*Introducing **Philip Americus** - a new guest blogger here on Think Verification. Phil is an ASIC veteran who's worked with every phase of ASIC design - from initial concept to tapeout, with an emphasis on verification, including management of both HW and SW engineers. This is his debut post on our website:*

When I first started my Engineering career we had Verilog-95 and VHDL. Combine this with some wrappers, in C, Perl, or Csh, and we accomplished Verification just fine. As designs became larger and more complex, we continually felt a need to go to higher level programming models. In came Specman, SystemVerilog (Superlog!), SystemC, model checking, etc. to meet the challenge. Some companies embraced these new technologies and some stuck with the old.

My question is this: Now that we are in 2010 (the Year We Make Contact, for those who get the movie reference), where do you think Verification technology will be, or should be, on January 1, 2011? What big challenges do we face this year and how will we solve them?

1. Outsourcing - by the end of 2010 all small to medium sized companies will outsource all of their verification needs. Verification has become such a specialty that it will be more beneficial to the company to outsource it. These verification companies would have several benefits over internal verification teams. Their engineers would be super-specialists in high level verification. They would also have at their disposal a large base of reusable test & environment code. This would allow them to setup top level chip testing for a complex chip in an hour. Lastly, the outsourced verification company would have a large server farm. I have been at companies where we almost needed a full time manager/referee just to decide who gets to use which machine and for how long.

2. Say Goodbye To Design - by the end of 2010 all design teams are totally restructured. These teams have all the design engineers replaced with software engineers who code the chip in C. Then they bring in one ASIC Design Superman who takes all the code and converts into Verilog. I say Design Superman, because this person has to really understand the limitations of both C and Verilog and how to make them work together correctly.

3. Database - The Verification environment should be built around a database. I think some of this capability is already available but it should be much simpler and easier to implement. The test plan is created in the database, then the tests can be linked to the test plan, and the regression can also be linked into it. This provides an easy way to monitor progress. Also, for test re-use the database would require a check to see if the test still matches the requirement. How many times have you written a test and forgot to see if it was still valid and/or actually did what it was supposed to do? The database would force you to check this detail.

Now it's your turn. What do you think about the year 2010?