CS 193: Design Project - 1
University of California, Riverside
Summer Study Abroad 2015
Report for Roberto Pasillas

Instructor: Dr. Philip Brisk (email: philip@cs.ucr.edu)
Teaching Assistant: Jeffrey McDaniel (email: jmcda001@ucr.edu)

Project Summary: The basis of this project is to learn the MIPS, which is a Reduced Instruction Set Computer (RISC), architecture and mimic the functionality through a Hardware Description Language. Furthermore, the functionality is implemented in both VHSIC Hardware Description Language (VHDL) and Verilog. The project is split into seven labs to focus on different functionality with each mini project. Following is the lab summary with useful implementation notes.

Lab 1 - ALU

Create an Arithmetic Logic Unit (ALU) that supports the following operations: unsigned add, signed add, ,unsigned sub, signed sub, bitwise AND, bitwise OR, bitwise XOR, and divide the first input by two. The carryout bit was set by placing the result into a container one bit larger than the two inputs and connecting the carryout bit to the Most Significant Bit (MSB). This was obtained easier with the use of Verilog as the basic arithmetic operations are unsigned operations unless stated otherwise, whereas every operation must be explicitly cast.

Lab 2 - Binary Coded Decimal ALU

Create an ALU with binary coded decimal inputs. The method chosen to implement this lab was to implement translation logic to convert the binary coded decimal into binary , process the values, then translate the result back into a binary coded decimal.

Lab 3 - Converting Fixed Point to Floating Point

This lab is useful to learn the details of floating point and fixed point value representation due to the conversion from one format to the other. The implementation included fixed to float and float to fixed. Floating point has a defined structure composed of a sign bit, 8 exponent bits , and 23 mantissa bits. Fixed point is 32 bits with an implementation specific bits before the decimal point.

Lab 4 - The Datapath, and ALU control units

This lab creates the control units that sets the various components of the MIPS architecture to execute the required processes called by the given operation code (opcode). Basically set up memory, registers, and ALU to execute the given operation. The implementation was to use a switch statement to set different bits based on the input, in this case the opcode.

Lab 5 - Single Cycle Datapath

The basis of this lab was to connect all the components required for the MIPS

architecture giving us an indepth understanding of the datapath. Connecting the components was a detailed task , but straightforward with exception to to the memory module. The program counter must count by incrementing the previous value by 4, yet the memory module is word addressable. In effect the memory module does not need the first 2 bits of the program counter giving us a word addressable memory module with the program counter incrementing by 4.

Lab 6 - Generic Flip-Flip CAM Design Space Exploration

In this lab the implementation required the creation of three distinct Content Addressable Memory(CAM) modules that work slightly different from each other. The Binary CAM cell stores a bit and when not storing, compares the stored bit to the input. The Stored Ternary CAM (STCAM) cell also stored the comparison bit and include a don't' care bit that if set bypasses the comparison. The last cell implemented is the Ternary CAM(TCAM) cell , similar to the STCAM cell compares against a search bit and checks the don't care bit, but the TCAM cell does not store the comparison bit.

Lab 7 - Caches

In this lab simulate a cache and load in data to compare the functional benefits of different cache sizes(1024, 2048, 4096, 8192, 16384) in bytes, set associativity(direct mapped, 2 way, 4 way, and 8 way), and replacement policy(Least Recently Used [LRU] , First In First Out [FIFO]). Implementing the cache helped with the understanding of how Blocks of data are loaded into the cache and how to address the data. This was implemented in both C and in Python.

Summary: Through the project my understanding of the MIPS architecture , VHDL, and Verilog has increased greatly. Verilog was my preferred language as it was straightforward and didn't require multiple casts and recasts to carry out the desired operations. Learning a new architecture or hardware will be more approachable.