ECE369A Computer Organization
LAB 1

Objectives

- Implement basic datapath components
- Synthesis of the datapath components
- Integration of datapath components

Folders and Files Provided in the "Resource" Folder

1- InstructionFetchUnit: Templates for testbenches and design modules for Lab1. You will modify these modules and turn in. Submission instructions are provided below.
2-Vivado_Simulation_Synthesis: Tutorial for installing Vivado 2019.2.1 and running simulation and synthesis
3-Nexys-A7: Datasheet, pin assignment information
4-ClockDivider: Module to slow down the clock when displaying values on the 7-segment display on the Nexys-A7
5-Two4DigitDisplay: Module to interface with your designs to send two values to the 7-segment display
6-LAB1_helper.pptx: Top level diagrams for the InstructionFetchUnit
7-LAB1_Verilog_Recap.pptx: Lab1 relevant Verilog higlights
8-Verilog Synthesis.pptx: Lab1 relevant synthesis higlights

Other Resources

For all Verilog and Synthesis related guides, tutorials, refer to D2L → Content → Verilog folder that has resources that are extremely helpful to refresh your digital logic background with a focus on Verilog based design.

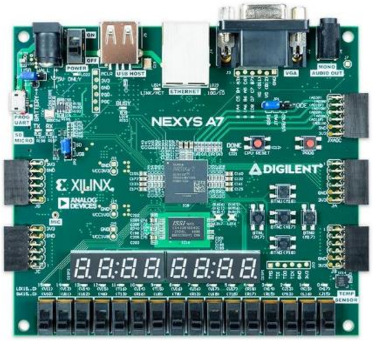
verilog_lecture1_comb.ppt: combinational logic design in verilog
verilog_lecture2_seq.ppt: sequentical logic design in verilog
verilog_lecture3_dp.ppt: datapath componenets in verilog
verilog_lecture4_rtl.ppt: state machine and controller design in Verilog
retaining_signals_in_post_synthesis_simulation_guide.zip: Tutorial on how to pull internal wires to monitor their values during post-synthesis simulation.
timing_analysis.pdf: Explains how to conduct timing analysis
Nets.pdf: a detailed discussion in wires and nets in Verilog
CummingsSNUG2000SJ_NBA.pdf: blocking and non-blocking assignments in verilog

Description

- You will implement three datapath components called PCAdder (adder), ProgramCounter (register) and InstructionMemory (memory). You will then glue these components together and implement the InstructionFetchUnit so that data from the instruction memory is read out using the address specified by the ProgramCounter register in each clock cycle. Templates (design and testbench) for these datapath components are in the "Resources→InstructionFetchUnit" folder.
- Refer to "Resources→6-LAB1_helper.pptx" for high-level diagrams for the InstructionFetchUnit design.

- Refer to “Resources→7-LAB1_Verilog_Recap.pptx and “Resources→8-Verilog Synthesis.pptx for Lab1 related potential Verilog related highlights.
- Demonstration due during your designated lab section.
- Demonstration requires post-routing simulation followed by executing on the FPGA using the seven segment display.
- A signup sheet will be available to choose your demonstration time slot at the beginning of the lab.
    - Suggested method:
        - First conduct functional verification for each datapath component (PCAdder, ProgramCounter and InstructionMemory) with behavioral simulation to make sure your logic is correct. Specification of each component is provided in the “Resources→InstructionFetchUnit” folder. Then referring to “6-LAB1_helper.pptx”, instruction fetch unit datapath (slide 3), implement the InstructionFetchUnit and conduct functional verification in behavioral simulation. Refer to the template testbenches provided for each component.
        - Then conduct post-routing functional verification, and practice bringing internal signals (wires) from your design to the simulation waveform. In your simulation show the values for the 32-bit Instruction being read from the InstructionMemory unit in each clock cycle along with the 32-bit current program counter value (PCResult). Refer to:
            - “D2L → Content → Verilog → retaining_signals_in_post_synthesis_simulation_guide.zip”
            - power point files provided in the Resource folder of Lab1.
        - Finally after functional verification of the I Asweszanstruction tech unit in post-routing simulation, load your bitstream onto the FPGA and display the 32-bit current program counter value and the 32-bit Instruction value on the FPGA as the datapath sequences through the instructions.
            - You need to use the “**Two4DigitDisplay**” and “**ClockDiv.v**” modules provided in the Lab1 folder. This module displays two 4-digit numbers.
            - It will be convenient if you initialize the instruction memory with values that can be represented with at most 4 digits (base 16).
            - Note that the values that will be displayed in post-routing simulation for the test code above are base 10. When displaying values on the FPGA , they will be hexadecimal form.

Notes:



- We will use the **Nexys A7-100T board** https://store.digilentinc.com/nexys-a7-fpga-trainer-board-recommended-for-ece-curriculum/
- "4-ClockDivider" and "5-Two4DigitDisplay" folders are provided as a reference for this lab. You will need these modules for displaying the values on the seven segment display of the FPGA.
- All interfaces that we provide as part of the labs will be based on Nexys A7-100T board. **We will strictly use Vivado 2019.2.1 for the labs.**

- You are welcome to use another **Xilinx** board if you wish as long as it has a display where you can show two 4 digit values. If you use another board, you may need to implement your own module for displaying numbers.

Submission Instructions

- Submit all verilog source files including testbenches (.v files), along with any constraint file or data file if applicable on D2L to the designated dropbox. Include the overall percent effort of each team member in the "InstructionFetchUnit.v" file.
- Penalty Conditions
  - Percent effort not reported (20% penalty)
  - Late submission or late demonstration (20% per day)
  - Submitting files in a folder or in compressed form (zip/tar). (25% penalty)
  - Changing the file name or extension. (25% penalty)
  - Failing to demonstrate (90% penalty)
  - Design works in behavioral simulation but fails to synthesize (80% penalty)
  - Design works in behavioral simulation, synthesizes with warnings but post-routing simulation fails (70% penalty)
  - Design works in post-routing simulation but fails to display Instruction and PC values on the FPGA (40% penalty)
  - Unable to answer questions during the demo (up to 75% penalty and mark for competition eligibility)