This chapter discusses tools and methodologies for managing the design database for macro design and for system design. The topics are:

- Data management
- Project management

12.1 Data Management

Data management issues include revision control, bug tracking, regression testing, managing multiple sites, and archiving the design project.

12.1.1 Revision Control Systems

A strong revision control system is essential for any design project. A good revision control system allows the design team to:

- Keep all source code (including scripts and documentation) in one centralized repository that is regularly backed up and archived
- Keep all previous versions of each file
- Identify quickly changes between different revisions of files
- Take a snapshot of the current state of the design and label it
RCS, SCCS, and Clearcase are examples of tools with which revision control systems can be built. Additional scripts and processes are typically used to create complete revision control system.

The most common paradigm is for each designer to be able to check out the entire design structure and recreate it locally, either by copying files or creating pointers to them. The designer then works and tests locally before checking the design files back into the central repository.

There are two different models for controlling this check-in process: the always-broken and always-working models.

**The Always-Broken Model**

In the *always-broken* model, each designer works and tests locally and then all the designers check in their work at the same time. The team then runs regression tests on the whole design, fixing bugs as they appear.

There are two problems with this model. First, when regressions tests fail, it is not clear whose code broke the design. If there are complex inter-dependencies between the modules, debugging regression failures can be difficult and time consuming.

The second problem with this model is that there tends to be a long integration period during which the design is essentially broken. No new design work can be done during this integration and debug phase because designers cannot check out a known-working copy of latest version of the design.

**The Always-Working Model**

The *always-working* model overcomes the major problems presented by the always-broken model. For the initial integration of the design, when separate modules are first tested together, the always-working model is the same as the always-broken model. Everyone checks in the initial version of the blocks and a significant debug effort ensues. In some designs, it may be possible to integrate a subset of the whole design, and then add additional blocks once the first subset is working. This approach greatly reduces the debug effort.

Once an initial baseline for the design is established, the always-working model uses the following check-in discipline:

- Only one designer can have a given block checked out for editing.
- When a block is being checked in, the entire central repository is locked, blocking other designers from checking modules in.