CADENCE

Design SUPERCON97

Signal Integrity Engineering in High-Speed Digital Systems

Donald Telian

Cadence Design Systems
2655 Seely Road
San Jose, CA 95134

1997
High-Performance System Design Conference

Abstract

Far from a well-understood science, Signal Integrity Engineering methods and practices are still being defined. And yet, ensuring electrical integrity is the next critical piece of the high-speed digital design puzzle. This paper addresses the 7 roles the Signal Integrity Engineer must play at various stages of the hardware development cycle. We'll study the correct application of Signal Integrity theory, tools, and models to engineering tasks ranging from originating new standards to fire-fighting bugs that arise on the assembly line. Roles are defined, and practical design tips are suggested from real-life successful execution of these tasks.

Authors/Speakers

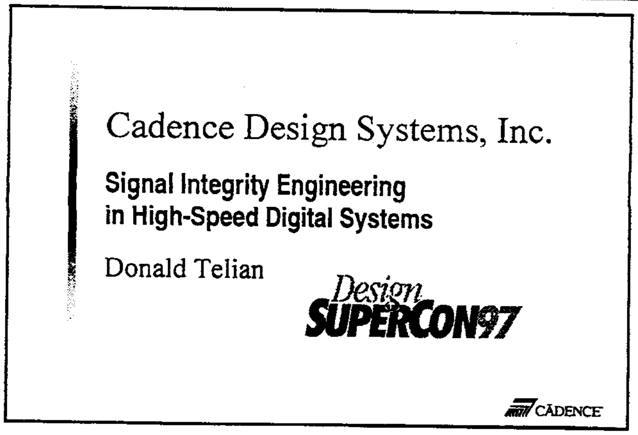
Donald Telian

Current Activities

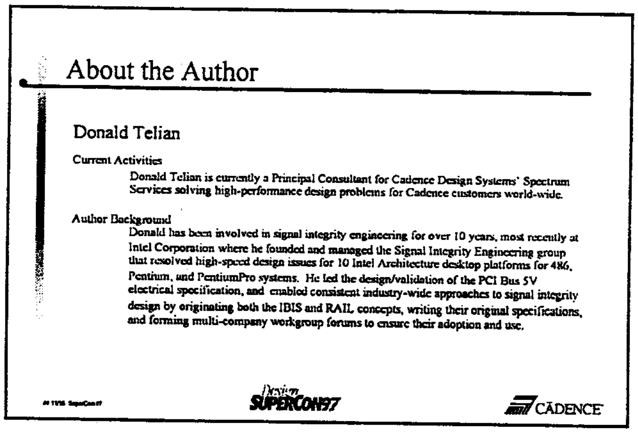
Donald Telian is currently a Principal Consultant for Cadence Design Systems' Spectrum Services solving high-performance design problems for Cadence customers world-wide.

Author Background

Donald has been involved in signal integrity engineering for over 10 years, most recently at Intel Corporation where he founded and managed the Signal Integrity Engineering group that resolved high-speed design issues for 10 Intel Architecture desktop platforms for 486, Pentium, and PentiumPro systems. He led the design/validation of the PCI Bus 5V electrical specification, and enabled consistent industry-wide approaches to signal integrity engineering by originating both the IBIS and RAIL concepts, writing their original specifications, and forming multi-company workgroup forums to ensure their adoption and use.

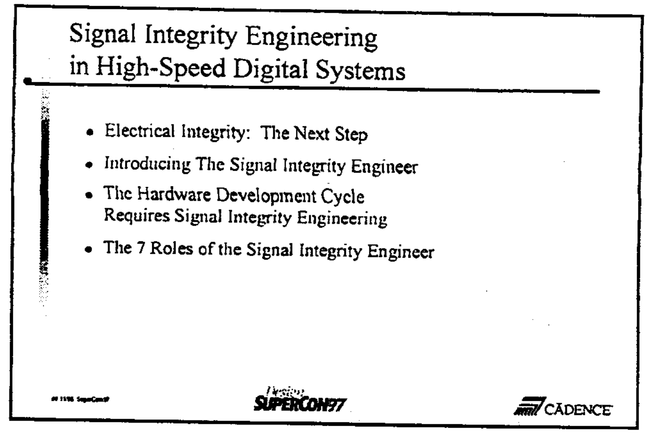


Slide #1



Slide #2

This paper also references comments from five managers of Signal Integrity groups at leading computer manufacturers across the country. My thanks for their valuable comments and insights into the task of Signal Integrity Engineering.

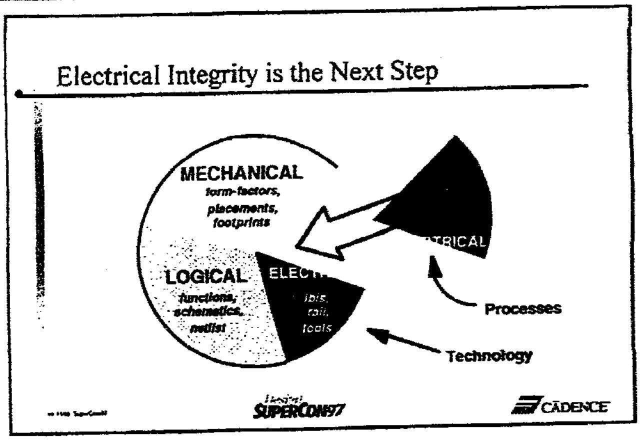


Slide #3

This paper will show how Signal Integrity Engineering has become an important part of today's high-speed digital system design. As a point of reference, we'll define "high-speed" as digital signals running faster than 25 MHz that are not internal to Integrated Circuits.

We'll briefly discuss how system design has changed over the last 10-15 years, and show how that change has caused the need for a new type of engineer: the Signal Integrity Engineer.

We'll then look at how this engineering function must be deployed throughout the Hardware Development Cycle through the proper execution of what I've termed "The 7 Roles of the Signal Integrity Engineer."



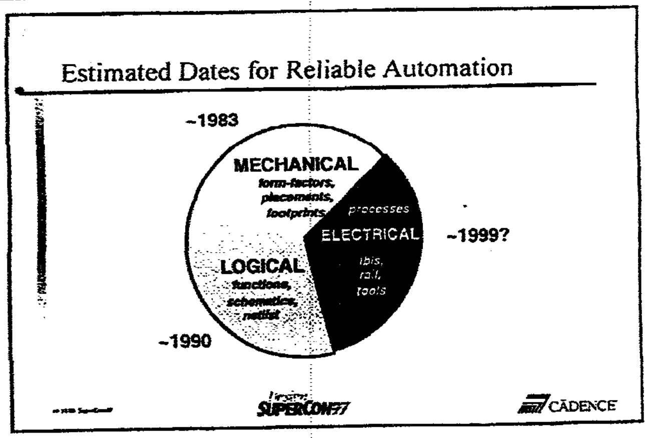
Slide #4

Digital system design has three primary areas: Mechanical, Logical, and Electrical. While Mechanical and Logical aspects have certainly had their share of change, the Electrical portion is going through an interesting metamorphosis.

This change in Electrical Design is largely due to the introduction of higher switching speeds. In slower systems, perhaps running at 1 MHz, signals spent over 95% of the cycle in the static condition. Consequently parameters of electrical interest described that static state; quantities such as I_ol or V_ih.

In contrast, signals in today's 66 MHz systems typically use about 1/3 of the cycle for switching. Often, these signals never even reach a "static" condition before they are required to switch again. Consequently new data types such as IBIS models and RAIL files, and robust and complex simulation tools have arisen to address this new electrical paradigm.

However, even though this new technology has become available, the processes by which they are used and judged are inconsistent and still developing across the industry. One Signal Integrity Engineering group manager put it this way: "Engineering is inherently skewed toward defined, stable tasks and processes, yet the business need for today's [signal integrity] practitioner is to produce acceptable results with scanty data, unstable process and multiple simultaneous dimensions." The process piece of the pie is simply missing.

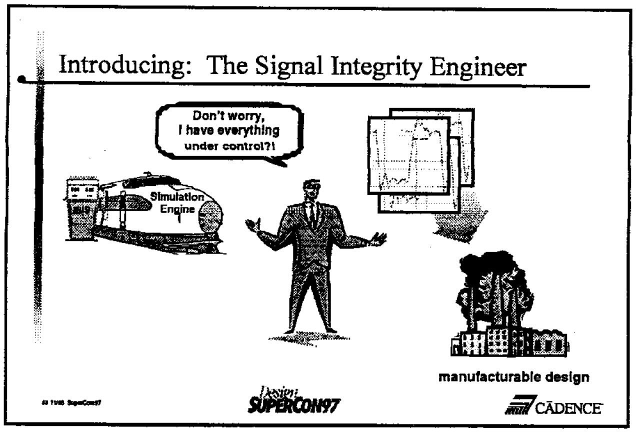


Slide #5

So it's reasonable to ask when a sufficient environment for ensuring Electrical Integrity through the proper use of tools, data, and process might emerge. That's a difficult question, but we may get some clues by examining what has occurred for ensuring Mechanical and Logical integrity.

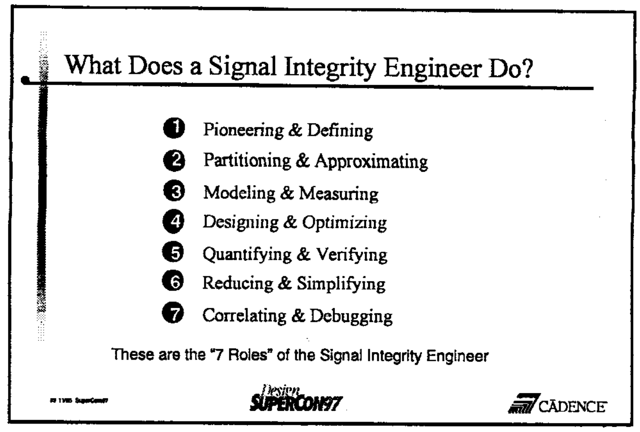
While these dates are certainly debatable, most can remember a time when PCB layouts were hand-taped and manually checked with yellow highlighters. First hand-taping went away, and later as confidence was built in design automation software so did the highlighter. The result was automated Mechanical Integrity. And Logical Integrity has followed a similar path, both at the PCB and IC level.

Though we're not there yet, it's reasonable to assume that in the next few years ensuring electrical integrity will become a well-defined task. Until then, a good amount of expertise and craftsmanship is required on the part of the "Signal Integrity Engineer."



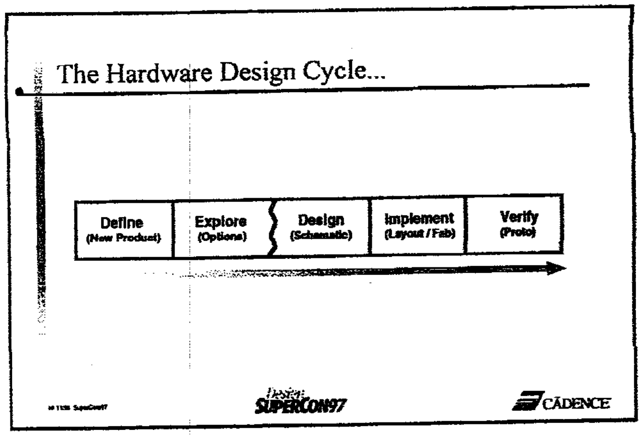
Slide #6

The "Signal Integrity Engineer" function continues to emerge across the industry. This engineer continues to grapple with simulation tools and lots of data, trying to translate it all into a manufacturable design. His job is to craft solutions even though the design processes and data types he must use are still maturing.



Slide #7

Perhaps the best way to characterize the Signal Integrity Engineer is to talk about what he does or should do. I have broken these tasks into "7 Roles" that we will discuss in more detail. Some are obvious, and commonly performed. Others are often overlooked, but deserve careful consideration. These "7 Roles" transition from "Pioneering" to "Debugging" as we move through the design cycle.

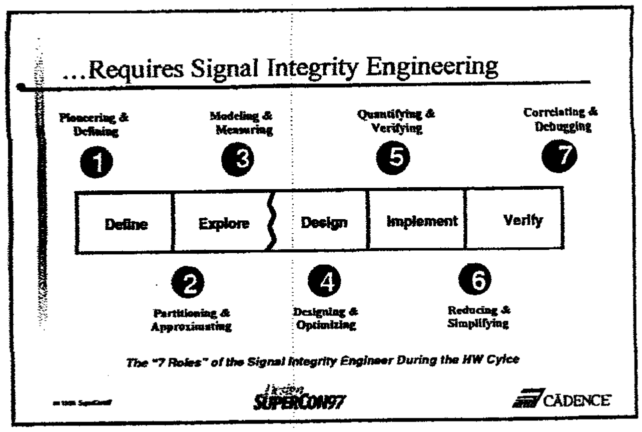


Slide #8

Before showing how the "7 Roles" map onto the Hardware Design Cycle, let's consider the stages that normally occur in the development of new products.

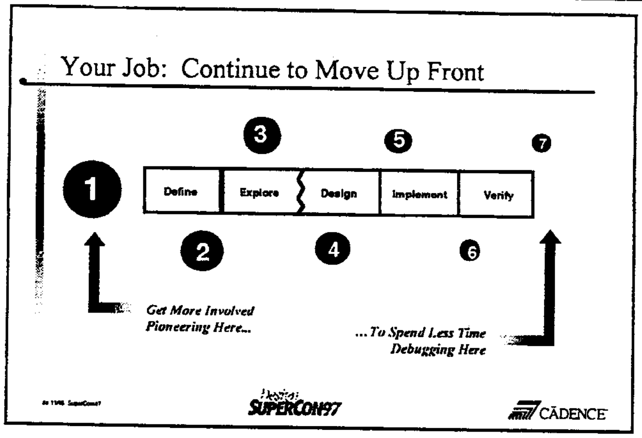
Any product must begin with a clear definition of what it should be. From this, time would be spent exploring the various implementation and technology options available.

During the design phase those choices are carefully organized and analyzed, leading to a more physical implementation phase. Once the product is prototyped and verified, it would transition into manufacturing.



Slide #9

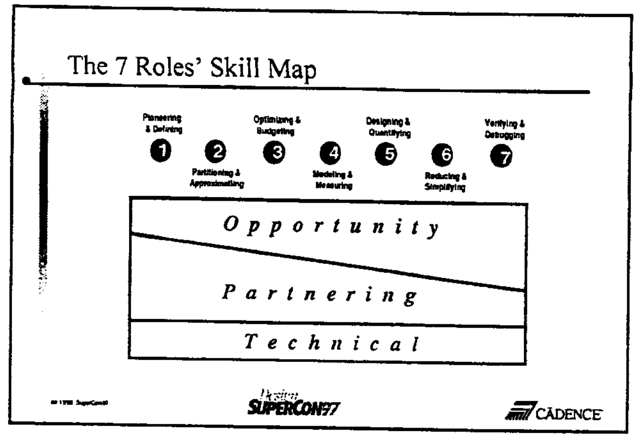
Interestingly, successful "Signal Integrity Engineering" should occur throughout the entire development cycle during nearly every phase and phase transition. There should be impact not only on ensuring design implementations are properly verified, but even on the types of products that are defined. For example, high-bandwidth functions can be defined and added to your desktop computer via PCI cards. This is a result (in part) of signal integrity "pioneering" work that moved open market bus signaling from 8 MHz to 33 (and later, 66) MHz.



Slide #10

Carrying the PCI example further, without the deliberate step to "pioneer" the electrical environment in which reliable switching could occur, there would have been a lot more time spent "debugging" cards before they could go to market. At worst, they may not have even been able to work together at the electrical level.

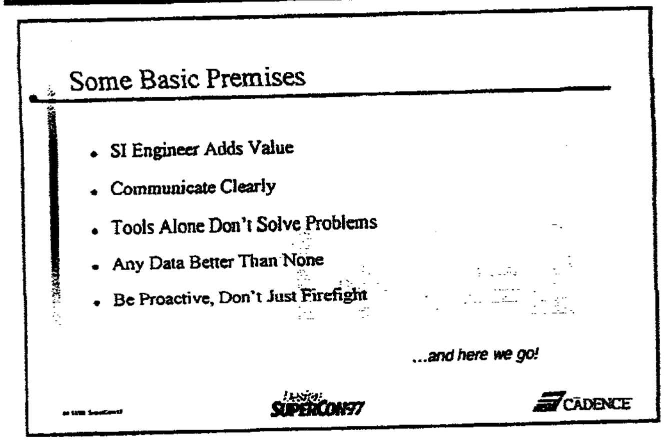
And so, it's important to point out that the Signal Integrity Engineer must continually focus on moving his expertise up-stream in the design cycle in order to add more value, ensure greater reliability, and get to market more quickly with high-speed products. This often does not occur because many have the view that empirical analysis is the only way to address signal integrity. But debugging physical hardware is actually the worst way to solve problems because it couples development with manufacturing cycles. It is also becoming increasingly difficult to adequately probe today's fine pitch components, forcing empirical analysis to be very tedious at best or even impossible at worst.



Slide #11

But we must also point out that there is less opportunity for "pioneering". This diagram points out that while solid technical skills are required to execute each of the 7 roles, gaining opportunity to perform the upper roles rests on your ability to partner with other individuals, design groups, and even companies.

For example, there is plenty of opportunity to debug noise problems that require little partnering with other individuals. However, the opportunity to "pioneer" generally is only realized with a much larger amount of partnering with the various organizations involved in deploying your new ideas.



Slide #12

Before jumping into the specific "7 Roles", here's some basic premises that apply to all of them.

First, the Signal Integrity Engineer's job is to **add value** in every interaction with the project team. Too often the perception is that you are a traffic cop whose only job is to slow everyone down, and are consequently avoided.

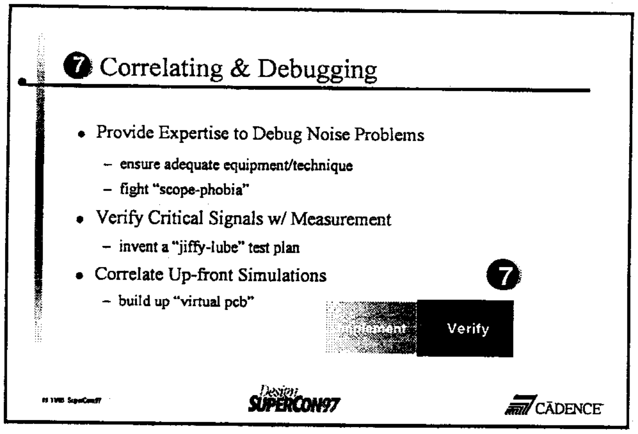
The Signal Integrity Engineer must also **communicate clearly**. The temptation is to be overly technical, expound the virtues of the 3-D via model, and amaze everyone with your incredible (irrelevant) information. I've seen too many engineers just confuse the team and walk away without really accomplishing anything at all. Avoid this at all costs.

Next, remember that **tools alone don't solve problems.** You drive the tools - they do not drive you. It has been well said that "a fool with a tool is still a fool." The job isn't done when the simulation is run. What does it mean? The value comes from interpreting the results. Simulation is the means to an end, not an end in itself.

One manager of Signal Integrity Engineers put it this way: "I've worked with a few engineers in the past that were considered technical experts in SI related areas. Some of those folks were ineffective simply because they got bogged down studying details. These "scientists" studied and analyzed everything, worked to the nth degree of simulation accuracy, and were unable to come up with the answer in a timely fashion (if at all)."

In every case, **any data is better than none**. Too many will freeze in their tracks when they can't get a certain model and accomplish nothing at all. Make assumptions when you must and go on.

Be proactive. When you see something that will not work right, deal with it then. Saying "I told you so" later really doesn't accomplish anything but proving that you don't know how to effectively make a point at the right time.



Slide #13

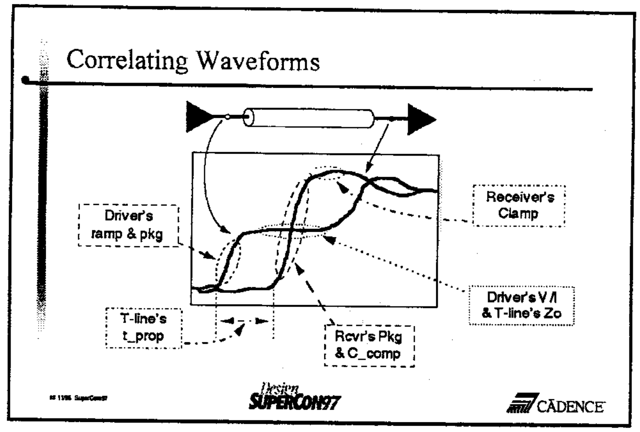
To cover the roles, we'll start at the back of the design cycle with #7 and work our way forward. This is because most Signal Integrity Engineers probably got their start right here in role 7 debugging noise problems. This unfortunately occurs because many design shops do not have the foresight to correctly plan and staff the analysis task on the front end of the development cycle. It may be that they were unaware of the issues associated with running at high-speeds, or they were concerned but didn't have (or know where to get) a methodology to address them.

Nevertheless debugging is a good and valuable skill. The most common problem here is not having sufficient equipment or technique. Most problematic noise spikes require at least a 2 GHz bandwidth scope and careful probing without ground leads to really see what's happening. One manager stressed: the Signal Integrity Engineer "must pay close attention to details in making the observations (measurement system, probe location, ...). It is very easy to draw the wrong conclusions."

And then, even if you have good equipment, somehow there's a real terror associated with using a scope. People prefer to work with logic analyzers in this digital world and are reluctant to get the scope out. (I'm told that 15-20 years ago, the opposite was true.) Fight this "scope-phobia".

A better idea might be to find the problems before they find you. Be proactive and write a test plan to examine critical/risky signals in a minimal amount of time. Take a "jiffy-lube" approach, and just quickly examine the top 14 signals needing attention.

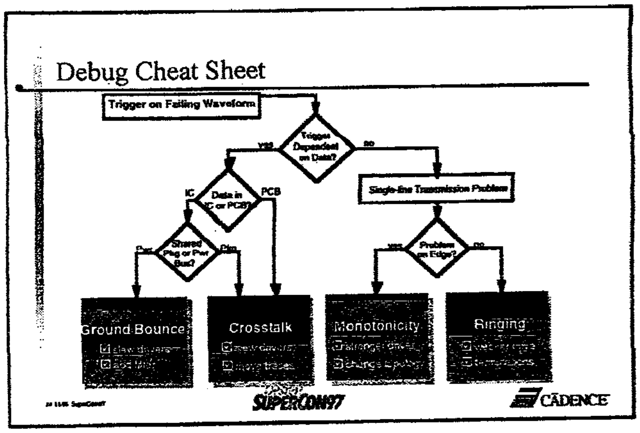
One other way to add value in this phase is to have a "virtual pcb" built up in your board-level simulator. Spend some time correlating your simulation environment, and then use it to validate solutions to problems found in the lab.



Slide #14

This diagram is a useful reference to help correlate measured waveforms with your simulations. At this point in time, most of the simulators on the market have accurate simulation algorithms. Problems and miscorrelations normally result from poor models that can be corrected by carefully examining measured waveforms.

This example shows a simple driver and receiver interconnected by a transmission line (or pcb trace), and a typical waveform captured at both ends (at driver and receiver). Assuming your oscilloscope and probing techniques are correct, the diagram shows what section of a circuit model to adjust to get specific portions of a waveform to match simulation. IBIS format (IO Buffer Information Specification, EIA-656) models are the simplest to adjust relative to the parameters shown. Once correlated, your models and simulation environment is much more efficient at solving problems on the current or next design.



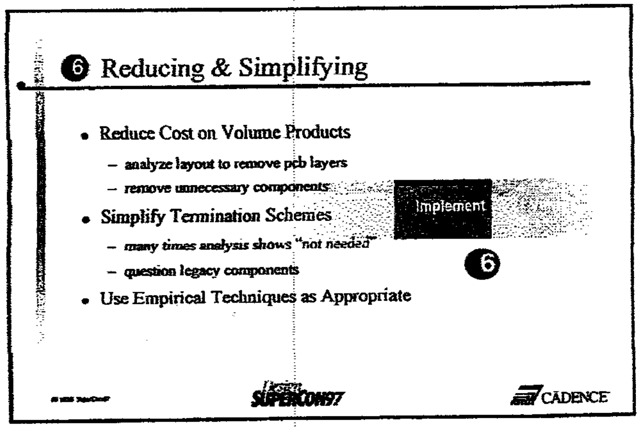
Slide #15

This diagram is designed to help you quickly discover the root cause of noise problems. The diagram shows the initial trigger leading into the decision tree. All decisions resolve into one of four failure modes: Ground Bounce, Crosstalk, Monotonicity, and Ringing. Generally speaking, all of these failure modes manifest themselves as some type of signal sampling or timing related problem. Under the failure modes are a (non-comprehensive) list of common fixes.

Initially, a trigger must be found to trap on the failing waveform. Sometimes this is not trivial, but normally can be done using logic analyzers. Once an oscilloscope is attached to the failure point, experiments should be made to determine if the failure mode is dependent on data patterns. If it is, then the waveform should look different over time and it is not a single-line transmission problem (refer to left half of diagram).

If the dependent data is only physically associated with the failing signal on the PCB, the problem is PCB crosstalk. If the data is more physically associated in an IC, more investigation is required to see if the dependent signals share IC package area or power buses with the failing signal. If the signals are related in the power bus, the problem is ground (or power) bounce. If in the package, the problem is crosstalk within the IC.

Single-line transmission problems are simpler to debug and understand. These failing waveforms should look consistent over time. These failures cause data corruption due to either non-monotonic edges, or ringing that transitions back into device threshold regions.

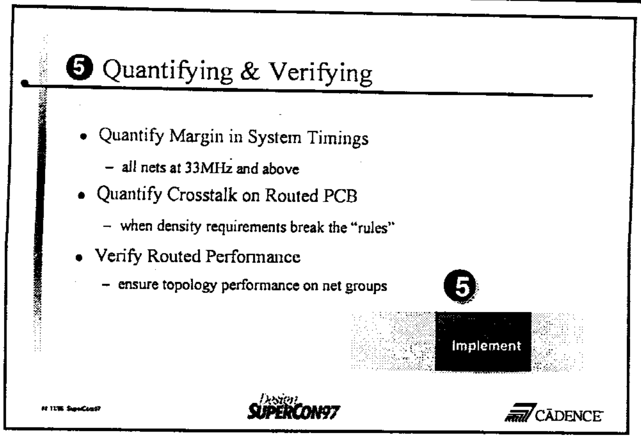


Slide #16

This role is probably the easiest one to overlook. However, it should be pointed out that (assuming you've developed a correlated simulation environment) you have the tools and skills to impact the production cost on volume designs.

Because Signal Integrity Engineering is still relatively new in the digital domain, there's a lot of designs out there that have hundreds of unnecessary components. These were added because "rules of thumb" or textbook approaches were used rather than analysis. Many times analysis shows that extra terminations really weren't needed. But that's data that simply wasn't available. Removing 100 resistors and capacitors from a design that runs 50k_units/month can save $1,000,000 per year in materials and assembly costs.

However, on many low-volume designs cost is often not an issue, so don't waste your time.



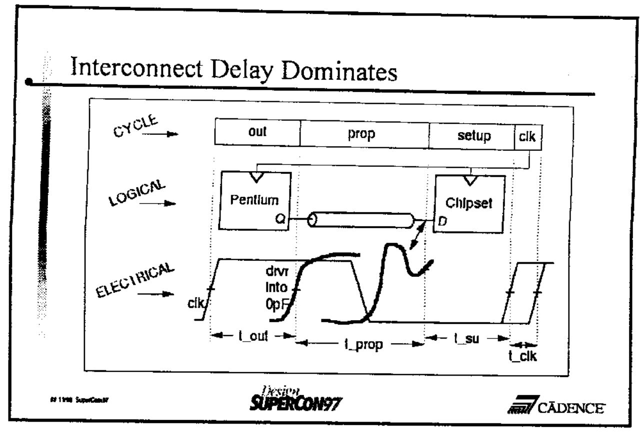
Slide #17

Role #5 is another familiar role. This role is applied during the implementation phase, concurrent with PCB layout. If Role #4 was done correctly, the effort here should be minimal.

Simulation tools are used heavily in this role. Now the task is to Quantify and Verify **groups** of nets as they become realized in the PCB artwork. Work should be done concurrently in order to provide feedback to layout as soon as possible. Some simulation software is now integrated with layout software in order to provide immediate feedback to the layout engineer.

If crosstalk is believed to be a critical issue, it generally must be quantified during this stage. Rules and budgets can be described prior to layout, but normally will need to be verified against the physical implementation.

The only way to confirm a working routed topology is to quantify performance measures associated with it. The most common measure is timing, but other factors such as monotonicity and overshoot may be important. The current industry-accepted benchmark for when to work system timings is for nets operating at 33MHz and above. For these nets it is important to allocate a certain timing margin of the total path to PCB signal propagation, or "interconnect delay", as shown on the next diagram.



Slide #18

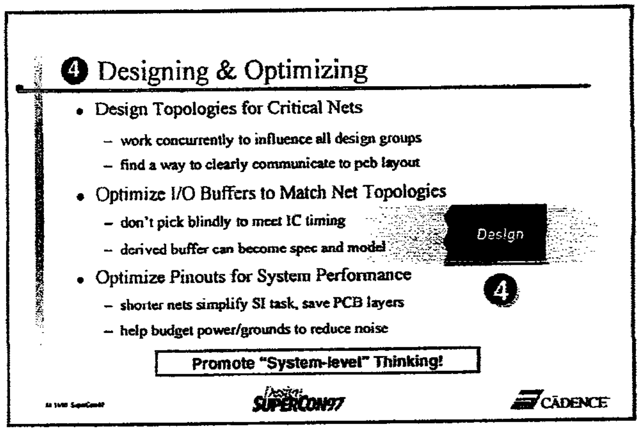
Signal propagation time across a PCB now represents a significant part of the cycle in today's high-speed digital systems, typically requiring about 1/3 of the available nanoseconds. Yet there is often confusion surrounding how that portion of the timings is allocated to the PCB.

This diagram gives the cycle, logical, and electrical view of a synchronous PCB timing path. A normal cycle consists of the elements shown at the top: time to transition "out" of the driver, time to "prop"-agate across the PCB, time to "setup" to the receiver's clock, and a minimized "clk" skew that results from variance in the clock signal as it arrives at the driver and receiver. The logical view helps illustrate this interaction.

The electrical view shows how the transmitted signal is synchronized using the rising edge of the clock signal (marked "clk"). The thicker waveforms show the exact hand-off points between the "out", "prop", and "setup" timing specifications. The driver's "out" timings are specified from the rising edge of the clock to a specified voltage into a rated load (shown here as 0 pF).

The Signal Integrity Engineer's task in this phase is to quantify the portion of the cycle time required by the interconnect (normally PCB) layout. This parameter (shown as "t_prop") begins where the driver specification left off, and ends when the receiver sees a stabilized waveform. Most common simulators do a good job of reporting this value.

The setup and clock skew parameters now round out the complete cycle.



Slide #19

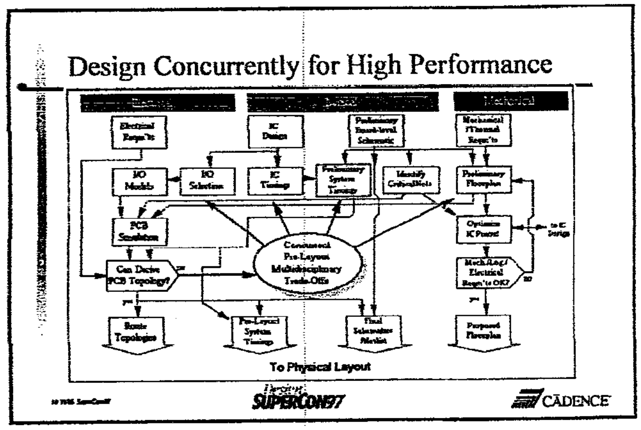
Role #4 represents the heart of the design phase. Here, the Signal Integrity Engineer adds value by effectively interfacing with design groups (mechanical, board-level, and IC) to ensure that the high-speed sections of the design will work properly when physical implementation occurs.

Prior to PCB layout, critical nets must be identified. It is important to explore implementation options on these nets and derive PCB topologies that will work across production and system environment tolerances. When options cannot be found, it is important to engage with other design groups early enough to negotiate changes. Unfortunately, this level of interaction rarely occurs during the design phase causing poor or non-optimal system-level performance.

Optimizing I/O buffers on ICs for a required physical topology is well worth the effort. Often, drivers designed or selected without comprehending the PCB environment are inappropriately sized. At worst, very strong buffers are used to satisfy an IC timing but destroy the system-level timing and performance by injecting too much noise into the system (as well as wasting die space).

Optimizing IC pinouts has a significant impact on net lengths and hence signal integrity and the number of PCB layers required for routing. Part of this optimization should include ensuring proper quantities and placement of IC power and grounds.

It must be noted that a working design is not just a route scheme on a PCB, but rather a robust organization of logical, mechanical, and electrical elements - and this requires all groups to exercise "system-level thinking".



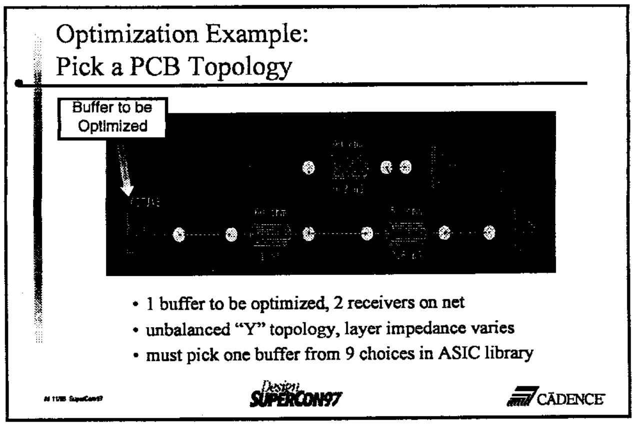
Slide #20

This diagram presents a detailed process flow for the proper way to concurrently derive PCB topologies (shown at left). The successful Signal Integrity Engineer knows how to drive this process in conjunction with the larger system design process.

The key challenge in working this process effectively is using your data to drive and coordinate system-level trade-offs and changes amongst the design disciplines (occurs in the central oval in the diagram). Usually the simplest design adjustment (feedback arrow path) is obvious when viewed at the system-level.

The key deliverable from this process is a set of "Route Topologies" that are known to work well in order to ensure a rapid and effective process of physical layout. Deriving good topologies drastically reduces the risk of having signal integrity problems arise when the design becomes physical and problems become harder to fix. Strive to find an automated way pass these topologies to layout to eliminate miscommunication and enable electronic archiving of successful design solutions.

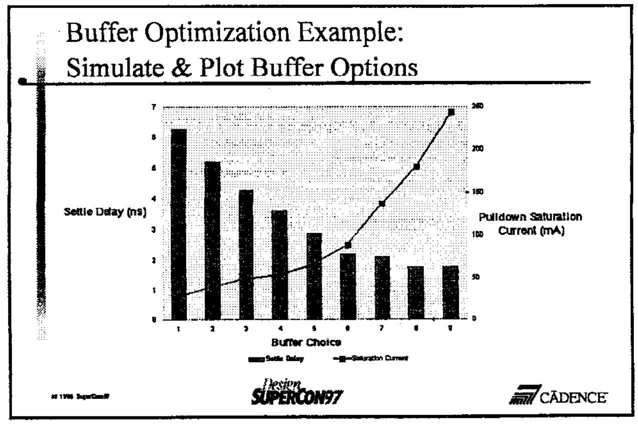
When done properly the process shown correctly influences I/O buffers, IC timings, system timings, IC pinouts, final schematics, and the proposed floorplan to achieve proper high-speed operation. Significant production risk is reduced by coordinating these aspects during the design phase. Obviously, a methodology that waits until the implementation phase to derive topologies misses the opportunity to influence and optimize the larger system-level environment for high-performance.



Slide #21

Now we'll examine a typical design scenario and show how to pick an optimal buffer for a given PCB topology (shown in the diagram).

Because of mechanical restrictions, the orientation of the ICs on this net will cause a large address bus to be routed in an unbalanced "Y" topology with varying PCB impedances. Other topologies/ impedances could perhaps be forced, with greater cost, but let's see if acceptable performance can be achieved through buffer optimization

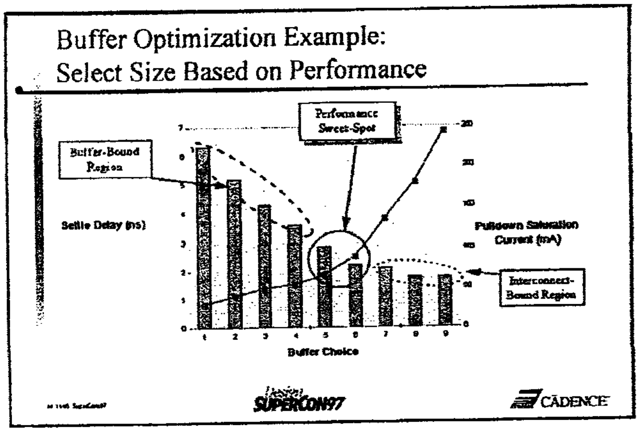


Slide #22

The address bus buffer will be driven by an ASIC which has nine different buffer options in it's library. The first step is to simulate and plot the measured PCB signal propagation settle delays, or "interconnect delays", for each buffer.

The nine buffers were simulated and yielded the various timings shown in the bar graph. Note that the various buffers yield settle delays that range from a little over 6 nS to 2 nS. Though all plots of this type do not look exactly like this, the results shown are common.

To illustrate the wide range of buffer sizes, the line chart superimposes each buffer's saturation current (relates directly to transistor size) on top of the measured settle delay. From this line we see that the buffers range in size from 30 mA to almost 250 mA of saturation current. In common ASIC terminology this represents buffers that might be referred to as "2 mA" to "24 mA" drive strength.



Slide #23

Once the results are plotted, we're now prepared to select the best buffer.

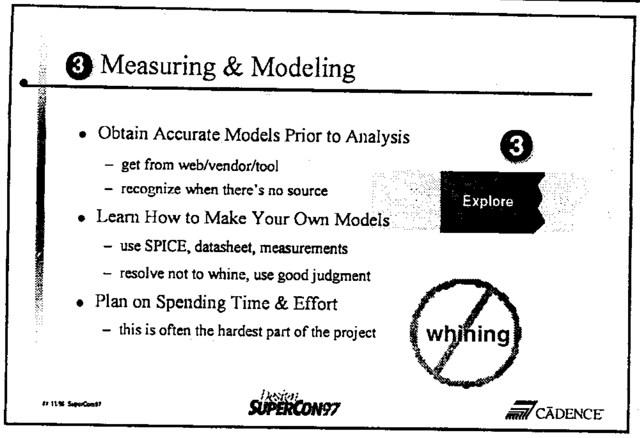
I should first point out that if system timings allowed 8 nS for this signal's propagation, buffer number 1 would be a great choice. However, since that amount of time is rarely available in today's high-speed systems we'll have to take a closer look.

First, let's examine the left half of the plot. On this side, minor up-sizing of the buffer strength produces big improvements in the settle delay. In this region the net's performance is heavily dependent on the buffer size. Because of this dependence, performance in this region could be referred to as "buffer bound."

In contrast, performance in right half of the plot is "interconnect bound". That is because huge changes in buffer size have only very minimal impact on performance. In this region, the performance of the interconnect dominates. This behavior is almost always observed, and dispels the myth that using a larger buffer produces better timings and basically "solves everything".

By now, it should be obvious that either buffers 5 or 6 are the best choice for this net topology. Slightly weaker buffers (to the left) suffer from poor timing performance, while stronger buffers (to the right) only waste die space and inject extra switching noise without improving performance. Usually, a "performance sweet-spot" is obvious on plots of this type.

This is just one example of the value the Signal Integrity Engineer can add by concurrently designing/interfacing with other design groups in Role #4.



Slide #24

Often, the stalemate in the whole signal integrity process is models. One manager surveyed cited the need for "good VALIDATED concise models" as his number one issue related to successful Signal Integrity Engineering.

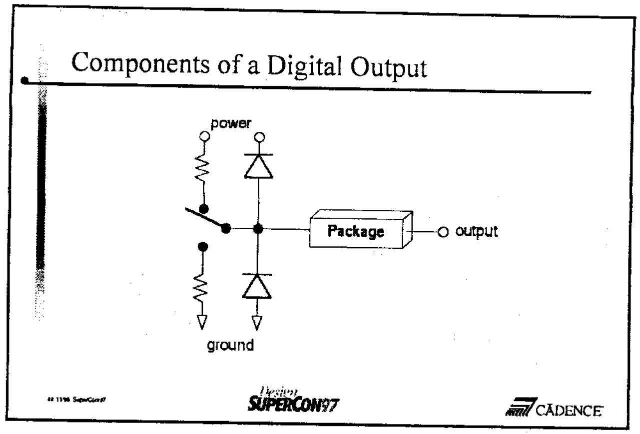
The effective Signal Integrity Engineer must ensure an adequate set of models is available prior to analysis in the design phase. This is a task which must be expected and planned.

Many sources for models exist. Some can be downloaded from web-sites, others might be found in the library purchased with your design software. An increasing number of component vendors supply models in addition to datasheets, and a few third party model suppliers exist today.

However, often when you are working on new products (particularly with new silicon) the models can not be simply obtained. In these cases learn to quickly recognize that no good source is available - you'll just need to develop the model yourself.

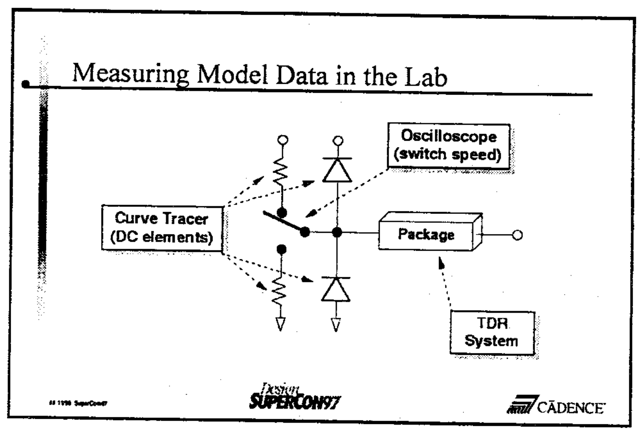
Today's Signal Integrity Engineer must also be a skilled model developer. Sometimes a SPICE model can be obtained and converted to IBIS. In other cases, you'll need to learn how to go into the lab and make the key measurements yourself. Ideally, highly accurate curve tracers and oscilloscopes are available for this task. But if not, even simple equipment and measurement techniques can be used.

Whatever the case, resolve not to whine, do the best you can, and don't be afraid to extrapolate and use good engineering judgment to develop the most accurate model possible within the time allowed. Never allow a project to be stalled because "I don't have the model."



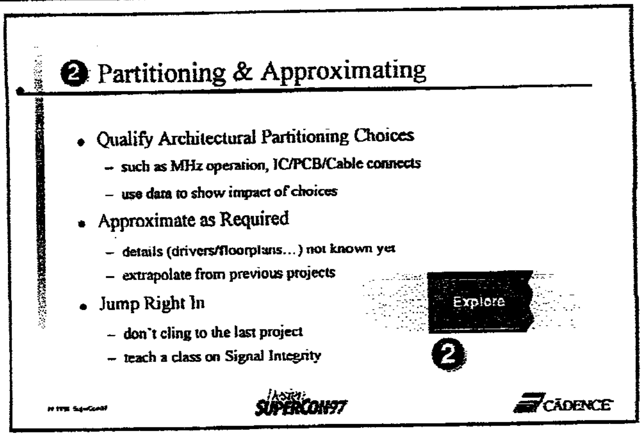
Slide #25

Most common digital IC drivers simply switch the output to either power (for logic "1") or ground (for logic "0"). This connection is through the non-linear resistance of the output transistors. In CMOS, these outputs have parasitic diodes that can affect signal behavior during switching. And in most cases, the IC driver is connected to the PCB through some kind of external packaging. These elements are shown in the diagram.



Slide #26

Today's measurement equipment does a decent job of capturing these key components of driver behavior. The different components can be characterized by using the equipment shown in the diagram. With some minor modifications, the data captured can be used directly to construct a model in the IBIS format.



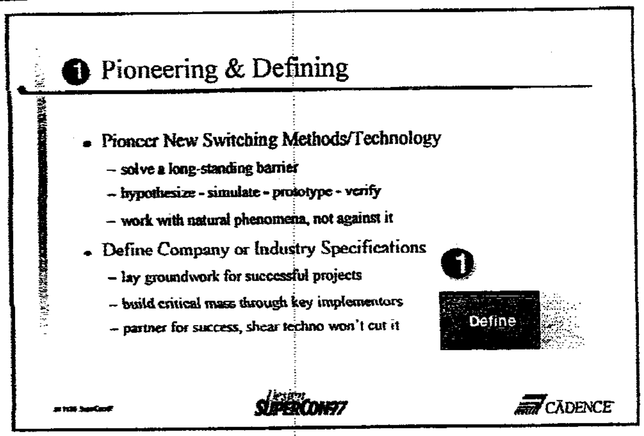
Slide #27

During the "Explore" phase of the Hardware Development Cycle, the Signal Integrity Engineer should engage in role #2: Partitioning and Approximating.

This work normally goes on, but too often without the Signal Integrity Engineer's input. You should be consulted in all choices regarding the feasibility of desired interconnects and operating frequencies for the technologies being considered.

Qualify all requests for high-bandwidth interfaces; many engineers are enamored with running faster than they need to. However, it's important during this phase to be optimistic, and think over proposed ideas before claiming they will not work. When you're not sure, do some approximate simulations to test the options. Make assumptions, and generate some preliminary data.

So jump right in and add value. Don't be hindered by clinging to the last project and oversimulating it for no reason. And, if you find that the team doesn't understand or isn't considering your inputs, teach a class on Signal Integrity and properly explain the impact of architectural choices. Engineers love to go to classes.

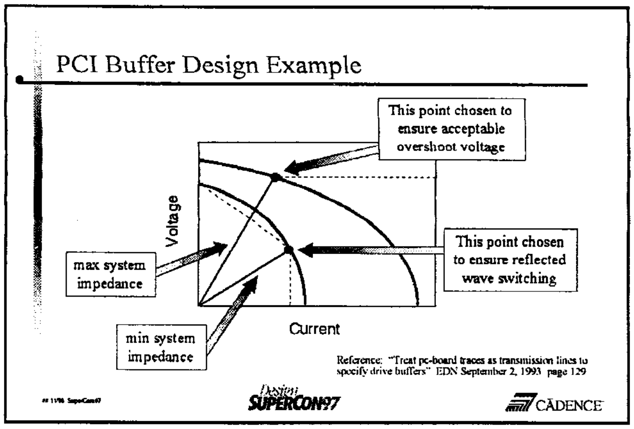


Slide #28

The door is always open for Pioneering and Defining. There are lots of problems still to solve. Hypothesize new ideas, then simulate, prototype, and verify them. The best ideas are surprisingly simple ones that use natural phenomena to accomplish something new.

But, as stated earlier, you must partner with key implementers to be successful here. Many great ideas go nowhere because the inventor fails to get others involved.

And remember that high-speed design still lacks a lot of process framework. If you come up with an idea that helps define the task and get it done, you'll probably find that many in the industry are ready to implement and use your solution.

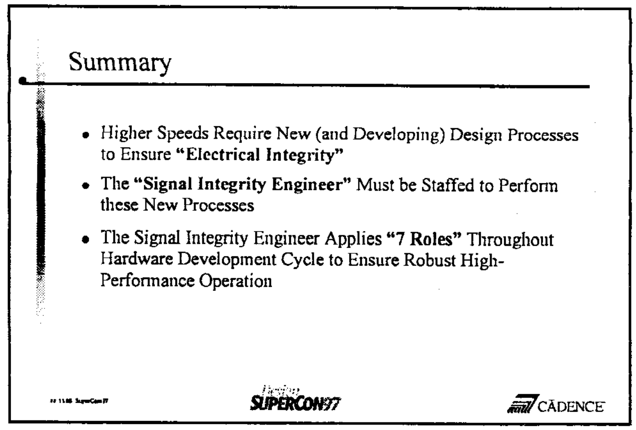


Slide #29

Defining the PCI bus was one example where Signal Integrity Engineers played a significant role. At the end of 1991, there was a desire to create an open market bus that was driven directly from ICs, connected more devices, and operated four times faster than the current bus.

Through lots of up-front engineering, we were able to craft a well-defined environment to ensure success prior to any implementations. The PCI drivers were optimized for best performance, once the system environment was understood and bounded. The driver's description then became both the spec and the model for further development and simulation. "Reflected-wave" switching was introduced as a way to work with natural phenomena (reflections... usually considered something bad) to allow the interconnect to be driven directly by low-power ASICs.

The diagram shows, in technical terms, how the system environment (impedance) was overlaid with the IC characteristics (output V/I curves) to define robust operation. This process can be used to define an acceptable design space for any interface. Once the key parameters are identified, a bounded region for IC design can be constructed, as shown. (Actually, the PCI bus chose another method to bound the maximum characteristics, rather than containing overshoot.) For a more thorough description, consult the reference listed.



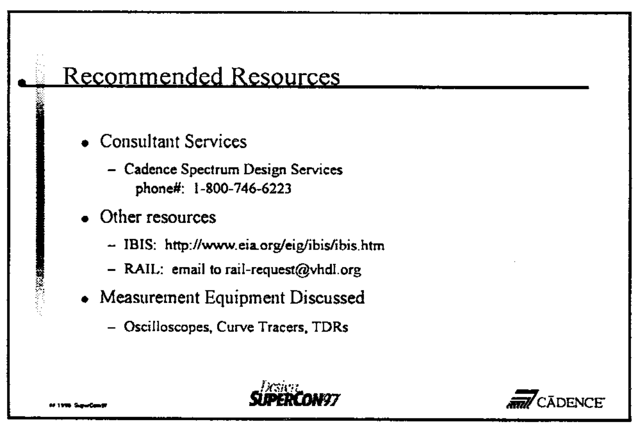
Slide #30

In this talk, we have described how to apply Signal Integrity Engineering to today's high-speed digital designs.

Ensuring Electrical Integrity now requires careful planning and analysis of the dynamic (rather than static) operation of digital designs. However, the processes used to do this task are still being developed.

The Signal Integrity Engineer has emerged in recent years to address these new requirements, and now must be staffed as an integral part of the Hardware Development Team.

Throughout the Hardware Development Cycle, the Signal Integrity Engineer adds value by executing seven unique roles. When performed effectively, these roles can dramatically increase a new product's performance, reliability, and time to market without adversely affecting cost.



Slide #31