EE345 Computer Organization: Project, First Deliverable

Background:

The 2015 EE345 Computer Organization project is to construct a relatively simple CPU on a C5G FPGA Board. The CPU should be sophisticated enough to do "useful" work, about on the order of complexity of the SAP-2 or a bit better. The project includes instruction set design as well as implementation of the CPU and its demonstration running a short demonstration program. This document describes the first deliverable for the project, a description of the instruction set and a test program for it.

The instruction set and programmer's model:

Define the various formats and instructions to be used for the computer. The wordlength, instruction memory size, and data memory size, need to be defined. (It is satisfactory to simply have separate instruction memory (ROM) and data memory (RAM) for this project. (Implementing these as cache for a larger main memory would be an interesting project, but well beyond the scope of what is expected with the limited time and resources available.) Memory does not need to be byte addressable; word access is sufficient. We will not support floating point. The machine may support either a single accumulator or a register file, though if a single accumulator is used the machine must support indexed memory referencing instructions.

The instruction set needs to include several basic capabilities:

1. Basic arithmetic instructions: add and subtract, with provision for immediate data.
2. Support for conditional branches or jumps.
3. Support for loads and stores. If a single accumulator, these must be indexed.
4. Support for subroutine/function calls (and returns)

The sample program:

Write a short program in C that performs some useful task. The program should make use of a function call, and shoulkd somewhere use iteration, such as to access successive elements of a vector. Performaing vector or matrix manipiulation is an example type of task that might be apoproprioate, say, using a subroutine to perform multiplication. The sample program should end by writing a piece of data to a "special" memory address which is to be an output register (say, address 0). The sample program should be run in C to ensure that the correct output is computed. All data in should come from ROM (although, adding an input port at that same special data memory address, say address 0, would also be an option).

Compile the program into assembly language for your computer:

This step gives the assembly and machine language programs that will actually be executed on your computer, which will be loaded into ROM on the FPGA.

Report:

The report will include a short paragraph opf introduction, describing the machine expected and giving an overview of its data path. That is to be followed by the instruction set design, Sample program in C and assembly and machine code. (The code is to be well commented.) A short conclusions paragraph wraps it up.