UW/CSE Hardware Design & Implementation

Course Description

Draft of May 19, 2009

Structural place in the curriculum

- 4 credits (3 weekly lectures, 1 weekly lab)
- Pre-requisites: Hardware/Software Interface, Foundations of Computing I
- Courses with this course as a pre-requisite:
  - 466 Embedded Systems
  - 467 Digital Design
  - 471 Computer Architecture
  - Hardware capstones
- Required for CE, optional for CS
- Catalog description: To be determined

Course Overview

A "bottom up" sequence from (1) circuit design to (2) CPU design to (3) embedded-systems (systems integration) design, including a significant lab component.

Description of work, etc.

While the lab sequence would constitute the largest portion of the coursework, there would also be:

- A midterm and a final exam
- Four Aldec Active-HDL design-tool tutorials to complete
- Possibly a few additional homework exercises to reenforce lecture material

Possible Textbooks

- Computer Organization and Design by Patterson and Hennessy
- Contemporary Logic Design (2nd Edition) by Katz and Borriello
- A reference on C

Approximate Topic List

The topic list is organized by weeks and is carefully structured to account for the labs. See the next page.

Week	Lecture	Lab
1	Introduction, implementing combinatorial logic circuits, nomenclature	Constructing Simple Logic Circuits; Introduction to Altera's Terasic DE1 prototyping board and Aldec's Active-HDL design tool for schematic entry
2	Verilog, building blocks, larger combinatorial logic circuits	Creating Verilog modules and test fixtures: Use multiplexers, decoders, and FPGAs to create a full adder
3	Implementing registers and sequential logic in hardware designs	Introduction to Registers: Edge-trigged D-type flip-flops and registers as sequential building blocks
4	Finite State Machines: Moore, Mealy and other hardware variants; FSM partitioning; FSMs in Verilog	Implementing Finite State Machines: Using a finite state machine to implement the controller logic for the game of Simon
5	Introduction to CPU Design, MIPS, Datapath, Memory / Control	Introduction to the Datapath: Constructing a datapath and some control components for a processor
6	Control / Strings & Pointers / Functions, Procedures, Machine Language	Taking Control: Adding new instructions that are essential for subroutines; constructing a control unit that supports normal operations. Implement a recursive program that solves the "Towers of Hanoi" for the case of 3 discs
7	Performance, Intro to Pipelining, Pipelined Datapath and Control	Pipelining: Breaking the datapath into multiple stages by adding registers between stages
8	Interrupts, I/O, Buses, Storage	Interrupts: Adding interrupt hardware to handle asynchronous events. Writing C code to handle button presses in the processor
9	Embedded systems, Microprocessor implementations, Microprocessor I/O	Embedded hardware output: Adding an LCD display to the processor; implementing hardware and C code to write data to it
10	System Performance, RTOS, Review	I/O and sensors: Adding a sensor to the system; implementing hardware and C code to read, process, and display sensor data