DIGITAL INDUSTRIES SOFTWARE

Automated compliance analysis of serial links reduces schedule risk

Executive summary

Most high-speed serial links don't get verified once routing is complete because the process is time consuming and skill-intensive – and SI experts are in short supply. As a result, most serial channels are laid out according to rules, verified through manual inspection and released to fabrication without thorough analysis. Unverified channels can result in lengthy (and hectic) prototype debugging, board spins and schedule slips. Until now, there has been no other choice. This paper discusses an automated post-route verification process with HyperLynx that can verify all the channels in a design for detailed compliance with a SerDes protocol standard – automatically, overnight. This allows designers to find problems early in the layout process, when they're easier to correct, and release designs for fabrication with confidence, knowing all their serial channels have been verified.

Todd Westerhoff, Siemens EDA

Contents

Introduction

The Wikipedia article on Murphy's law (https://en.wikipedia.org/wiki/Murphy%27s_law) is a fascinating read. Edward Murphy was a test engineer at what is now Edward's Air Force base in the late 1940s, providing, among other things, instrumentation for rocket sled testing. The apocryphal story of Murphy's Law involves Edward Murphy blaming someone else for a wiring mistake that ruined test results – despite the fact that he had been asked to verify the test setup and had refused. The actual problem was small, but the seed had been planted, and the legend of Murphy's law grew, and thrives to this day.

Murphy's Law still gets discussed today because things that shouldn't go wrong do go wrong, and small mistakes that go undetected can have disastrous consequences. It happens with PCB design, too. Even after extensive pre-route simulation has been performed, layout rules have been defined in meticulous detail, and the PCB designer has done their absolute best to follow those routing rules, Murphy's Law is still in play. That's because no amount of pre-layout analysis can anticipate all of the changes and tradeoffs that occur during layout, and people still make mistakes. Things that shouldn't happen during layout do happen. The operative question is: "What should we do about it?"

We simulate the design in pre-layout as we believe it should be laid out. But at some point, we have to analyze the design as it actually was laid out. This is why systems engineering teams need to run post-route analysis and analyze all of their serial links before the design is sent out for prototype fab. Chances are something is wrong, somewhere. However, complete post-route analysis typically does not happen, which means design errors go undetected, leading to costly design respins and schedule delays.

In this paper, we explain why complete post-route analysis is typically considered to be too time-consuming and too expensive. We will compare traditional methods to automated compliance analysis with HyperLynx, showing how design teams can verify all of the serial links in their designs overnight, thereby improving design performance and reducing schedule risk.

The signal integrity analysis bottleneck

If post-route verification of serial links is so important, why are so many PCB designs sent to prototype fab without full verification? Part of the issue is the prevalence of serial links in modern products. Everything is packed with serial links today – computers, phones, smart watches, cars – and the list goes on. So there are a lot of designs, and a lot of links to verify. This leads into the second, and larger issue: there are simply not enough signal integrity experts to handle this much work. Signal integrity experts are often like artists – each one has their own style and approaches the task a bit differently. Much of what they do is based on detailed knowledge and experience, and it's individual. There really isn't anything like a signal integrity analysis assembly line – analysis flows aren't standardized, and as a result, they're not scalable. So it's like anything else with limited, highly skilled labor – too much work and too few people capable of practicing the art.

The result: companies must decide which sections of which designs merit an expert's time and attention. Those projects get expert assistance, the others must do without or wait until an expert is available. Even on a single PCB layout, this can create costly bottlenecks. Companies can't afford the resulting delays. Yet, they can't afford to let random errors slip through undetected into prototypes in the lab, where finding, isolating, and debugging signal integrity problems takes longer, costs more money, and is notoriously difficult. So, what to do?

Until now, PCB design teams have typically followed one of four paths for analyzing their designs after layout.

1. Send the board out for fabrication and hope for the best. The theory is that if manufacturer's guidelines have been followed, the design should work. However, how can anyone be sure all the design guidelines have been followed?
2. Visually inspect the layout to ensure design guidelines and best practices have been followed. This is certainly better than option 1, but visual inspection is tedious and time consuming, making it highly error prone. Design errors can be found this way, but it's still a hit or miss proposition.
3. Submit the design to an internal signal integrity expert for analysis. There are two requirements here: (a) there must actually be an internal signal integrity expert, and (b) the expert must have the time and tools available. Since there are too many designs and too few experts to go around, this is usually not the case. Yet even when an expert is available and their analysis shows problems that need to be corrected, the updated layout has to go back to the end of the queue, causing further delays.
4. Send the layout to an external signal integrity consultant. This is a way to bypass an internal analysis queue or run analysis when no internal expert exists. This will get faster attention, presumably, but any design changes will cost both time and money because consultants won't run that second set of simulations for free.

None of these are particularly great options. They either take on too much risk in order to get the design to fabrication earlier or impose lengthy delays in order to perform detailed signal integrity analysis. What's needed is a fast, reliable way to validate designs after layout, without having to wait for a signal integrity expert or external consultant.

So is it possible to validate all of a design's serial links for protocol compliance before sending the board out to fabrication, without a lengthy and highly skilled and labor-intensive process? To answer that question, we need to take a closer look at how channels are typically verified and see how that process can be improved.

Post-route verification of serial links

There are three essential steps in validating serial links before sending a design out to fabrication:
1. Electromagnetic modeling
2. Analysis
3. Results processing

Electromagnetic modeling: For any kind of analysis to be performed on the layout, an accurate interconnect model must be created for every channel model that will be analyzed and any channels that couple energy into it. Because the signal frequencies associated with serial channels are high, areas where the signals change layers will need to be modeled using a full-wave electromagnetic solver. A detailed model of everything in the signal's path, from device pin to device pin, will need to be created.

Analysis: Once we have a simulation model for the channel, we need to predict how it will behave. Analysis combines the channel model with suitable representations of the transmitter (Tx) and receiver (Rx) devices, along with details of the channel protocol (bit rate, encoding, etc.), to determine what the signal will look like at the end of the link. This process answers the question: "If we build this channel and operate it this way, what will the signal inside the receiver look like?"

Results processing: Oddly enough, analysis by itself is not the end of the line, because it doesn't tell us what we really want to know: whether our design will pass or fail, and by how much. We want our design to work, and we want it to have enough extra margin to be reliable in volume production. Thus, being able to definitively and quantitatively determine design margins is just as important as being able run analysis in the first place.

Let's take a closer look at each of these areas to determine how these steps are typically performed, and how HyperLynx can make the process better.

Channel modeling

If you're not familiar with full-wave electromagnetic solvers, you'd probably reload the entire board database(s) and try to solve the entire channel at once. The problem with this approach is that it is prodigiously expensive, both from a compute time and resource point of view. While it's possible in limited scenarios, it's rarely practical from a cost/benefit point of view.

The standard method for modeling serial channels is referred to as cut-and-stitch. It's based on the observation that, for the vast majority of a channel's length, signals travel using transverse electromagnetic mode (TEM) propagation. An effective way to model the channel is therefore to "cut" it into regions of TEM and non-TEM propagation, solve the different regions independently, then combine, or "stitch" the regions back together to create a model of the complete channel. It's true that this method involves critical assumptions and is less accurate than modeling the entire channel in a full-wave solver, but it's also true that, properly performed, the inaccuracies are small, and the cut-and-stitch method is much more efficient than its counterpart.

The trick with the cut-and-stitch method is correctly picking the points where the channel model will be cut. If you pick a point too close to a discontinuity, where the signal flow isn't TEM, then the combined channel model won't have the correct behavior, because you've violated the fundamental assumption of the method. The area in the cut region needs to include the cause of the discontinuity (a via, for example), the traces that lead away from it, and the signal's complete return path. If you cut too far away from the discontinuity, then the area you solve in the full-wave solver will be larger than it needs to be, and the solve process will take longer. Typically, the cut-and-stitch process is performed manually by an experienced signal integrity (SI) engineer, because they have the insight needed to make those tradeoffs.

When cut-and-stitch is performed manually, the number of channels that can be modeled is limited, because a manual process like this is time-consuming, and only a few people are capable of performing it. So, while cut-and-stitch is the typical method for analyzing serial channels post-route, typically only a few channels are modeled: the shortest, the longest, and a few other channels that are suspected as potential problems. It's also performed only a few times over the course of a board's design cycle because the process is too labor-intensive to be performed any other way. This brings us back to the problem we posited at the start of this paper: if all channels aren't verified, then there are potentially problems left undiscovered before fab out.

Automating the cut-and-stitch process makes it possible to model all the serial channels in a design for verification. The trick is still cutting the channel in the right places, but instead of requiring expert intervention, HyperLynx automates the process using its DRC engine to identify the extent of the return path around a signal discontinuity. HyperLynx then prepares the area for full-wave simulation by automatically creating the signal ports and setting up other parameters for the solver. With this automated process, HyperLynx can identify and set up hundreds of areas per hour for full-wave simulation. Once the areas are created, full-wave simulations can be run in parallel across multiple computers to reduce the time needed to solve all the areas in the system, which are typically a few hundred areas or more.

HyperLynx automates the "stitch" part of the process as well. Once 3D areas have been solved, channel models are assembled by combining lossy transmission lines of the proper length with solved models. Because some of the signal trace is represented inside the 3D areas, the length of the transmission line must be adjusted accordingly. With traditional, manual cut-and-stitch this is tedious and error-prone. With HyperLynx, this process is automatic. HyperLynx knows how much of the signal trace was included in each area and it adjusts transmission line lengths to compensate.

The result: HyperLynx can create interconnect models for hundreds of serial channels, automatically, overnight.

Analysis

IBIS-AMI simulation

There are two ways to analyze serial links after layout: IBIS-AMI simulation and standards-based compliance analysis. They both produce results that indicate whether the design will work or not, but they differ dramatically in terms of how hard they are to perform, how they model success or failure, how long they take, and how often they can be performed during the design cycle.

IBIS-AMI simulation is the more accurate method because it uses simulation models of the actual Tx/Rx devices that will be used in the system and how those devices will be configured. The Tx/Rx models are obtained directly from the Tx/Rx device vendor(s) as IBIS-AMI models. Unfortunately, it is often be difficult to get accurate, complete, well-documented IBIS-AMI models from a device vendor, and often they're not available in time to meet a project's schedule. That's the rub: an analysis that's more accurate isn't much help if it can't be performed when you need to make the design decisions that analysis is supposed to help you with.

IBIS-AMI simulation is more time-consuming and expertise intensive than compliance analysis. Individual IBIS-AMI models can support three different simulator interface methods: “Init,” “Getwave,” or both. Since there's a Tx and an Rx in every channel, there are 3*3 = 9 different combinations of model types, or “flows,” that an AMI simulator must support. These different flows offer different levels of accuracy, device modeling details, and statistical coverage. The possible flows for any given simulation will be determined by the capabilities of the particular vendor models, which will vary from vendor to vendor and model to model. AMI model completeness and documentation is also an issue. It can be hard to know if the model includes everything needed for analysis (for example, are jitter budgets included, and for which types of jitter?), which device behaviors the model does or doesn't represent, and how to configure the model's control inputs. Last, but not least, because AMI models are meant to reflect actual hardware, the user must provide the equalization settings to be used for simulation. For a model with five different sets of control inputs, that can lead to hundreds of combinations of device settings that have to be explored through simulation.

If that sounds complex – well, it is, because that's the price of accurately modeling vendor-specific device behavior in high-speed serial links. AMI models give vendors the ability to showcase how well their proprietary equalization algorithms will work on a customer's channel – without having to expose trade secrets. From the system designer's perspective, simulation with AMI models is often limited to the same people who run full-wave solvers: dedicated SI engineers who use these tools for a living.

A final problem with this approach is that it often gets delayed until layout is complete. Because the analysis is expensive, there's a strong desire to perform it only once with the hope that everything passes. The problem is that when the channels don't pass, changes to the board are more difficult. There's more to rip up and rework, and that means a hit to the schedule.

Protocol compliance analysis

The second way to analyze a serial channel is standards-based compliance analysis. This focuses on the channel itself and does not depend on the designer's choice of devices for the Tx and Rx. Where the behavior of the Tx and Rx are considered, compliance analysis assumes that these devices are merely compliant – they contain the minimal functionality required by the associated standards specification. Unlike AMI simulation, which can't be performed if IBIS-AMI models aren't available, compliance analysis is always possible, because it is based on the channel requirements in the protocol spec itself.

Another advantage of compliance analysis is that it runs quickly. IBIS-AMI simulations can run for a half hour or more per channel, depending on the model and analysis setup, while compliance analysis typically completes in less than a minute.

Still, significant challenges remain because there are dozens of protocol specifications, each of which is often hundreds of pages long. Just reading one spec is a huge task. As a result, performing the associated channel compliance tests has traditionally been a manual process that varies from company to company. Each company uses a different collection of tools and therefore creates their own modeling and analysis process steps. To complicate matters, there are several ways that compliance analysis is performed: some protocols apply a time-domain mask to simulation results, some apply frequency-domain masks and metrics to the channel model itself, while others use public-domain tools that take the channel model as input and perform their own analysis. Notable examples of the latter include IEEE COM (Channel Operating Margin), JEDEC JCOM (literally, JEDEC COM), and the PCI-SIG's Seasim.

And so – with dozens of protocols, hundreds of pages each, at least five different analysis methods – compliance analysis begins to look like less of a panacea and more like just another set of complicated analytical processes for use by dedicated SI engineers; albeit one that isn't reliant on Tx/Rx vendor simulation models.

As with channel modeling, HyperLynx addresses this problem through automation. The HyperLynx SerDes Compliance Wizard provides a single, consolidated workflow that supports all the different analytical methods for determining SerDes Compliance. The user specifies which protocol / variant to use, which tells HyperLynx the correct analysis methodology and which parameters to check. From the user's point of view, the compliance process is always the same: specify the channels to be analyzed, select the protocol / variant to use, and press Run. HyperLynx provides the broadest protocol compliance capability of any vendor, with support for 210 different protocol / variants in the 2.11 release. The parameters used for each protocol /variant are defined in a control file supplied with HyperLynx. Those parameters can be displayed during set up, and the user can adjust any of those values if desired.

Hundreds of protocol / variants supported with a single, automated flow. That's HyperLynx.

Results processing

When we run analysis there's really only one question we're ever trying to answer: "Will it work or not, and by how much?" Modeling and analysis might yield useful insights, but they don't tell us what we want to know: "If we build it, will work or not?" And answering that question isn't as simple as one might think.

In order to answer that fundamental question, explicit, measurable pass/fail criteria must exist, and analysis must produce results that can be directly compared to those criteria. That may sound simple, but it isn't. Let's dig into some of the details. Results processing is specific to how the analysis was performed, so we'll discuss results for IBIS-AMI simulation and compliance analysis separately.

IBIS-AMI Simulation

IBIS-AMI simulations predict signal behavior and, therefore, the eye opening at the sampling latch inside the receiver – where the received data is ultimately captured as ones and zeroes. So how hard could it be to figure out if the design works? After all, an open eye is an open eye, right? The answer is, not exactly. A latch requires margin to ensure data is captured correctly. The signal has to exceed an input threshold for long enough before (setup) and after (hold) the associated clock signal to ensure the data is captured correctly. Although IBIS-AMI simulations produce equalized waveforms and clock ticks, they don' t produce clock waveforms, so conventional methods of comparing clock and data signals for margin don't apply. Typically, what an IBIS-AMI simulator does is use clock tick information to center the equalized signal in the bit period and produce an eye diagram from that data, assuming the clock sampling time is in the middle of the UI. Then an eye mask is compared to the eye diagram, with the mask centered horizontally in the UI. If the inner portion of the eye doesn't impinge on the mask, the test passes. The smallest distance between the inner eye and the mask is the margin of the measurement (assuming the test passes). The eye mask, and the probability level it should be measured against, is model-specific.

But we're not done yet. Serial channels pass billions of bits per second, and the target bit error rate is often 1e-12 or lower. So plotting an eye diagram for a few million bits and comparing it to an eye mask doesn't answer the question of whether the design will work as intended because the sample size is too small. In order to determine if the channel works with the reliability (bit error rate) desired, we need to increase the sample size of the simulation. That can be accomplished by running the simulations in statistical (Init) mode, which effectively simulates an unlimited (or very large) number of bits. But the vendor models may not support that, or their accuracy in Init mode may be reduced. If we need to run simulations in time-domain mode, it may be possible to extrapolate the results from the actual number of bits simulated, but that extrapolation can be tricky. We'll typically only simulate a few million bits in the time-domain, but we'll often want to see BER figures of 1e-12 or less. The extrapolation must predict margins at 1e-12 from a sample of only 1e-7 or so – that's five orders of magnitude. Can it be done? Sure! For better or worse, simulations always produce a result. Will that extrapolation be accurate enough to put a design into production? That depends.

And then there's the issue of jitter. All Tx and Rx devices introduce jitter that reduces design margin. For a margin estimate to be accurate, it must include a complete set of jitter budgets for both devices. . Jitter information isn't always included with the simulation models, so often the user has to research and add it. Finally, and perhaps most importantly, signal post-processing and reporting isn't standardized by the IBIS-AMI specification; it varies from simulator to simulator, and the details matter. Thus, the way to determine if an IBIS-AMI simulation passes or fails also varies with each model, and this parameter is not part of the IBIS-AMI specification itself. That means signal integrity experts need to understand the details of how the simulator post-processes and presents data so that they can make the right pass/fail measurements.

None of this is to say that AMI models shouldn't be used for post-route verification; but there are a lot of details that must be correct to produce a worthwhile result, and those details vary from model to model. That means that meaningful AMI analysis is normally the realm of a full-time SI engineer, and someone who knows the IBIS-AMI spec and the simulator they're using inside and out. There's a big difference between simply running an AMI simulation and running one that produces accurate, actionable results.

Protocol compliance analysis

With compliance analysis, both the analysis process and results processing requirements are fully defined at the outset. Since there are no vendor Tx/Rx models, the analysis process doesn't vary between devices and projects. Since the compliance requirements are defined as part of the protocol spec, the results that need to be produced and how they need to be interpreted is also known.

That's a huge benefit! IBIS-AMI simulations have the potential to be more accurate, but compliance analysis is more reliable, because you can always run it, even when vendor models aren't available. Furthermore, because compliance analysis with HyperLynx is also faster and easier than IBIS-AMI simulation, it makes sense to always run compliance analysis first, before investing time and effort with IBIS-AMI. If compliance analysis reveals a design problem, you can find and fix it faster. If compliance analysis shows your design has plenty of margin, you may be able to defer or skip IBIS-AMI simulations, because a design that works with a nominal (spec compliant) device will most likely work with a device that exceeds the spec – and what device vendor doesn't claim that their device is better than the standard?

Most importantly, compliance analysis with HyperLynx produces a detailed report that shows which signals passed, which signals failed, and by how much – and those are the fundamental questions that need to be answered. The report includes plenty of other details as well, providing supporting and diagnostic data for further use once the most important questions have been answered.

Automating compliance analysis with HyperLynx

In this paper, we've examined traditional methods for post-route verification of serial channels and compared them to automated compliance verification with HyperLynx. Traditional flows are largely manual, consist of multiple steps, and are run by SI experts. If we combine the three steps (EM modeling, analysis, and results processing) into a process chart for a traditional flow, it looks something like what is shown in Figure 1.

The red arrows indication parts of the flow where data must be examined for accuracy and the parts of the process repeated if things need to be adjusted. Again, this diagram shows a compliance analysis flow using a traditional methodology. An IBIS-AMI flow would have fewer elements, but the simulation step itself would be more complex.

HyperLynx can automate the entire post-layout verification process because Siemens provides all the necessary EDA tools in the HyperLynx family, integrated with a single, automated workflow. This includes the automated identification of critical areas that need to be modeled with a full-wave solver (cut), assembly of the full channel model from individual pieces once everything has been solved (stitch), analysis of the resulting channel models for compliance (analysis), and formatting the results to show

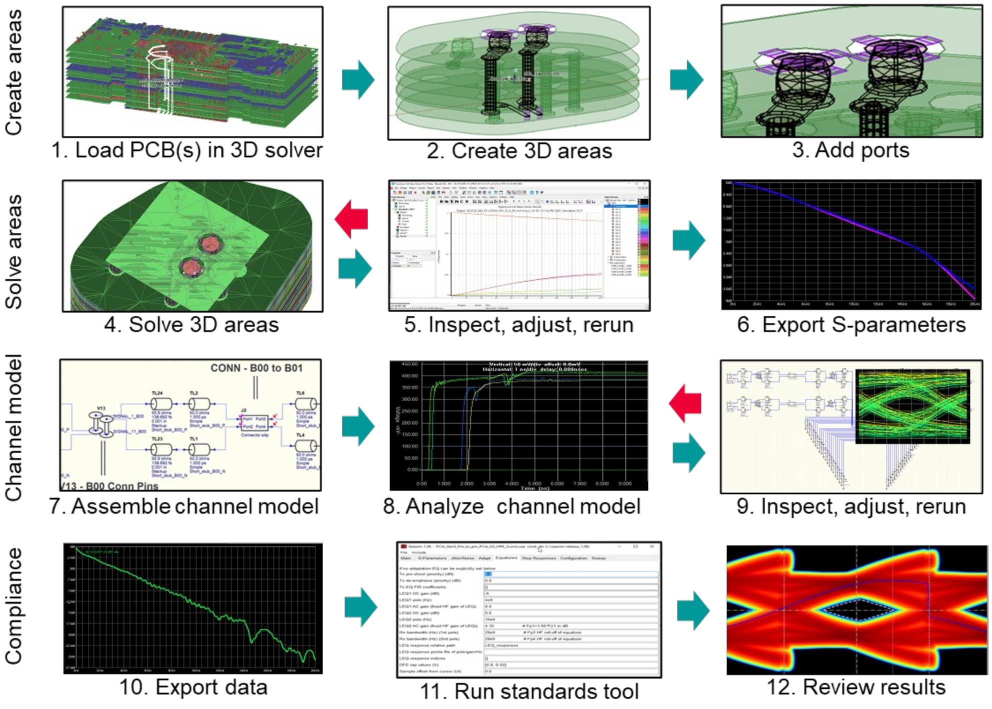


Figure 1: Process chart for a traditional compliance analysis flow.

which channels passed, which channels failed, and by how much (results processing). The HyperLynx process for post-route serial channel protocol verification looks like what is shown in Figure 2.

This automated process means all the channels in a large system design can be modeled and analyzed. The EM modeling process can be accelerated by running multiple solvers in parallel, so users can control the run time versus required-resource tradeoffs based on their project needs. Most importantly, HyperLynx tells you exactly what you want to know: which channels pass, which channels fail and by how much – all in a detailed report that includes frequency and time domain plots and eye diagrams.

Everything you need to know, in one place, organized and cross-referenced.

That means you can analyze all the channels in your design for protocol compliance – automatically, overnight. It's fast and easy enough to analyze your channels to find problems while the design is still in layout, instead of waiting until layout is complete and rework is more expensive.

Can HyperLynx really make it that easy? Contact us to see for yourself – we'd love to show you how automated compliance analysis with HyperLynx can help you verify your entire design overnight!

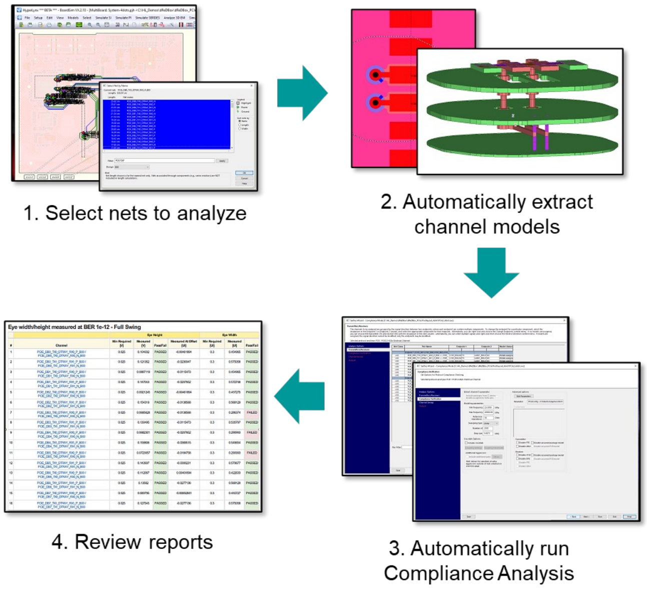
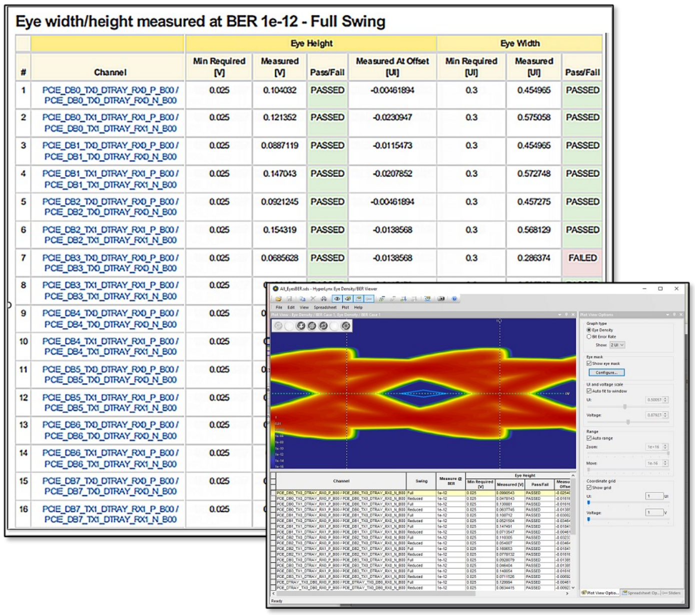

1. Select nets to analyze
2. Automatically extract channel models
3. Automatically run Compliance Analysis
4. Review reports

Figure 2: HyperLynx process for post-route serial channel protocol verification.



Eye width/height measured at BER 1e-12 - Full Swing

		Eye Height				Eye Width		
#	Channel	Min Required [V]	Measured [V]	Pass/Fail	Measured At Offset [UI]	Min Required [UI]	Measured [UI]	Pass/Fail
1	PCIE_DB0_TX0_DTRAY_RX0_P_B00 / PCIE_DB0_TX0_DTRAY_RX0_N_B00	0.025	0.104032	PASSED	-0.00461894	0.3	0.454965	PASSED
2	PCIE_DB0_TX1_DTRAY_RX1_P_B00 / PCIE_DB0_TX1_DTRAY_RX1_N_B00	0.025	0.121352	PASSED	-0.0230947	0.3	0.575058	PASSED
3	PCIE_DB1_TX0_DTRAY_RX0_P_B00 / PCIE_DB1_TX0_DTRAY_RX0_N_B00	0.025	0.0887119	PASSED	-0.0115473	0.3	0.454965	PASSED
4	PCIE_DB1_TX1_DTRAY_RX1_P_B00 / PCIE_DB1_TX1_DTRAY_RX1_N_B00	0.025	0.147043	PASSED	-0.0207852	0.3	0.572748	PASSED
5	PCIE_DB2_TX0_DTRAY_RX0_P_B00 / PCIE_DB2_TX0_DTRAY_RX0_N_B00	0.025	0.0921245	PASSED	-0.00461894	0.3	0.457275	PASSED
6	PCIE_DB2_TX1_DTRAY_RX1_P_B00 / PCIE_DB2_TX1_DTRAY_RX1_N_B00	0.025	0.154319	PASSED	-0.0138568	0.3	0.568129	PASSED
7	PCIE_DB3_TX0_DTRAY_RX0_P_B00 / PCIE_DB3_TX0_DTRAY_RX0_N_B00	0.025	0.0685628	PASSED	-0.0138568	0.3	0.286374	FAILED
8	PCIE_DB3_TX1_DTRAY_RX1_P_B00 / PCIE_DB3_TX1_DTRAY_RX1_N_B00	0.025						
9	PCIE_DB4_TX0_DTRAY_RX0_P_B00 / PCIE_DB4_TX0_DTRAY_RX0_N_B00	0.025						
10	PCIE_DB4_TX1_DTRAY_RX1_P_B00 / PCIE_DB4_TX1_DTRAY_RX1_N_B00	0.025						
11	PCIE_DB5_TX0_DTRAY_RX0_P_B00 / PCIE_DB5_TX0_DTRAY_RX0_N_B00	0.025						
12	PCIE_DB5_TX1_DTRAY_RX1_P_B00 / PCIE_DB5_TX1_DTRAY_RX1_N_B00	0.025						
13	PCIE_DB6_TX0_DTRAY_RX0_P_B00 / PCIE_DB6_TX0_DTRAY_RX0_N_B00	0.025						
14	PCIE_DB6_TX1_DTRAY_RX1_P_B00 / PCIE_DB6_TX1_DTRAY_RX1_N_B00	0.025						
15	PCIE_DB7_TX0_DTRAY_RX0_P_B00 / PCIE_DB7_TX0_DTRAY_RX0_N_B00	0.025						
16	PCIE_DB7_TX1_DTRAY_RX1_P_B00 / PCIE_DB7_TX1_DTRAY_RX1_N_B00	0.025						

Figure 3: HyperLynx detailed reports present everything you need to know, in one place, organized and cross-referenced.

Siemens Digital Industries Software

Americas: 1 800 498 5351

EMEA: 00 800 70002222

Asia-Pacific: 001 800 03061910

For additional numbers, click here.

Siemens Digital Industries Software helps organizations of all sizes digitally transform using software, hardware and services from the Siemens Xcelerator business platform. Siemens' software and the comprehensive digital twin enable companies to optimize their design, engineering and manufacturing processes to turn today's ideas into the sustainable products of the future. From chips to entire systems, from product to process, across all industries, Siemens Digital Industries Software is where today meets tomorrow.

siemens.com/software

84725-D4 5/23 A