

Open Systems Speed Easy EDA Tool Integration: Adding ClioSoft DDM to Synopsys Custom Designer

Authors:

- Srinath Anantharaman, ClioSoft
 - Les Spruiell, Synopsys
-

Historic challenges of tool integration

Electronic design flows are very complex. No single electronic design automation (EDA) vendor has all the answers. The Design Automation Conference in 2008 had nearly 250 exhibitors. It is no wonder that tool interoperability and integration is one of the most significant problems of EDA.

Traditionally, the major EDA vendors have tried to protect their turf with closed systems and proprietary data formats. Integrating a third party tool into a design flow has some major challenges. Integrating a design data management (DDM) system has additional challenges because it affects all the tools in the design flow.

The foremost issue for a DDM system is accessing and interpreting the design data. Typical EDA design flows often have decades of legacy. Design tools save design data in proprietary formats. Data for a design is often saved in multiple files with program-generated file names. Design tools in the flow also create several run files that are mixed in with the design data.

Providing an integrated user interface is very important for a good user experience. EDA tools are typically configurable and allow customers and partners to add additional functionality and GUI. However, major EDA vendors typically each have their own proprietary extension languages.

A DDM solution is not a point tool. It controls all the design files produced by the tools in the design flow. This presents the additional challenge of making all the tools in the design flow aware that a DDM system is being used so that they take appropriate action before modifying the design data.

The rise of open systems in EDA

Early in its history, the EDA industry primarily delivered single-purpose point tools created by widely disparate organizations. During this time, little thought was given to the use of these tools in concert with each other. With the rise of more complex semiconductor processes in the early 1990s, it quickly became apparent that more integrated flows were required and the development of tools had to change to accommodate this need.

Getting two disparate tools to work together requires a significant investment in interface programs to move data between the tools. The first-order industry response to the problem was the development of a number of "standard formats" for representing libraries, netlists, extraction results, design rule checking (DRC) and layout versus schematic (LVS) errors and so on, and served well for a number of years. It served well until the number of tools in a

flow increased, causing the number of unique interfaces to grow rapidly, causing a significant drain on both EDA vendors and internal CAD teams.

A major step towards solving the integration problem was the efforts on "CAD Frameworks." This approach sought to implement a standardized interface, command and control layer for each application so that each could be "plugged" into the top-level framework. The framework approach met with good success for the framework company with their detailed knowledge of the interface and their ability to extend it when needed to simplify the addition of a new tool.

Frameworks still didn't deliver on their promise, however, because they were implemented (much like the point tools of old) as disparate, proprietary systems. The rest of the EDA industry was still faced with creating an interface for each different framework and, because of the proprietary nature of the integration languages, very little code could be reused. Legions of programmers were still required to develop and maintain the interfaces between tools in spite of the framework's benefits.

During the late 1990s, EDA customers began to settle on one or two freely available integration languages. Most notably, Perl, Python and TCL became the popular choices for developing the bulk of the "glue" that stitched EDA tools together. These languages are well documented and supported and provided the industry with an alternative to the previous choices of closed, arcane languages for integration.

In the early 2000s, two additional approaches caused a radical change in the integration problem. The first of these was introduction of the OpenAccess database (OA) that allowed both customers and EDA vendors to resolve the bulk of the data interchange problem. Two programs written to the OA specification can interchange data without translation. The second approach to the problem was the rise of core-level application programming interfaces (APIs) that allow developers to more tightly integrate new products, with better control.

Today, integrators are offered a new alternative to the world of closed frameworks. This world provides the tools needed to develop easy integrations through well-understood languages and documented interfaces and to bypass the need for excessive effort in developing such systems.

Open architectures of Synopsys Custom Designer and ClioSoft SOS

Synopsys Custom Designer

In keeping with the above new approaches to tool integration, Synopsys released Custom Designer, a complete custom and analog/ mixed-signal (A/MS) integrated circuit (IC) design solution. Architected from the ground up for productivity and usability. Custom Designer is also based on OpenAccess, supports the TCL scripting language, and offers planned APIs for tool integration. Custom Designer provides everything a tool integrator needs.

- **OpenAccess:** Custom Designer is natively based on the OpenAccess database from Si2. This provides tool integrators with the important feature that the in-memory model is also native OpenAccess, which means that a tool written for one OA application can be directly adapted to another OA application, usually with minor modifications.
- **TCL Extension Language:** Custom Designer uses vanilla TCL as an extension language and provides a well-documented API to its internal functions. Additionally, Custom Designer's TCL interpreter provides for command and argument auto-completion which reduces the time developer's spend looking up command syntax.

- **Extensible Menus and GUI:** Custom Designer's menus are shipped as open source files that allow both developers and users to add to or modify the menu system as needed to complete tool integration. The GUI is also extensible through a set of documented commands allowing developers to craft new dialogs as needed.
- **Netlist-based Tool Integration (NTI):** Many EDA tool integrations are based on netlists and Custom Designer provides a complete, programmable netlister out of the box. The netlist formatting procedures are open source and shipped with the system for easy modification. Additionally, command and control is provided to allow triggering of user-defined actions before, during and after the netlist creation.

ClioSoft SOS Design Data Collaboration platform

Design teams often use EDA tools from multiple vendors. Some companies also have their own in-house tools to meet certain specific needs. However, all the design data for the entire design flow should be managed under the same DDM system. ClioSoft's SOS Design Data Collaboration platform has an open architecture that makes it easy to integrate it with all the tools in the design flow. SOS has the following integration points:

- **Command line:** SOS supports a full command line interface to execute commands or query status. It does not require a proprietary or even an open source interpreter such as TCL. Therefore SOS command can be invoked from any shell, script, or program.
- **Event Triggers:** Post- and pre-event triggers can be associated with any SOS command. This allows for automatic invocation of programs or processes when an SOS command is executed.
- **Custom Attributes:** Project administrators can track and monitor design metrics using custom-defined attributes. For example, project leaders could track the number of issues found in a module, code coverage metrics, power measurements, etc.
- **TCL/Tk GUI:** The SOS GUI is built with the open standard TCL/Tk that allows users to easily add additional functionality to the SOS window.
- **Universal DM Adaptor (UDMA):** A unique rule-based technology allows the SOS administrator to quickly and easily define the relationship between design objects and the data files produced by any design tool. Using the specified UDMA rules, SOS automatically determines and packages co-managed file sets into user-recognizable composite design objects. This UDMA technology enables version control of design data from almost any application at abstraction levels that designers use (libraries, cells, views, etc.) while eliminating the tedium of sorting through a complex set of tool-generated files.
- **Programming API:** A published C API implemented in a shared library can be used to create a really tight integration with any design flow.

ClioSoft's architecture provides several options for integrating SOS DM into the design flow. A loose but quick integration can be built into SOS using the Universal DM Adaptor enhanced with triggers, attributes and scripts. A really close integration, where the DM commands are directly plugged in to the tools in the design flow, can be developed using the API but would require more effort and an open architecture for the design flow.

Integration effort

The open architectures of Custom Designer and SOS provide a lot of flexibility in creating an integrated solution. ClioSoft's Universal DM Adaptor technology allows a quick integration to manage Custom Designer's OpenAccess database.

Universal DM Adaptor

Custom Designer saves data in the OpenAccess standard data format. An OA library is a directory with sub-directories for each cell. Each cell has a sub-directory for each view (schematic, layout, etc.) which contains a set of co-managed files that together form the view. Managing this data presents two major problems for the designer:

- A cell view is made up of multiple files. For example, a schematic cell view is made up of files such as 'master.tag,' 'sch.oe,' 'data.dm.' Additionally the design tools create temporary run files for backups, locking, etc. such as 'sch.oe%' and 'sch.lck.' These are files generated by the tools and each tool produces a different set of files. The designer is working at the higher level abstractions of libraries, cells and view and cannot be expected to know which files should be placed under revision control and which should not.
- A single cell view is made up of multiple co-managed files. With each design iteration/version, some or all of these files may be modified, new files may be added and some files may be removed. Cell views are composite objects that must be managed and versioned as such.

If designers attempt to manage these collections of files manually then they are certain to make errors. These problems were solved by writing Universal DM Adaptor rules that do the following:

- Recognize directories that are OpenAccess libraries.
- Determine the views in the cells.
- Include the set of files that make up the cell view.
- Exclude the files that should not be managed.
- Package and present the set of view files as a composite design object that the user recognizes as a cell view.

Figures 1 through 3 illustrate how UDMA is used to manage Custom Designer design data.

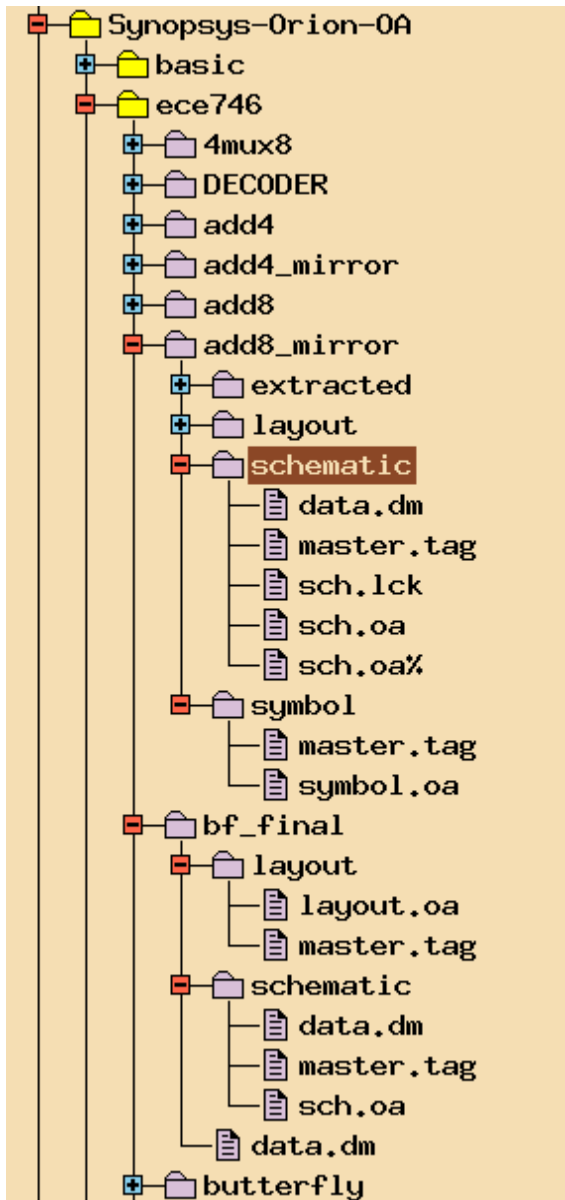


Figure 1: Custom Designer library viewed before applying UDMA rules

```
-- UDMA rules for Synopsys Custom Designer
```

```
package CustomDesigner {
```

```
  libid matchany
```

```
    "*/data.dm";
```

```
  basename globany
```

```

    "{*}/master.tag",
    "{*}/*.oa";
packname use "$1";
include globplus "$1/*";
exclude glob
    "*lck",
    "*%";
}

```

Figure 2: Example UDMA rule for Custom Designer OA libraries

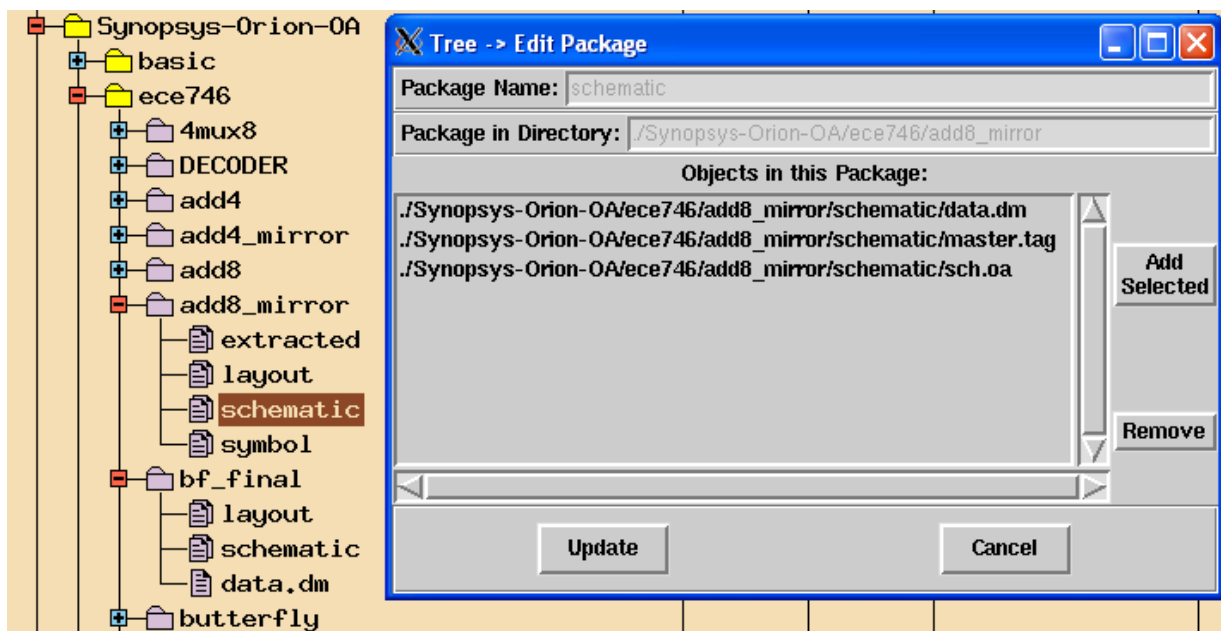


Figure 3: Custom Designer library viewed after applying UDMA rules

Using the UDMA rules specified, SOS is able to automatically recognize the Custom Designer libraries, pack the necessary files to present a cell view as a single object instead of several files and also exclude the files that should not be managed. Custom Designer can now easily check-out, modify, and check-in the cell views without worrying about the physical files.

This UDMA-based integration requires no programming and is available now.

Summary

Designing tools for interoperability has become mandatory for the EDA industry. The initial effort required to add interoperability in early tool development is clearly outweighed by the time savings later. Now, EDA vendors, CAD groups and even individual designers have the capabilities they need to deliver highly productive design flows with reduced effort and errors.

With ever increasing design complexity, engineering teams have to constantly adjust design flows, upgrade software and make the best-in-class tools from different vendors work together. Without a well integrated flow, designer productivity will be impacted. Adherence to standards enhances tool inter-operability. Standards and open architectures allow internal CAD teams to create the 'glue software' to quickly integrate a new tool into the flow. Published APIs help EDA vendors create a tight integration with popular tools and flows. Custom Designer's support for the OpenAccess database and ClioSoft's Universal DM adaptor enables version control and management of design libraries with just a simple configuration file. The open API and customizability built into Custom Designer and SOS may allow Synopsys and ClioSoft to create a tighter integration in the future.

About the authors:

Srinath Anantharaman is the CEO of ClioSoft, Inc., a company he founded in 1997. ClioSoft's SOS Design Data Collaboration Platform enables efficient management of design data from concept through tape-out and improves global team productivity. Srinath has over 25 years of experience in the electronic design automation industry. He was previously a partner at Proxy Modeling, a consulting company focused on front end design and methodology. Prior to that, Srinath held engineering and management positions at Synopsys, ViewLogic, Vantage, and Silvar-Lisco. He has an MSEE from Washington University in St. Louis and a BTech from the Indian Institute of Technology in Kanpur.

Les Spruiell is a member of the custom design marketing group at Synopsys, Inc. Les's experience includes 30 years developing, marketing and supporting EDA tools in a variety of roles at a variety of companies. Most recently, Les was director of corporate applications at Xoomsys, a member of the Virtuoso Platform marketing team at Cadence and co-founder and vice-president marketing at Antrim Design Systems.