Maximizing Team Productivity through Efficient Design Data Management

Authors:

- Paul Beard, Manager CAD, AMCC Canada.
- Srinath Anantharaman, Founder & CEO, ClioSoft, Inc.

Need for Design Data Management

Several factors such as exponential increase in design complexity and shrinking market windows are leading to larger design teams. Additionally, the increased competition due to globalization requires the use of the best available engineers irrespective of location.

The volume of data and the need to share and communicate rises exponentially with the team size. Communication across different sites, different time zones and different countries further increases the complexity significantly. This rising complexity drastically erodes engineering efficiency and increases the possibility of errors. At a certain point adding more engineering resources may actually become a detriment to the project.

Therefore, the biggest gain in productivity can be achieved by streamlining communication and the access and sharing of data. Significant improvements in efficiency can be achieved by defining processes and deploying tools to manage data and collaborate seamlessly. Primarily the following needs should be addressed:

- Exchange large volumes of data reliably and efficiently between team members irrespective of geographical location.
- Manage changes so engineers get the right set of changes at the right time.
- Identify and back-off incorrect changes.
- Remember and recreate project configurations at specific milestones or time points.

Challenges of Collaborative Design

A design project involves several people working on different aspects of the design such as:

- Specification
- Architecture
- Digital design
- Analog design
- Simulation and functional verification
- Test program generation
- Timing and power analysis
- Custom layout
- Layout verification
- Firmware
- CAD Support (scripts, make files, etc.)
- Documentation

Each phase of the design process requires different skills, different tools and quite often different platforms spread over multiple design centers. Design teams face the following major challenges:

- **Team Size**: Size of engineering teams may range from tens of engineers to hundreds. Each engineer may have a different skill set and level of experience. There may be several sub-groups working on different phases of the design. All these engineers need to be able to share data reliably and efficiently.
- **Geographical Separation**: Design teams are often separated across multiple sites making face to face meetings difficult. Differences in time zone and language add further complications. Network latency and bandwidth limitations aggravate the problem of sharing of large volumes of data.
- **Data Explosion**: Files in a project may easily number in the hundreds of thousands distributed across many thousands of directories. Files which have graphical data such as schematics and layout are binary and very large. Of course, files are constantly being added, deleted, moved and modified. Users must be able to reliably synchronize with changes made by team members.

- **Complex Flows**: An SoC flow has several stages from front end design and simulation to back end layout and verification. Often there is also an analog component. Multiple tools from different vendors may be in use, each producing large volumes of data in different formats, organized in different library structures with complex relationships and dependencies.

Efficient Collaboration

Ad hoc processes naturally evolve as the design team needs to collaborate. Such ad hoc processes may solve a problem at that time but do not scale well when the size of the group grows and the project is close to the tapeout deadline. This leads to several problems such as:

- Project delays impacting time to market.
- Design errors leading to re-spins or compromise in quality.
- Decreased predictability with lack of control and visibility.
- A complete nightmare for the CAD support group.

The biggest gains in productivity can be achieved by streamlining the process of sharing and managing design data. The problem must be tackled on four important fronts:

- **Infrastructure**: Engineers should be able to efficiently access any project data they need. A high availability infrastructure consisting of performance workstations, fast and abundant storage, and high bandwidth networks will easily pay for itself with increased designer productivity.
- **Software**: A design data management platform is the software backbone that enables efficient management and sharing of design data. The design data management system should support efficient multi-site collaboration and work seamlessly with the EDA tools in use. All the design data – from specs to layout – should be managed in the same DM system.
- **Process**: To be effective the infrastructure and software have to be complemented by well-defined processes and procedures. The processes should meet the requirements of the project but should be simple enough that engineers can learn the rules easily and be able to follow them everyday without undue burden.
- **Training**: Engineers must be well trained on the use of the infrastructure, DM software and processes. Without adequate training, users are likely to continue to use methods they are accustomed to and not realize the full benefits of the system.

This paper will explore the features of a design data management system and the processes that need to be implemented to streamline team development.

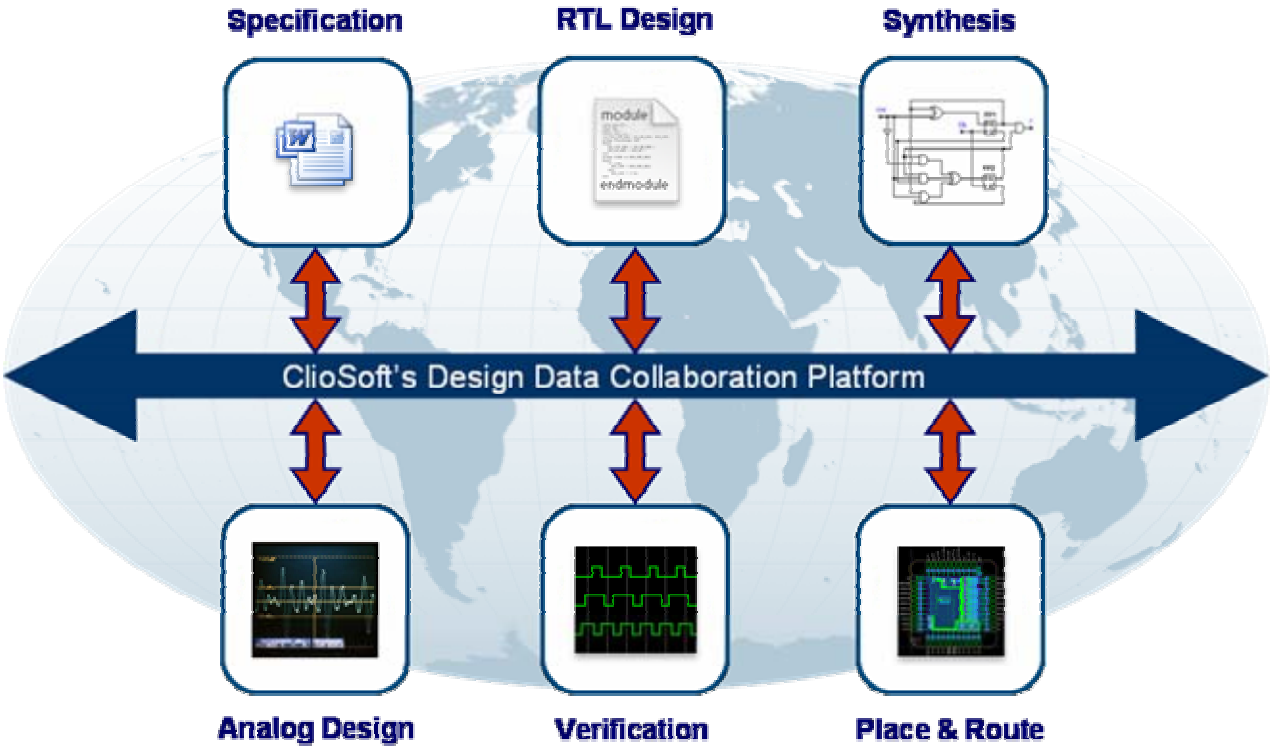

Figure 1: Streamlining the SoC Design Flow

Special Requirements for Managing Design Data

Often a software configuration management system, already in use in the company, such as IBM's Clearcase or public domain CVS is adopted for managing design data. This may work for the front end digital design flow but is typically unsuitable for other flows such as analog or custom IC design.

Differences from Software Development

IC design data and development process is different from software data and development in several ways:

- Users are working at higher abstraction levels such as libraries and cells and are not necessarily aware of the files that store the data.
- Design tools create files that contain design data as well as temporary run files that need not be managed. End users do not know which files should be managed and which should not.
- A design flow has several complex tools sometimes from different vendors. Each tool organizes data files in different ways.
- A design object such as a schematic or layout is usually made up of multiple files created by the design tool with software generated names.
- The size of the data is much larger. There are many more files and several of them are large binary files.
- Designers often think in terms of design hierarchy that is not apparent when looking at the file system.

Design Aware Integration with EDA Tools

Electronic design data tends to be very complex. The relationship between the design abstractions that the engineer is working with and the physical files is one to many and changing. For instance, a designer using Mentor's ICstudio or Cadence's Virtuoso Custom IC platform thinks of libraries, cells and views. Each cell-view consists of several files. Some files are log files, backup files or run files and these should not be managed. Sometimes the set of files that represent a design unit change between revisions.

The relationship between the design units and the physical files is known only to the EDA tool. If engineers try to version control data at the file level then they run a high risk of missing some required files or placing run files under version control preventing others from running simulations etc. To effectively deal with this issue the DM system should be seamlessly integrated with the EDA tools. The integration should be aware of and manage the sets of files that are part of a design unit. Users should be able to perform DM operations such as check-out and check-in at the abstraction level they are familiar with and the integration should translate that to the physical files that must be operated on by the DM system.

Additionally, engineers are used to working in their EDA environment. An Analog designer may be editing schematics and running simulations. A place and route expert may be editing the layout. Going to a separate DM tool for version management would add a significant hurdle to deployment. Making DM commands directly available in the EDA tool environment will enhance usability and correct use of DM methodology. The EDA tool is aware of when a user may want to edit a design unit and can prompt the user to check-out the design unit when needed.

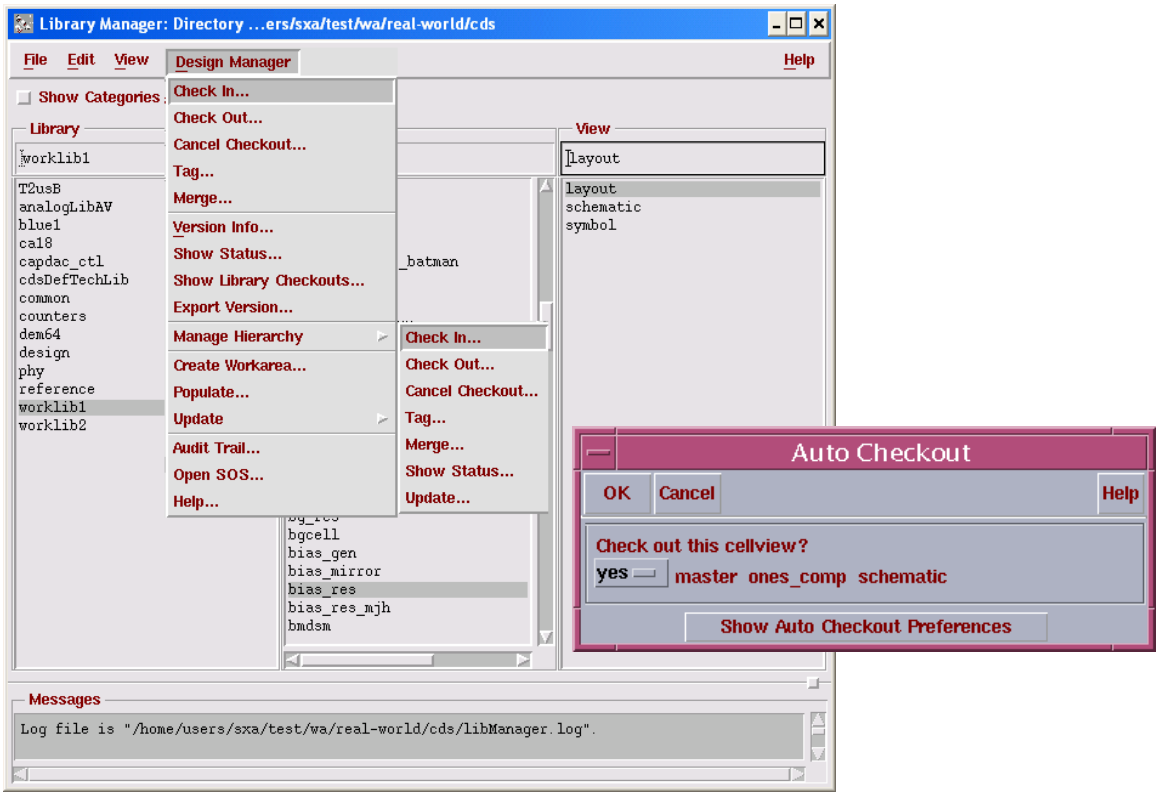

Figure 2: ClioSoft's Design-Aware integration with Cadence Virtuoso Platform

Design Hierarchy

Designs are created hierarchically. Usually the project is partitioned into modules and a designer is assigned a module or sub-module. Each module is again made up of several levels of hierarchy.

A library is a flat list of cells often in the thousands where the design hierarchy is not apparent. An individual designer is most interested in the module he/she is developing and the hierarchy of cells below. For instance, the user may want to work on a module and may want check-out for editing not just the module but the sub-modules as well that make up the design hierarchy.

A hierarchy aware DM integration allows designers to easily manage their work. ClioSoft's integrations with both the Cadence and Mentor IC flows allows users to do all the common DM operations such as check-in, check-out, tag, update and show status of all sub-modules in the design hierarchy.

Composite Objects

An engineer works with a design object such as schematic, symbol, or layout. The design tool usually saves this data as a collection of multiple files. To further complicate matters the list of files that make up

an object may change as modifications are made to the object making it impossible to manage the object as separate files.

To maintain the integrity of the data the design management system should be able to support revision control of such composite objects. All the files that make up a composite object should be managed together as a single object within the system. This has the following advantages:

- You can easily revert back to a previous revision of the object.
- All DM operations on the object are atomic and data integrity of the object is maintained.
- The object is identifiable as itself in the DM system instead of a set of files.
- The number of objects under management is reduced.
- DM operations are more efficient.

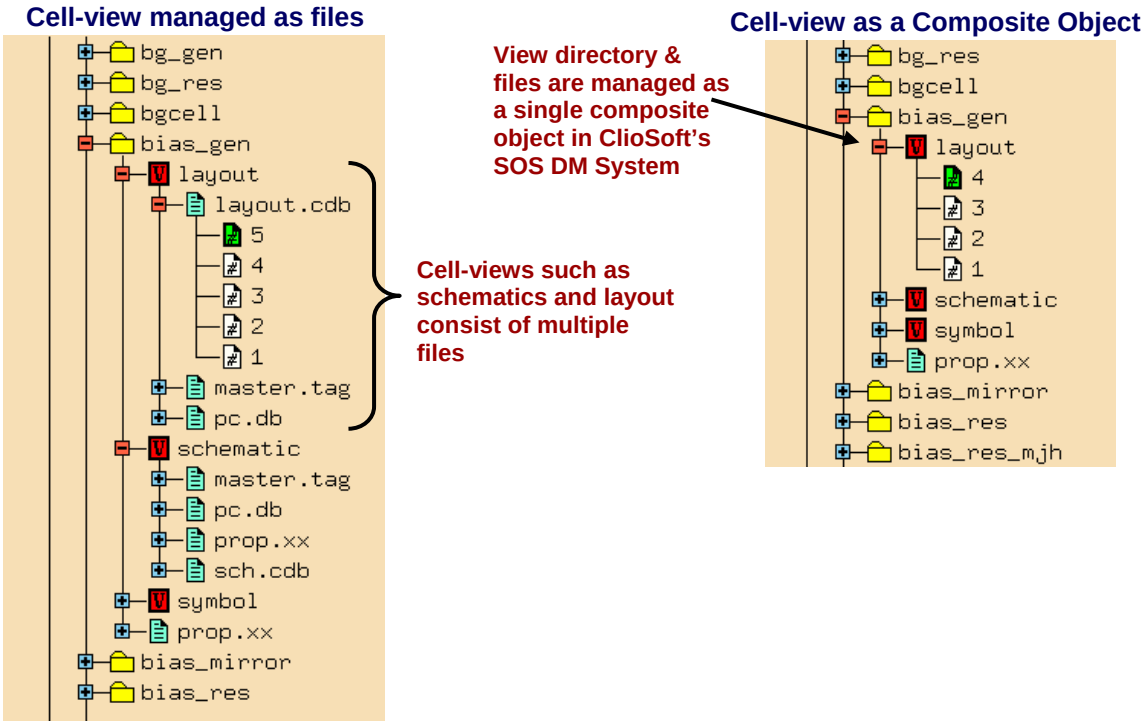

Figure 3: Composite Design Object

Optimizing Disk Space Usage

Design files may vary in type and can be quite large. Although disk space is relatively inexpensive the IT expenses to provide services, such as reliability and backup, add up. Therefore it is important to optimize use of disk space.

In a typical software development environment each user has a workarea with a physical copy of the latest revision of each file. This approach is not optimal for large teams with graphical design data such as schematics and back-end layout in large libraries.

A practical approach is to create a workarea where the directories are physical but the files are symbolic links to a shared area. When a file is checked out for editing, the link is broken and a physical copy is placed in the user's workarea. Engineers share the files except for the ones they have checked out for editing. A significant drawback to this approach is that the user loses control of which revisions are being accessed. The physical file in the shared space may change right in the middle of a big verification job thereby nullifying the result.

ClioSoft's SOS DM solves this quandary with a workarea model where files are symbolic links to a cache managed by a 'smart' cache server. The cache server maintains a copy of revisions of files being used by all the workareas and keeps a count of how many symbolic links point to each version. A file revision is purged from the cache only when there are no links pointing to it. This model provides users with a stable workarea that they have full control over, and yet optimizes disk space by using links to shared revisions.

Collaborating Across Sites

Limited network bandwidth and increased latency are two significant factors that affect data collaboration when development is spread over multiple sites. The most important requirement to facilitate development spread across multiple sites is to have a reliable and secure network with sufficient bandwidth to meet project requirements. Recently used data should be cached at the remote sites so other users can get the necessary data from the local LAN instead of going over the WAN. This can easily be automated by using a DM solution that is optimized for multi-site development. For instance, ClioSoft's SOS DM platform supports a Smart Cache Server that keeps the latest revisions. If a user gets a new revision of a file at a remote site then it is automatically stored in the cache and subsequent users transparently get this file revision from the cache.

Defining a Process

A well defined process and methodology is as important as the infrastructure and tools. The methodology should define how the data is organized, how change is propagated, who has what type of access to what data and when. Users should understand and follow the methodology.

What should be version controlled?

As a rule of thumb any data that users directly create and modify should be managed and placed under version control. This includes specifications and documentation, design data files from RTL through GDSII, Makefiles and automation scripts, stimulus vectors, environment settings, etc. Keeping all source data under the same data management system makes it much easier to administer the system and train the engineers.

It is not necessary to track every change to derived files such as simulation output or synthesis DB files. These files tend to be very large and can be generated if the tools and source files are available. However, at some critical time points you may want to archive these derived objects in the DM system. It is best to archive the derived objects in a separate directory or under a different name so it does not interfere with the normal tool runs that output these derived objects.

Setting up Workareas

Each phase of the design process has different requirements on how workareas should be set up. Design engineers may create workareas with links to cache to optimize disk space. Release engineers will create workareas with physical copies of files. Multiple layout engineers may need to work very closely together and share the same sandbox so they can verify the interdependent changes before checking in the layouts.

ClioSoft's DM supports a variety of workarea models – isolated or shared sandbox – with physical copies of files or with symbolic links. Each user or group working on the project can choose the model most suitable for their needs.

Directory Versioning

During a project, designers not only modify source files, but will also add files/directories, delete or rename files, and sometimes even reorganize the entire project structure. It is just as important to version control changes made to the directory structure as well so that:

- Designers can change the directory structure and have the changes propagate seamlessly to other team members on demand.
- If there is a need to restore a previous time point or milestone, the directory structure will also be reproduced exactly and all scripts and automation will work as before.

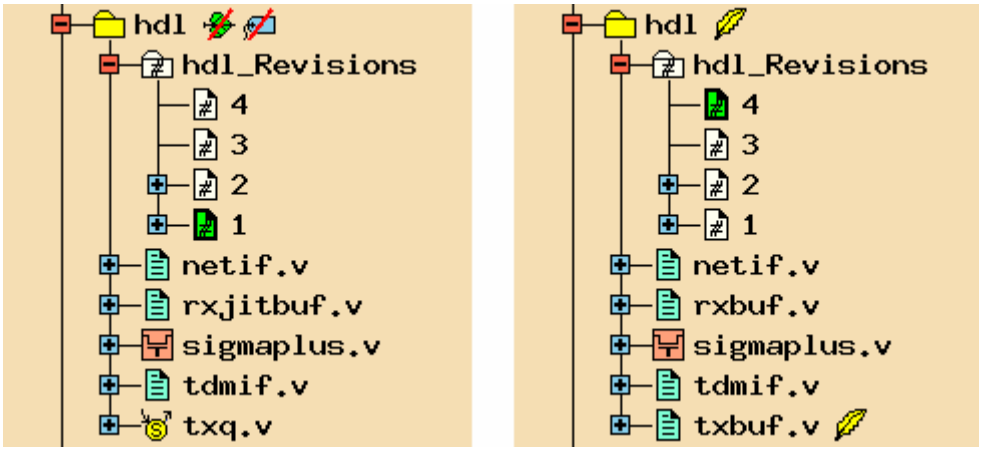

Figure 4: Two versions of a directory after files have been deleted, added, and renamed

Propagating Changes

When sharing thousands of files between several engineers it is imperative that each engineer gets the right set of files and versions. In an SoC design with multiple stages and groups it is impractical for everyone to be always accessing the latest versions of files. Such a methodology will be very restrictive and will discourage users from checking in changes for long periods of time. Instead a well defined but simple version tagging and promotion methodology should be adopted.

One simple methodology would be to define a tag for each stage of the design. For instance let us say there are three stages in a design flow – design, verification, and layout. You could then define three tags say – design_done, verify_done, and layout_done. The design group will tag a consistent set of revisions of design files to 'design_done'. The verification engineers will use the 'design_done' tag to create workareas for verification while the design engineers continue development. The set of revisions that successfully passed verification is then tagged 'verify_done'. The engineers further down the design chain can then use this tag to create their workarea.

Once the rules of engagement are understood and followed, data can be shared without confusion and errors. Keeping the rules as simple as possible is important otherwise they will either not be correctly followed or will add undue overhead. If more complex procedures are required, then scripts can be written to simplify the day to day DM tasks for users. With ClioSoft's SOS, pre and post event triggers can automatically run user-defined scripts to help streamline the design process. For instance, a pre-event trigger on the check-in command can run a lint check on Verilog files before they are checked in. This can prevent changes with syntax errors from being propagated to others thereby improving productivity.

Access Controls

Access controls are typically required for the following reasons:

- To protect intellectual property especially when outside consultants are also working on a project.
- To prevent accidental changes by other team members.

The access control requirements for each team are often unique and different. The DM system should provide a flexible model that can be used to control who has read and modify access to which parts of the project. Furthermore each engineer plays a role in the team and has some privileges. For instance, a verification engineer may have the privilege to tag all the design files but not to checkout and modify the files. The DM system should be set up with assigned roles and privileges to prevent users from accidentally performing an operation they are not supposed to do.

Recording Milestones

It is prudent to record design data configurations when certain milestones are reached. The most important milestone to record is of course the tapeout. A snapshot of the design data should be taken so this exact configuration which includes the right set of files and directories and their current revisions can be recreated in the future even if the project continues development to create other variants of the design.

ClioSoft's SOS DM system can create the state of the project at any time in the past. However, taking a snapshot ensures that particular milestones are recorded and can be exactly recreated. A snapshot results in a frozen label being assigned to the current revision of each file and directory. Since it is not expensive in terms of disk space or time, snapshots can be taken as often as necessary.

Usage Scenario at AMCC

SOS has been used by AMCC (formerly Quake Technologies) for more than 5 years. It was initially used for revision control of Verilog RTL code within the digital design team. When ClioSoft introduced SOS via DFII the concept of revision control was introduced to analog and mixed signal designers using the Cadence Virtuoso flow.

The introduction of SOS within an analog-mixed signal design team required changes in methodology in handling the design data. The use of separate libraries owned by individual designers with revisions maintained by using ad-hoc naming conventions was replaced by the concept of shared libraries under SOS control. This allowed partitioning of the design data into functional blocks with multiple user access.

Creating groups of team members and assigning access controls is essential in preventing accidents. Users are assigned to RTL, Analog, or Layout groups. All project members are given read access to all the design data but only group members are allowed to checkout and modify the data owned by their group.

SOS is also used for revision control of the design reference libraries, IP libraries & design documents. The CAD group developed a concept of SOS mirrors which provide read-only access to these libraries. SOS mirrors have also been used to provide a controlled environment for design documentation published to the internal intranet.

The SOS tagging feature is used extensively during the design process to indicate major milestones. At tape out, the hierarchy trees of all the design data is populated and tagged. Tagged data is then used to select the design data for creation of a snapshot for permanent archive. Use of these tag and snapshot features allows designers to continue creating and enhancing the design data without the fear of being unable to recreate the taped out version.

Branches have been used for parallel development. SOS support for branching folders has also been used to present a simplified or pruned view of the project data to different members of the team depending on their needs. For instance, this method is used to hide analog and digital design databases from Managers, Marketers or others who only want to access documentation, datasheets, and evaluation areas within the project repository.

Conclusion

Managing design data efficiently and accurately is critically important to the successful completion of a design project. The combination of a reliable infrastructure, with a powerful data collaboration platform such as ClioSoft's SOS, its integration with the EDA environments and a well defined methodology along with adequate training can help streamline the design flow. This will soon show a significant ROI by dramatically increasing team productivity, helping the design team meet project deadlines and significantly reducing the chance of expensive re-spins.

ClioSoft, Inc.

43551 Mission Blvd, Suite 107

Fremont, CA 94539

510-440-8424

www.cliosoft.com

AMCC

215 Moffett Park Drive

Sunnyvale, CA 94089

408-542-8600

www.amcc.com