**CPU Design Project –Part 6**

Catherine Brogden

What did you learn from this project?

I gained a better understanding about how to design and operate a single cycle datapath. I also gained a better grasp of how difficult it can be to build around limitations in design due to design specifications. For example it would have been nice to have a larger offset range for branching or immediate adding/subtracting, but it would have required either increasing the size of the processor from 16-bits, which violates the design limitations, or cutting in half the number of available registers, which reduces the operation capability of the processor.

What would you do differently next time?

I would have started the construction of the completed datapath earlier, as it was difficult to get all of the possible instructions working on the datapath as I had a couple of design flaws that were unnoticed in the implementation of single component testing. I would also back up my data in more than one location frequently; my folder containing the project files was corrupted after I submitted part four and I had to rebuild it from my component list from part three in order to complete part five. This caused me to use more time on reworking completed sections than I would have if I had possessed a more recent copy of my work. I would also have better utilized component declaration and not designed as many multiplexors as I did, as I could have used three types, because of how I built other components, instead of six, each multiplexor being its own unique component.

What is your advice to someone who is going to work on a similar project?

Remember that components can be declared multiple times, which reduces the number of components that need to be created. Start as early as you can on the project as debugging issues takes time. Always keep multiple backups of your project. Keep the design and implementation as simple as possible, the more complex designs may be able to do more but they also take longer to debug as they require more effort to implement, which means errors are more complex and much more difficult to spot and correct.