CPE480 Assignment 2: Multi-Cycle Tangled

Implementor's Notes

Gerard (Jed) Mijares
Ben Luckett
Nick Santini
Department of Electrical and Computer Engineering
University of Kentucky, Lexington, KY USA

ABSTRACT

This is an multi-cycled implementation of a processor for the Tangled instruction set.

1. GENERAL APPROACH

Our AIK specification of Tangled is largely based on the sample solution provided by Dr. Dietz, with a few changes. Namely, we added two additional possibilities for the first 4 bits of the instruction: `Start` and `Decode`. `0xd` was available, so we used that for `Start`. However, at this point, we were out of hex numbers for the first 4 bits. So, we combined the instructions that were originally categorized under both `0x7` and `0x8` to only begin with `0x7`, leaving `0x8` available for `Decode`.

Our processor is structured as a state machine. The first step is to use a case statement on the first 4 bits of the instruction. Sometimes, that is sufficient to distinguish which instruction we should use, but if not, we nest a second case statement on the second 4 bits of the instruction to further determine which instruction to use.

Since some instructions will require a second 16-bit word to process, we elect to simply store both of the next two words in the `Decode` state.

As recommended by Dr. Dietz, we make use of Verilog's `‘define`s to name ranges of bits we frequently use and the OP code for each instruction.

We decided to run our Verilog locally, so a few commands differ from Dr. Dietz's web interface. A Makefile is included to facilitate running the project. To run different assembly programs, the `readmemh` instructions must be changed to the appropriate filenames.

2. TESTING

Not very fleshed out yet, there's some code to test the branch instructions.

3. ISSUES

Qat instructions originally didn't increment the pc when we needed two words.

Branch instructions were a bit difficult.

ACM ISBN N.A..
DOI: N.A.