Chris, I know most of these are late/overdue. I’m turning them in regardless in hopes of you helping me out. I’ve missed most of your lectures, and have failed to turn in assignments on time, due to my lack of time management and over commitment to classes and my job this semester. I wouldn’t be satisfied with myself if I didn’t try to reconcile. I will be turning in these “quarter sheet” responses, and as many assignments as I can. If you choose to not accept them, I’ll understand, it’s my own fault for getting into this predicament.

Regards,

Jeremy Winterberg

**Speed**

The podcast on speed was interesting, yet all over the place in its organization. One avenue which this relates to my life would have to be the choices I make. The immediate results are very fast, yet the long-term effects of a decision remain to be seen, sometimes not till further into the future. E.G. Choosing to eat a donut today is satisfying, yet in 20 years it may have been the catalyst for a heart attack.

**Future of Computers**

This series of articles continued a topic I first heard of during a guest lecture in Junior Seminar. The notion that we’re currently hitting a performance wall on three fronts is both discouraging and inspiring. Without a challenge, such as this, there wouldn’t be pressure to innovate and take our technology to the next level.

**The Free Lunch is Over**

*What impact do Sutters’ statements have on you as you consider your future career in technology?*

Stutter’s article brings up the notion of a stalemate with computational advancements. I see it in two ways, the first being reasonable. The need to be efficient with programming to maximize performance per clock cycle. And secondly, an opportunity to change the way modern computers’ architecture. If what we currently have is hitting a wall, we must find a way to demolish the wall, climb it, or go around it. Perhaps this is what will spark quantum computing into more than a research capacity that it currently resides in.

*Abstractions have hidden many low-level aspects of technology away from software developers. Can they do the same for concurrency?*

Yes, in certain situations. I don’t see concurrency abstractions being mainstream for all programs. But, with something such as extensions for an existing application or building a library for a language, concurrency could be abstracted as to guarantee these efficiencies throughout future additions to a project.

**FPGA’s**

* Unlimited (digital) possibilities for combinations of gates you can create with them is insanely cost efficient.
* The speed of which they can perform actions allows specialized flexibility.
* Parallel processing with the high quantity of I/O blocks opens many channels to play with in a project’s design.