

Project 1: Analysis and Simulation of a Simple Sequential Machine

Instructor Dr. Drew Phillips

Created by: Dr. Eylem Ekici

This project assignment assumes you are familiar (at a basic level) with the elements of the *Xilinx ISE* software. If you are not familiar, review the steps in the Practice Project regarding opening a new project, creating schematics in the schematic editor, setting up inputs for functional simulation and doing the simulation. You may use departmental laboratory facilities or your own computer to complete this project.

Create a Project and invoke the Schematic Editor

Invoke the Project Navigator, and create a new project by selecting **File** → **New Project** with the name **ECE3561Proj1**. In the New Project window, specify *Schematic* as the *Top-Level Module Type*. After clicking *Next*, specify *XC9500 CPLDs* as the *Device Family* and *XST (VHDL/Verilog)* as your *Synthesis Tool* and *Preferred Language* as *VHDL* from the pull-down menus. Please note that *Xilinx* cannot read blank space in directory names such as “*My Documents*” and do not save *Xilinx* project files on Desktop.

We will not create new sources for this project at this point. Continue clicking *Next*, *Next*, *Next* and then *Finish* to end the new project setup process.

To create new source, click on **Project** → **New Source**. In the pop-up window, select the entry *Schematic* and give the name *top* in the *File Name* field. Click *Next* and in the next window click *Finish*. Once you do that a blank schematic *Xilinx - ISE - [top.sch]* will open in the Schematic Editor window.

Add Inputs and Outputs

Start by adding six I/O Markers to the design. Click on **Tools** → **Create IOMarkers**. A window will pop up, requesting you to specify the input and output port names. Specify inputs as **INPUT**, **PRE**, and **CLK** and outputs as **Q0**, **Q1**, and **RCO**. Separate the IOMarker names by commas. This will add 3 inputs and 3 outputs with wires attached to them in your schematic.

Now add input buffers corresponding to each input. In the left side of your schematic editor window, under the symbol sub-window called *Categories*, click on the entry *IO*. If you don't see the sub-windows, click on **Add** → **Symbol**. Now click on the

entry *ibuf* under the *Symbols* sub-window. This will select an input buffer. To place the buffer on the schematic, just click the mouse in the schematic sheet to place the part. Repeat this for all inputs. Add the output buffers to the outputs. Click on the *obuf* entry under the *Symbols* sub-window and repeat the same procedure as you did for the input buffers.

The Schematic Editor is now in symbol placement mode. To get back into selection (or pointer) mode press ESC key (escape key) and the mouse pointer becomes normal again. You can select symbols and move them around in this mode only. To move a symbol, click on the symbol and move it to the desired location while keeping the left mouse key pressed down.

Place some devices

Place two *and2* devices, two *inv* devices, and two *or2* device. These devices can be found under the *Symbols* sub-window, when you click on the entry *Logic* under the *Categories* sub-window. Place two *fjkrse* devices. This can be found under the Category *Flip_Flop*. Now add a *d3_8e* device in the design. This device is found under the Category *Decoder*. Your schematic is as shown in the figure.

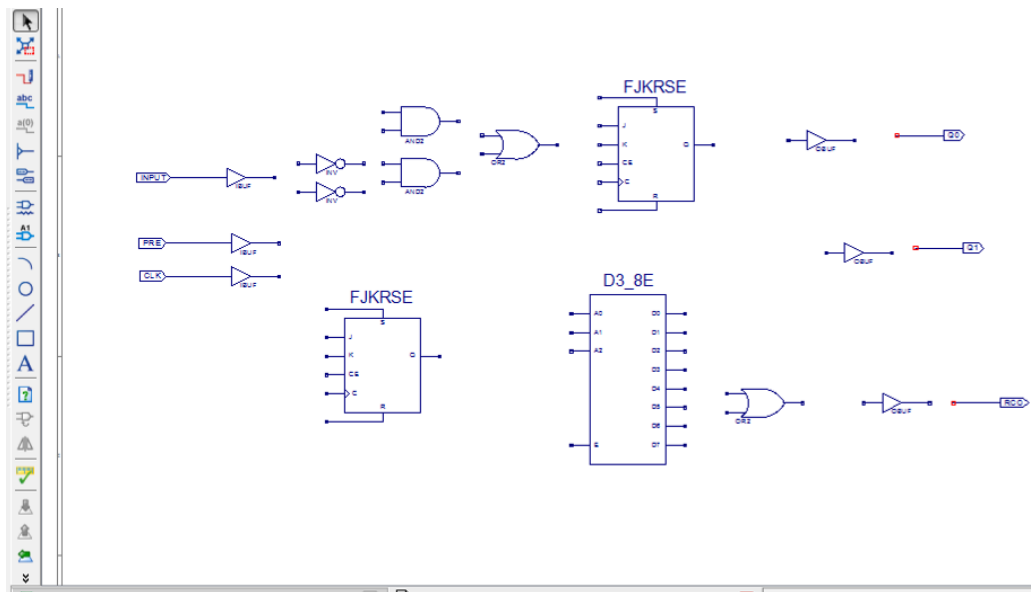



Figure 1: Schematic Editor window

You don't have to place the items exactly as shown there. If you need to move an item after placing it in the design, click the arrow-shaped tool button at the top of the toolbar on the left edge of the window. It allows you to select one item or a group of them.

Add some wires

Now you will 'wire up' the circuit as shown in Figure 2. Click the tool button  to enter the wiring mode. Begin a wire by clicking on one of the ends of a device in the sheet, and terminate the wire by double-clicking on its destination. Wire the logic using this technique. To route a single signal to multiple destinations, complete one connection, and then click on this wire to start the other connections. You can delete a wire by selecting it and pressing the delete key. You can reroute a wire (sometimes with difficulty) by clicking and dragging it around. The final design layout is in Figure 2.

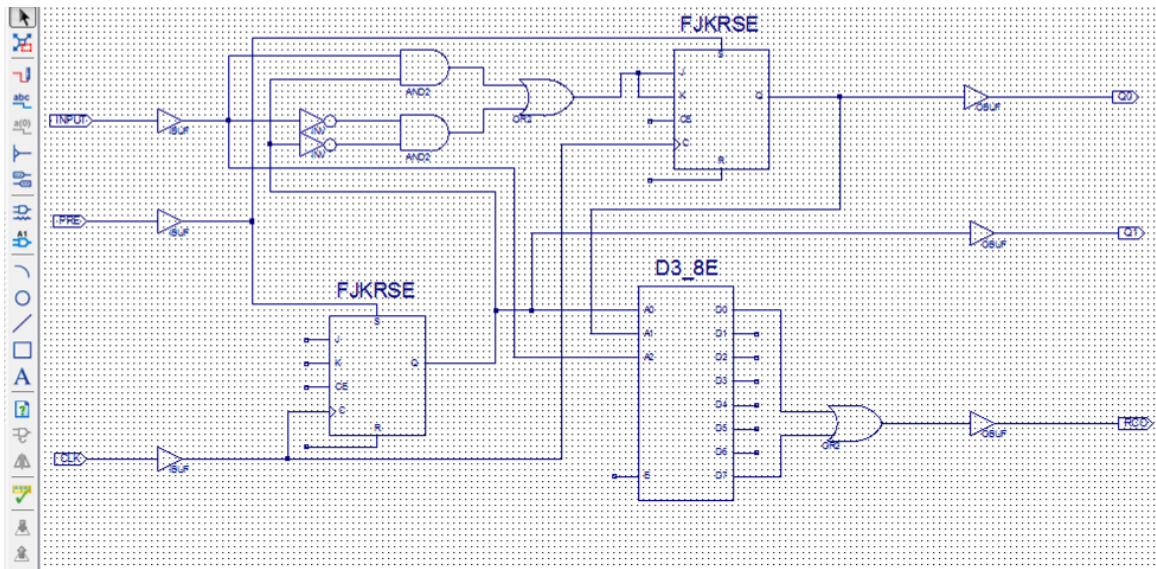


Figure 2: Schematic Editor window with partially-wired circuit

Note

In the schematic shown above, only at the positions where there is a blue dot, is a connection between wires. Otherwise, wires that cross each other don't have any connection. For example, the input **CLK** is connected only to the **C** inputs of the flip-flops *fjkrse*. Even though this connection passes through the wires from the **A0** and **A2** inputs of *d3_8e*, there is no connection between them. The input named **INPUT** is actually an input in your circuit; don't change that.

Add constant signals

The remaining dangling inputs need to be connected to ground (logic 0) or high (logic 1) signals. This is done by clicking on the *General* item in the *Categories* in your Schematic Editor window. Now click on the entry named *gnd* in the Symbols sub-window. This selects ground (logic 0). You can click to place the symbol wherever you like. To place high (logic 1) select the entry named *vcc* from the device list and click on the required location in the schematic to place it. Be careful to attach all dangling inputs. The **R** (reset) inputs of flip-flops should be tied to *gnd*. The **E** (enable) input of the decoder, the **CE** (clock enable) lines on the flip-flops, and the **J K** inputs on the lower left flip-flops should be tied high. The final schematic is given in Figure 3.

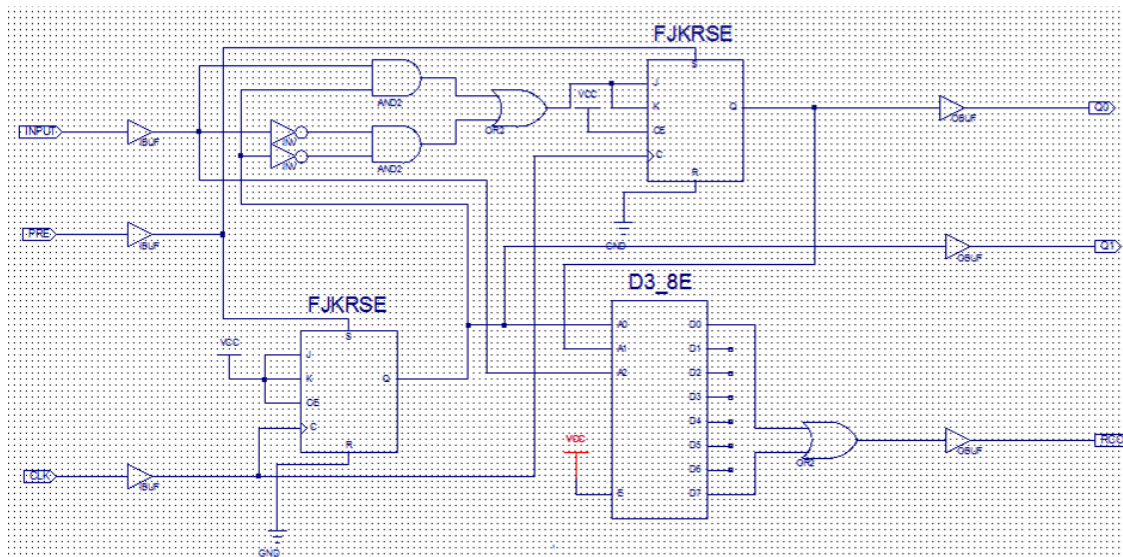


Figure 3: Final design

Checking Errors

Click on Tools → Check Schematic. If there are errors, they will be displayed in the Transcripts window at the bottom of Project Navigator. If there are no errors, then the Console tab will display *No error or warning were detected*. Note: successful passage of the integrity test does not mean the circuit is correct!

Functional Simulation and Analysis

Now you can invoke a functional simulation of the design. This simulation assumes ideal timing and does not reflect the realities of finite propagation delays and setup/hold times. In your Project Navigator window, click on your schematic file *top (top.sch)* to make it active. Now select Project → New Source. In the window that opens up select the option *VHDL Test Bench*. Specify a name in the *File Name* field and click on *Next*. In the following window click on *Next* and then finally click *Finish*. We choose the file name *wave* in the example illustrated in Figure 4.

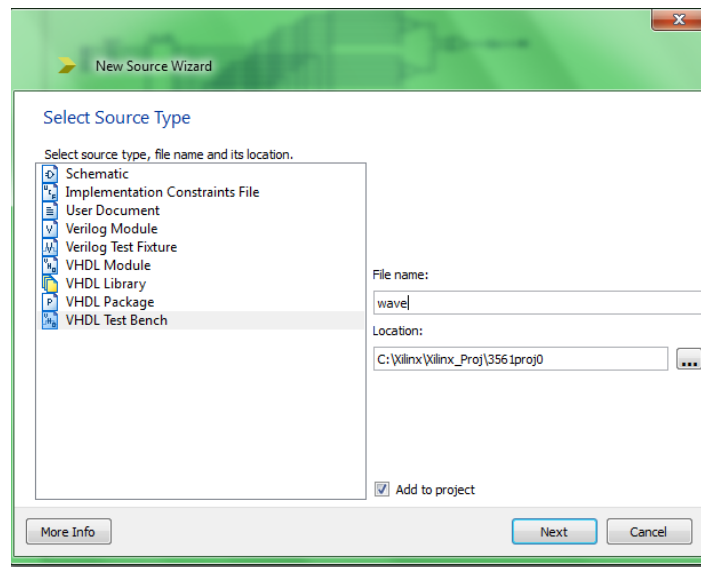


Figure 4: New Source window

Assigning Input Values

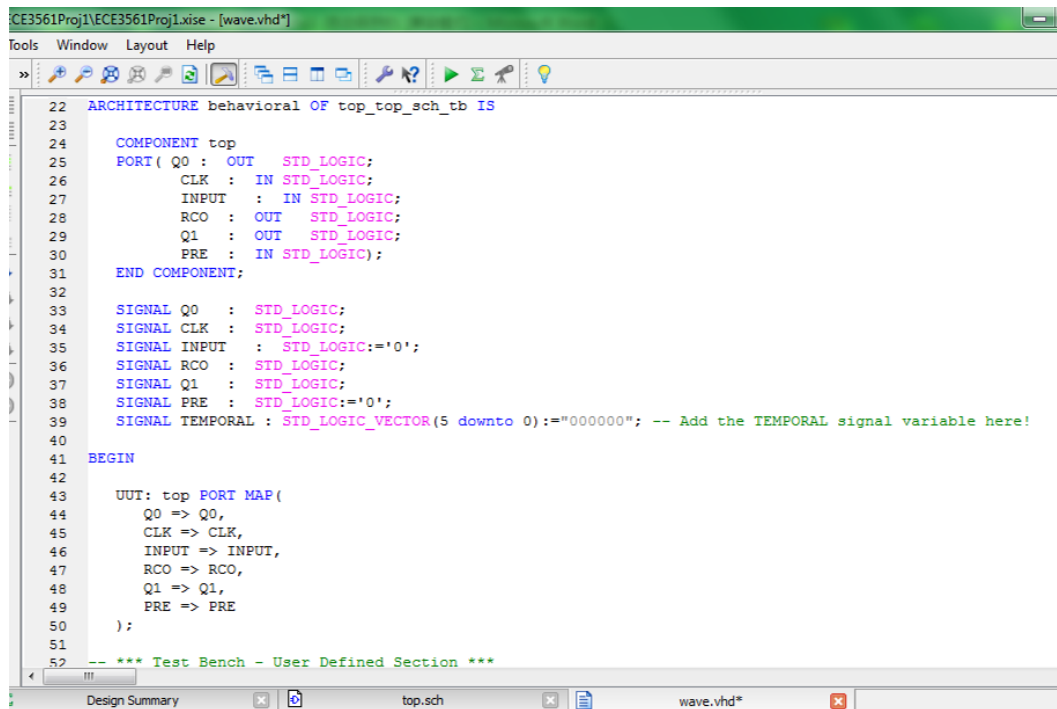
This will open the VHDL Test Bench *wave.vhd*. Go along the text and define a 6-bit long signal variable *TEMPORAL* to represent a timer, i.e., the timer increases from 0 to 63 when *TEMPORAL* loops from the state (*TEMPORAL*="000000") to the state (*TEMPORAL*="111111"). Specifically, the signal *TEMPORAL* is defined as follows:

```
SIGNAL TEMPORAL : STD_LOGIC_VECTOR(5 downto 0):= "000000";
```

For an illustration of signal definition, please refer to line 39 in the Editor window in Figure 5. In the above code, the signal *TEMPORAL* is initialized to 0.

Now assign a zero logic level to **PRE**. We also initialize the **INPUT** signal to zero logic level. Specifically, we have the following codes (also illustrated in Figure 5):

```
SIGNAL INPUT   :      STD_LOGIC:= '0';  
SIGNAL PRE     :      STD_LOGIC:= '0';
```

Figure 5: Define Signal *TEMPORAL*

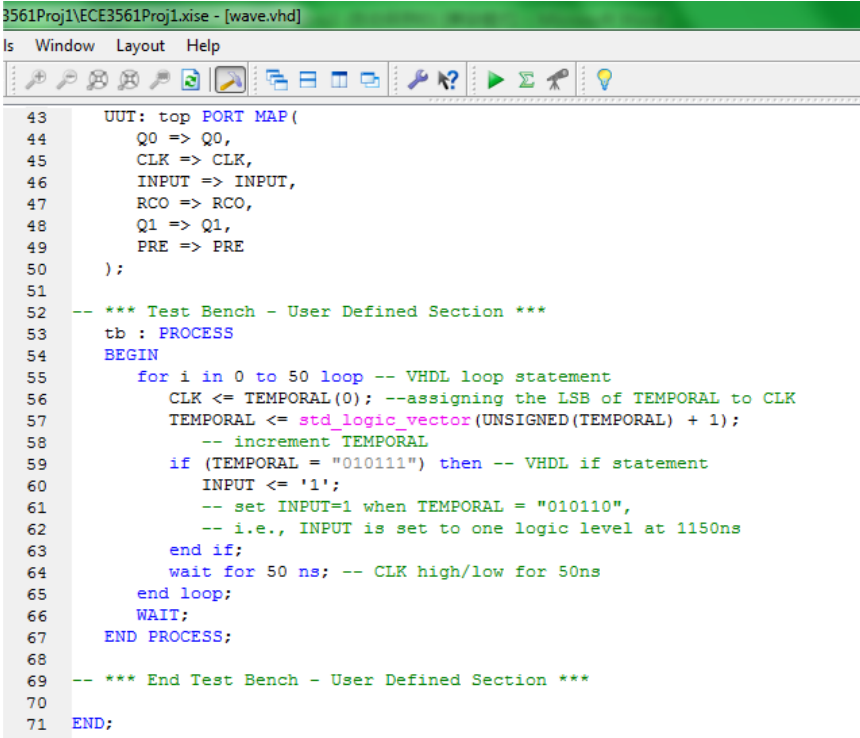
Now we model the CLK signal using *TEMPORAL*. We let CLK be the LSB of *TEMPORAL*. Thus, CLK switches its state (between '0' and '1') with each increment of *TEMPORAL*. Specifically, in the *wave.vhd* file, we find the *User Defined Section*, and copy and paste the following codes:

```

tb : PROCESS
BEGIN
    for i in 0 to 50 loop -- VHDL loop statement
        CLK <= TEMPORAL(0); --assigning the LSB of TEMPORAL to CLK
        TEMPORAL <= std_logic_vector(UNSIGNED(TEMPORAL) + 1);
        -- increment TEMPORAL
        if (TEMPORAL = "010111") then -- VHDL if statement
            INPUT <= '1';
            -- set INPUT=1 when TEMPORAL = "010110",
            -- i.e., INPUT is set to one logic level at 1150ns
        end if;
        wait for 50 ns; -- CLK high/low for 50ns
    end loop;
    WAIT;
END PROCESS;

```

We also assign one logic level to **INPUT** at 1150 ns. Since 1150 ns consists of 23 ("010111") clock periods, we let INPUT <= '1' when TEMPORAL = "010111". The complete codes for the User Defined Section are illustrated in Figure 6.



```

3561Proj1\ECE3561Proj1.xise - [wave.vhd]
Is Window Layout Help
43      UUT: top PORT MAP(
44          Q0 => Q0,
45          CLK => CLK,
46          INPUT => INPUT,
47          RCO => RCO,
48          Q1 => Q1,
49          PRE => PRE
50      );
51
52  -- *** Test Bench - User Defined Section ***
53  tb : PROCESS
54  BEGIN
55      for i in 0 to 50 loop -- VHDL loop statement
56          CLK <= TEMPORAL(0); --assigning the LSB of TEMPORAL to CLK
57          TEMPORAL <= std_logic_vector(UNSIGNED(TEMPORAL) + 1);
58          -- increment TEMPORAL
59          if (TEMPORAL = "010111") then -- VHDL if statement
60              INPUT <= '1';
61              -- set INPUT=1 when TEMPORAL = "010110",
62              -- i.e., INPUT is set to one logic level at 1150ns
63          end if;
64          wait for 50 ns; -- CLK high/low for 50ns
65      end loop;
66      WAIT;
67  END PROCESS;
68
69  -- *** End Test Bench - User Defined Section ***
70
71  END;

```

Figure 6: Codes for User Defined Section

Simulations

Now we are prepared to run the simulation via Xilinx's ISim simulator. Proceed with the following steps for a complete simulation.

Step (i): Click on **Simulation** over the Design Window and highlight the VHDL Test Bench created (*wave.vhd*).

Step (ii): Double-click on Behavioral Check Syntax on the Processes Window. A **green mark** should indicate the correct completion of the operation. If getting a red cross, check the errors, save and try again.

Step (iii) Right click on **Simulate Behavioral Model** and select **Properties**. In Process Properties window illustrated in Figure 7, change **Simulation Run Time** to 2500 ns and click OK. Double-click on **Simulate Behavioral Model**. The simulator will open, and the simulation results are illustrated in Figure 8 after appropriate zooming in or zooming out.

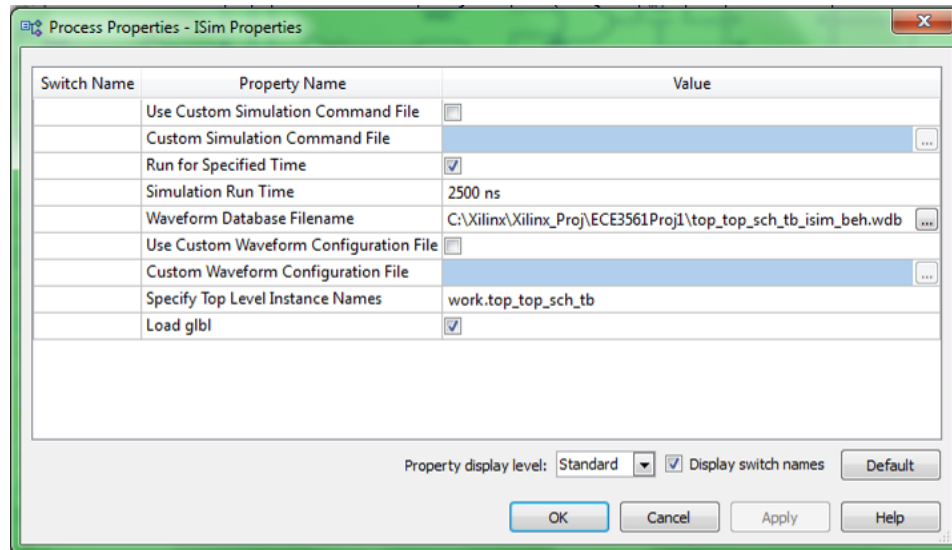


Figure 7: Process Properties window

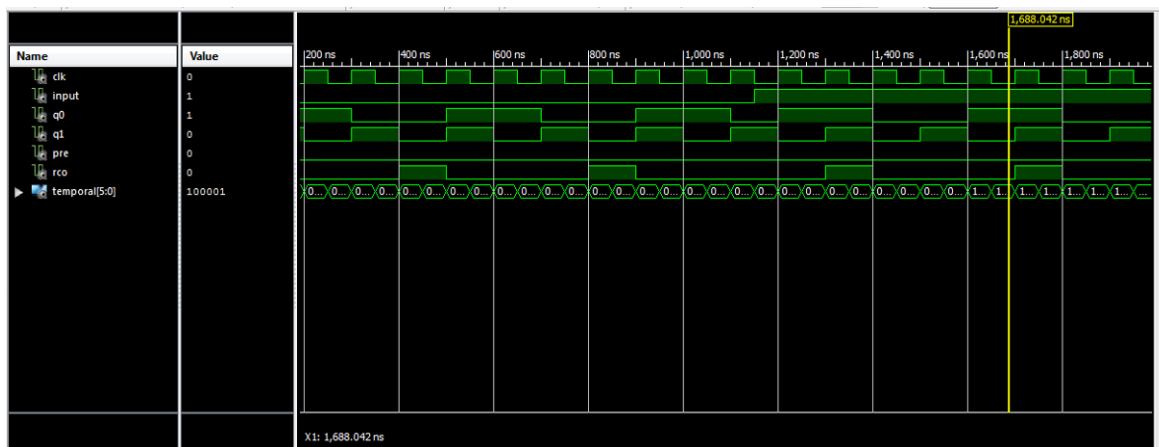


Figure 8: the ISim Simulator

You might be able to figure out what this machine does from this single simulation. Consider the two outputs **Q0** and **Q1** as a single 2-bit bus with **Q0** forming the MSB and **Q1** forming the LSB.

Problem 1: Run a simulation containing ten clock periods with **INPUT = 0**, followed by ten clock periods with **INPUT = 1**. What does this circuit do? Make sure you describe what the **PRE** and **INPUT** inputs do, and what the **RCO** output does. Attach a printout of the simulation waveforms and write on the printout as necessary to document your conclusions.

Problem 2: Based on your answer to Problem 1, suggest a technique to make the circuit operate "in reverse" given the same inputs. Modify the schematic to implement your technique, simulate the resulting circuit, and attach a simulation printout as in Problem 1. Did your modification work?

Note: If the definition of "in reverse" operation is unclear. Please clearly specify your definition of "in reverse" operation and the modification should follow your specification.

Problem 3: Using the analysis techniques covered in class, obtain a state/output table and a state diagram for this circuit. Is this a Mealy or a Moore machine? Why?

Timing Simulation

Remove the modification you added for Problem 2 before continuing.

Now you will use default parameters for the PLD devices to determine the maximum possible clock speed for this device. To do this in the *Xilinx* environment, you must take your design all the way through the "implementation" process and then run a timing simulation.

In the Design sub-window and the design tab, select *Implementation* for *View* and select the schematic file *top* (*top.sch*). You can analyze your design by looking at two analysis reports as follows.

In the Processes sub-window and the Processes tab, click on the (+) sign in front of *Implement Design*, Double-click on *Synthesize – XST*. In the *Design Summary* window, you can view *Synthesis Report* under *Detailed Reports* (illustrated in Figure 9).

Under *Implement Design*, Click on the (+) in front of *Optional Implementation Tools*, and double-click on *Generate Timing*. In the *Design Summary* window, you can view *CPLD Timing Report* under *Design Overview*.

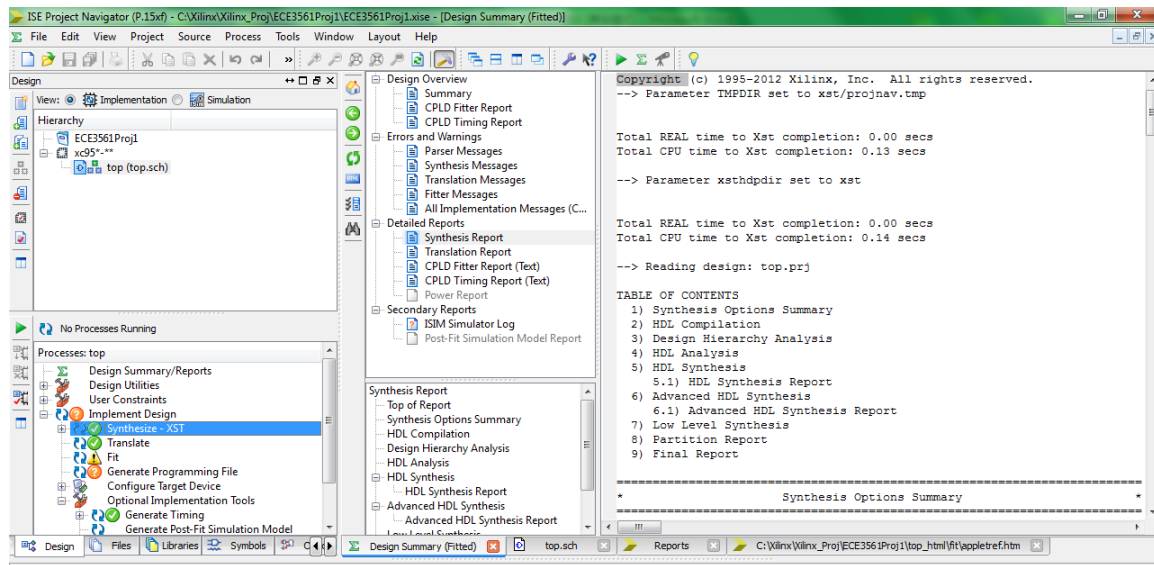


Figure 9: Process for Source window with Synthesis Report selection

Problem 4: According to the Timing Report analysis, what is the maximum clock speed for this circuit? Attach the Timing Report.

Report

The report should be typed. Be sure to include:

- Title page including course number, project number, your name.
The content should start on a separate page.
- Answers to the questions in Problems 1-4
- A printout of the schematic of your circuit (original and modified)
- A trace output from each simulation (waveforms)
- State/output table and state diagram for Problem 3
- Timing Report in Problem 4