ECE 552 – Project Plan

MIPS: Meaningless Indication of Processor Speed – Team

Members

Ryan Bambrough (rbambrough@wisc.edu)

Mitch Eberle (eberle3@wisc.edu)

For ECE 552 we are building a pipelined processor. This will require a lot of work and coordination in order to ensure that the final product operates with in specs. To do this we, as a team, have come up with a schedule and outline to follow through out the remaining portion of the semester. This plan is not final and can be changed to meet any expected occurrences that may pop up during the project time. The bulk of the work will be done on weekends, the details of what we will do each week will be covered later in this plan.

In order to execute this pipelined processor we plan on building a single cycle processor first and then implement pipelining into this processor. We will work primarily off designs provided in class and in the text. While designing our processor we will make an effort to optimize each module before moving onto the next module, but we will not waste time trying to implement an overly complex module if we believe that it will compromise a large section of the plan or not operate correctly with the rest of the design. The decision of modules that will be implemented into the processor will be on a per module basis and will be the result of agreement between both of us.

For our processor we plan on implement the following main modules with each being designed and created to operate within the whole with the prescribed interfaces. We hope to test each module as best as we can before moving onto the next module. Dividing up the work will be done on a per module basis with each of hoping to focus on either control or data path elements within each design. Our division of modules is based around the idea that later in the project we will be pipelining so sections are split up by their, would be, pipelining boundaries. The base modules as we see them currently are:

**Instruction Fetch:**

PC register – Contain the current address of instruction

Instruction Memory – Already implemented for us

**Instruction Decode:**

Control Unit – Create the signals for other modules in order to do the correct instruction. Signals for: ALU, Register File, Mux selects for forwarding and select lines, and Memory File.

**Execute:**

ALU – The ‘meat’ of the processor, will implement all necessary arithmetic and logic operations that will be used in the processor.

**Memory:**

Memory File – Long term storage for data created and used by the processor and program.

**Write Back:**

Register File – Short term storage for data created and used by the processor and program

To begin with we plan on putting together a Design of our processor following roughly what was outlined above. We will use schematic tool to help us track each module, which will help with the implementation of each module for Demo 1. The Basic design will be done by the **Design Review** date of **March 3rd**. After the design review we will take any input from our reviewer and integrate it into our design and start working on implementing our modules.

For a time break down of processes we plan on implementing 2 to 3 modules, based of difficulty and complexity, per week/weekend. This will leave us with all the modules implemented the weekend before **Demo 1** on **March 17th**. For this demo we will have all the base modules completed and working with each other. If we correctly implemented our design all that should be left to do is connect the modules together and ensure the single cycle processor is working correctly.

Once we have a single cycle processor completed we will start moving toward the pipelined processor that will be the result for **Demo 2** on **April 12th**. This stage should be fairly straight forward was we will try and split the processor such that it can be pipelined easily. Additionally for this demo we will write at least two tests that demonstrate the pipelining that we implemented. For the pipelined processor we will have a plan to take care of any forwarding that needs to be implemented in our processor. On top of this we are going to test the different memories that are part of the demo 2.

From this stage we will move into implementing the cache and ensuring that it is working with our pipelined processor. We will have the cache completed and working with the processor by the **Cache Demo** date of **April 21st**. We will also be taking a closer look at our processor and start moving toward the final iteration of the processor for the final demo. This includes ensuring that all features of pipelining have been checked and are functioning correctly. Also, we will begin finalizing our report which will be due on May 9th.

Finally we will have the pipelined processor finalized by May 8th the day before the **Final Demo** on **May 9th**. The **final report** will be done in parallel with the processor and competed before its due date of **May 10th**.This concludes our outline for the ECE 552 project. We will do our best to stick to the plan though small adjustments may need to be made in order to ensure that we deliver a complete working project by the due date on May 9th.