Computer Structure Lab - Fall 2000

Dr. Guy Even, Marko Markov, and Eran Eidinger

Handout #0: The Xilinx Foundation Software

The purpose of this lab is to get the students acquainted with the main tool in the lab, the Xilinx Foundation software.

1 General Background

The lab consists of: (a) software - mainly the Xilinx Foundation Software, and (b) hardware - a special computer called RESA built for the purpose of a hardware lab. The RESA computer consists of three boards: a CPU board, a Memory board, and an I/O controller. These three boards communicate via a bus. A 4th board, called the I/O Interface, is in your PC. The PC can therefore communicate with the Memory board and the CPU board via the I/O Interface board (which happens to be connected to the RESA's I/O Controller)

The CPU board is a special type of board. It does not contain a microprocessor in the usual sense. Instead, it has an FPGA (Field Programmable Gate Array) chip. Loosely speaking, an FPGA is a device that can be programmed (or configured) while it is on the board (i.e. no need to take it out of the board and plug it into a special programming device). The FPGA's used in the RESA can be programmed to function like any circuit that has at most $25,000$ gates (this is not an accurate figure). The programming of the FPGA in the RESA is done by executing a special program on the PC which instructs the FPGA to configure itself according to a "file" that is stored either in the PC or in the RESA's Memory board. Such a configuration file defines the functionality of the FPGA, which in our lab will be a simplified DLX CPU. The DLX CPU accesses the memory for fetching of instructions, and load/store instructions. We need, therefore, a means to store programs in the RESA's Memory board (which will serve as the main memory of the CPU). Copying of RESA programs from the PC to the RESA's Memory board is done using a special program on the PC.

In order to do all that, the students need to know the following:

- How to design a DLX CPU in detail (all the gates).
- How to use the Xilinx software to design the DLX.
- How to test the correctness of designs (i.e. simulation).
- How to create a configuration file for the FPGA.
- How to configure the FPGA.
- How to run the RESA and check if it is running properly using a monitoring program.

The first Lab is aimed to teach basic skills in drawing schematics, VHDL design and simulation. Other tasks such as: creation of configuration files (called HEX files), downloading of configuration files, and operation of the RESA will be taught later in the lab. This write-up attempts to give a full description to ease your work in the future.

2 Work flow

The Students' activity in the lab should follow these steps:

1. The first stage in the lab is called design entry. This is the stage in which you edit (or draw) the design using the software. There are two main methods in design entry: (a) schematic design: in this method a logical design is drawn using blocks (i.e. gates, flip-flops) and nets (i.e. wires). The software has a graphical interface which enables you to choose the type of block, name the ports of a block, and connect between the blocks.

   It never hurts to think before designing, but in the case of this lab it is highly recommended! You should have a clear picture of what you plan to design and how it will work. Design entry is time consuming. If you cherish your time, make sure you know what you are designing before you start design entry.

   A general advice is to use a top-down approach: First, draw the "big blocks" and the interconnections between them. Only then proceed to design the details of the blocks. Do not crowd your design with too many blocks, lest your designs will become unintelligible spaghettis. Add remarks to your designs so that you (and the grader) can figure out what you had in mind. In general, clear drawings require shorter explanations and help one find mistakes much faster.

2. The second stage is called simulation. We are actually referring here to logical simulation (or functional simulation) that ignores timing issues (which you will learn later are very important). The simulator works under the "zero propagation" delay model. Namely,all the propagation delays are zero. This can sometimes be a bit confusing (e.g. what signal causes a change in values?) Simulation is software that enables you to choose input values and view the expected output values. This is the first and most important method of testing your design. Typical errors that you are supposed to detect at this stage are: mistakes in connections of wires, wrong polarities (i.e. connection of signal x instead of $not(x)$), incompatible names of signals, etc.

3. The third stage is implementation. During implementation a configuration file is created from our design. This stage is fully automatic in the easy cases (i.e. small designs), and all you do is ask the software to create the configuration file (called a HEX file). In larger designs, the automatic process fails (for example, the delay of the critical path is larger than the clock period). Some guidelines need to be given to the software to assist it in fitting the design into the resources of the FPGA. The configuration file (which resides on the PC) can be copied to the RESA using a special software on the PC, called the RESACTRL - the RESA Control Software.

   After implementation, a simulation that does not ignore timing issues is possible. This simulation is called timing simulation, timing verification, or delay analysis. The main goal here is to see if the delay of the critical path is not larger than the clock period. We will elaborate more on this type of simulation later in the lab.

4. The fourth stage is running your design on the RESA. This stage is performed with the aid of the RESACTRL program. There are many options in this software, the simplest of which are writing to the RESA memory and reading values from the RESA memory. A DLX program can be executed by the following stages: copy program to RESA's

memory, start execution of DLX, and read RESA's memory to see the outcome of the executed program. More options which allow monitoring of signals and registers in your design will be explained later.

3 Assignment in Lab #0

The assignment in this lab is to practice design entry (schematic and VHDL) and simulation on a design which is fully given to you. Please do not submit your work. It will be assumed that you have successfully completed it within a week.

Detailed Instructions:

1. We want to draw a project which is identical to the `decoder` project which already exists in your PC. You may view the `decoder` project to see what your design should look like. Start Xilinx Foundation application. The project consists of a counter (designed using VHDL), an AND-gate, and a FLIP-FLOP. The decoder design (which uses a counter, gate, and flip-flop) is designed using schematic entry. A printout of the project appears in the end of this writeup.

2. Activate the simulator. Choose the signals using the menu option Signal $\rightarrow$ Add_Signals. In the "Chip Selection' window, double click to select the chip, then double click on the block the signals of which you want to select, and then on the right hand window, double clock to select the signals you want to watch. Choose the following signals:

   (a) Inputs: `CNT, RESET, CLK`

   (b) Outputs: `CNTOUT[3:0], PULSE_C-F, PULSE_D-0`.

   The stimulators (the input waveforms) are set in two stages. First, formulas are written. Then, each formula is matched with an input signal. Formulas are defined by selecting the menu option Signal_Add_Stimulators, and tapping the Formula Button, followed by double clicking the requested formula, typing the formula in the bottom row, and pressing Accept. Use the following formulas:

   (a) `f0 = L101H` to stimulate `CNT`. This means: Low for 101ns, then High forever.

   (b) `f1 = L2001H` to stimulate `RESET`.

   The Clock signal `CLK` is driven by using `b0` (this is simply a zero-one alternating signal). The matching between formulas and signals is done by selecting the signal and then selecting the corresponding formula.

   Note that the input signals transition only between clock ticks. The clock ticks occur in multiples of 100ns, and this is why the input signals change 1ns after the clock transition.

   Run the Simulation. The default presentation depicts the values of the busses in hexadecimal, but in reverse order (MSB on the right). A reverse interpretation of bus values is obtained by selecting the signal from the left window, and then choosing

the menu option Signal → Bus → Change_Direction. Change the direction of the `CNTOUT` bus.

Make sure that you understand the difference between the pulse output signals.

4 Xilinx Foundation 1.5 - Instructions

1. To start a new project: From menu option 'File', select 'New Project'. You are given a form to fill. Give a name to your project. Change the directory path to your working directory as given to you in the first lab. Additional fields in this form are: "Type" - select "Foundation Series v1.5"; "Flow" - check the 'Schematic' option; "Device Family" - choose XC4000E; "Part no," - choose 4025EHQ240. These additional fields need to be entered only once; in future projects the same values are used.

2. To add libraries to your project: From 'File' menu, select 'Project Libraries'. In our case, select the library 'xc4000e' from the 'Attached Libraries' window, tap the 'Add' button. Make sure the library is now displayed in the 'Project Libraries'. Do the same for 'lablib1' library. Tap the 'Close' button to exit. In future designs you will add the library 'lablib2' when required.

3. In order to start a schematic sheet, press on 'Schematic Editor' button (rightmost) in the 'Design Entry' square.

4. Using library elements: on the left bar of the editor, choose the 'Symbols toolbox' button (labeled with a gate icon), or select 'Symbols' from 'Mode' menu. A window labeled "SC Symbols" is opened. In this window, all the modules that exist in the different libraries of the project are listed. In order to enter one of them to your design, choose the component you want to draw (e.g. AND2 for a 2-inputs 'and' gate). Click it once with the cursor, drag the cursor to the desired point, and then click once again for insertion in that location. If you drag a component by mistake, press 'ESC' to cancel the selection. To search for a specific component, you can either use the scroll bar on the right of the window, or type the name of the component at the bottom of the window. Note that when you select a component from the list, its description is displayed in the lower part of the window. When you are done, close the 'SC Symbols' window by double clicking the minus sign at the top left corner.

5. To create your own component (i.e. macro): From the 'Hierarchy' menu, select 'New_symbol_wizard'. Tap the 'Next' button. Enter the new symbol's name. In the 'Contents' field select either 'schematic' or 'HDL code', according to the design method you want to use. In the following window, you should specify the input and output pins of your component. To add a new pin: tap the 'New' button, enter the pin's name, select the pin type (input, output or bi-directional), specify the bus range (if you are adding is a bus). Do the same again if you want to add more pins. Otherwise, tap the 'Next' button. In the following page, choose the "Label" for your component (in the "Reference" field) and mark a comment that explains the usage of this component. To finish the definition of the macro pins, tap "Next" and then "Finish". A symbol (i.e. block with pins) is now created for your component.

6. Editing an existing macro: A macro is not only its design but also its symbol (which can be regarded as an "envelope"). You can choose to edit only the symbol using the "Symbol_editor". Select the macro to be edited as follows. First, make sure you are in "select_and_drag" mode. You select this mode by choosing the menu option "Mode → Select_and_drag". Now, you can select the macro to be edited by double clicking the macro. The "Symbol_Properties" window opens. Choose the "Symbol_editor". There a several options now. You can do the following: Resize symbol - drag its sides or top or button using mouse. Moving pins - drag the pin to the desired location. Delete a pin- color it by clicking the left bottom of the mouse, then press the Delete key. Adding a pin- choose the menu option "Symbol → Add_Pin" and fill the "Add_pin" dialog box. To exit the Symbol Editor- click on "X" on top right, answer "yes" to save and press "OK" in "Symbol_Properties" window.

   To edit a macro itself: Choose "Hierarchy_push/pop" (on the left side) and double click to start editing the macro. Then edit the drawing or HDL file. These files can be accessed also directly by the menu option "File → Open" in the "Schematic _editor" window.

7. Connecting pins. Use the icon "Draw_wire" on the left side or the menu option "Mode → Draw_wire". Press the left mouse button at the starting point. Use the left button to add jogs (i.e. bending points) and to determine the end-point. Busses are connected using the icon "Draw_busses". Wires are connected to buses using the icon "Draw_bus_taps". First, touch the bus name, then the pins in the desired order. You can also connect one pin at a time and name the wire as desired by double clicking the wire in "Select_and_drag" mode.

8. Adding and removing sheets from a project. A project is a database that holds all the sheets. To add a new sheet to project you must be in the "Schematic_editor". Choose "File → New" to get an empty sheet which is automatically added to the project.

   To remove a sheet from project: In "Project_manager", select the file and choose "Document → Remove".

   To add a schematic to a project: In "Project_manager", choose "Document → Add", choose the document from the "Add window → Open".

9. From the Schematic Editor, activate the simulator by pressing the "SIM" button in the "Schematic_editor" or by choosing the menu option "File → Go_to_simulator".

10. Selecting the desired signals: In the Simulator: Signal → Add_signals mark the global signals on the left (signals selection window) by double click or by mouse and Add. For viewing specific signals, choose the chip and then the pins of the selected chip in the window on the right.

    11) Stimulating the signals: In the Simulator: choose the menu option "Signal → Add_stimulators", and select a desired signal in the left part of the simulator window. Match a stimulator to the signal by choosing one of the following options in the stimulator window:

    - 1= high, 0 = low;
    - Bc0= counter output (can be used as a clock) Bc1,Bc2,...,BcF = other outputs of the binary counter (used for simulation);

- NBc0,NBc1,...,NBcF = outputs of the counter after negation;
- formulas = a way of defining more complex signals (use Help to learn more about using formulas).

Start the simulator by pressing ON and then STEP and watch the signals.

5 Project DECODER