Macromodels for Generating Signal Integrity and Timing Management Advice for Package Design

Paul Franzon, Slobodan Simovich, Sharad Mehrotra, Michael Steer,
PicoLab
Department of Electrical and Computer Engineering
North Carolina State University,
Raleigh NC 27695-7911

Abstract

The electrical design of packaging for high speed digital systems requires intensive efforts on the part of signal integrity engineers. We have produced a set of tools that assist these engineers in efficiently producing PCB and MCM designs that meet timing and other electrical needs. This paper describes the most important aspect of this solution, the internal 'macromodels' that accurately capture the relationships between electrical/timing design and the package physical design (or layout).

1 Introduction

In high speed PCBs and MCMs it is necessary to constrain physical parameters such as cross-sections, lengths, via count and topologies of interconnection nets (MCM) to ensure that timing and noise (signal integrity) requirements are met [3].[1] The physical constraints are often referred to as wiring rules. Ensuring proper electrical design of PCBs and MCMs is a daunting task for the signal integrity and other packaging engineers. It often involves a lot of tedious simulation and analysis work. Also, the few experts who understand the process are often spread thin over many designs and are often even called in too late to influence the design into the correct direction. Our work has resulted in a set of tools that provide computer-aided assistance to the signal integrity engineer. These tools remove much of the drudgery and result in designs that are correct-by-construction. In this paper we describe the most important technical aspect of this tool set, the internal 'macromodels' used to accurately capture the relationships between electrical/timing design and physical design (or layout).

Currently, there are two approaches used to generate the physical constraints for each net in a design. The current mainstream industry computer-aided approach to managing timing and signal integrity requirements is illustrated in Figure 1. There are a number of problems with this approach:

- Even the most complete theory-based equations for predicting delay and noise assume ideal circuits including linear drivers and receivers, point to point nets, uniform transmission lines, and no simultaneous switching noise. Because these non-ideal circuit properties are not taken into account the theory-based equations are inaccurate for most nets and the post-layout simulator reports many delay and signal integrity violations that the theory-based equations did not predict. The extract-simulate-adjust loop must be iterated many times, increasing total design time.
- Rules of thumb, such as 'zero length stubs' over-constrain the router. The effective result is increased routing overflows and possibly additional layers.
- Specifying and adjusting the rules requires the attention of a signal integrity expert as well as design and layout engineers. This is inefficient. Furthermore, the signal integrity expert is often not called in until it is found that the prototype does not work! Correct-by-construction approaches are needed.

The other approach, reported mainly in the context of its use in designing high speed computers at IBM [3][6] involves a team of signal integrity engineers working concurrently with the computer design engineers (their methodology for signal nets only is described):

1. Through extensive simulation studies, the signal integrity engineers work out, for each net class of interest, the constraints on lengths and stub capacitances, etc., that guarantee first incident switching. Within IBM, these constraints are called 'wiring rules' which is a different meaning from that discussed in the first paragraph above and used throughout this paper.
2. The signal integrity engineers then conduct further simulation studies to empirically determine a delay equation for

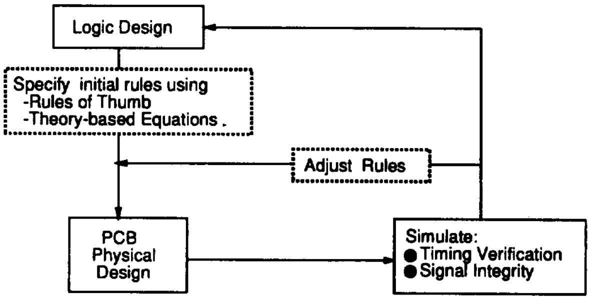


Figure 1: Current Approach to Managing Wiring Rules.

[1] This research was supported by the National Science Foundation under grant MIP-901704 and by Cadence Design Systems.

0569-5503/93/0000-0523 $3.00 ©1993 IEEE

each net class. The delay equation is only determined within the bounds on length, etc. that guarantee first incident switching.

3. These 'wiring rules' and delay equations are used by the circuit and package design engineers, through appropriate floorplanning, placement, and routing CAD tools. If the design engineers find the rules and equations too restrictive to produce an effective design they work closely with the signal integrity engineers to obtain more suitable rules.

Though highly successful, there are some drawbacks to this approach. It requires large amounts of human effort and expertise to generate and maintain the rules and equations. In order to make the process tractable to humans, certain simplifying assumptions have to be made. In particular, they specify that all signals must be first incident switching and construct the wiring rule to reflect that. This means that the resulting rules are somewhat more restrictive than they would otherwise be. The result is that during design, the designers make requests to the circuit analysis team for rule modifications realizing that the rules are conservative and simplified.

Central to either of these approaches is the development of a high level model that captures circuit delay and noise responses as functions of physical variables, such as net length. In the first approach simple analytic and qualitative functions are used. These functions are approximate and often inaccurate. In the second approach, polynomial functions are fitted over a restricted range of physical variables. While these functions are accurate, their tight ranges of validity artificially restrict the designs.

An improved approach is to develop a set of global models that apply over the full range of design variables (e.g. net lengths). These are called *macromodels.* The usual use of the term 'macromodel' is to signify a simplified model used to replace a more complex model in a circuit simulation, so as to speed the simulation up. Though our macromodels can be used as fast 'simulations' (justifying their name) their main use here is for obtaining wiring rules.

In this paper, we discuss the automatic production of macromodels and how they are applied to a design. In this approach, the macromodels are built from simulation studies, allowing them to be accurate, thus avoiding the problems of the first approach above. Because they are built automatically, not manually, they are cheaper to produce than in the approach used at IBM. Also, as they are built by computers, not humans, fewer simplifying assumptions are made in their construction.

The remainder of this paper is organized as follows. First we discuss what is needed from the macromodels in order for them to be useful. Then we show how the macromodels that meet these needs are generated. Two types of macromodels are generated, the mean-fit macromodel and the conservative-design macromodel. We show how the second type is used to generate wiring rules that can then be used by placement and routing tools.

An example is integrated throughout this paper. The example is taken from an MCM implementation of a small computer design. The MCM process is the MCM-D process developed at MCNC. The small computer design (provided by Cadence Design Systems) had seventeen classes of nets, where each class has a different number of loads and a different driver. In the example in this paper, we automatically develop the macromodel for one class of nets, the custom-CMOS driven, two-receiver nets. We then show how this macromodel is used to produce a wiring rule for one of the nets in the design. The same macromodelling process would be applied to the other sixteen net classes and the rule production process to all of the other nets in the design in order to obtain a full set of wiring rules.

2 Macromodels for Timing and Signal Integrity Design

In this section, we first list the electrical responses of interest which we need to macromodel. We then discuss why these electrical responses must be captured in equation form in order to formulate wiring rules. We also define and justify the macromodel types employed.

Fundamentally, data signals only have to satisfy the following requirements [4]:

1. The signal be stable, within a noise margin of the logic-0 or logic-1 nominal voltage, at each latch in the system during the maximum possible extent of the set-up and hold period of the latch.
2. The positive and negative overshoot be constrained in order to prevent programmable logic reset, etc.

Given a logical design, and the min/max timing of every gate in that design, a timing analyzer will return the min/max timing slack on every interconnect net in the system. In order to meet the requirements for signal nets, the *settling delay* (Figure 2) on each net must have a value within the bounds of the timing slack. Settling delay on an interconnect is defined as (Figure 2) the delay from the signal at the start of the net (driver end) passing through the 50% point in the voltage swing to when the signal settles to within the allocated noise budget for reflection noise at each receiver. (The requirements on clock signals are much more stringent. They require first incident switching, minimum or no porching, and minimum levels of overshoot.) Using settling delay as the interconnect net delay criterion has advantages and disadvantages.

The main advantage is gained from the realization that not all signals will need to be first incident switching to obtain a satisfactory design, though the interconnect nets with the smallest timing slacks are likely to require it. By dictating first incident switching as part of the wiring rule process, the designer is over-constraining many of the interconnections, specifying their longest length to be shorter than necessary.

As expected, settling delay is not a smooth function of the physical design parameters. For example, consider the two terminal net shown in Figure 3. Figure 4 shows part of a set of simulation results (defining a *response surface*) showing settling delay as a function of wire lengths for an unterminated MCM-D interconnection [5]. Figure 5 shows similar results for a matching terminated lossless PCB interconnection. In both cases, the response is a highly non-linear function with respect to lengths. The reasons for the non-linearities are quite simple. As the branch lengths change so do the magnitudes of the undershoots in the ringing. If a small change in a length causes the peak voltage of an undershoot to exceed the noise budget, the waveform feature that defines settling delay is switched to this voltage peak from another and settling delay changes significantly. In the case of the

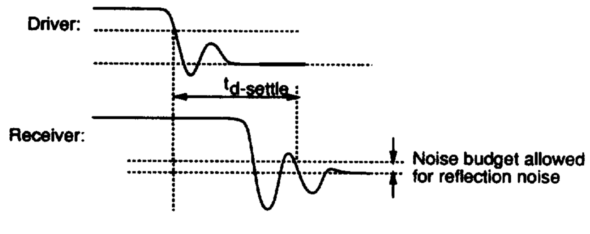


Figure 2: Definition of settling delay $t_{d-settle}$.

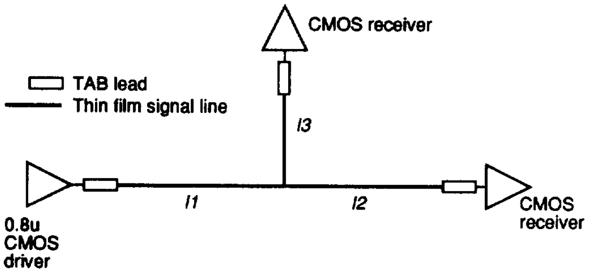


Figure 3: Two terminal net, showing the length design parameters.

lossy MCM interconnect, line losses suppress ringing in long lines and the first undershoot voltage becomes less than the reflection noise budget at some point, resulting in a sudden drop in settling delay.

Inspecting the response surfaces in Figures 4 and 5 shows that these irregularities make the fitting of linear and other polynomial macromodels difficult. However, such forms are needed for the macromodel to be useful in formulating wiring rules. For example, if delay is expressed as a quadratic function of length,

$$t(l) = a_0 + a_1 l + a_2 l^2, \tag{1}$$

then the length ranges that meet a delay requirement of say $t(l) =$ 1.2 ns can easily be found mathematically. One way around this problem would be to choose a portion of the response surface that is sufficiently smooth for fitting a single equation and model that portion only. If we did this, we would be artificially restricting the follow-on design tools (e.g. the router) to lengths, etc, that that were covered in the sufficiently smooth portion. As we are seeking a conservative design, another solution would be to form a macromodel for which the modeled response, say $t(l)$ would always be greater than the actual response (assuming we were interested in restricting maximum delays in the design). However, such a macromodel is actually very conservative, the difference between the modeled and true responses being large. Again, this artificially restrains the follow-on tools.

The method we use to deal with this issue is to capture the entire response surface as a piecewise linear macromodel. In this macromodel, the region for which the response surface was obtained is divided into a number of sub-regions and in each subregion a linear equation is fitted to the response. Generating this macromodel for the general case, where the settling delay or other circuit electrical response might be a function of six or more physical variables, requires an automated approach. In the next section, we present this automated approach, starting with a brief description of how the response surfaces are obtained.

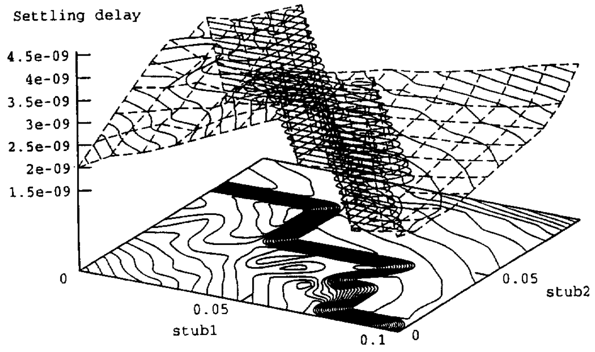


Figure 4: Response surfaces characterizing settling delay over a range of lengths for a lossy net on an MCM-D in the net class given in Figure 3. 'stub1' is labelled 'l3' in Figure 3 and 'stub2' is labelled as 'l2'.

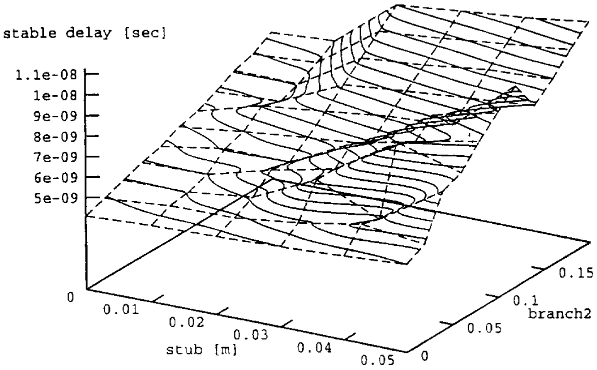
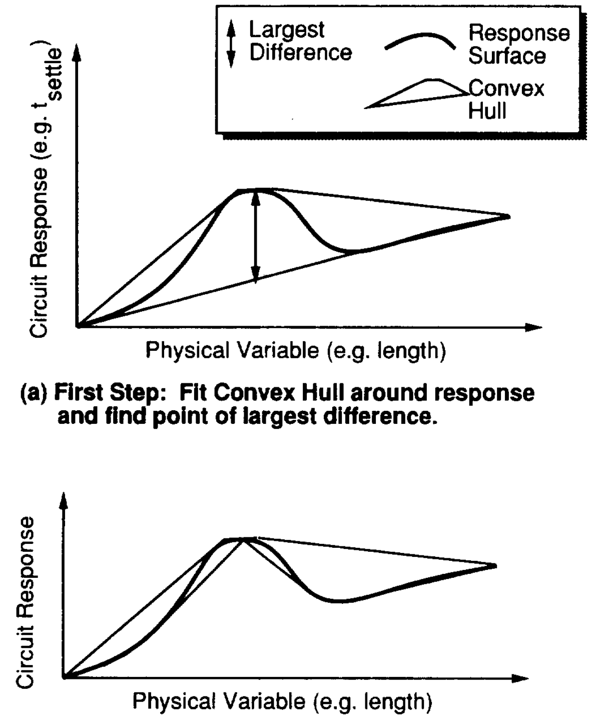


Figure 5: Response surfaces characterizing settling delay over a range of lengths for a lossless net on an PCB in the net class given in Figure 3. 'stub' is labeled as 'l3' in Figure 3 and 'branch2' is labelled as 'l2'.

3 The Generation of the Macromodels

The first step in generating the macromodels is to characterize the response surfaces for the timing and voltage parameters of interest over a certain physical design space. (A 'physical design space' is simply the ranges for the physical variables of interest, e.g. $l_1 < 10$ cm, $l_2 < 10$ cm, etc.) The response surface is characterized by automatically designing a *sequential computer experiment*. In each step of this experiment, a set of points are selected for sampling via simulation. After each step, it is determined if this set of points provides a sufficiently accurate characterization when interpolated over the entire design space. If it does not, more points are sampled. The details of this process are described elsewhere [8]. Using this approach, we are able to achieve sufficiently accurate characterizations with typically only a few hundred simulations. The responses surfaces that are partially shown in Figures 4 and 5 were each obtained with 200 simulations. The number of samples needed does not depend on the number of physical design variables, or their ranges, but on the linearity of the response surface.

The key to obtaining a piecewise linear macromodel is to make



(a) First Step: Fit Convex Hull around response and find point of largest difference.

(b) Second step: Divide into two sub-regions and fit convex hull around response in each region.

Figure 6: Example of basic steps used to generate a macromodel for a single-input-variable design space.

good choices for the boundaries between the linearly fitted sub-regions. In contrast to the approach where the regions are determined a-priori [2], based on the designer's previous experience, our approach is to perform an automated determination that does not require the use of the designer's knowledge. We have developed a heuristic 'greedy-cut' algorithm to do this. We illustrate the algorithm with a simple example before presenting its details.

Imagine a specified design space that consists of a single dimension and the characterized response is as shown at the top of Figure 6. The algorithm first finds the convex hull that completely encloses the data. This is also shown at the top of Figure 6. It compares this convex hull with the data and determines the point where the difference between either the lower or upper surfaces and the data is the largest. The design space is then split into two subregions, one on each side of this point, as shown at the bottom of Figure 6. A convex hull is then generated around each sub-region as shown. These steps are repeated until some stopping criterion is met.

More formally, the steps in the algorithm are as follows:

Step 1. Perform the following steps:

(**a.**) Read in the set S of N_p points that characterizes the response in the D-dimensional space R^D. The first $D-1$ dimensions are the dimensions of the physical design space R^{D-1} and the D-th dimension is the response variable. This dimension is referred to as the 'vertical' dimension and defines 'above' and 'below'.

(**b.**) The boundaries of the space that contains the set S of points is the initial 'current' sub-region. These boundaries are assumed to specify a convex sub-region.

(**c.**) Fit the convex hull around the points S.

(**d.**) Set the stopping criterion. Stop either when $N_{reg-max}$ ($N_{reg-max} > 1$) sub-regions are generated or the maximum vertical difference between a point and any facet of any convex hull is less than d_{max}.

Step 2. Count the number of points contained within the 'current' sub-region. If there are less than $D+1$ points then fit a plane (in D dimensions) through those points, remove the sub-region from further consideration, and proceed to step 4.

Step 3. For the 'current' sub-region, determine that point that has the furthest 'vertical' distance from any facet (surface) of the convex hull. Call that facet fw, and do the following:

(**a.**) Find the $D-1$ points above or below the facet fw that have the largest vertical distance from that facet. If there are less than $D-1$ points above or below facet fw then use those points and select others at random from the rest of the set in order to obtain $D-1$ points.

(**b.**) Construct the vertical *greedy-cut* plane that runs through these $D-1$ points.

(**c.**) Separate the set of points S_i into two sub-sets S_i^1 and S_i^2, defining new sub-regions that lie on either side of the greedy-cut plane.

(**d.**) If there are fewer than $D+1$ points in either sub-region then then remove that sub-region from further consideration.

(**e.**) Add the sub-regions with greater than D points containing S_i^1 and S_i^2 to the set of sub-regions $\{S_i\}$ and determine the convex hulls that contain the points in each sub-region.

Step 4. If the number of obtained subregions (including those set aside in Steps 2 and 3(d)) is about to exceed $N_{reg-max}$, then exit.

Step 5. Determine the sub-region that has the maximum vertical distance between a point and a facet fw of the convex hull. If this distance is less than d_{max}, or all the regions have been set aside as a result of steps 2 and 3(d), then exit. Else, make this the 'current' sub-region and go to step 2.

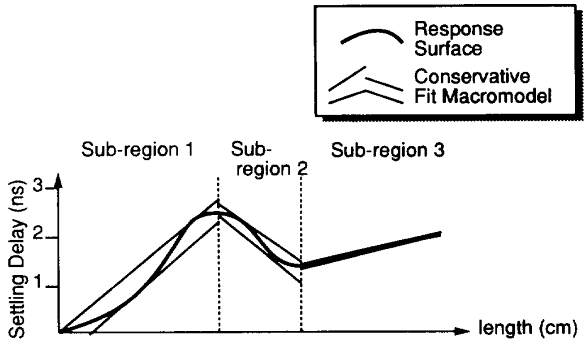


Figure 7: Simple example of a conservative-design macromodel.

Table 1: Equations for 'above' macromodel for the entire initial region and two of the sub-regions.

Region	Equation: t_{settle} (ns) =	Domain (cm)
Initial	$6.932 + 0.412l_1 + 0.242l_2 + 0.188l_3$	$0 \leq l_1, l_2, l_3 \leq 10$
subreg 1	$0.824 + 0.407l_1 + 0.332l_2 + 0.093l_3$	$0 \leq l_1, l_2, l_3 \leq 10$ $3.19 - 0.141l_1 + 0.545l_2 - l_3 \leq 0$ $2.00 + 1.13l_1 - 0.511l_2 - l_3 \leq 0$ $3.49 - 0.158l_1 + 0.791l_2 - l_3 \leq 0$ $-0.903 - 0.668l_1 + 1.99l_2 - l_3 \leq 0$
subreg 2	$2.67 + 0.586l_1 - 0.188l_2 + 0.195l_3$	$0 \leq l_1, l_2, l_3\ leq 10$ $3.19 - 0.141l_1 + 0.545l_2 - l_3 \leq 0$ $2.00 + 1.13l_1 - 0.511l_2 - l_3 \leq 0$ $3.49 - 0.158l_1 + 0.791l_2 - l_3 \leq 0$ $-0.903 - 0.668l_1 + 1.99l_2 - l_3 > 0$

The algorithm used for constructing the convex hull is the "gift-wrapping" algorithm by Chand and Kapur [1]. A detailed analysis of the algorithm has been conducted by Swart [9].

3.1 Macromodel Types

Two types of macromodels are obtained from the results of the process described above. The *mean-fit* macromodel is obtained by performing a least-squares fit on the points contained within each sub-region so as to obtain a single linear function therein. In the *conservative-design* macromodel, two linear equations are obtained for each sub-region. The 'below' equation is defined so that all of the points in each sub-region lie above the surface represented by the equation. The 'above' equation is defined so that all of the points in each sub-region lie below the surface represented by the equation. The conservative-design macromodel is obtained by finding the single 'above' facet and single 'below' facet of the convex hull in each sub-region that has the smallest worst case 'vertical' distance from the points within the sub-region. An example is given in Figure 7 based on the response surface, shown earlier in Figure 6. One more split was done on the regions shown in Figure 6 before the macromodel was generated.

Each type of macromodel has its own particular uses. The mean-fit macromodel is best suited for use at the high level design phase, for example in a floorplanning tool such as PEPPER [7]. In a floorplanner, the macromodels are used to determine the maximum practical spacing between chips, to decide on driver type, number of loads, etc. As only rough design is being done now, a single least-square fit macromodel is adequate and there is no need to use the conservative-fit macromodel.

The conservative-design macromodel is used is used to generate wiring rules. Wiring rules have to be conservatively specified. It is better to over-predict noise and delay (if the design requirement is expressed as a maximum delay) rather than under-predict. The mean-fit macromodel may over-predict or it may under-predict. The conservative-design macromodel will always over-predict, and thus leads to a 'safe' design.

As an example, the 'above' half of the conservative-design macromodel was generated using the MCM response for settling delay illustrated above. Eleven disjoint sub-regions were formed as a result of this process, the equations for two of which are given in Table 1. The maximum and average vertical distances between the macromodel and the of set of randomly sampled points in each region are reported in Table 2. In Table 2, the line titled 'Initial' provides the difference statistics obtained when the 'above' macromodel is generated without any region sub-division. It can be seen that with the division into sub-regions, the conservative-design macromodel provides a much closer fit. The manner in which these macromodels are used to obtain wiring rules is discussed next.

4 Obtaining Wiring Rules from the Macromodels

We illustrate the process first, through a simple example, before giving the algorithm used and showing a full example.

Consider the single variable design space and the response illustrated earlier in Figure 6 and its corresponding conservative-design macromodel shown in Figure 7. If the timing design indicates that delay must be less than 2 ns then the 'above' macromodel can be used to find the ranges of length for which this is guaranteed. If the length range for which overshoot is less than 1 V is obtained from the overshoot macromodel, then the resulting wiring rule is the overlap of these two lengths as shown in the bottom of Figure 8.

Table 2: The sub-regions generated for the MCM case with the differences between sampled points and the macromodels.

Region	Number of Points	Maximum Difference (ns)	Average Difference (ns)
Initial	1000	2.276	1.11
subreg 1	176	1.261	0.658
subreg 2	25	1.710	0.621
subreg 3	38	0.990	0.510
subreg 4	69	0.972	0.942
subreg 5	150	1.351	0.710
subreg 6	32	1.451	0.656
subreg 7	38	0.499	0.282
subreg 8	114	1.316	0.475
subreg 9	74	1.253	0.594
subreg 10	68	1.429	0.701
subreg 11	216	1.452	0.810

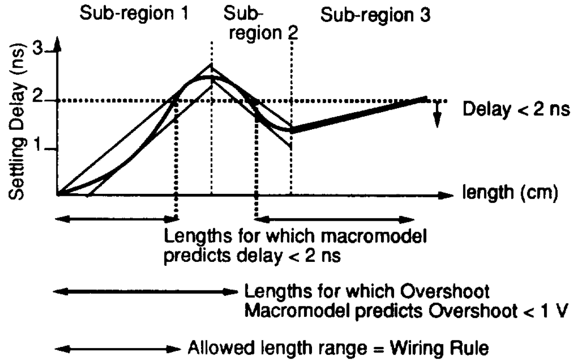


Figure 8: Simple example showing a conservative-design macromodel and how a wiring rule is obtained from it.

Written more formally, a wiring rule is obtained for a data signal net as follows:

Step 1. The maximum settling delay requirement t_{D_s-max} is obtained from the timing design. This is compared with the 'above' settling delay macromodel to determine the union of regions A in the physical design space R^{D-1} for which t_{D_s-max} is greater than the delay obtained from the macromodel. (Note $t_{high-low}$ and $t_{low-high}$ transition results are combined into a single macromodel.)

Step 2. The minimum settling delay requirement t_{D_s-min} is obtained from the timing design and compared with the macromodel, in the same fashion as above, to obtain region B.

Step 3. The maximum positive overshoot is obtained from the noise requirements and compared with the positive overshoot macromodel to obtain region C.

Step 4. Region D is obtained by considering the maximum negative overshoot.

Step 5. The wiring rule is the polytope determined by the intersection of volumes A, B, C, and D.

A wiring rule is obtained for a clock net by similarly applying this process using the requirements for, and 'above' macromodels for, 50% delay, positive and negative overshoot, positive and negative undershoot, rise and fall times, and the maximum width of any 'porching'. Note that all but the 50% delay requirement are requirements for first incident switching.

This process was applied to the two-receiver example illustrated above, using the MCM-D response surface (but only the 'initial' macromodel given in Table 1 to make the results easier to visualize), and the following timing and electrical requirements:

- Stable delay $\leq$ 5 ns.
- Peak overshoot (positive and negative) $\leq$ 2 V.

The resulting wiring rule is the three-dimensional volume illustrated in Figure 9.

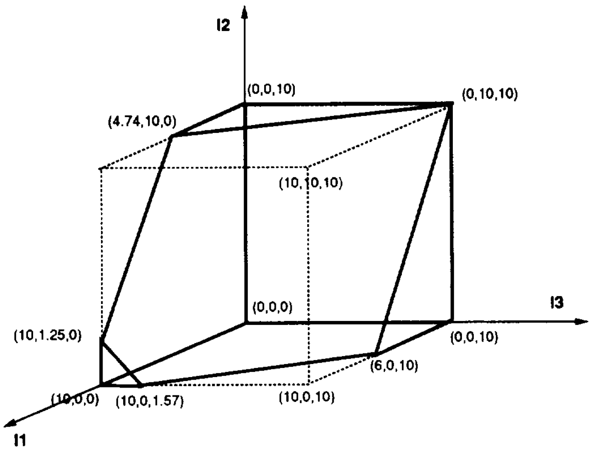


Figure 9: The acceptable design space, or wiring rule, is the volume enclosed by the bold-drawn edges.

5 Conclusions

We have established an automated process in which signal delay and reflection noise are managed without resorting to overly conservative and ultimately costly design practices. Central to this process is the production of macromodels that accurately capture the electrical responses of interconnect circuits over a wide range of physical design variables (including lengths, number of vias, layer assignment, number of loads, etc). The macromodels are obtained by fitting piecewise-linear equations to a set of simulation results. The macromodel must be obtained from simulation results because the available analytical expressions for circuit electrical responses are inaccurate. We obtain two types of macromodel, a piece-wise least-square fitted linear model and a piece-wise linear upper and lower bound model. (Only the second type is discussed in detail above.) The advantage of the second model type is that it is guaranteed to be conservative resulting in a 'safe' design without overly restricting that design.

Our solution greatly reduces the burden on the signal integrity and package design engineers. The automatic production of macromodels releases the signal integrity engineer from the need to conduct a large number of simulation runs. The automatic application of the macromodels to the generation of a package layout from the timing design releases the signal integrity engineer from the need to become involved in the detailed layout of a large number of designs. This automation also helps guarantee first pass success for high speed digital designs.

Acknowledgements

The authors would like to thank Zaki Rakib and C. Kumar of Cadence, George Katopis and Jay Diepenbrock of IBM, and John Grebenkemper of Tandem for discussions during the course of this work. We would also like to thank Garrett Simpson of Cadence for providing the small computer design and the preliminary processing of it.

References

[1] Donald Chand and Sham Kapur. An Algorithm for Convex Polytopes. *Journal of the ACM*, 17(1):78–86, January 1970.

[2] M.J. Chen and E.S. Kuh. Solving nonlinear resistive networks using piecewise-linear analysis and simplicial subdivision. *IEEE Transactions on Circuits and Systems*, CAS-24(6):305–317, June 1977.

[3] E.E. Davidson and G.A. Katopis. Package electrical design. In R.R. Tummala and E.J. Rymaszewski, editors, *Microelectronics Packaging Handbook*, chapter 3. Van Nostrand Reinhold, 1989.

[4] P.D. Franzon. Electrical design. In D.A. Doane and P.D. Franzon (ed), editors, *Multichip Module Technologies and Alternatives: The Basics*, chapter 11. Van Nostrand Reinhold (New York), 1992.

[5] P.D. Franzon, S. Simovich, S. Mehrotra, and M. Steer. Automatic a-priori generation of delay and noise macromodels and wiring rules for MCMs. In *Proc. 1993 IEEE Multi-Chip Module Conference*, 1993.

[6] G. Katopis. Wiring rules for electrical package design, 1992.

[7] D.P. LaPotin. Early analysis of multichip modules. In *Proc. 1990 International Electronics Packaging Symposium*, pages pp.557–563, September 1990.

[8] S. Mehrotra, S. Simovich, P.D. Franzon, and M.B. Steer. Automatic Circuit Characterization through Computer Experiments. *submitted for publication*, 1993.

[9] Garret Swart. Finding the Convex Hull Facet by Facet. *Journal of Algorithms*, (6):17–48, 1985.