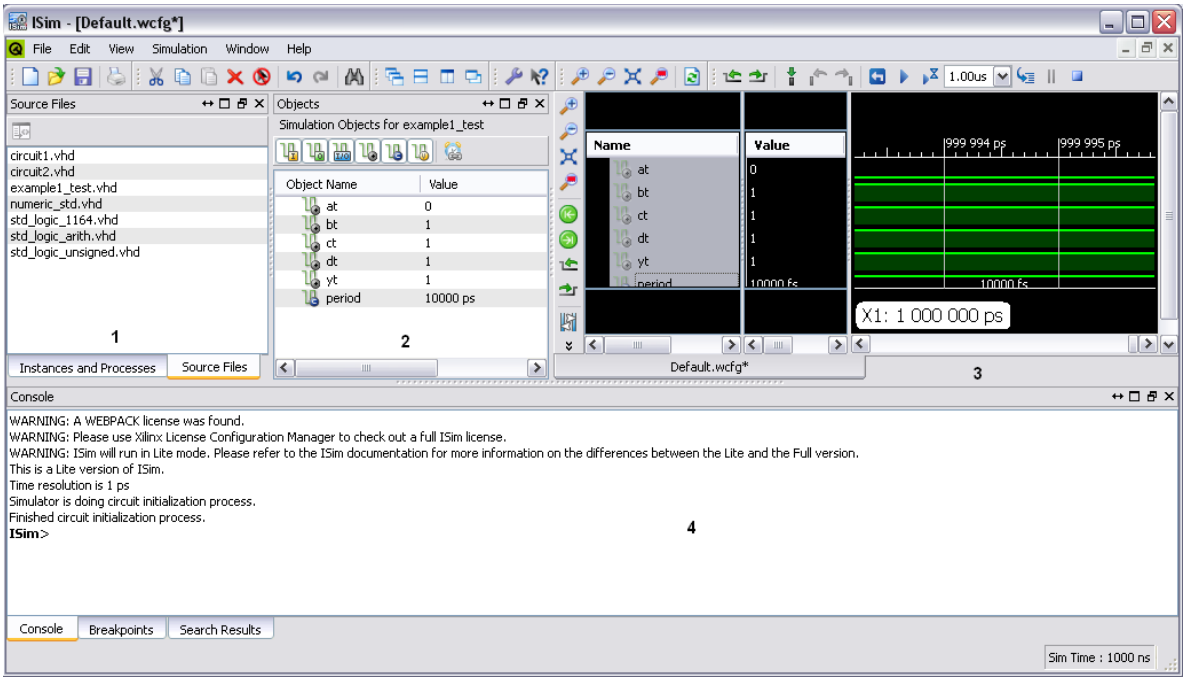
Different simulators provide various methods for designers to define input signals over time. Most simulators provide at least three methods, including a graphical interface, a text file based interface, and a command line interface. Any of these methods can be used with the ISE simulator included with the Xilinx ISE/WebPack CAD tools. Graphical interfaces are most useful when defining small numbers of inputs (up to 20 or so) that require relatively few changes over time (e.g., each of the 20 signals might need 20 or 30 changes between '0' and '1'). When dealing with a greater number of input signals (possibly numbering in the hundreds), or a greater number of signal changes over time (possibly in the tens of thousands), a graphical interface is too cumbersome. In this case, a text file based interface is used. The third method using the command line interface is most useful when changing a few signals once or twice to make some quick adjustments to the end of a graphical or text file based simulation.

The ISE simulator is a state-of-the-art tool that has many features to assist engineers in creating stimulus inputs, editing circuit descriptions, and analyzing circuit outputs. A document (dealing with creating HDL source files) will present creation of text-based stimulus files. For now, you will be given stimulus files that you can run on your own circuit.



Running a simulation

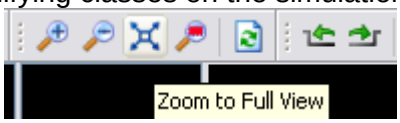
Running the Simulate Behavioral Model process will cause the ISim Simulator Window to appear.



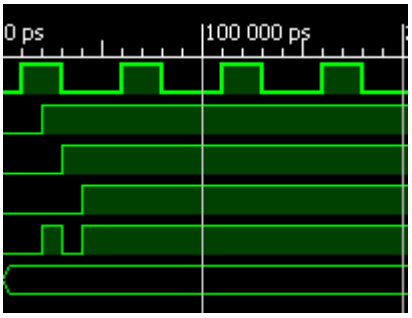
Some features of this window include:

1. Sources panel where source files to be viewed can be selected.
2. Objects panel where different signals can be added to the simulation.
3. Simulation panel where the state of signals can be observed.
4. Console panel.

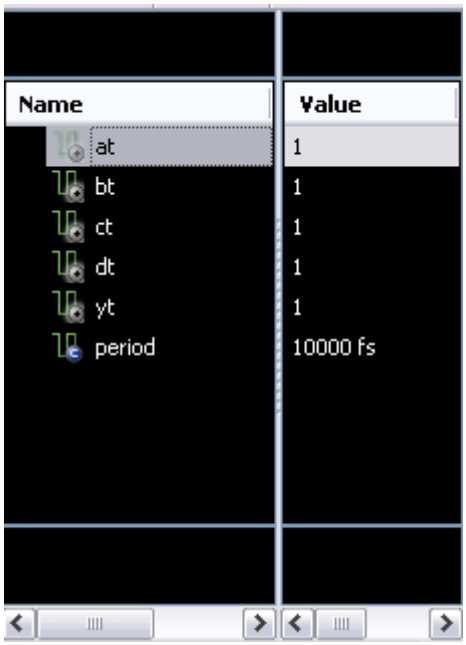
We will first use the Zoom to Full View tool to observe the entire full view of the simulation, which is located right beside the set of magnifying classes on the simulation panel toolbar.



This will bring the useful part of the simulation into scope. Use the magnifying glass with the plus sign to zoom in further on the useful part of the simulation as follows:



On the left side of the simulation panel there are columns labeled Name and Value:



For a given item on these columns, you can right click and choose options to delete, rename, or change the color of the signal color.
You may also use the scroll bars to observe the simulation at different times as well as observe more signals if you have a larger design.

The final step in the design process is to download the .bit file for your completed design as discussed in Module 2.

Appendix C: Programming your circuit board: connecting circuit nodes in your CAD project to physical pins and circuits on your board

After you have created a synthesizable circuit in the VHDL or schematic CAD tool, and before you can program that circuit into your board, you must associate signal names in your design with physical pin connections and circuits on your circuit board. This is required because I/O devices on your board (like buttons, switches, and LEDs) are physically tied to certain pins on the FPGA device, and the CAD tools must be told which pins are connected to which devices.

The Xilinx tools use an "User Constraints File" (.ucf file) to map circuit node names in source files to physical pins on the circuit board. If no .ucf file is present in a project folder, the Xilinx tools will randomly assign I/O port nodes to physical pins.

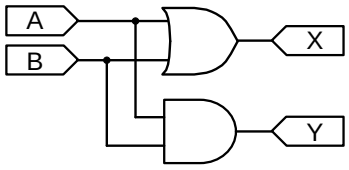
There are two ways to create a .ucf file in the Xilinx tools: a new module can be created and added manually (right click inside project navigator and select Add New Module -> User Constraints File), or the User Constraints processes can be run (double-click on the Edit Constraints process in the Processes window). In practice, manually creating a .ucf file is probably easier for small designs.

The .ucf file contains entries that specify pin assignments as shown below. You must create these entries, by entering signal names from your source files, and assigning those signals to specific pin numbers. Pin numbers for all I/O devices and connectors attached to the FPGA can be found in the board's reference manual and/or schematic.

The format of a pin mapping statement in a .ucf is as follows:

NET "<I/O name>" LOC = "<Pin Name">;

As an example, a .ucf file that will map the inputs and outputs shown in the logic circuit below to slide switches and LED's on the Basys board is provided (the # sign starts a comment field).




```
NET "A" LOC = "P38";  # maps circuit node A to Slide switch 0
NET "B" LOC = "P36";  # maps circuit node B to Slide switch 1
NET "X" LOC = "P15";  # maps circuit node X to LED 0
NET "Y" LOC = "P14";  # maps circuit node Y to LED 1
```

As a second example, the signals shown in the VHDL port statement below can be assigned to FPGA pins on the Nexys2 board using the .ucf file shown (note that the input A is a single bit wide while the output B is a four-bit wide bus).
.

```
Port( A : in std_logic;
      B : out std_logic_vector (3 downto 0));
```

```
NET "A" LOC = "G18"; # maps signal A to slide switch0
NET "B<0>" LOC ="J14"; # maps signal B(0) to LED0
NET "B<1>" LOC ="J15"; # maps signal B(1) to LED1
NET "B<2>" LOC ="K15"; # maps signal B(2) to LED2
NET "B<3>" LOC ="K14"; # maps signal B(3) to LED3
```

After creating the .ucf file, a .bit file can be generated and then download to your board.