# Cadence Design Systems, Inc.

## Designing High-Speed Digital Systems

### Using Today's Blazing Fast ICs

Donald Telian

version 1.7

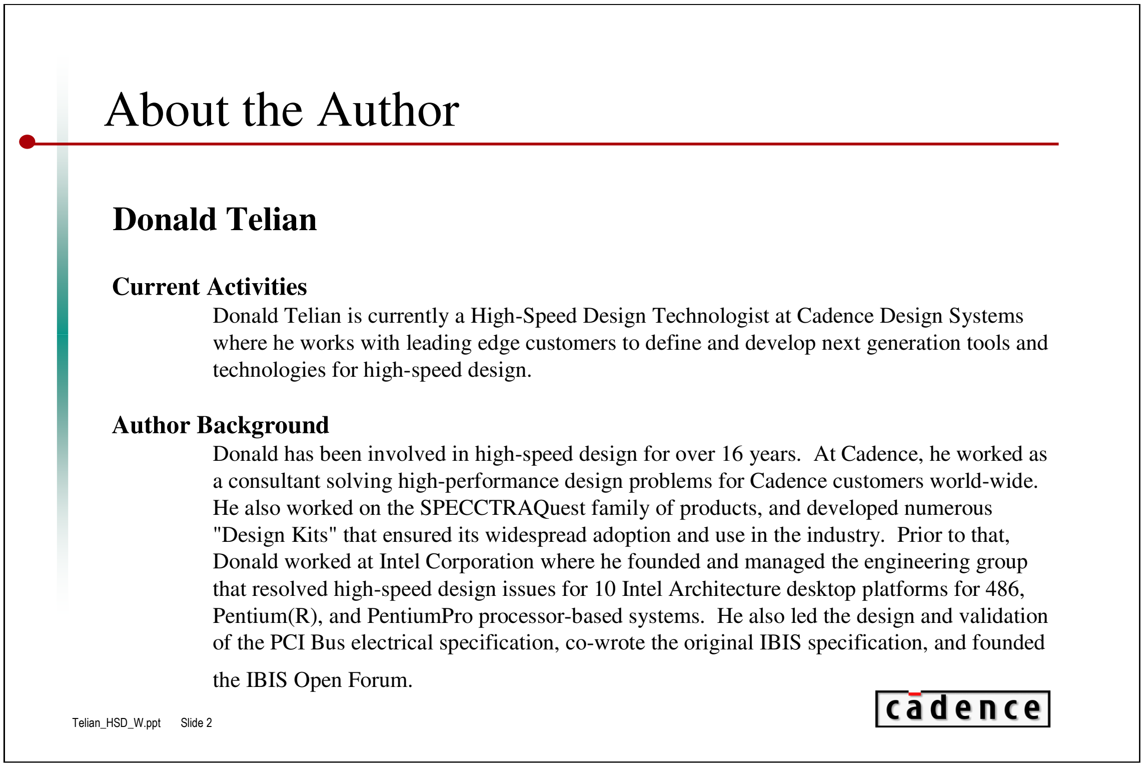
Design SUPERCON97 "Best Paper" Award

cadence

ABSTRACT:

Far from a well-understood science, High-Speed Design methods and practices are still being defined.  And yet, ensuring electrical integrity is the next critical piece of the high-speed digital design puzzle due the introduction of today's blazing fast mainstream Integrated Circuits.

This paper outlines the 7 points in the hardware development cycle where proper High-Speed Design must be performed. We'll study the correct application of theory, tools, and models to engineering tasks ranging from originating new standards to fire-fighting bugs that arise on the assembly line. Methods are defined, and practical design tips are suggested from real-life successful execution of these tasks.



# About the Author

## Donald Telian

### Current Activities

Donald Telian is currently a High-Speed Design Technologist at Cadence Design Systems where he works with leading edge customers to define and develop next generation tools and technologies for high-speed design.

### Author Background

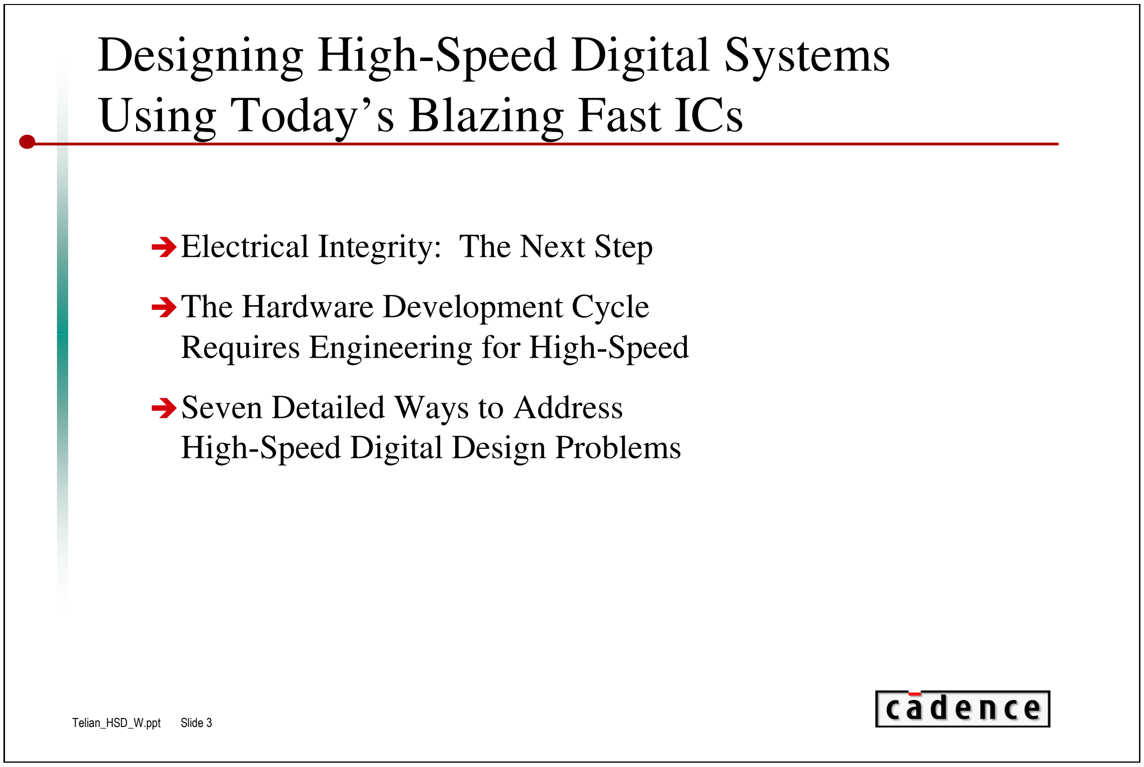
Donald has been involved in high-speed design for over 16 years. At Cadence, he worked as a consultant solving high-performance design problems for Cadence customers world-wide. He also worked on the SPECCTRAQuest family of products, and developed numerous "Design Kits" that ensured its widespread adoption and use in the industry. Prior to that, Donald worked at Intel Corporation where he founded and managed the engineering group that resolved high-speed design issues for 10 Intel Architecture desktop platforms for 486, Pentium(R), and PentiumPro processor-based systems. He also led the design and validation of the PCI Bus electrical specification, co-wrote the original IBIS specification, and founded the IBIS Open Forum.

Telian_HSD_W.ppt Slide 2

cadence

This paper also references comments from five High-Speed Design pioneers who manage development groups at leading computer manufacturers across the United States. My thanks for their valuable comments and insights into this topic.

This paper (in its original form) was voted “Best Paper” in the High Performance Design Conference at Design SuperCon97 based on attendee and conference organizer feedback. The conference occurred in January 1997 in Santa Clara, California.



# Designing High-Speed Digital Systems Using Today's Blazing Fast ICs

- ➔Electrical Integrity: The Next Step
- ➔The Hardware Development Cycle Requires Engineering for High-Speed
- ➔Seven Detailed Ways to Address High-Speed Digital Design Problems

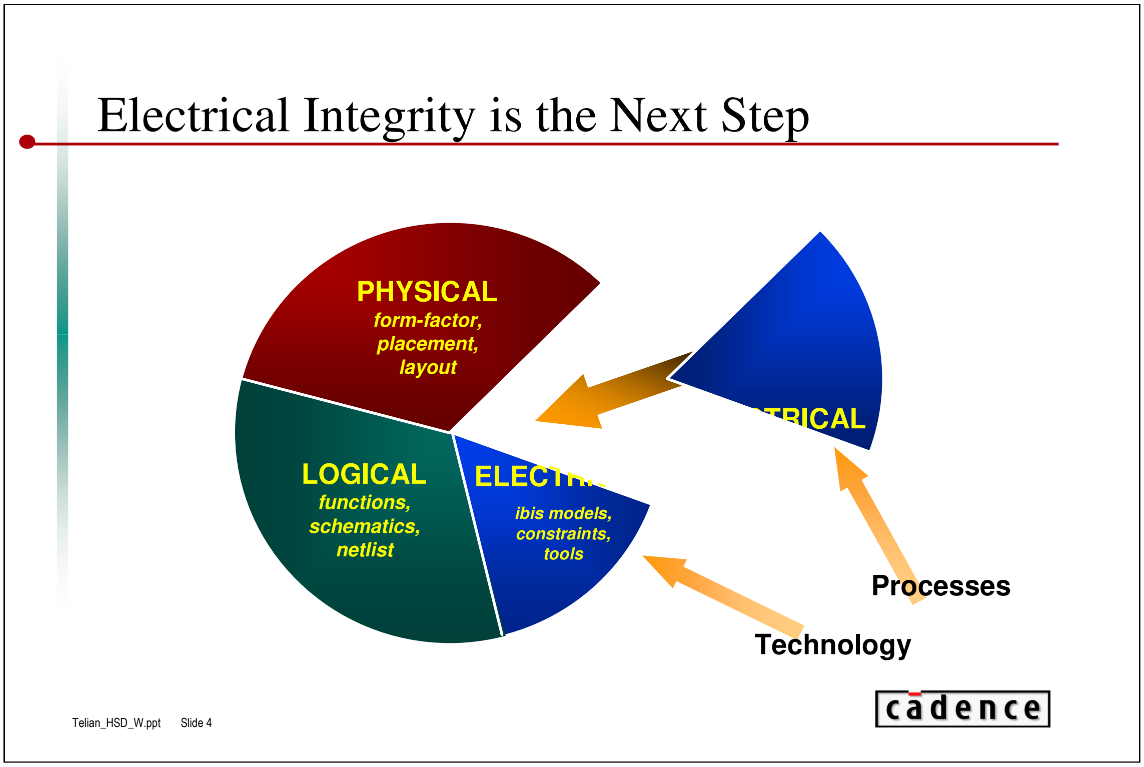
Telian_HSD_W.ppt Slide 3

cadence

This paper will show how High-Speed Design Engineering has become an important part of today's digital system design due to the introduction of very fast mainstream Integrated Circuits (ICs). As a point of reference, we'll define "high-speed" as digital signals running faster than 25 MHz that are not internal to ICs.

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 new engineering methods to address High-Speed designs.

We'll look at how these engineering functions must be deployed throughout the Hardware Development Cycle through the proper execution of seven detailed ways to address High-Speed design problems.



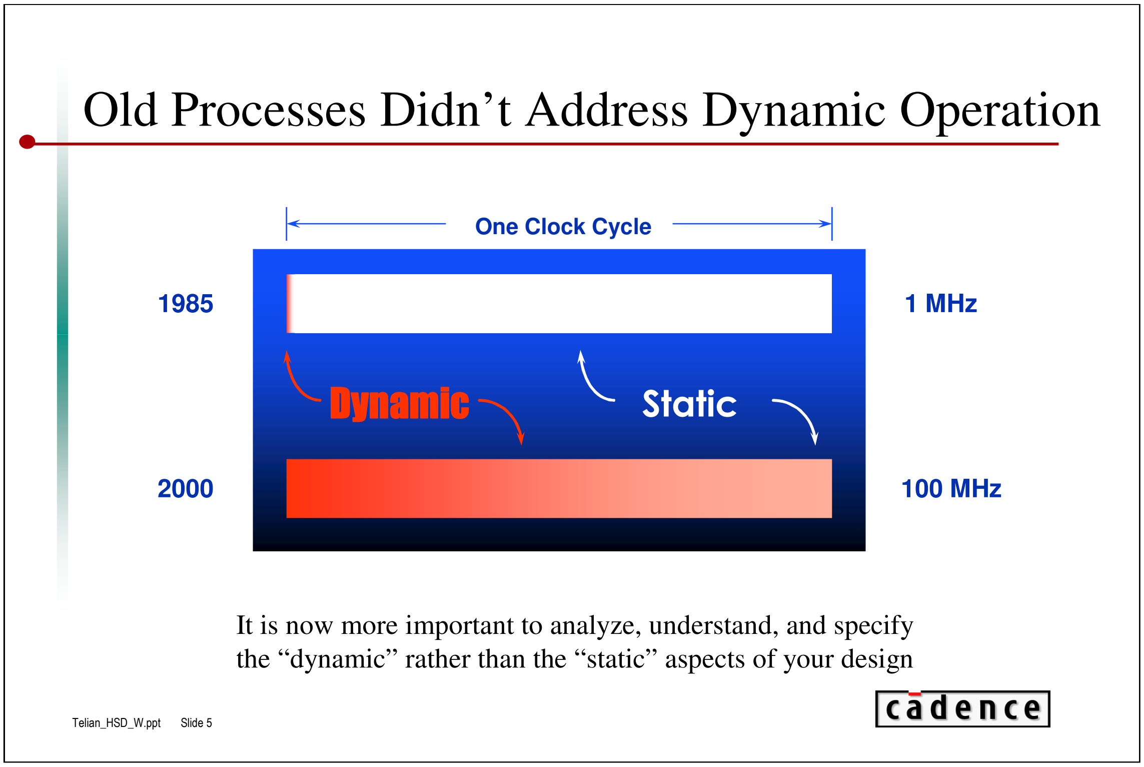
## Electrical Integrity is the Next Step

Telian_HSD_W.ppt Slide 4

Digital system design has three primary areas: Physical, Logical, and Electrical. While Physical 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 the higher switching speeds used in today’s blazing fast ICs. Faster external clock rates have caused new problems in the development of digital systems. And the problems are no longer isolated to the elaborate supercomputers of yesterday. Now the simple computer on your desk has interfaces running above 100MHz. Consequently, design shops all over the world are being challenged to approach design in new and different ways.

New data types such as IBIS models and constraint 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 High-Speed Design group manager put it this way: “Engineering is inherently skewed toward defined, stable tasks and processes, yet the business need for today's [high-speed] practitioner is to produce acceptable results with scanty data, unstable process and multiple simultaneous dimensions.” The process piece of the pie is simply missing.



# Old Processes Didn't Address Dynamic Operation

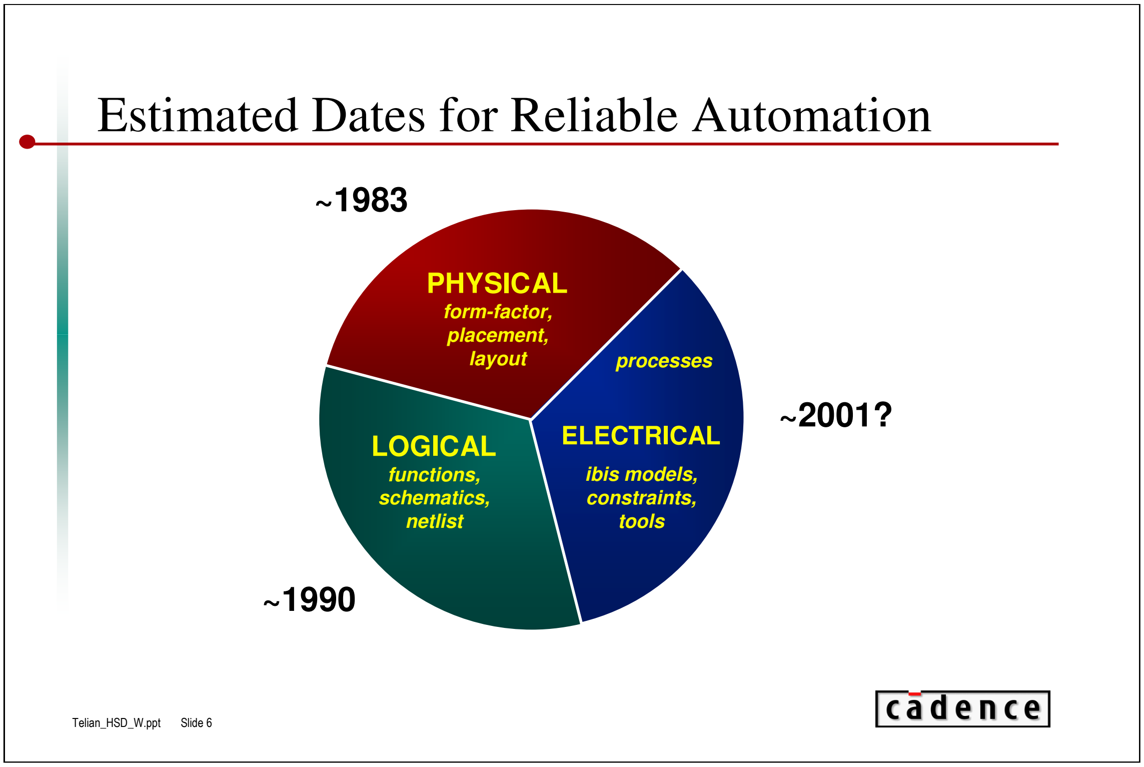
It is now more important to analyze, understand, and specify the "dynamic" rather than the "static" aspects of your design

Telian_HSD_W.ppt Slide 5

cadence

Granted, there have been plenty of design processes aimed at achieving Electrical Integrity, but these methods focused on static, rather than dynamic, operation. 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 100 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. Simulating and understanding these dynamic characteristics has become an essential part of determining how well a design operates.



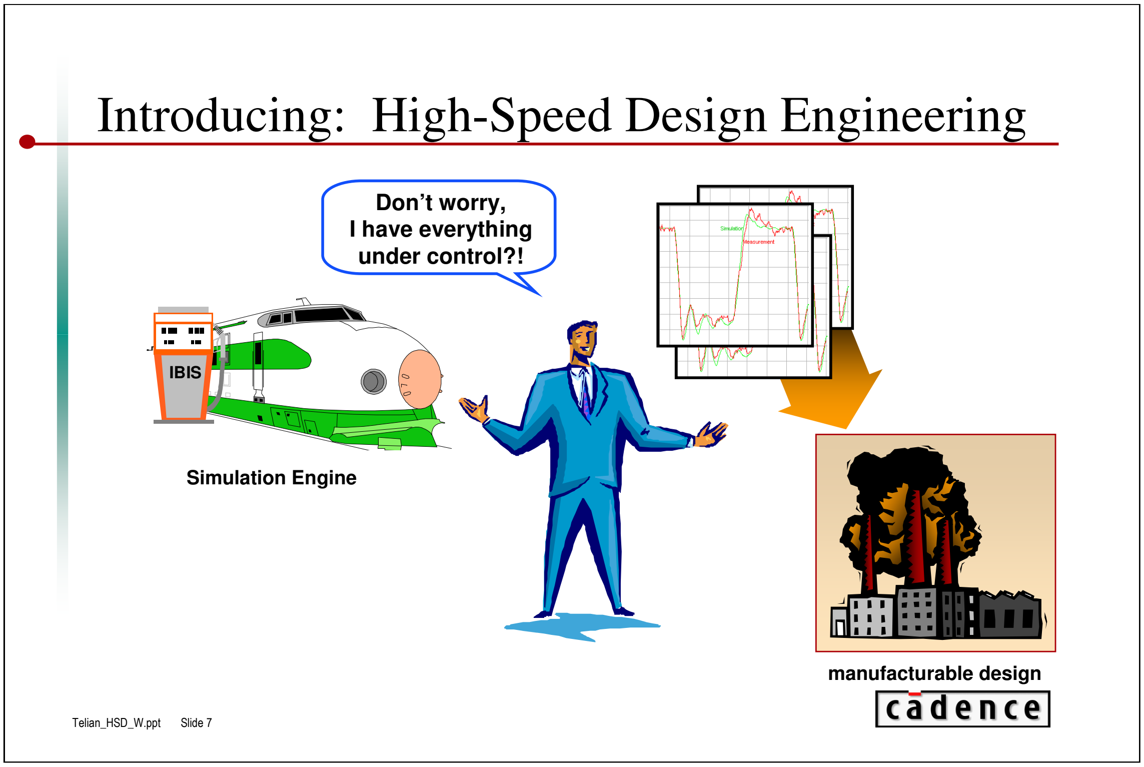
## Estimated Dates for Reliable Automation

Telian_HSD_W.ppt Slide 6

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 Physical 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 Physical 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 engineers to understand and solve High-Speed Design problems.



# Introducing: High-Speed Design Engineering

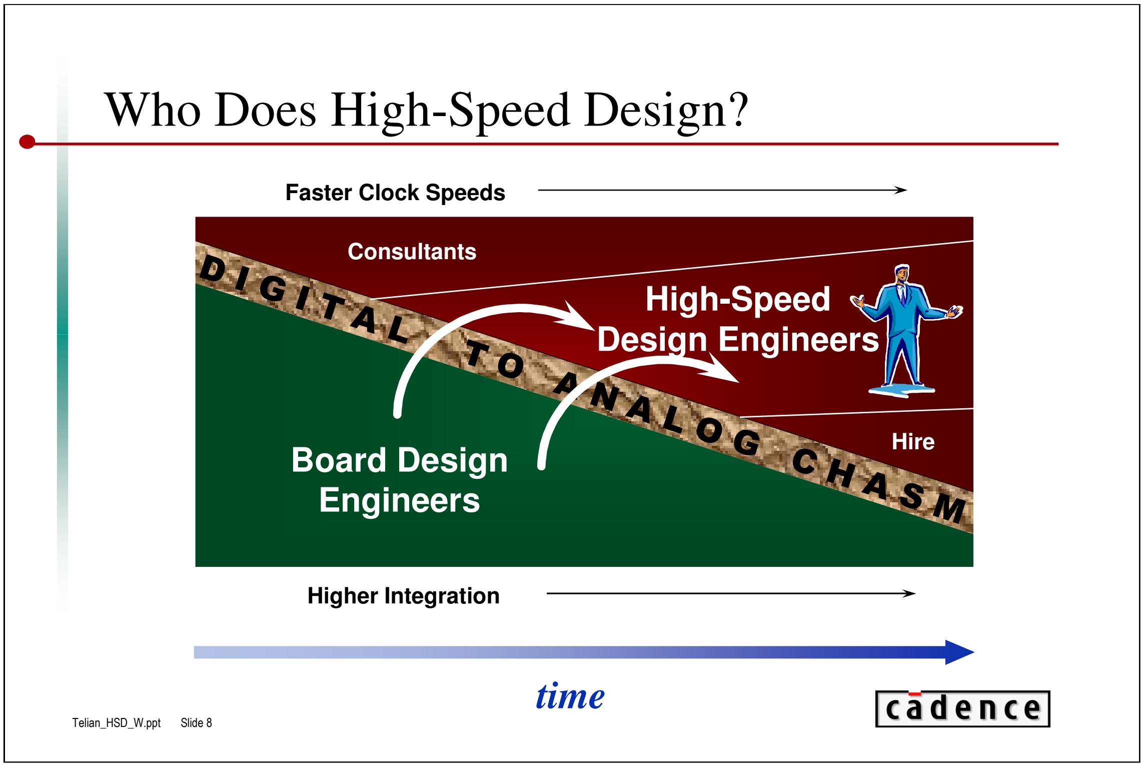
Don't worry,
I have everything
under control?!

Simulation Engine

manufacturable design

Telian_HSD_W.ppt Slide 7

Engineers across the industry are now focusing on understanding and solving High-Speed Design issues. These engineers continue to grapple with simulation tools and lots of data, trying to translate it all into a manufacturable design. Their job is to craft solutions even though the design processes and data types they must use are still maturing.



## Who Does High-Speed Design?

Faster Clock Speeds →

Consultants

DIGITAL TO ANALOG CHASM

High-Speed Design Engineers

Hire

Board Design Engineers

Higher Integration →

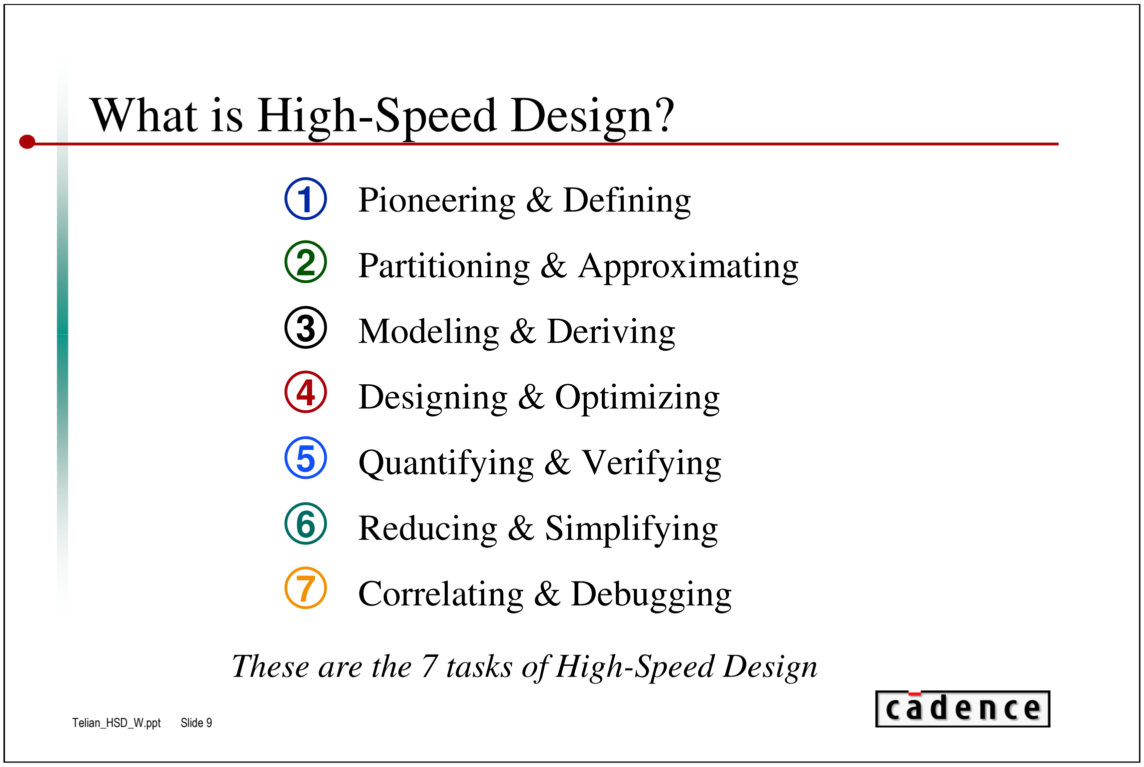
*time*

Telian_HSD_W.ppt Slide 8

A logical question is “Who is equipped to really DO High-Speed Design?”. In these early stages of the growing requirement for this kind of engineer, expert consultants with practical experience (primarily from the computer industry) are spreading their useful skills across the industry.

As clock rates continue to climb in a variety of new ICs, there is a growing need for high-speed design skills to develop telecom, wireless, network, and computer products. Consequently, many traditional board designers are beginning to tackle high-speed issues. This can be done if some deliberate retraining is planned in order to cross the “chasm” where those nice stable digital bits begin to have analog fits.

Many design shops are attempting to hire these skills, but it is too early for that. The demand is too high and the supply is too low. In the short term, retraining and working with consultants is a better plan. Growing the required skills internally often makes sense, since higher integration continues to greatly simplify the board design task.



## What is High-Speed Design?

① Pioneering & Defining

② Partitioning & Approximating

③ Modeling & Deriving

④ Designing & Optimizing

⑤ Quantifying & Verifying

⑥ Reducing & Simplifying

⑦ Correlating & Debugging

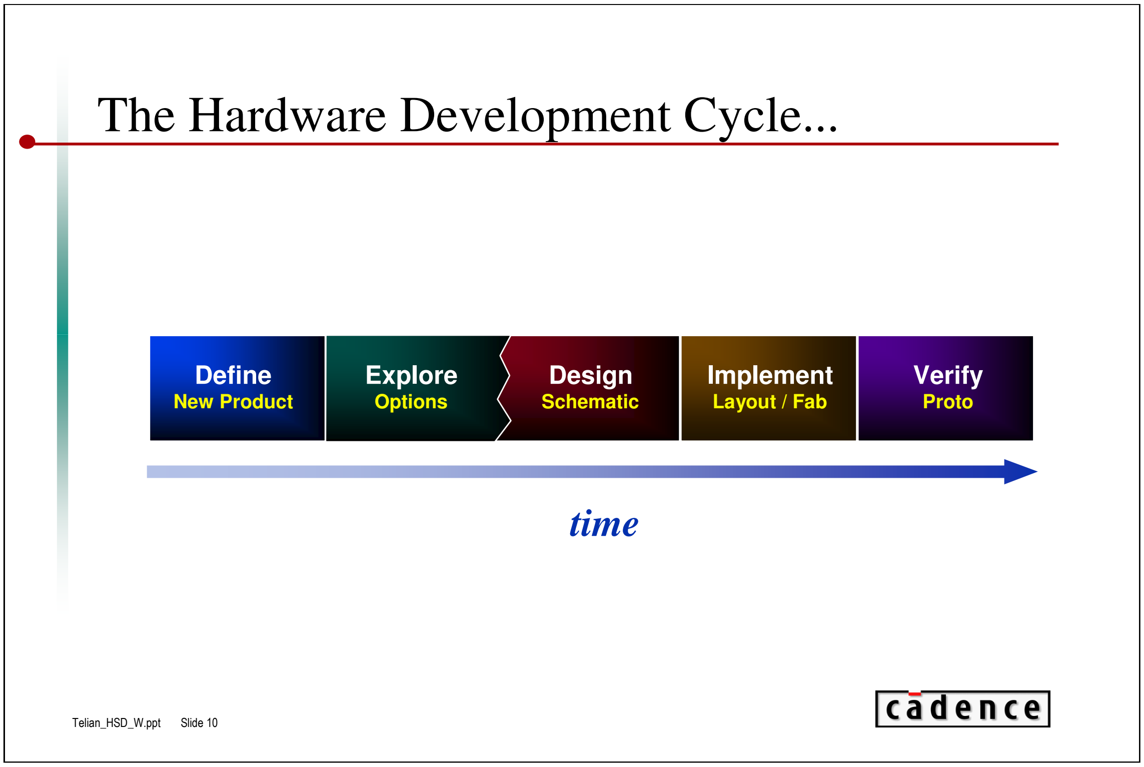
*These are the 7 tasks of High-Speed Design*

cadence

Telian_HSD_W.ppt Slide 9

What is High-Speed Design? How and when is it performed?

Perhaps the best way to characterize High-Speed design is to talk about how it is (or should be) done. Here, I have broken the process into 7 tasks that we will discuss in more detail. Some are obvious, and commonly performed. Others are often overlooked, but deserve careful consideration. These 7 tasks transition from “Pioneering” to “Debugging” as we move through the hardware development cycle.



# The Hardware Development Cycle...

Define
New Product

Explore
Options

Design
Schematic

Implement
Layout / Fab

Verify
Proto

*time*

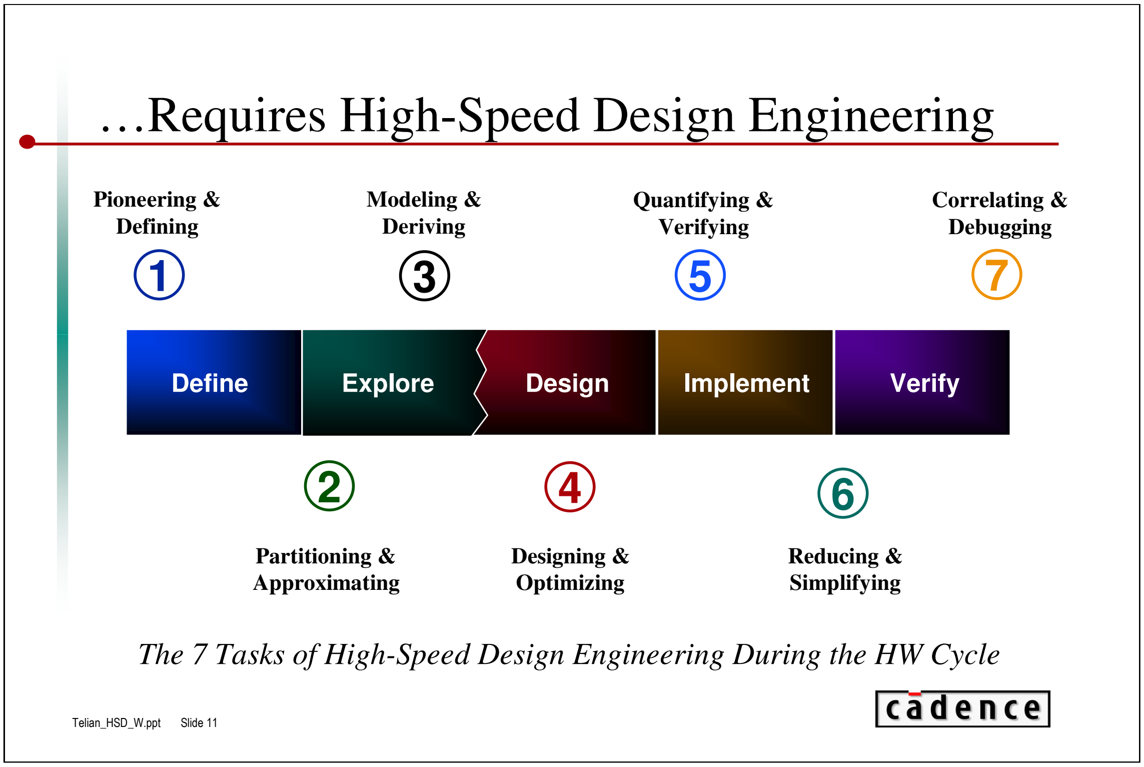
Telian_HSD_W.ppt Slide 10

cadence

Before showing how the 7 tasks map onto the Hardware Development Cycle, let’s consider the stages that normally occur in the process of developing 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.



## …Requires High-Speed Design Engineering

Pioneering & Defining ①

Modeling & Deriving ③

Quantifying & Verifying ⑤

Correlating & Debugging ⑦

Define | Explore | Design | Implement | Verify

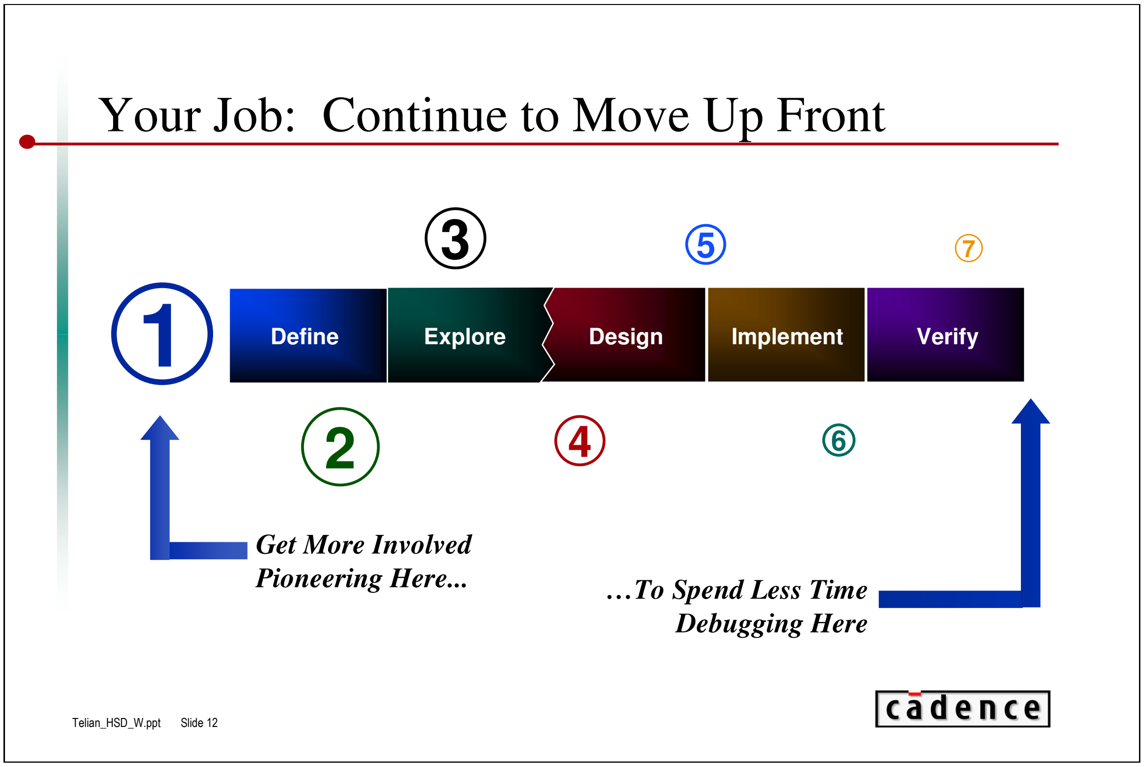
② Partitioning & Approximating

④ Designing & Optimizing

⑥ Reducing & Simplifying

*The 7 Tasks of High-Speed Design Engineering During the HW Cycle*

Interestingly, successful High-Speed Design 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 high-speed “pioneering” work that moved open market bus signaling from 8 MHz to 33 (and later, 66) MHz.



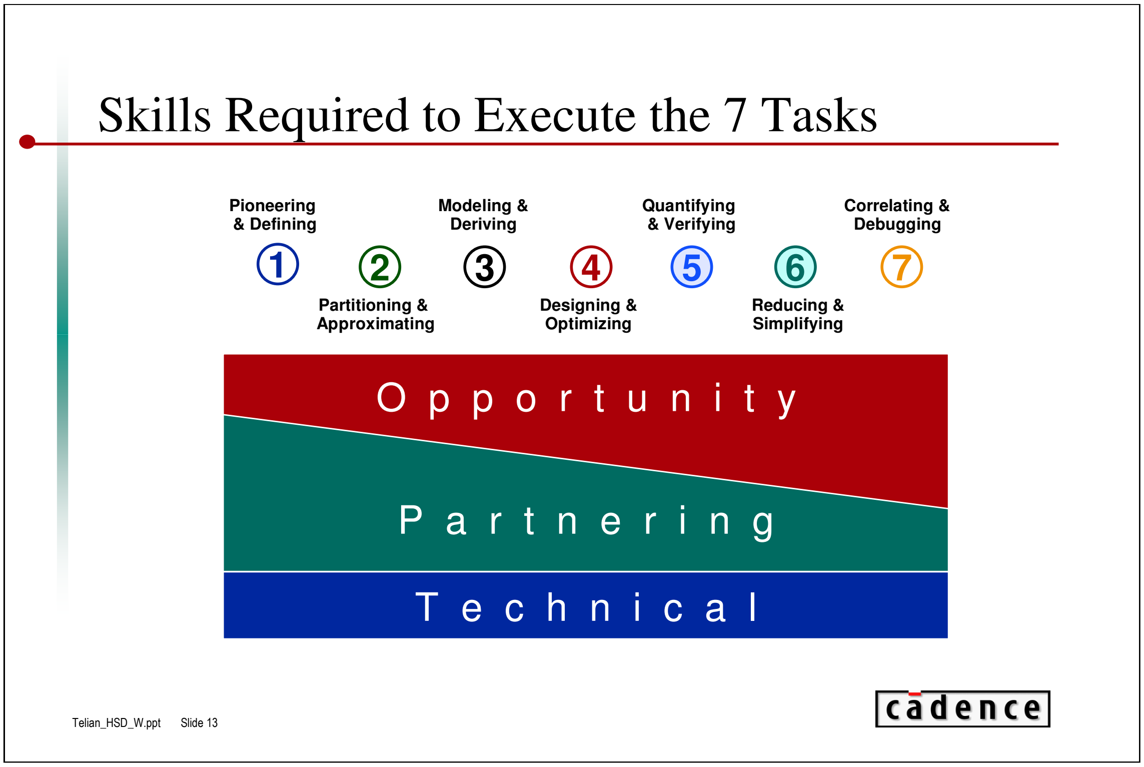
## Your Job:  Continue to Move Up Front

*Get More Involved Pioneering Here...*

*...To Spend Less Time Debugging Here*

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 High-Speed Design Engineering adds the most value when performed up-stream in the design cycle in order to confirm proper operation, ensure greater reliability, and get to market more quickly.  This often does not occur because many have the view that empirical analysis is the only way to address high-speed issues.  But debugging physical hardware is actually the worst (and most time consuming) 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.



# Skills Required to Execute the 7 Tasks

① Pioneering & Defining

② Partitioning & Approximating

③ Modeling & Deriving

④ Designing & Optimizing

⑤ Quantifying & Verifying

⑥ Reducing & Simplifying

⑦ Correlating & Debugging

Opportunity

Partnering

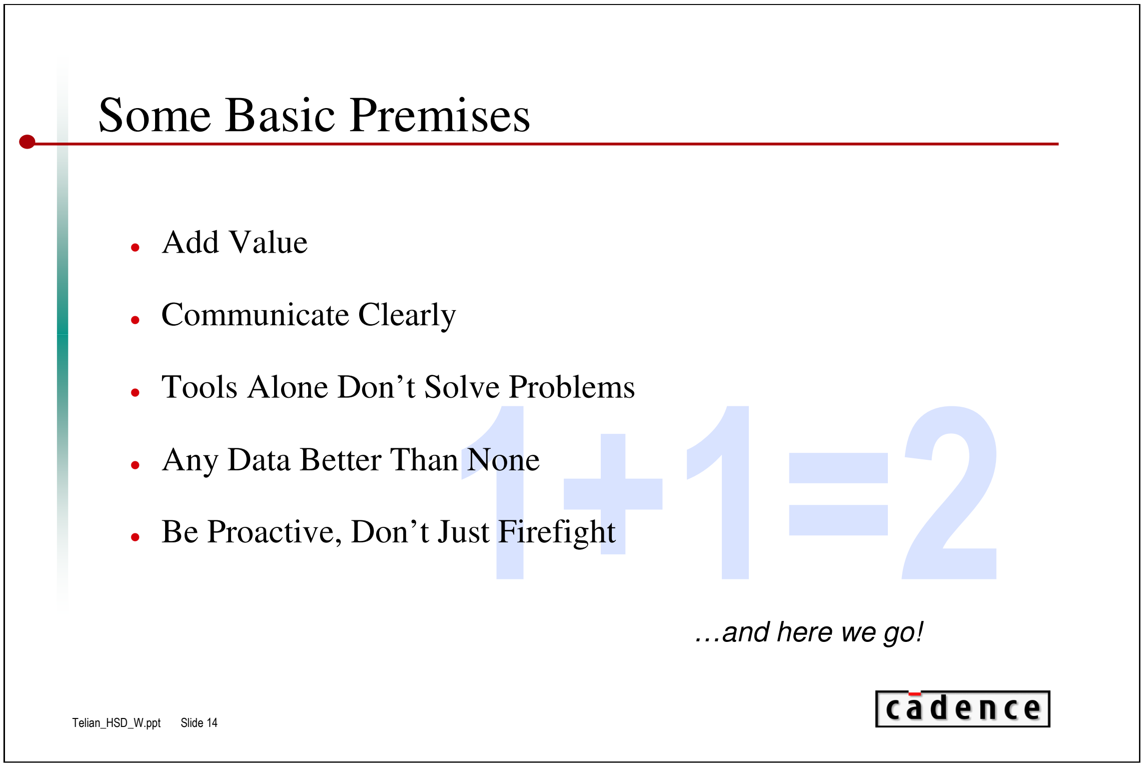
Technical

Telian_HSD_W.ppt Slide 13

cadence

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 tasks, gaining opportunity to perform the earlier functions 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.



## Some Basic Premises

- Add Value
- Communicate Clearly
- Tools Alone Don’t Solve Problems
- Any Data Better Than None
- Be Proactive, Don’t Just Firefight

1+1=2

*…and here we go!*

Telian_HSD_W.ppt Slide 14

Before jumping into the specific 7 tasks in detail, here’s some basic premises that apply to all of them.

First, be sure to **add value**. High-Speed Design has many, many side roads to travel that really don’t help the project team at all.

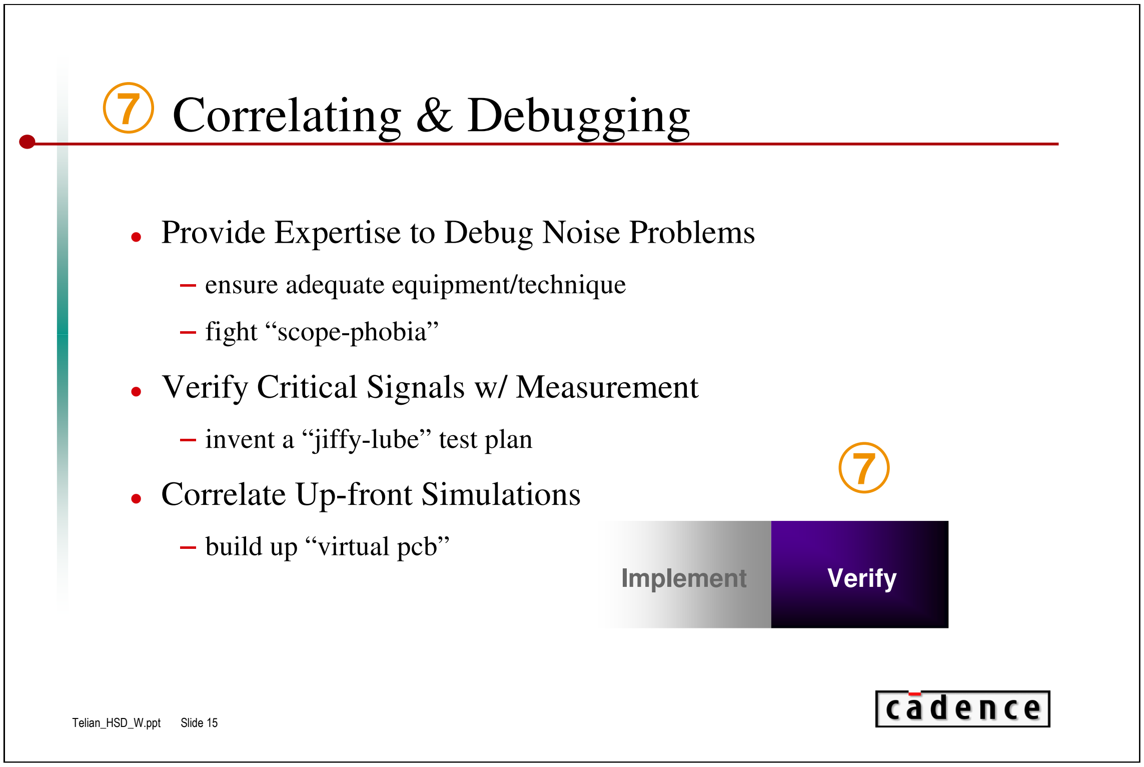
One manager of High-Speed Design Engineers put it this way: “I've worked with a few engineers in the past that were considered technical experts in [high-speed] related areas. Some of those folks were ineffective simply because they got bogged down studying details. These "scientists" studied and analyzed 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).”

Also **communicate clearly** and concisely. 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.

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.



## ⑦ Correlating & Debugging

- Provide Expertise to Debug Noise Problems
  - ensure adequate equipment/technique
  - fight “scope-phobia”
- Verify Critical Signals w/ Measurement
  - invent a “jiffy-lube” test plan
- Correlate Up-front Simulations
  - build up “virtual pcb”

⑦
Implement
Verify

Telian_HSD_W.ppt Slide 15

And now, we’ll cover the 7 ways to address High-Speed issues in detail.

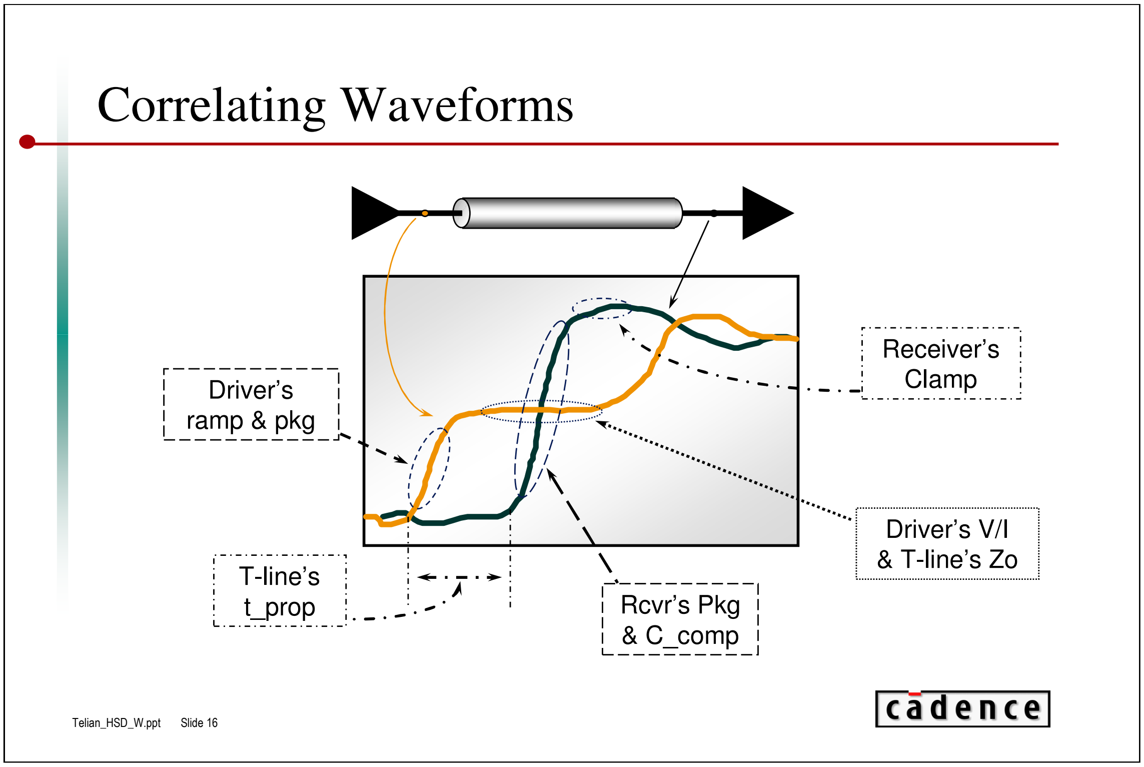
To cover the 7 tasks, we’ll start at the back of the design cycle with #7 and work our way forward. This is because this is the task most engineers have some familiarity with - debugging noise problems. Unfortunately, excessive debugging often occurs because many design shops do not have the foresight to correctly plan and staff the analysis task on the front end of the design 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 high-speed 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 is 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.



# Correlating Waveforms

Driver's ramp & pkg

T-line's t_prop

Rcvr's Pkg & C_comp

Receiver's Clamp

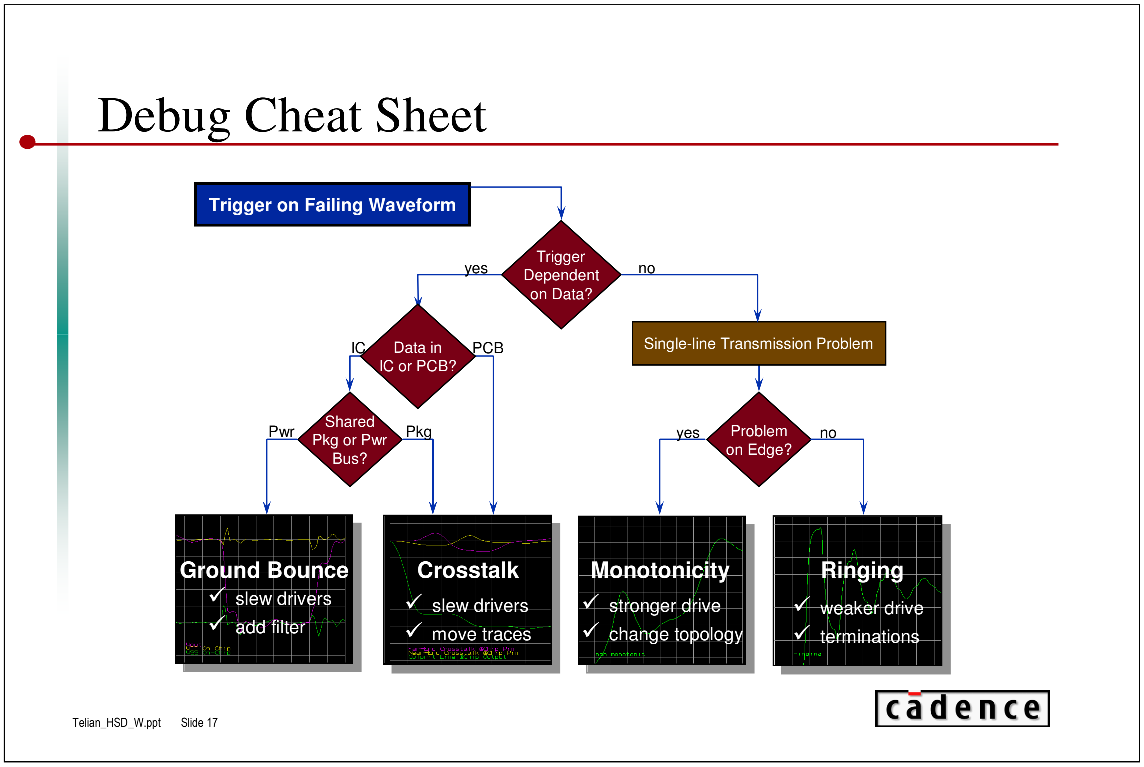
Driver's V/I & T-line's Zo

Telian_HSD_W.ppt Slide 16

cadence

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, http://www.eia.org/eig/ibis/ibis.htm) 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.



# Debug Cheat Sheet

Trigger on Failing Waveform

Trigger Dependent on Data?

yes

no

Data in IC or PCB?

IC

PCB

Single-line Transmission Problem

Shared Pkg or Pwr Bus?

Pwr

Pkg

Problem on Edge?

yes

no

**Ground Bounce**
- ✓ slew drivers
- ✓ add filter

**Crosstalk**
- ✓ slew drivers
- ✓ move traces

**Monotonicity**
- ✓ stronger drive
- ✓ change topology

**Ringing**
- ✓ weaker drive
- ✓ terminations

Telian_HSD_W.ppt Slide 17

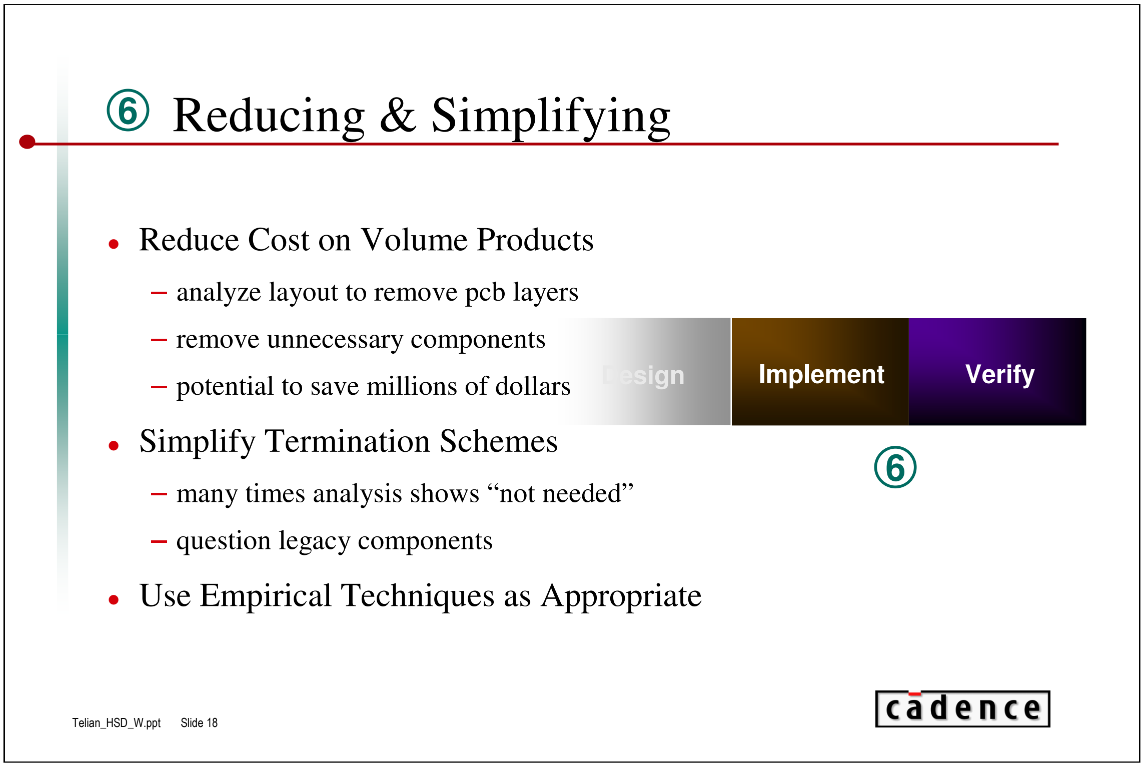
cadence

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 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.



## ⑥ Reducing & Simplifying

- Reduce Cost on Volume Products
  - analyze layout to remove pcb layers
  - remove unnecessary components
  - potential to save millions of dollars
- Simplify Termination Schemes
  - many times analysis shows “not needed”
  - question legacy components
- Use Empirical Techniques as Appropriate

Design
Implement
Verify
⑥

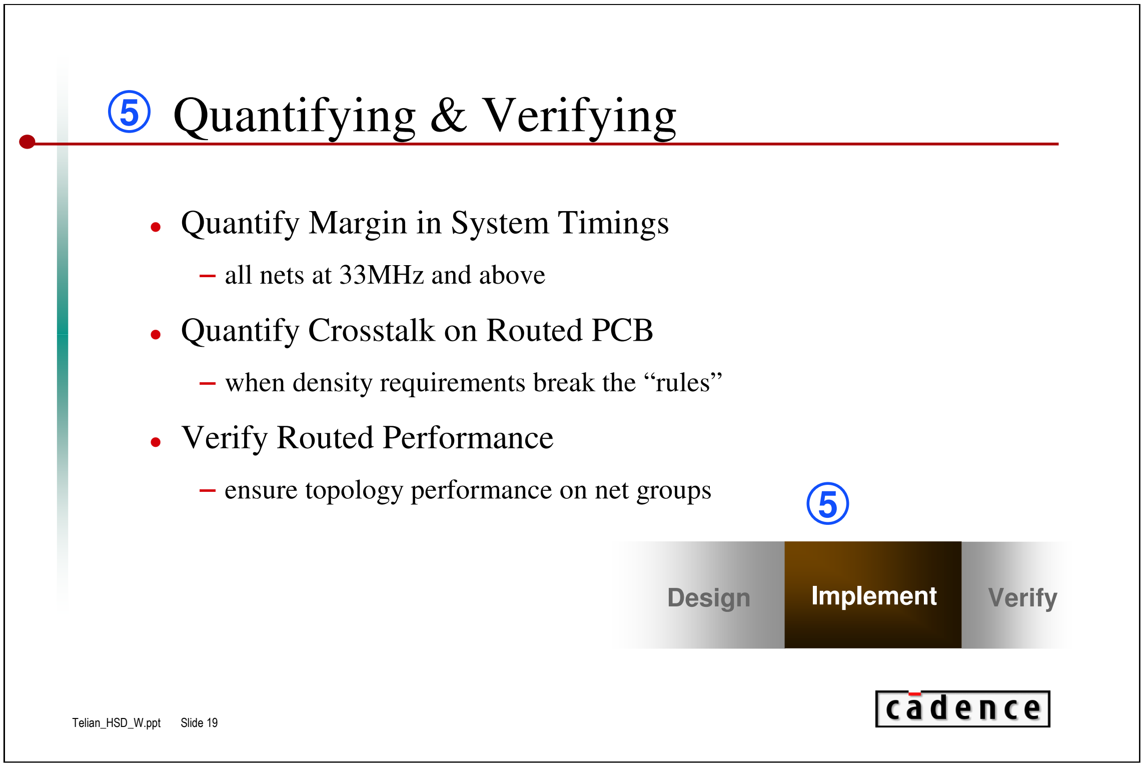
Telian_HSD_W.ppt Slide 18

cadence

This task 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 High-Speed Design 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.



## ⑤ Quantifying & Verifying

- Quantify Margin in System Timings
  - all nets at 33MHz and above
- Quantify Crosstalk on Routed PCB
  - when density requirements break the “rules”
- Verify Routed Performance
  - ensure topology performance on net groups

⑤

Design Implement Verify

Telian_HSD_W.ppt Slide 19

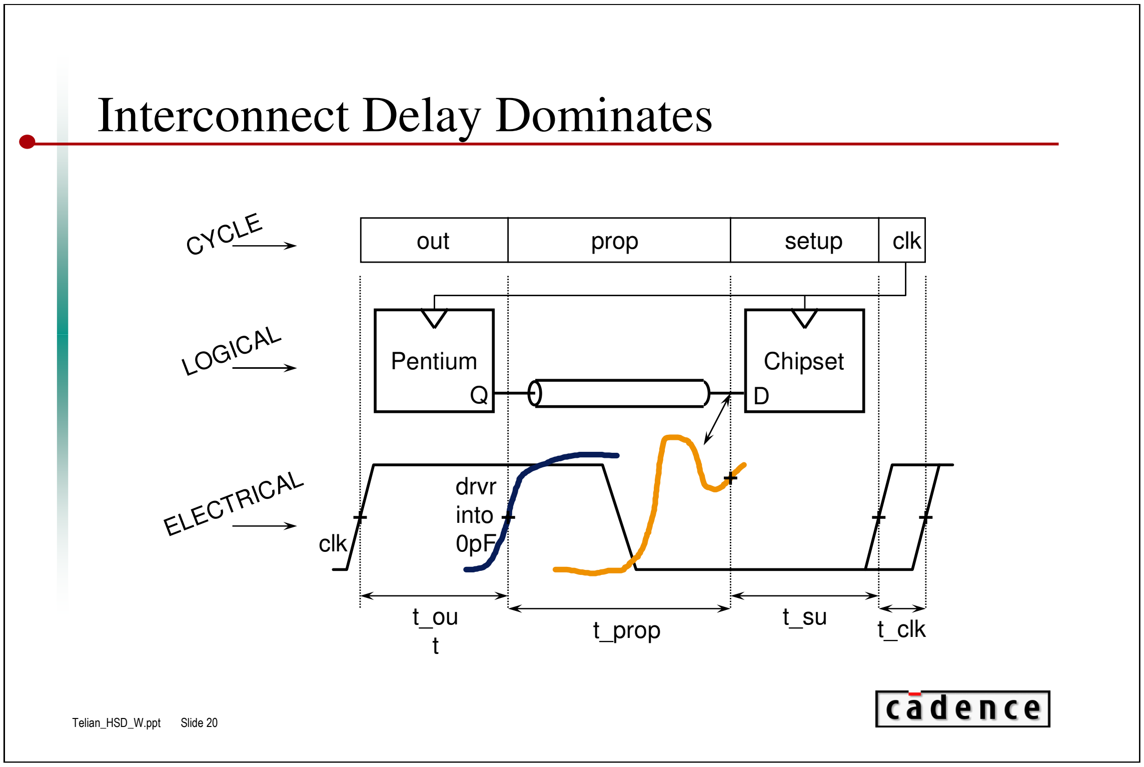
cadence

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

Simulation tools are used heavily in this role. Here 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 simulate system level 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.



## Interconnect Delay Dominates

Telian_HSD_W.ppt Slide 20

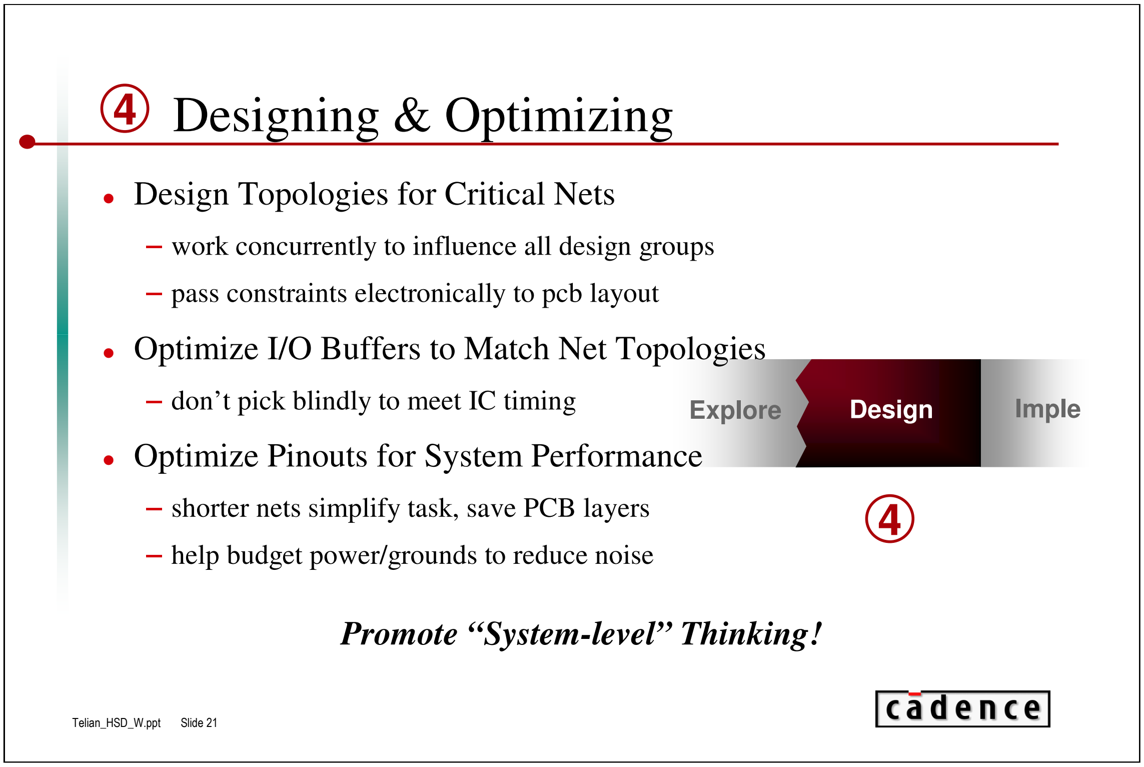
Signal propagation time across a PCB now represents a significant part of the cycle in today's high-speed digital systems, typically requiring about 1/3 of the available nanoseconds. Yet there is often confusion surrounding how that portion of the timings is allocated to the PCB.

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 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.

Simply said, the interconnect delay is a measure of the difference between the IC's testload and the actual system environment.



## ④ Designing & Optimizing

- Design Topologies for Critical Nets
  - work concurrently to influence all design groups
  - pass constraints electronically to pcb layout
- Optimize I/O Buffers to Match Net Topologies
  - don't pick blindly to meet IC timing
- Optimize Pinouts for System Performance
  - shorter nets simplify task, save PCB layers
  - help budget power/grounds to reduce noise

***Promote "System-level" Thinking!***

cadence

Telian_HSD_W.ppt Slide 21

Task #4 represents the heart of the design phase. Here, all groups (mechanical, board-level, and IC) must interface effectively to ensure that the high-speed sections of the design will work properly when physical implementation occurs.

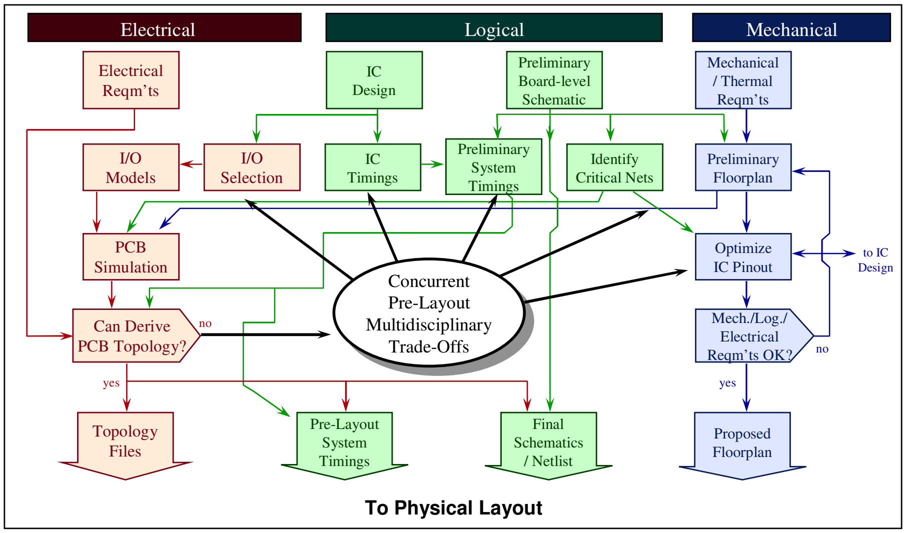
Prior to PCB layout, critical nets must be identified. It is important to explore implementation options on these nets and derive PCB topologies that will work across production and system environment tolerances. When options cannot be found, it is important to engage all design groups early enough to 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".

# Design Concurrently for High Performance



Electrical

Logical

Mechanical

Electrical Reqm’ts

IC Design

Preliminary Board-level Schematic

Mechanical / Thermal Reqm’ts

I/O Models

I/O Selection

IC Timings

Preliminary System Timings

Identify Critical Nets

Preliminary Floorplan

PCB Simulation

Optimize IC Pinout

to IC Design

Concurrent Pre-Layout Multidisciplinary Trade-Offs

Can Derive PCB Topology?

no

Mech./Log./ Electrical Reqm’ts OK?

no

yes

yes

Topology Files

Pre-Layout System Timings

Final Schematics / Netlist

Proposed Floorplan

To Physical Layout

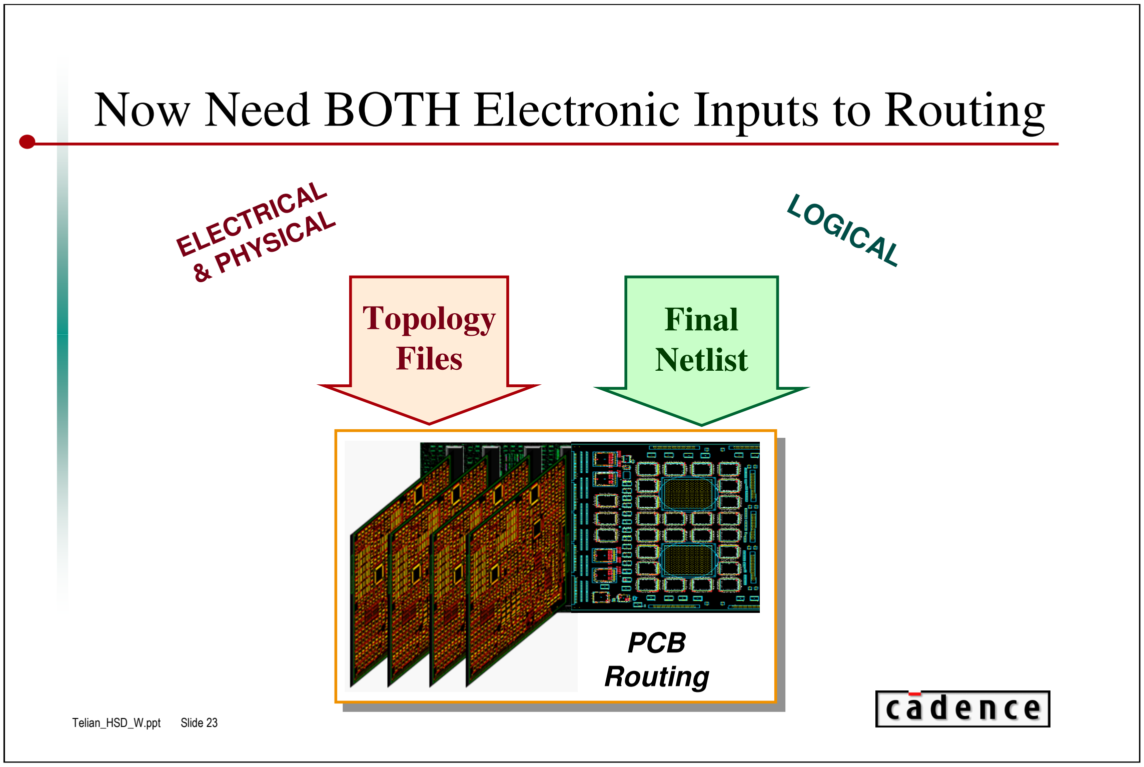
Telian_HSD_W.ppt Slide 22

This diagram presents a detailed process flow for the proper way to address high-speed issues and concurrently derive PCB topologies (shown at left). Successful High-Speed Engineering requires driving 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 Files” 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 high-speed problems arise when the design becomes physical and problems become harder to fix. Some design software can now read in these topology constraint files electronically to help guide the layout process. Automating this step helps to accelerate the design process, 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.



## Now Need BOTH Electronic Inputs to Routing

ELECTRICAL & PHYSICAL

LOGICAL

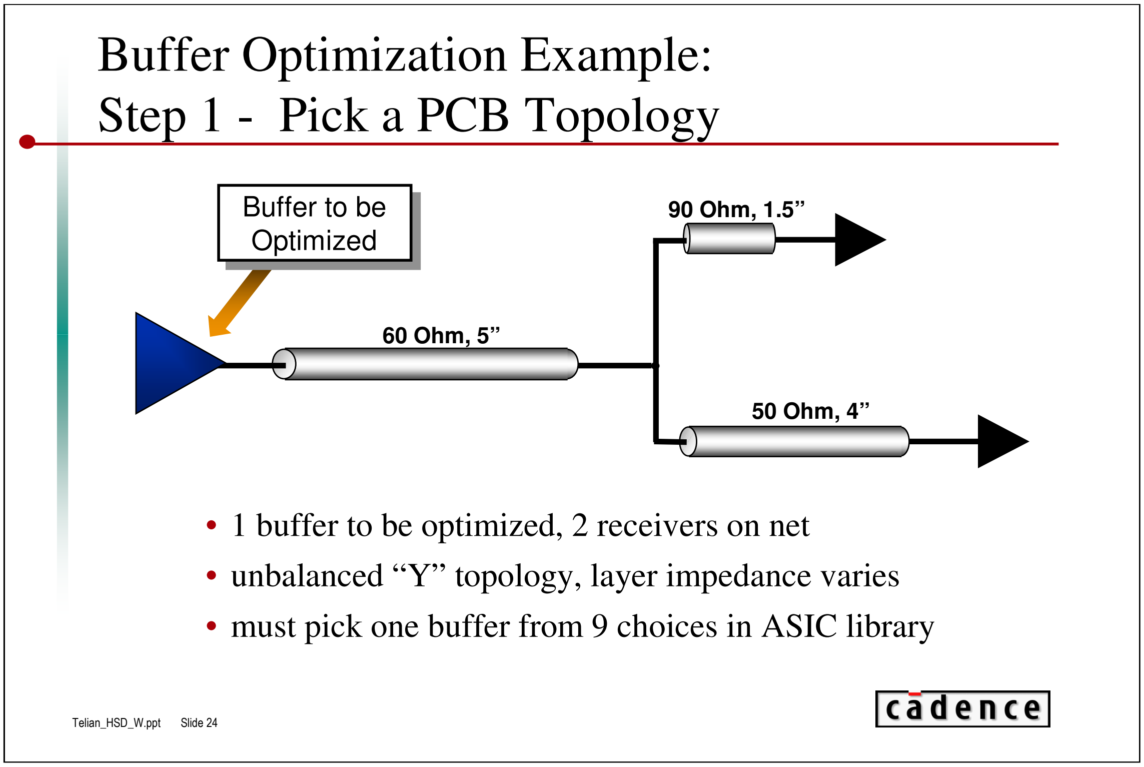
Topology Files

Final Netlist

PCB Routing

Telian_HSD_W.ppt Slide 23

So the engineering process must be modified to design high-speed PCBs. A successful route is no longer simply adherence to the logical connectivity requirements in a netlist, it must also satisfy electrical and physical requirements as well. Actually, that’s been fairly obvious for awhile, but the fact that now *most of the board* has high-speed ICs and nets has made the automation of this task imperative. To guide the routing of high-speed PCBs, the layout designer needs to load both netlist AND topology files to ensure the artwork implementation is robust.



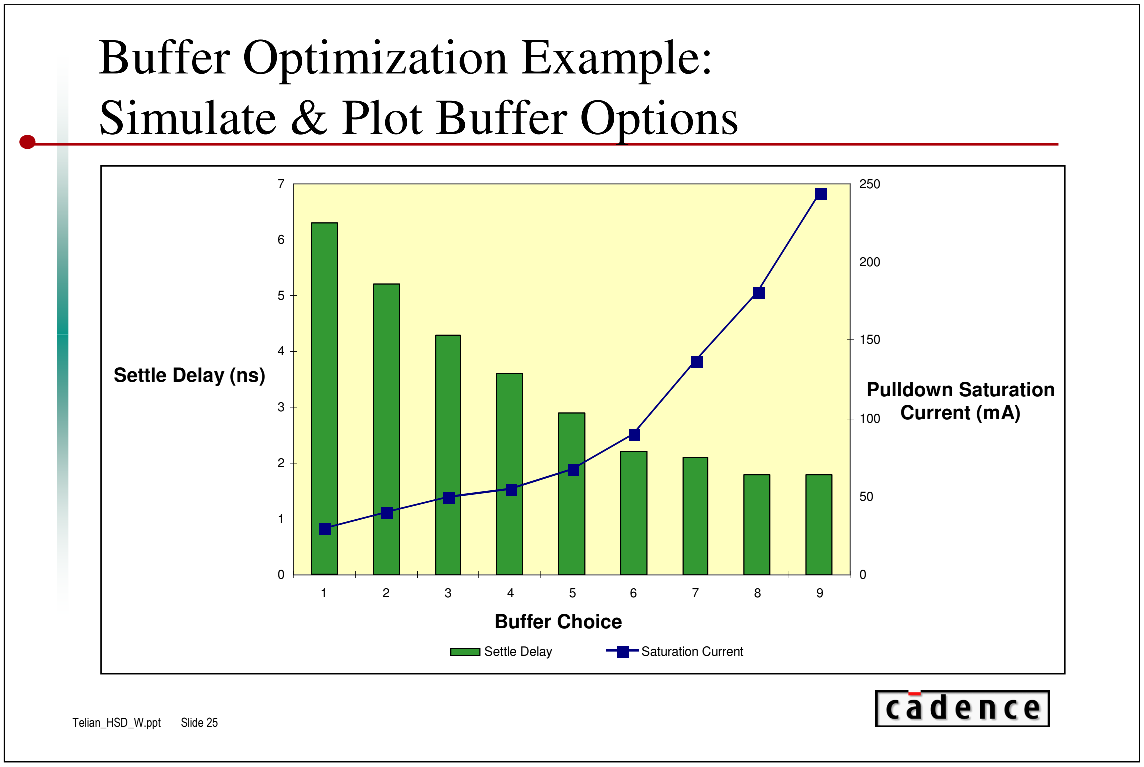
# Buffer Optimization Example: Step 1 - Pick a PCB Topology

- 1 buffer to be optimized, 2 receivers on net
- unbalanced “Y” topology, layer impedance varies
- must pick one buffer from 9 choices in ASIC library

Telian_HSD_W.ppt Slide 24

Now let’s approach the problem in another way. Sometimes mechanical, thermal, or other restrictions may require a certain physical topology. The question arises: “Can the IC’s buffers be optimized to support a certain required topology?”. Though most engineers are used to being handed ICs and trying to find a way to design them into a system, more and more companies are designing their own custom ICs and ASICs. This change makes it possible for many engineers to pick an optimal buffer for a given PCB topology, instead of picking an optimal PCB topology for a given buffer. In many situations, it is preferable to solve the problem from this perspective in order achieve the highest possible performance at the lowest cost.

Consider the 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.



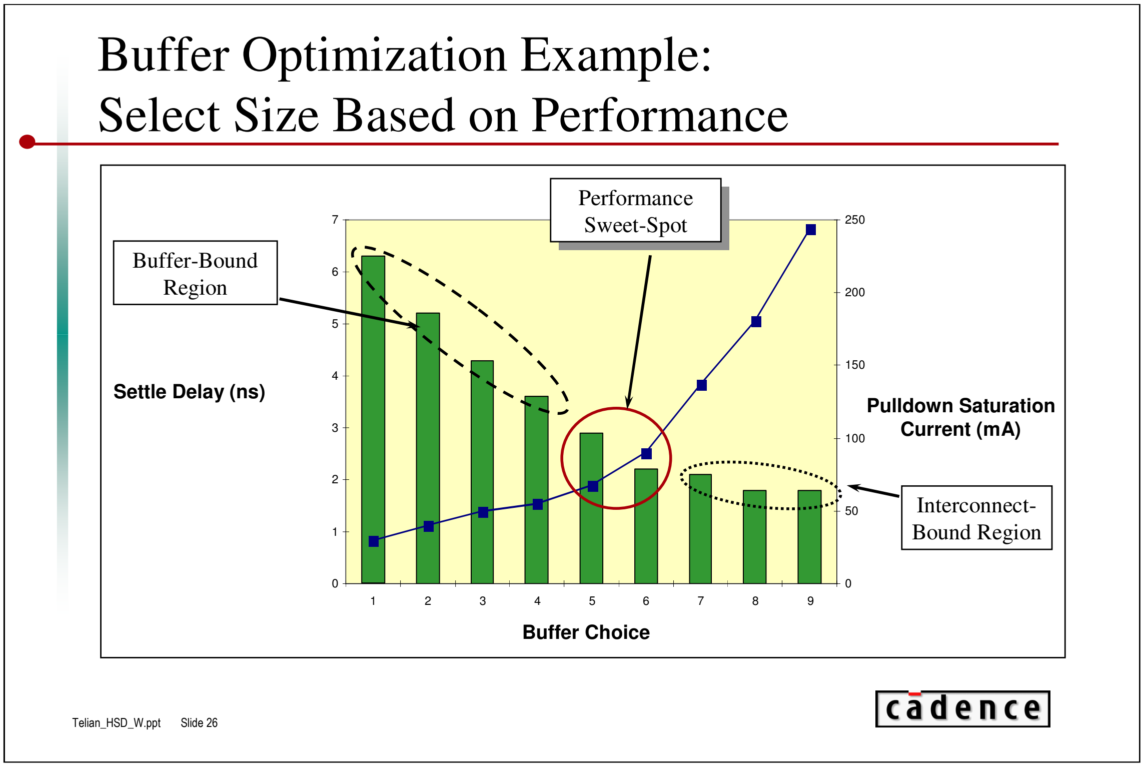
# Buffer Optimization Example: Simulate & Plot Buffer Options

Telian_HSD_W.ppt Slide 25

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.

Once simulated, the nine buffers 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 tested, 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 ranged 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.



# Buffer Optimization Example: Select Size Based on Performance

Buffer-Bound Region

Performance Sweet-Spot

Interconnect-Bound Region

Telian_HSD_W.ppt Slide 26

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

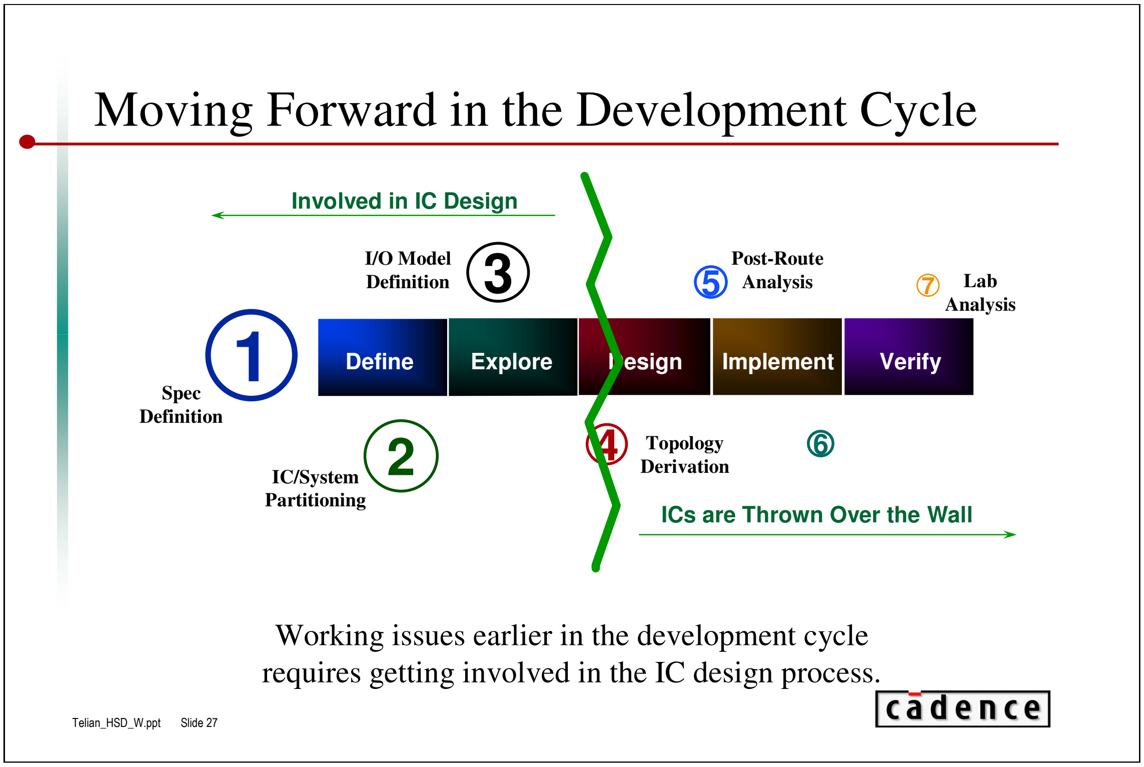
I should first point out that if system timings allowed 8 nS for this signal's propagation, buffer number 1 would be a great choice. However, since that amount of time is rarely available in today's high-speed systems we'll have to take a closer look.

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 the 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 design groups can add working together to concurrently address and solve high-speed issues.



# Moving Forward in the Development Cycle

Involved in IC Design

1 Spec Definition

2 IC/System Partitioning

3 I/O Model Definition

Define | Explore | Design | Implement | Verify

4 Topology Derivation

5 Post-Route Analysis

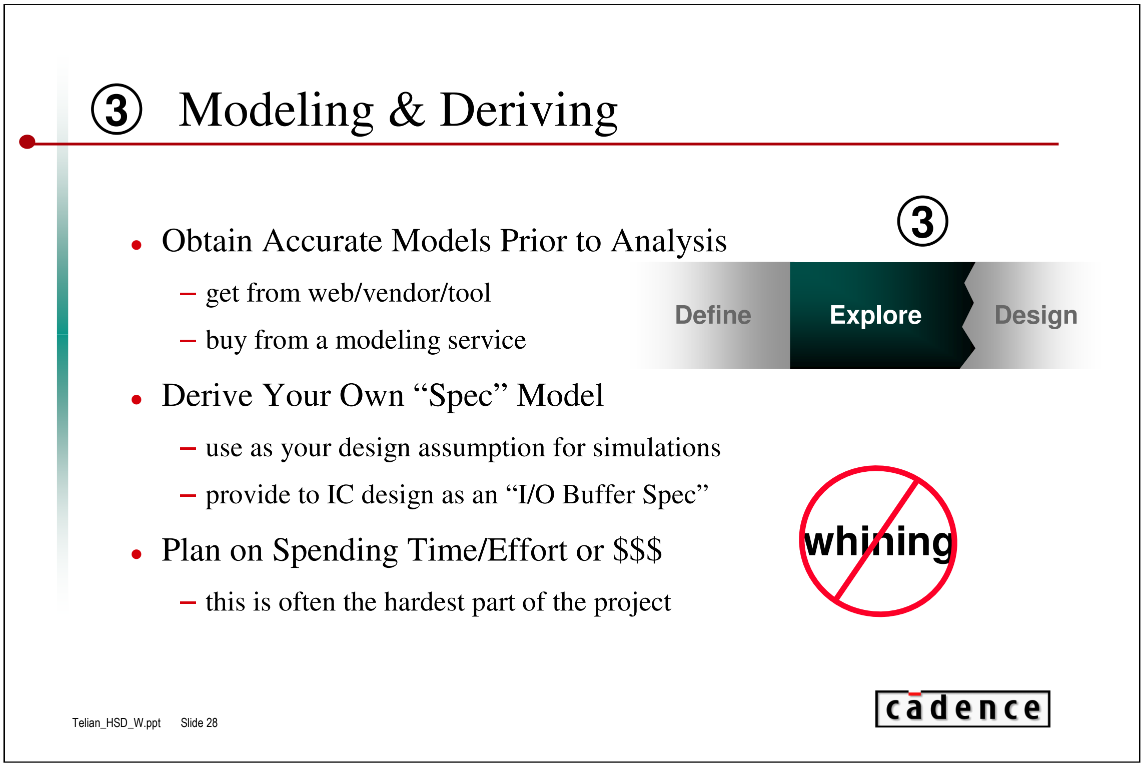
6

7 Lab Analysis

ICs are Thrown Over the Wall

Working issues earlier in the development cycle requires getting involved in the IC design process.

Before proceeding to task #3, I should point out that in task #4 we've crossed an interesting barrier. The quest for successful High-Speed Design of Digital Systems Using Today's Blazing Fast ICs has carried us into influencing the **design of the ICs themselves.** This is a natural progression in the effort to solve the problems earlier in the development cycle. It's important to recognize that engineers skilled in high-speed system issues have a lot of value to add in the development of ICs. Many "high-speed issues" are simply fallout from incorrectly designed ICs. Note that any IC will never make it into volume production if it passes test vectors and works on a silicon tester, but fails when it's soldered onto a PCB.



## ③ Modeling & Deriving

- Obtain Accurate Models Prior to Analysis
  - get from web/vendor/tool
  - buy from a modeling service
- Derive Your Own “Spec” Model
  - use as your design assumption for simulations
  - provide to IC design as an “I/O Buffer Spec”
- Plan on Spending Time/Effort or $$$
  - this is often the hardest part of the project

③

Define Explore Design

whining

Telian_HSD_W.ppt Slide 28

cadence

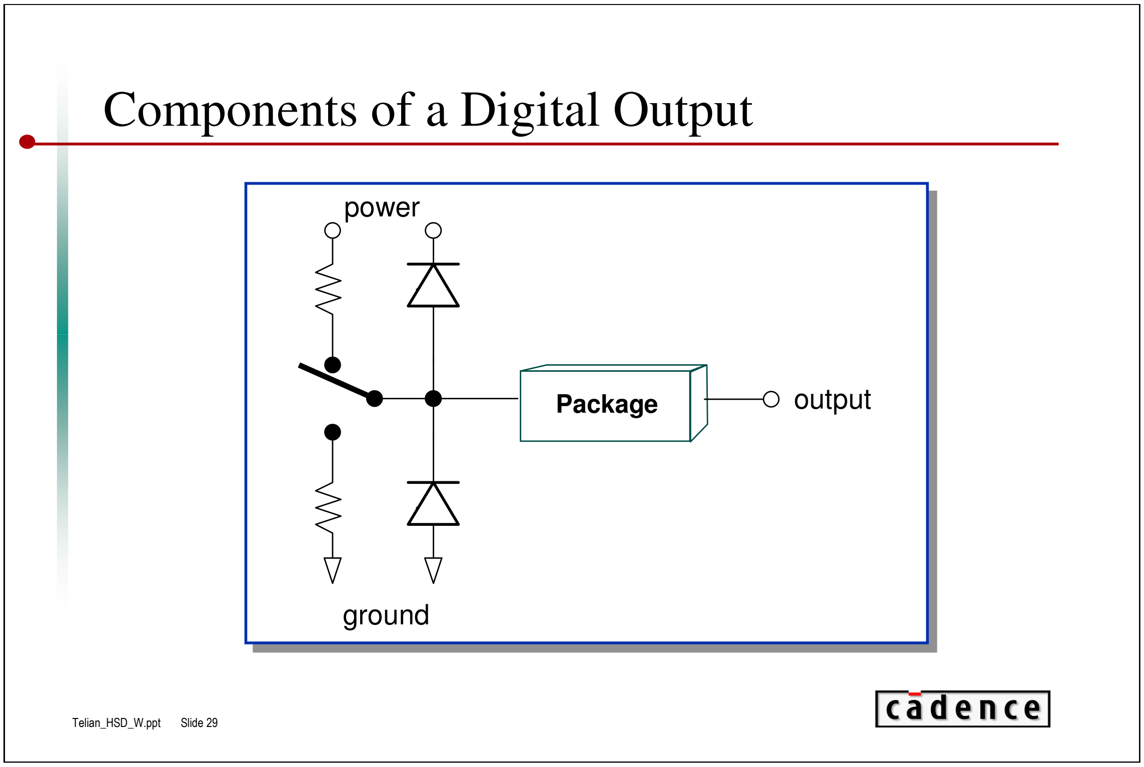
Often, the stalemate in the whole development process is models. One manager surveyed cited the need for “good VALIDATED concise models” as his number one issue related to successful High-Speed Design Engineering.

The effective design team 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. EDA vendors serious about enabling high-speed design should be able to direct you to an effective source for the models you need.

Another approach, as we just outlined in the buffer optimization example, is to derive a model of the buffer characteristics your design needs and pass that to the IC design team as a “spec”. Use this “spec model” to design your boards. Working this way optimizes performance, and gets you out of the loop of waiting for a model from the IC design team. However, be sure to verify that the final IC implementation matches your spec model.

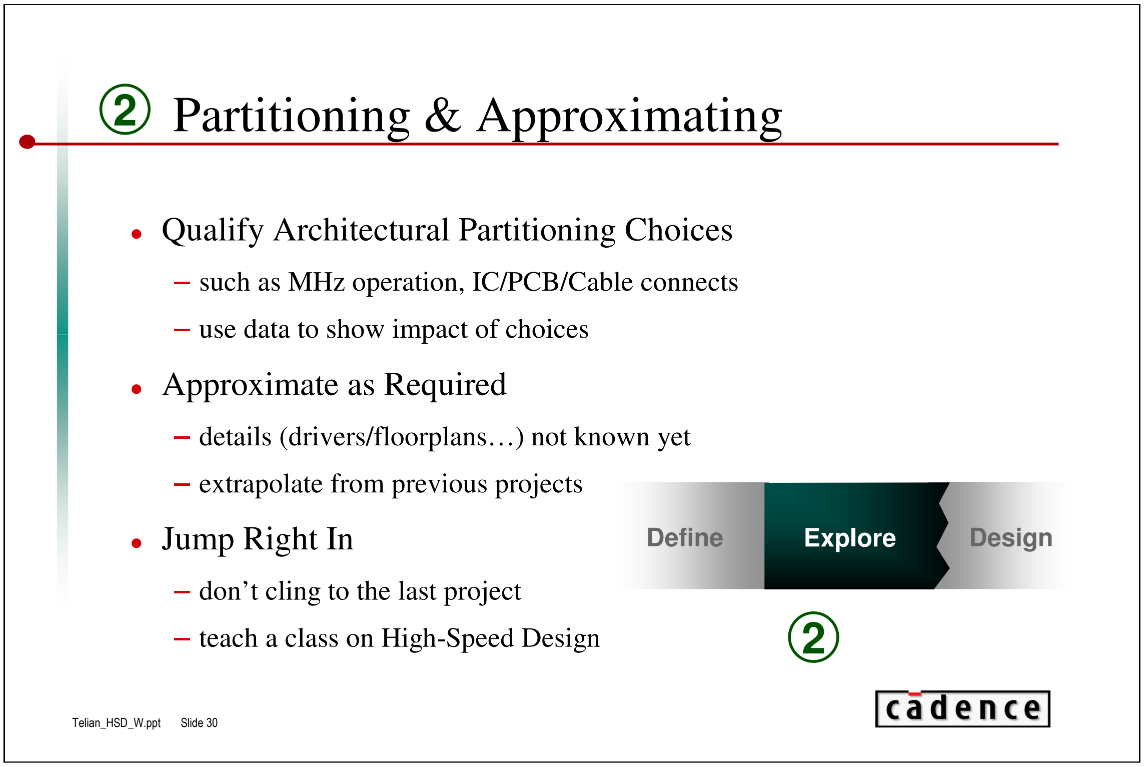
However you approach the problem, be sure models are in place prior to the design phase. Resolve not to whine, do the best you can, and don’t be afraid to extrapolate and use good engineering judgment to approximate the most accurate model possible within the time allowed. Never allow a project to be stalled because “I don’t have the model.”



# Components of a Digital Output

Telian_HSD_W.ppt Slide 29

High-Speed PCB design models normally just need a few key elements shown here. 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. Characteristics for each of these elements can be directly entered into a model in the IBIS format.



## ② Partitioning & Approximating

- Qualify Architectural Partitioning Choices
  - such as MHz operation, IC/PCB/Cable connects
  - use data to show impact of choices
- Approximate as Required
  - details (drivers/floorplans…) not known yet
  - extrapolate from previous projects
- Jump Right In
  - don't cling to the last project
  - teach a class on High-Speed Design

Define
Explore
Design
②

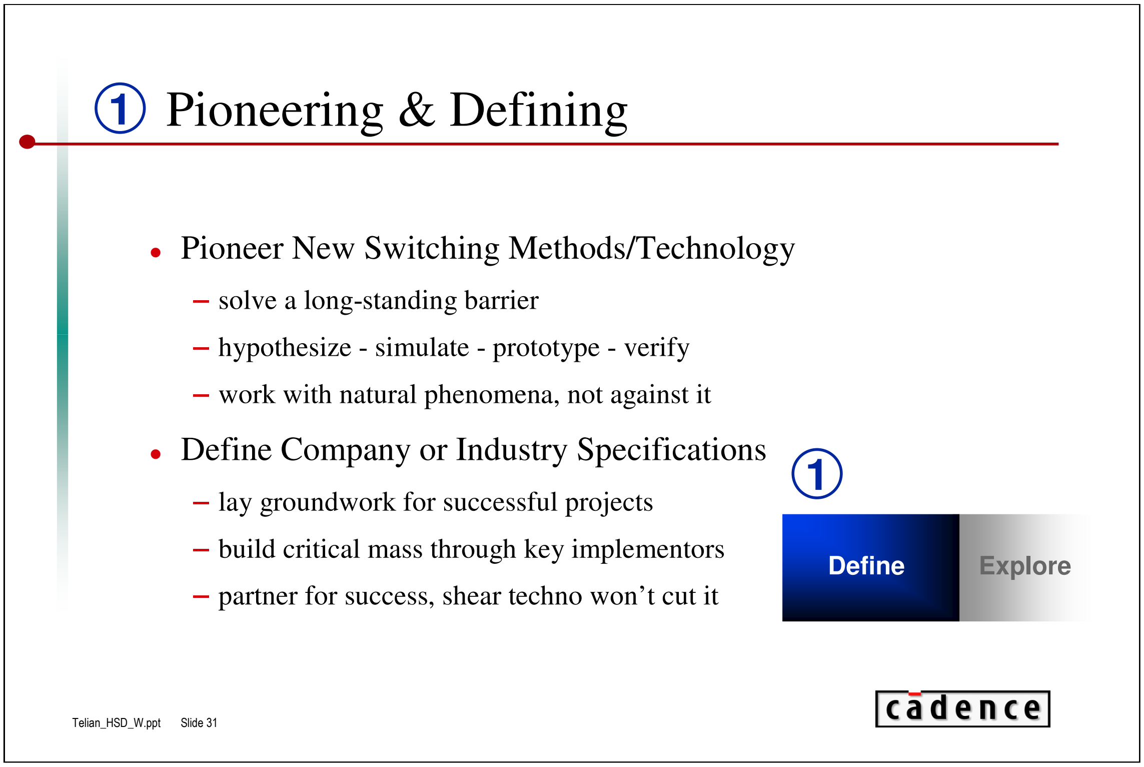
Telian_HSD_W.ppt Slide 30

During the "Explore" phase of the Hardware Development Cycle, High-Speed Design engages in task #2: Partitioning and Approximating.

This work normally goes on, but too often without the high-speed issues being addressed. The feasibility of proper high-speed operation must be factored into the discussion 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 high-speed issues, teach a class on High-Speed Design and properly explain the impact of architectural choices before it's too late. Engineers love to go to classes.



## ① Pioneering & Defining

- Pioneer New Switching Methods/Technology
  - solve a long-standing barrier
  - hypothesize - simulate - prototype - verify
  - work with natural phenomena, not against it
- Define Company or Industry Specifications
  - lay groundwork for successful projects
  - build critical mass through key implementors
  - partner for success, shear techno won’t cut it

①

Define Explore

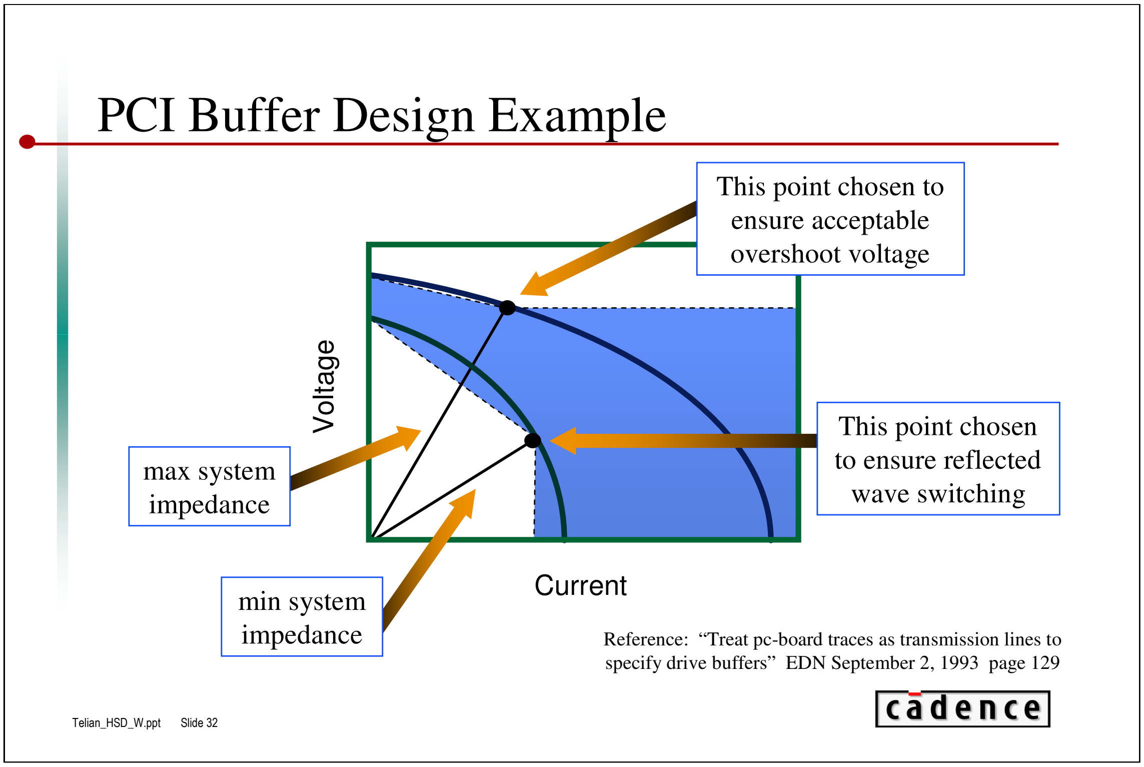
Telian_HSD_W.ppt Slide 31

cadence

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 implementors 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.



# PCI Buffer Design Example

This point chosen to ensure acceptable overshoot voltage

This point chosen to ensure reflected wave switching

max system impedance

min system impedance

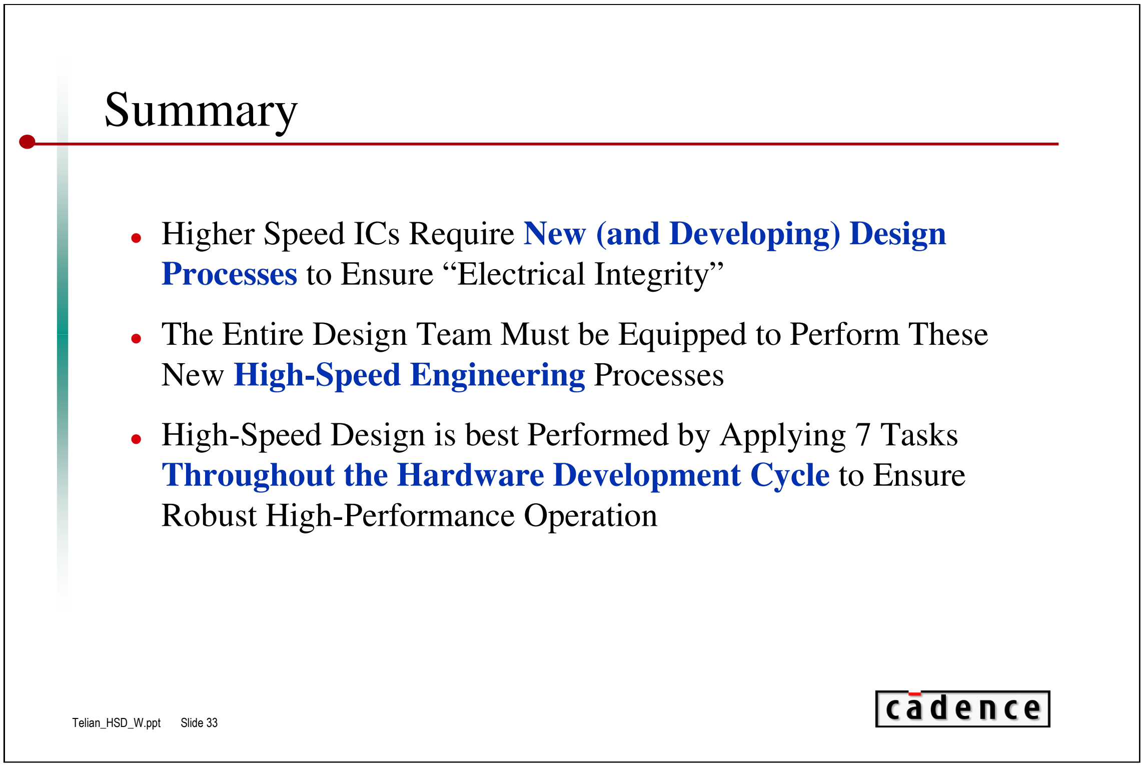
Reference: "Treat pc-board traces as transmission lines to specify drive buffers" EDN September 2, 1993 page 129

Telian_HSD_W.ppt Slide 32

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

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. As a result, the dynamic characteristics of all PCI devices are defined and understood. This has enabled a variety industry implementations because the design problem is bounded. "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.



## Summary

- Higher Speed ICs Require **New (and Developing) Design Processes** to Ensure “Electrical Integrity”
- The Entire Design Team Must be Equipped to Perform These New **High-Speed Engineering** Processes
- High-Speed Design is best Performed by Applying 7 Tasks **Throughout the Hardware Development Cycle** to Ensure Robust High-Performance Operation

Telian_HSD_W.ppt Slide 33

cadence

In this talk, we have described how to Design High-Speed Digital Systems Using Today’s Blazing Fast ICs. This has required us to approach the problem with an understanding of a variety of areas: digital, analog, IC design, PCB layout, System design, simulation, modeling, partnering, mechanical - a whole system-level approach.

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.

Design teams working on high-speed projects must be well equipped to address these new requirements and perform the task through a combination of tools, processes, training, and expertise.

Throughout the Hardware Development Cycle, 7 unique tasks were outlined and illustrated. When performed effectively, these roles dramatically increase a new product’s performance, reliability, and time to market without adversely affecting cost.