Addendum # 2
Guidelines to the Design Project

1. Motivation:

Throughout the semester, you have done various lab exercises and built simple circuits. In the process, you have learnt many different techniques to design, implement and debug digital circuits. However, so far you did not have the freedom to design a complete system of YOUR choice. End-semester project is your chance to come up with an idea, design the circuit that implements your idea and prove your digital design skills.

2. Overview and Scope:

EE201L requires you to perform a final "Design Project". Your design should be coded in Verilog, simulated and proved in ModelSim, and finally implemented on the Nexys-2 FPGA board. At the end of the project you need to present your project to your TA and the rest of the students in your lab during the last week of classes. Before starting your project you should write a proposal that should be approved by your TA. You will also have to submit a final report on your project.

As EE201L students, we do not expect you to design a complete state-of-the-art microprocessor for your project! By the same token, we also don't want you to implement a simple clock displaying real-time. Three weeks have been dedicated for your end-semester project, so the expected complexity of your design is roughly three times the complexity of a regular lab exercise. This document contains a list of possible projects, some of which have been done by EE 201 students in the past. We have attached brief description of their implementation for you to read and understand the scope of the project. You are encouraged to come up with your own project. Your TA will discuss your project proposal before allowing you to start working on it.

3. Grading:

Your project constitutes 10% of your course grade, i.e. just over three labs worth of points. Keeping in view tough competition in EE 201L, the difference between a below average and excellent project could well be the difference between an A and a C course grade[1]. Tentative grading policy is given below:

Difficulty and Completeness	60 points
Project Report (Proposal + Final Report)	5+15 = 20 points
Project Presentation	20 points

1. Please note that this DOES NOT imply that an excellent project will earn you A grade automatically. Course grade will be based on your overall performance.

4. Groups:

End-semester projects are to be done in groups of two students each. Exceptions can be made to allow three students to work in a group. Also, you are not obliged to work with your regular lab partner. In most cases both members of a team will get equal or similar grade on their project. Your TA will ask each team member several questions to gauge his or her contribution to the project.

5. “Blast from the Past”:

Listed below are some of the projects that students have done in the past along with their brief summaries from the students who did them.

5.1 Sudoku Puzzle Solver

Urmila Mahadev, Fall 2007

The aim of my project was to design a machine to solve a Sudoku puzzle. A Sudoku is a logic puzzle in which numbers must be placed in a grid such that no row, column, or sub-block has repeated digits. The intention of my project was to solve such a puzzle using a state machine and logic symbols to abstract the squares of the Sudoku puzzle.

There are three main layers to the design-- the top level, the state machine, and the logic symbols. In order to properly implement the state machine, two logic symbols were needed -- a symbol representing an individual square in the puzzle grid and a larger symbol that would allow comparisons between different squares (the “brains”). The state machine performs a process that solves the puzzle by iteratively checking each column, row, and sub-block to see if values can be eliminated from squares that have yet to be solved. The finish state occurs when each square has a set value (a “win”) or no squares change after an iteration (a “stalemate”).

5.2 Matrix Determinant

Violet Chu and Wunna Kyaw, Spring 2008

This project was designed and implemented to calculate the determinant of an arbitrary positive integer matrix with 3-bit entries in Verilog. Our calculation of the determinant for a matrix with dimension k is based on summing the products of the element with the determinant of its minor (a smaller k-1 square matrices excluding the row and the column the element is in).

First we broke down the steps necessary to calculate the determinant of a matrix (the range of the matrix size is from 2x2 to 8x8). We then created a state diagram to implement the sequential computation. Our design does not require reconfiguration as the size of the input matrix changes. It uses recursion to solve larger matrix determinants by using itself to iteratively calculate the determinants of its respective minors.

5.3 Polygon Wizard

Prithvi Balaram and Daniel Hilderman, Fall 2008

Our polygon wizard allows a user to draw a polygon on a small display field made up of LEDs. As the user uses a joystick to move the cursor, which is represented by a lit LED, the user can create a point at that location, which will be a vertex of the polygon. As the user proceeds the tool draws an approximated line will be drawn between points. Once the user has created the desired amount of points needed for the polygon, the tool will fill the LEDs inside the polygon. Input is received from a joystick and the execution will be done on a Digilent FPGA.

Inputs are received from a Digilent PmodJSTK, which is a peripheral joystick with 3 buttons. The joystick allows the user to move the cursor in four directions: left, right, up, down. Pushing in on the joystick creates a point (a “vertex”) at the cursor’s current location. Our output is a grid of LEDs mounted on a breadboard. This breadboard is connected to a Digilent Nexys 2 FPGA. The final design uses a grid height seven and width seven.

5.4 Morse Code Translator

Nicholas Haskell and Phebe Greenwood, Fall 2007

The purpose of our design is to construct a device that accepts a series of signals of different lengths and then translate the signals into Morse code. It then displays its result on the Xilinx board.

The input block is the linchpin of the design. Without it there will be no signal to translate. It accepts inputs from a timing calibration block as a pair of 16-bit wide busses. These inputs specify the average length of a short signal and the average length of a pause. The allows our machine to “adapt” to a users skill by calibrating itself to the average length of the users buttons presses and pauses. Once calibration is completed the system is receives input a single button. If the signal is shorter than the calibration value we read it as a short. If longer it is considered a long. The same thing happens with the pause values except shorter sends nothing and longer sends an endchar. After receiving an endchar (end character) the machine consults a lookup table for the character. This requires some translation since morse characters have varying lengths -- so we cannot simply call shorts 0 and longs 1. After the lookup the value is displayed to the user by accessing a second lookup table that gives the proper SSD segment codes.

6. Digilent peripheral modules (Pmods) for Nexys 2

These modules are designed to work with Nexys-2. You can consider using them in your project. Inform your TA if you are planning on using any of these as he/she needs to procure these for you. See details of these at the Digilent website:
http://www.digilentinc.com/Products/Catalog.cfm?NavPath=2,401&Cat=9

PMOD-BTN - Module with 6-pin header and pushbutton switches
PMOD-SWITCH - Four slide-switch Peripheral Module
PmodJSTK - Two axis joystick
PMOD-ENC - Rotary shaft encoder with integral push-button
PMOD-LED - LED Peripheral Module
Pmod8LD - 8 high bright LEDs
PMOD-SSD - Single two-digit seven-segment display device
PmodCLP - Character LCD w/ parallel interface
PmodAD1 - Two 12-bit A/D inputs
Pmod-DA1 - Four 8-bit D/A outputs
Pmod-DA2 - Two 12-bit D/A outputs
PMOD-TPH2 - Two 12-pin pass-through headers, one 12-pin test header
PMOD-BB - Solderless breadboard
FX2BB - Digilent breadboard with 100-pin Hirose FX2 socket connector

7. Additional resources for your projects from the EE454L Lab:

A few years back (in Fall 2005) in the EE454L lab, we have acquired 4 different boards listed below. Your TA can help you to understand how these boards operate and will discuss with you how they can be interfaced to the FPGA board and can be integrated into your project.

Seven Segment Display Interface
http://www-classes.usc.edu/engr/ee-s/454/ESA_Trainer/Peripheral_boards/7SEGMENT_SCH.pdf
Keyboard Interface
http://www-classes.usc.edu/engr/ee-s/454/ESA_Trainer/Peripheral_boards/KBD_SCH.pdf
Elevator Interface
http://www-classes.usc.edu/engr/ee-s/454/ESA_Trainer/Peripheral_boards/ELEVATOR1_SCH.pdf
Stepper Motor Interface
http://www-classes.usc.edu/engr/ee-s/454/ESA_Trainer/Peripheral_boards/STEPPER_SCH.pdf

8. List of some old projects:

Given below is a list of some simpler project ideas used in the past when we did not have FPGAs. It is better to check with your TA what level of project he or she expects you to perform. Suggest to your TA an interesting project.

- Number conversions using a number of methods. Example: 8-bit binary to packed BCD-Packed BCD to 8-bit binary. Implement in ePD and demonstrate on the FPGA board.
- Handshake control between two state machines running at different clocks
- Security alarm - Arming and disarming
- Wrist watch MODE selection button and SET button implementation
- Automated digital IC function tester for a specific MSI chip.
- Digital programmable (configurable) divide-by-N counter
- Implementation of digital watch/clock
- Implementation of a simple calculator.
- Part of a serial transmitter responsible for transmitting a character, (say 7 bits). Append with start, parity, and stop bits and transmit serially.
- Fast multiplication by bit-pair recoding
- Compute the GCF (Greatest Common Factor) of given two numbers.
- Finding factors of a given 8-bit number.
- Finding the first few prime numbers.