Writing Design Documents

CS150 Fall 2010

Writing a good design document can be a somewhat subtle art, especially if you have never done it before. Here are some guidelines to get you started.

Modularity

Modularity is the single most important concept to understand when writing a design document (and possibly in digital design in general). DDCA section 1.2 gives a good overview of what exactly modularity means in this context, and it is highly suggested that you read it. Make sure when you are designing your system that you are primarily thinking in terms of the modules you need and the interfaces between them. If any single module in your system is complex enough that you can't easily reason about its behavior, then it should be split up into multiple modules.

A Picture Is Worth 2^12 Bytes

Trying to write an entirely textual description of a circuit quickly becomes unmanageable for a design of any significant size. Well organized diagrams should always come first, and then small textual blurbs should be added for any piece of the diagram which may not be immediately clear to the viewer. When creating diagrams, it is important to organize blocks in some logical fashion. A typical strategy is to have inputs on the left and outputs on the right, and then to have data "flow" through your module from left to right.

Know Your Audience

You can assume that anybody looking at your design will know everything from lecture, but not that they have any prior knowledge about how your design is organized. Therefore is is perfectly acceptable to use primitive blocks such as the various boolean gates and flip-flops without showing their transistor level implementation. In fact, it will actively hurt the clarity of your document if you decide to show implementation at such a low level. It is even fine to use high level blocks such as adders that we have covered in class without further elaboration. If you have a block that is a combinational function, it is best to give a truth table or boolean expression for the function rather than showing a harder to understand gate-level implementation of the function.

It is not, however, okay to give a high level description of a block such as "Control Logic" and never give any further elaboration.

The Algorithm

We recommend the following process for making your document:

1. Draw a top level schematic for your circuit, using the coarsest blocks you can think of as your building blocks.
2. For each block on the top level, draw a new diagram showing the implementation of that block. Again, use modules to implement any non-trivial functionality in this model.
3. Recursively apply this process everywhere until you only have primitive blocks (see Know Your Audience).
4. If there are any places where your solution is not made immediately obvious by the diagram, add text annotations to explain what is happening.

Useful Tools

Many of you have never had to make circuit diagrams before and may be wondering what the best way to make them is. There are a variety of options:

1. **Visio.** This is a diagramming tool by Microsoft. It's a pretty popular option and works fairly well, despite being frustrating for some people. It's installed on the iservers and possibly other department systems.
2. **Omnigraffle.** Some consider it to be a better option than Visio, but it has the disadvantage of only running on OS X. Additionally there is no way to get licenses from the department.
3. **XFig.** Tried and true diagramming solution for the X11 windowing system. Should run (with varying degrees of effort) on most Unix-like systems (Linux, OSX).
4. **A pen(cil).** Has the advantage of being compatible with almost all environments. It is acceptable to turn in hand-drawn design documents, but they're expected to be as clean as legible as a computer-generated document.