Scripting an IC Flow

Developing a standard cell integrated circuit is a long and complicated process. Even after the RTL code is found to be essentially bug-free, a enormous number of steps still remain to be taken to reach the point of silicon “tape-out”. Many of the steps will be iterative ones, requiring possibly dozens of passes before the full design criterion is met. Using IC design tools via their graphical user interface would be extremely tedious and since many of the operations would be the same, valuable engineering time would be lost. Fortunately, computers provide an excellent solution for doing repetitive tasks.

Instead of telling the computer what to do with mouse and keyboard, we utilize a command interpreter such as a OS “shell” to read a textual list of commands for running different tools as well as feeding the tools the information they need to run. Nearly all large professional IC development is fully scripted. Having the development of the IC scripted has two important advantages:

- The script itself gives a concise and exact methodology of how the chip is built. All the steps are in the script. Nobody can forget who did what or how it was done. All the build information can be in the script.
- The scripts commands are repeatable and should give identical results. This is important when comparing slightly different versions of a chip or when using a different version of a tool. The resultant design built by the script is deterministically identical if nothing has been changed.
- The script provides a orderly way to manage the utilization of a large number of tools. A script could utilize multiple workstations to work on different parts of a design project.

What scripting language do we choose?

Traditionally, some type of UNIX shell scripting has been used. Shells used can be csh, (C shell) tcsh, (TC shell) sh, (Bourne shell), and more lately bash, (born again shell) and PERL (Practical Extraction and Reporting Language). The best candidates are most likely the bash shell and perl.

Entire scripts can be done in just about any scripting language. However, some data manipulation can become very hard to do in common shells. PERL however, provides extremely comprehensive data manipulation tools. For very

quick an dirty scripts of 1-5 lines, csh or tcsh shell provides an easy to use, quick way to automate work. For more complex work, bash is arguably the best shell to use. Finally, if there is a need for sophisticated data manipulation, perl is unmatched in its power.