Use of System Link Design for Multi-board systems

IBM

Samuel Yang

Session # 8.6

Presented at

cadence designer network

cdn LIVE™

Silicon Valley 2007

Abstract
Today's complex system architecture calls for a design process that can manage an entire system from processor card to backplanes to I/O cards. Bus specifications need to be met across the entire path of the net, not only on a single board file. Limitations of the current tool include constraint management—so far, constraints can only be managed one board at a time. A dynamic solution is needed so constraints can be changed and updated in real time across all the board files. This gives the session a high level control over the system constraints. The session created scripts to extract the information from a system link. This information was analyzed and new constraints were created and imported back to the Allegro board file. Further enhancements to the import/export routine will significantly improve efficiency. The session can save hours a day using an automated extract, import/export routine in conjunction with the system link. The session can work the system design in one sitting, versus working on each board individually which easily double or triples the amount of time spent in a design, based on the number of boards in the design.

Introduction

In the engineering industry today, the card design process is extremely complex. There are usually many circuit cards that come together to form one major system. The Cadence environment allows users to manage the electrical constraints using Allegro Constraint Manager. However, the current software is limited to a board by board analysis. Cadence utilizes a system design link to analyze an entire system. This gives the user an overall view of the system, but does not allow constraint management over the entire system. This paper explores using automated scripts to extract information from the system link, analyze it, and import changes back to the Allegro board. The new method can easily cut the amount of time it takes to analyze and create new constraints in half.

Prior Art

Often in the design process, the system is broken up into different cards: backplane, processor card, I/O cards, etc. This is done so that multiple groups can work on separate cards in parallel. Constraints taken from published specifications are usually meant for the full length of the signal path. The full path includes the length from the backplane +

processor card + I/O card + etc. Since the system is broken up into those different cards, the constraints need to be sized on a per board system.

A complete solution for the problem does not exist today. Allegro's system link allows the user to take a snap shot of the entire system. It provides net names, their constraints, and any associated errors. Yet, there is no way to edit each card in the system from the system link view. New constraints must be manually entered in each independent board. This can be difficult when there are multiple designers working on the separate cards.

The current process has a "trial and error" feel to it. Each card is worked on independently and the result can be improper skew on certain nets. The problem occurs when the engineer designs his or her board to the specification, but over looks the fact that the specification is in reference to the entire path of the net, not only on one card. As a result, communication needs improvement over different teams to make sure the specification is met as a whole (over the entire system). Teams need to have a process to make sure the constraints are in sync. The skew for each net needs to be divided up into different boards. When the physical placement of the nets is done, the engineer may find out that there is a lot of slack in the constraints. On the other hand, a different team may be in need of more skew in their design. A decision is made to cut back length on one board and give more length to the other board. Next, the constraints need to be updated for each board and imported back into their respective designs. Then the constraints need to be rechecked and the board rewired to fit the new parameters. The "trial and error" portion of this method resides with the constant back and forth between different teams in order to reach the best compromise net lengths across the system.

There is a lot of churn in the process. Although the system link provides an overall view of the constraints in the system, the user cannot change the constraints as needed. The constraints must be divided among all the board files and implemented again, after which the design link must be refreshed in order to see the new results.

The biggest disadvantage to the current method is time. Even though the teams are working in parallel, the same task (updating the constraints) is repeated as many times as the number of boards in the system. (Eg. The entire system includes 1 backplane, 1 processor card, and 2 I/O cards. Therefore, the constraints need to be updated 4 times.)

Another disadvantage is readability. Each board file may have different naming conventions for net names. There is no easy way to tell which net on Board 1 continues to which net on Board 2. Constraint Manager gives the user a generic net name and the total etch of the net, but does not show each individual net lengths per board.

All of these things make the system link difficult to manage.

New Method

The proposed method uses a series of scripts to automate the process of analyzing an entire system, finding the problem areas, and inputting new constraints. It builds upon the current system link process and provides improvement. From there, the user can chose

which nets he or she wants to change. The script will automatically adjust the topology file for the user. Then, with a simple update, the new rules are applied to the design. The scripts use a combination of Allegro Skill[1] code and Perl code[2].

The first step follows the old method: create a system link using Cadence Allegro. Then, the first script is executed. The design link is analyzed and a net list of the entire design is formed. The list is broken up into different columns, showing all nets within each board file and the nets' associated etch length. The end columns keep a cumulative total the individual boards' net lengths.

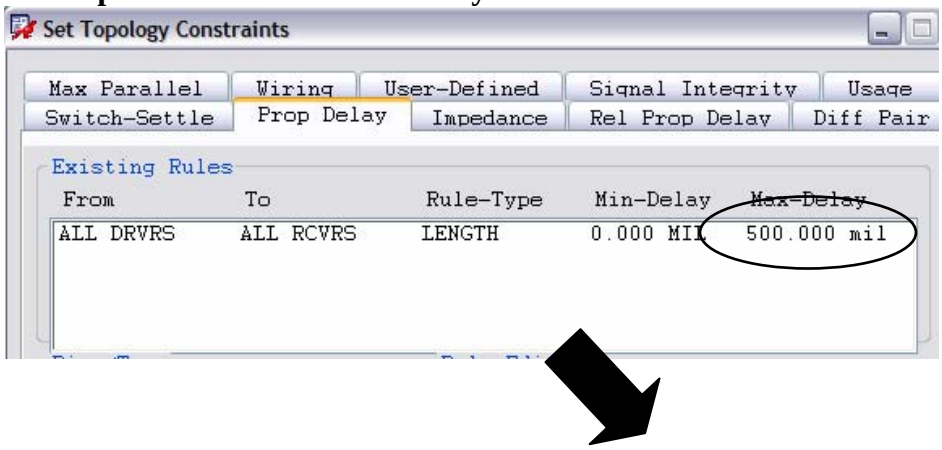
As shown in Figure 1 below, the spreadsheet includes all the nets in each board file, as well as their lengths, connector pin lengths, and the total length.

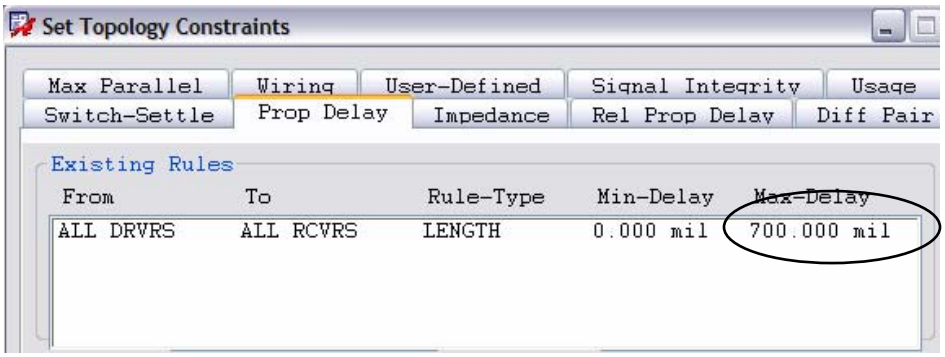
Figure 1. An example spreadsheet with a 2 board design.

Board 1	Etch Length (mils)	Board 2	Etch Length (mils)	System Etch Length (mils)	System Pin Length (mils)	System Total Etch Length (mils)
Net 1	12000	Net_2 1	1330	13330	300	13630
Net 2	8500	Net_2 2	1350	9850	1000	10850
Net 3	11000	Net_2 3	1280	12280	500	12780
Net 4	12500	Net_2 4	1400	13900	400	14300
Net 5	11800	Net_2 5	1180	12980	600	13580
Net 6	21000	Net_2 6	1300	22300	900	23200

A given board design usually has hundreds or thousands of nets. The user may only be concerned with a certain bus, or group of nets. The second script prompts the user for which bus he or she is interested in. A file is generated, deleting unwanted nets from the spreadsheet, and leaving the user only information that pertains to his or her needs. At this point, the engineer can decide whether the specification was broken or not. If so, the user has the option of changing values in the spreadsheet. Finally, the third script is executed and the new constraints are updated to their respective topology files as shown in Figure 2 below.

Figure 2. Before and after pictures of the changes the script made. In this case, the user inputted a different max-delay value of 700 mils.





At this point, the new topology files can be imported to their respective board files and the process is complete.

The greatest benefit by far is obviously the automation. Creating a spreadsheet of all the nets in the system by hand would be tedious and time consuming. The best time saver is automating the constraint updates. Anyone with experience creating constraints knows that updating each constraint by hand could take hours—especially if you have thousands of constraints to update. This automation process is incredibly faster than the traditional "by hand" method (see Figure 3 below).

Figure 3. Comparison of old versus new method.

	# of nets	Create spreadsheet	Update topology
By hand	300	1 hour	1 hour
Script automation	300	**1 second**	**1 second**

3600 times faster!!!

Better readability is also addressed in the new method. The first script creates an easy to read spreadsheet that lines up each net's entire path. Every board in the entire system is listed and the net names fall under each one in alphabetical order. The net names are also paired with their etch lengths.

There are, however, a few weaknesses in the new method. One weakness is that there are multiple scripts that need to be executed. This can provide the user with problems if he or she doesn't have the latest version of the script, of if the user loses track of where the scripts are stored.

Another weakness is the process is not fully automated. The user has to change the values of the constraints when a problem arises. The user also has to import the new topology files into the board file. But by far, the advantages outweigh any weaknesses in the proposed method.

Conclusions and Future Work

This method greatly enhances the system link process and saves a lot of time by automating many steps. All the time spent on data gathering and constraint updating is basically gone. The user also has an easy to read spreadsheet of the net list.

Work still needs to be done in order to fully automate the process. The Session is already working on converting all the scripts into Allegro Skill. This way, the entire process can live within the Cadence environment providing a better flow.

Additionally, introducing a mathematical algorithm can take out the human portion of the problem. As mentioned above, there is a human step involved with figuring out how much length to distribute among the board files. If there was a smart auto-algorithm, the calculations for determining the length for each board could be automated as well.

Finally, the overall system link software can be improved. Currently in the system link setup, the user can only view the system configuration. There is no option to modify constraints across all boards, and DRC errors are not updated in real time.

The ultimate goal is for a complete automated system link flow. The Session is confident that with these added features, the goal can be met.

Acknowledgements
[1] Allegro Skill code created by Larry Bowman, Cadence Application Engineer
[2] Perl Code created by Daniel Rodriguez and Samuel Yang, IBM Signal Integrity Engineers