Project - Behavioral Model

Homework 2 - April 24, 2005

Jason Nemeth

I. Introduction

The purpose of this assignment was to implement a working mock-up of the microprocessor using a behavioral VHDL model.

II. Requirements

The microprocessor must have the following capabilities:

1. Direct access to internal memory with at least 256 words.
2. An instruction set capable of executing at least the following two benchmarks:
   a. Output 10 signed integers in sorted (largest first) order where the numbers are input in random order.
   b. Output the mean (to the nearest integer) of 16 signed integers read into the processor.
3. One 8-bit input port, and one 8-bit output port.
4. Optional one single-bit Input enable line, and one single-bit Output ready line

The overall layout of the microprocessor is shown below in *Figure 1*. The behavioral model was created such that all these characteristics were met. The model is capable of modeling both provided benchmarks.

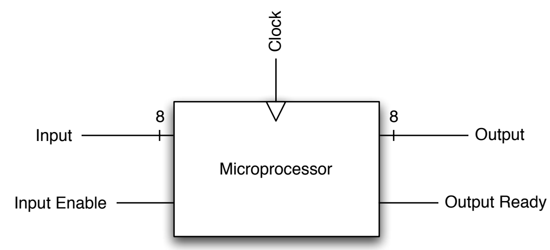

Figure 1. Layout of the Microprocessor

III. Design Decisions

Opcodes

The first major design decision made was regarding the opcodes required. The averaging and sorting operations have been implemented in psuedocode, and the necessary operations have been identified. However, due to possible optimizations made prior to the implementation of the structural model, the required opcodes may change. So far Add, Shift Right 4 bits, Move (from one memory location to another), and Jump (code execution) have been identified as necessary instructions. As noted, more may be needed as optimizations increase.

ALU Implementation

The ALU will be memory-mapped, and one of its two inputs will be designated as directly mapped to the input port (location 254 as noted in previous report). This will allow the addition operation to take place as the numbers are entered. Then, only a single clock-cycle will be required for the shift operation before the result can be presented on the output lines.

Behavioral Model

The behavioral model was developed with the above requirements and decisions in mind. The model makes a number of assumptions regarding the delay times of operations, and in general assumes synchronous operation in all cases. The model accurately models an average and sort operation, producing the output along the output lines and setting the Output Ready bit high whenever an operation completes.

Room for Improvement

Since memory can generally be written to in approximately 2 ns, and a clock cycle may be closer to 30 ns, it may be advantageous to asynchronously perform memory operations as it would greatly increase the speed of the sorting operation, which requires three memory writes per swap. This could provide a significant improvement if it can be implemented.

IV. Results

The results of the behavioral model are what one would expect from a simple behavioral system. The model produces the correct averaged or sorted output in all tested cases. The model estimates an averaging operation will take approximately two clock cycles on top of 16 clock cycles taken to input the numbers. The model also estimates a sort operation will take approximately 128 clock cycles on top of the 10 clock cycles needed in read in the numbers. The results are shown in *Table 1,* with the values representing the number clock cycles beyond the data input. In this behavioral model the clock cycle was set to 30 ns, a speed derived from the speed of the adder created in Homework 1. This obviously leaves lots of room for improvement, and shows that the most complicated portion of this project will be coordinating how information flows into and out of main memory.

Average	Sort
2.1	128.3

Table 1. The results of the behavioral simulation

V. Conclusions

The results of the behavioral model will be used to test for the proper operation of the structural and RTL models, as they were in the first Homework of the quarter. As such, the behavioral model will remain useful as a benchmark of the future implementations. In addition to proving the proper functionality of the system, the model provides a pretty good indication that the majority of the work optimizing the system needs to be directed at reducing the computation time of the sorting benchmark.