



USING INTELLIGENT AUTOMATION IN ADVANCED PCB DESIGN

Eliminate Manual Processes While Preserving
the Designer's Intent



GREGORY M. HORLICK – CADENCE DESIGN SYSTEMS, INC.

DR. KEN WADLAND – CADENCE DESIGN SYSTEMS, INC.

INTRODUCTION

With the emergence of each new interconnect design challenge—from surface-mount technology to differential pairs to complex high-speed constraints—EDA technologies have generally kept pace by adding features with new levels of automation. More recently, however, this has not been the case. Advances in industry interfaces, larger and denser component packages, and complex constraints have leapfrogged the capabilities of conventional routing automation. Currently available design and autorouting technologies are also unable to capture designers' intent when designing today's state-of-the-art, highly constrained, and highly dense PCBs. The design and actual routing of these designs is, in fact, so plagued with manual and often archaic processes that designers must spend a significant amount of time hand-routing their designs.

Overcoming long design times and delays in getting products to market requires more than a continued evolution of existing interconnect design technologies. Designers need an entirely new paradigm—one that will deliver a significant leap in designer productivity by applying intelligent global automation that includes the capability to preserve and adhere to designers' intent.

A GLOBAL APPROACH TO THE COMPLEXITY PROBLEM

Looking at the problems PCB designers face in designing today's highly complex system interconnect reveals several common threads that could form the basis of a new usage model. However, in order to maximize the benefits to the user, this new model must provide benefits throughout the design process. It must also provide benefits across all design disciplines (e.g., not just for PCB design).

Therefore, developing a holistic solution requires looking at each discipline to determine not only what are the high-level processes but also what processes are common across disciplines. Doing this, we find these common functions include placement, routing, verification, and fixing errors reported by verification.

Each of these high-level processes is a highly interactive and labor-intensive step with its own requirements and goals. Moreover, these processes pit the user with their knowledge/experience and manual capabilities against the spatial/connectivity problem; and each individual process can potentially require the user to make thousands of manual edits in the effort to complete a single iteration.

From a high-level view, a flowchart of the entire process would something like *Figure 1*. From this perspective, users make their best attempt to get the data defined correctly and then perform some sanity checks (DRC cycles, or other reviews). They then take this information and feed it back through a refinement phase and loop until satisfied with the result.

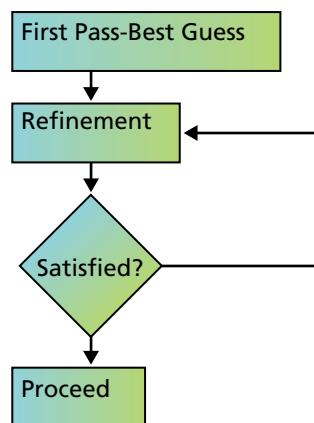


Figure 1: Basic user iteration

ISOLATING AND SIMPLIFYING THE PROBLEM

Each pass through this loop requires the user to visualize in their mind what all of this data means, and resolve it down to a solution(s) that will work. The problem is that the number of viable solutions is potentially very large and the time required to manually obtain any single solution—either good or bad—grows with each iteration.

Without intelligence to understand what is in their mind, and a route engine that understands these desires, users are left to do this manually. This leaves users in the position of “hoping” that their best guess is going to be right.

This brings us to the fundamental problem, data overload of the user, which is compounded by tools that don't provide a way to conceptualize the problem, visualize it—let alone resolve it—or quickly assess whether a potential solution is good or bad.

With the problem defined (data overload) we can start to look for places within the design process in relation to the data as a starting point for simplification of the design problem. This can then be combined with data presentation, manipulation and design intention support to reduce the overall design cycle time for today's design teams.

INTERACTION BETWEEN DATA AND THE USER

No matter what design discipline we are working in (PCB, SIP, and possibly custom IC), there are some common aspects to the data presented to the user. This data may include geometric data, a connection list, and constraints to govern those connections.

The interplay between all of this data and how the users sees it greatly influences how they interact with it and, ultimately, the selection of the first step in each process—the “Best Guess”. It is here that making careful logical changes to the data can have a great impact on the user.

Looking at the data and how the user works with it has some benefit so it helps to understand the interaction between data and the user before defining the presentation and how to modify the use model. Depending on the state of the design at any given time, the amount and type of data presented to the user will vary. However if we look at where the user is in the design cycle in relation to the types of data required we can start to define what they need.

Initially while placing components, they will probably only need the geometric data and connectivity. From the perspective of the EDA design software, geometric data is anything that ultimately will end up as metal in the final design. This may include pins, vias, plane shapes, and interconnect lines. In these early stages constraints may or may not exist yet. Users are trying to solve the geometry problem—i.e., how to fit all of the components into the allowed space while maintaining a logical connectivity pattern.

However, the amount of data is again a problem. The connectivity may have thousands of elements obscuring the view that is demonstrated in the *Figure 2* on the following page.

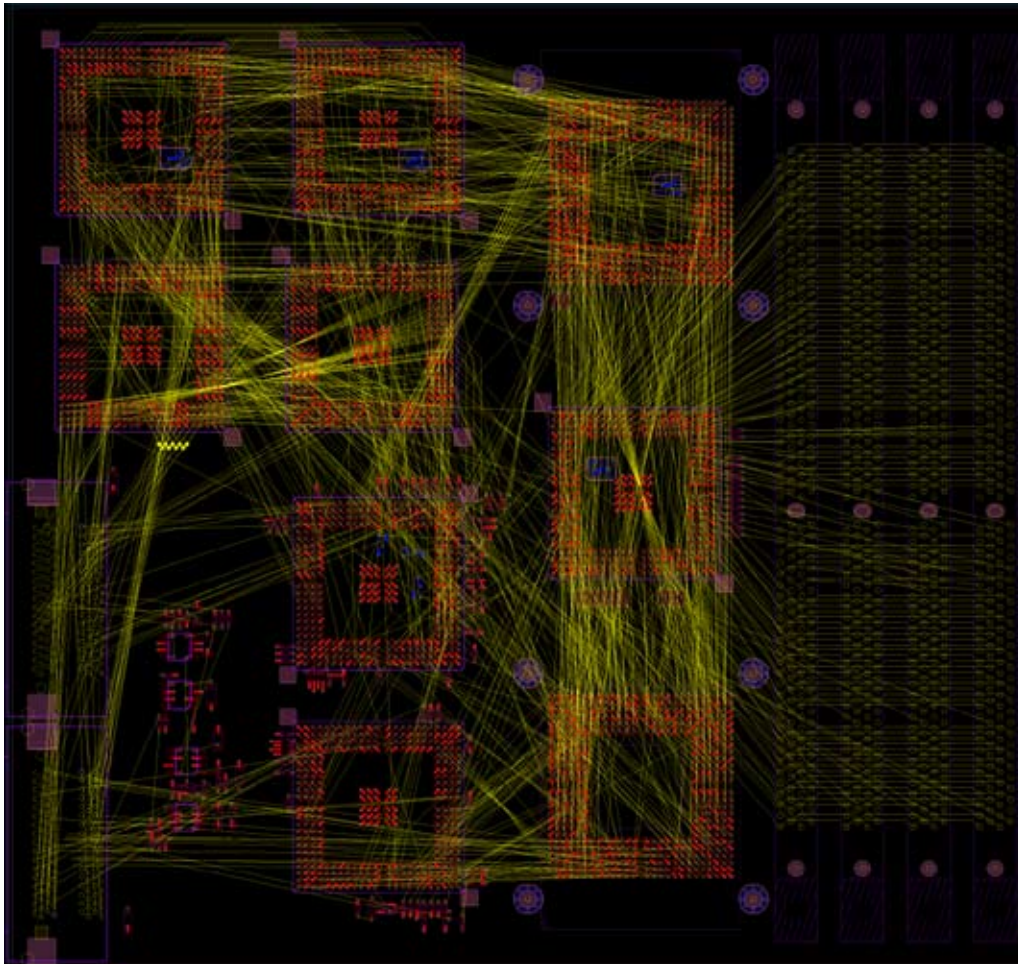


Figure 2: Example of placed design connectivity

This is where current solutions fall far short of solving the problems. Users want to see the connectivity of different bus structures, so they resort to color coding methods in an effort to clear the screen of data while still providing the means to identify where objects are connected.

Clearly, this is a place where abstraction of the data can be leveraged to reduce the design problem. Abstraction would provide a “clustering” effect to the connectivity display—providing what they need to see while not overloading the display with data.

THE ROUTING PROCESS

Generally, after placement is complete routing can begin. This does not mean that some exploratory routes haven’t been performed to do spatial analysis; it means the route process is ready to proceed toward design completion.

By this stage the constraints have typically been more firmly defined and can now be implemented in interconnect. It is at this stage that the user must hope that the “vision” of the design they had in their head—that “Best Guess”—was correct. They won’t know this until the interconnect pattern starts to develop and they realize that they had enough space to implement it or not. Bad guesses at this point can prove disastrous. If you look at the interconnect pattern in *Figure 2* and imagine routing much of this only to find out that you have to move a component in order to fit the interconnect in, it is obvious that an early understanding of problem can be a huge time saver.

Clearly, this is a place where abstraction of the data could be of great value.

DATA ABSTRACTION

We have alluded to data abstraction several times. It is now time to define exactly what this means to the system and the user. This is the first place where abstraction will greatly simplify the problem. If we look at *Figure 2* we can clearly see that there are large groups of things (ratlines) that connect to common places or components. These clearly could be grouped together and represented by a single larger entity—a bundle.

In order to understand the power this provides to the user we need to look at an example—a closer look at a bundle. The following diagram illustrates a simple branched topology similar to what might be found in memory sub-systems. It has a driver to the left, a T-Point in the middle, and the load pins to the right.

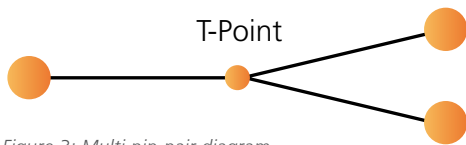


Figure 3: Multi pin-pair diagram

Although fairly simple, when we complicate the problem by replicating it across many address and data lines it will start to look more like *Figure 4*. If we extrapolate further, the problem will quickly become much worse.

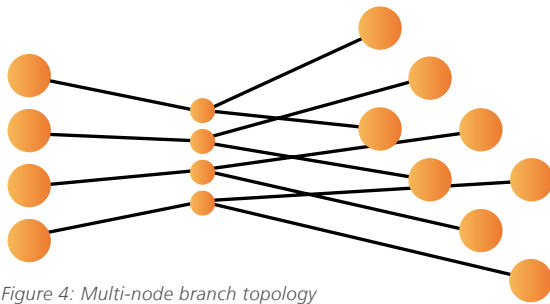


Figure 4: Multi-node branch topology

However, if we allow the user to represent the connectivity abstractly the problem becomes simplified. Our multi-node group in *Figure 5* has taken on the characteristics of the original example shown in *Figure 3*.



Figure 5: Multi-node topology — bundled

So if we apply the bundle approach to our original design, we will end up with a much cleaner view of the design connectivity but without all the screen clutter (*Figure 6*).

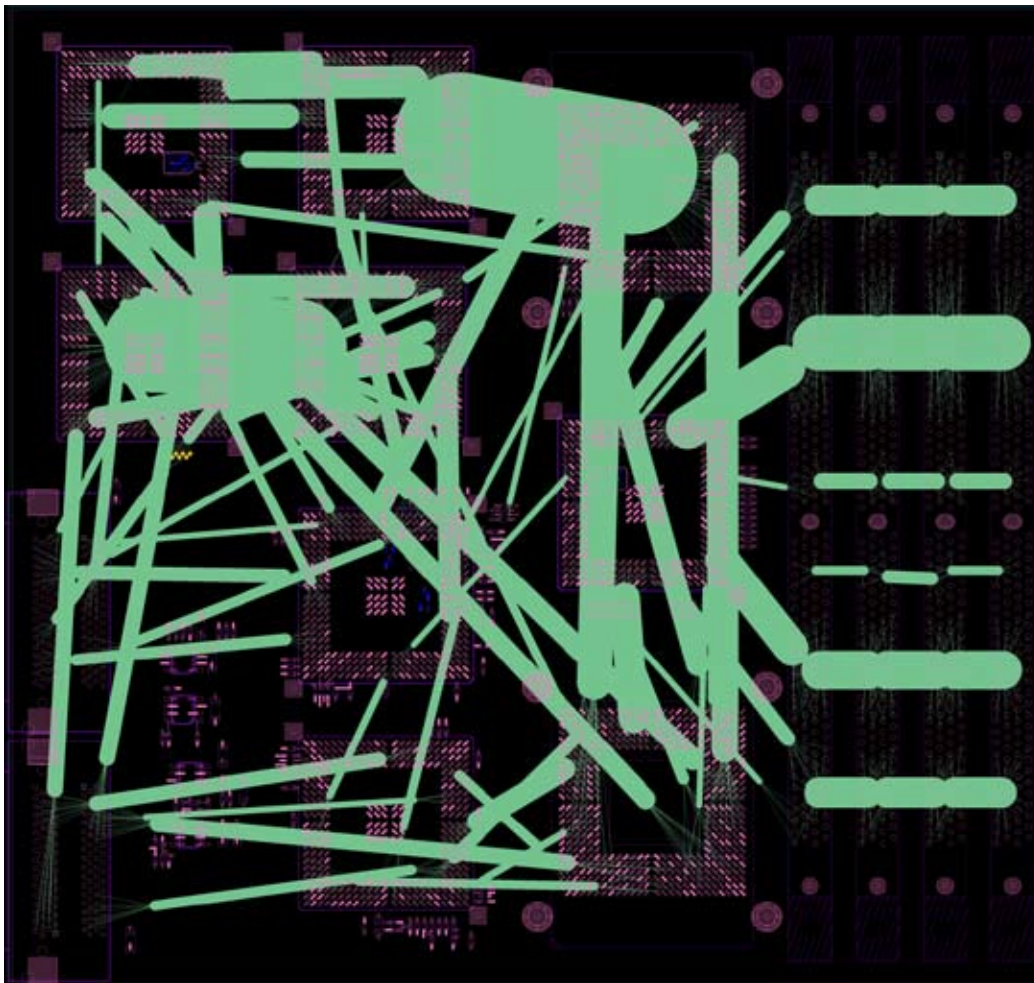


Figure 6: Bundled design view

ABSTRACTION OF THE INTERCONNECT PATTERN

If the process works for the basic connectivity map, it should also work at the interconnect level, or we have left the user with only a partial solution. Therefore, we need to look at the interconnect itself—where the problems are a little larger. We must represent the interconnect pattern in 3-D space in which the connectivity map technically doesn't exist.

This leads to more new objects that the user has to deal with, but if we keep the objects in “familiar” terms and in a connectivity/use model that the user already understands we can reduce the learning curve to a minimum.

In the context of the user's interconnect model, during the placement phase the user has tried to pre-allocate the space necessary to implement the interconnect pattern. In short, they have mentally developed a “Flow” plan—a concept of how they want the interconnect pattern to route. In order to keep the user in a common frame of reference, objects used in the graphical interconnect flow planning architecture should have similar names, concepts, and attributes that they are already accustomed to when working within the current Cadence® Allegro® PCB environment.

This means that we should adopt the same basic terminology and choose the name “flow line” as the graphic representing the interconnect itself. And just like standard design terminology (vias, pins, etc.), there should be representative flow objects that the user can manipulate. For example, in current EDA tools, the co-ordinate set that any particular route or group of routes takes is considered its path. It is represented by a graphic line that the user can see and manipulate.

In the bigger (more abstract) world of the graphical interconnect flow planning architecture, the bundle represents a group of things yet it still has a similar path that it will take from source to target. This path or flow is represented by a line, the flow line, which the user wants the router to follow.

Again this means that we have succeeded with two of the primary goals of the graphical interconnect flow planning architecture project—data abstraction with visual acuity and spatial representation of the required space.

DEFINING A FLOW

During the placement phase, users try to pre-allocate space for developing the interconnect pattern. Historically, they have done this by using conventional routers, route voids, fences, and a host of manual techniques. Unfortunately this process does not reveal the success or failure of this “Guesstimate Technique” until it is too late.

Now using the abstract data we can accelerate this process by providing a visual/spatial map of the open area in relation to the larger bundle structures. But to do this, we must turn bundles into flow lines, vias, and so on, which requires a set of tools/commands that lets users define an x, y and z path for each of the bundles in their design.

In order to keep things in user terms we have considered the use of standard routing models, ones in which the user selects the bundle to define the flow and then defines a path subject to standard routing angles, rules, etc. This allows designers to use a familiar interface—just at a more abstract level. Plus, it allows them to quickly generate the paths, vias and transition points as they deem necessary.

PROPERTIES OF FLOW PLAN OBJECTS

As part of the overall flow plan, the objects can have properties attached to them to help expedite the route process, change the way the route engine will perceive the data, or make any other further intent definitions that we choose to put on them. This property set can grow over time allowing further route pattern control and refinement controls that the user may need to address to converge on an interconnect solution.

Of course, all of this will require an intelligent route engine in order to converge on the final interconnect pattern.

FINAL ROUTING OF THE INTERCONNECT PATTERN

In keeping with the overall goal of simplification of the data, the bundle/flow methodology has greatly simplified the user’s data and significantly reduced the overall screen clutter. However, this has not addressed the final step of getting the design completely routed. This will require an entirely new route engine that understands what the user has in mind—their design intent.

We had previously stated that the placement and flow plan are significant pieces of the overall design process and therefore are also significant pieces of the designer's intent. However there are other things that can be considered part of design intent. These may include return path management, layer reduction exploration, and willingness to explore "what-if" alternatives that can all be accommodated with this new Bundle/Flow/Route Methodology more easily than ever before.

Once the basic data abstraction has been done, further design intent functionality can be applied to Flow plan objects as needed to solve specific design needs.

KEY TO THE NEW PARADIGM—A HIERARCHICALLY AWARE GLOBAL ROUTING ENGINE

As stated previously, route engines have fallen behind the technology curve in terms of their ability to follow the user's needs and desires. Compound this with the density issues of today's components, new signaling levels, and topology requirements and we can clearly see the need for a new route engine.

Clearly a new route engine must understand the requirements of today's designs. This includes not only the topology and timing relationships but also the user's design intent. This would include the guidance of the flow plan objects the user has provided, which would represent a big departure from traditional methods of route restrictions, route a section and manual cleanup.

Finally, after all of this time, the user can capture their intentions in the flow plan, route the design and let the route engine deal with the details of the routing without having to solve the spatial problem at the same time.

We also want the route engine to do this without causing the user to have to deal with "real" interconnect in case there are problems. So what we propose is to use a method where the route engine knows where all the routes need to be—or can go—but are not truly implemented in metal until feedback from the route engine indicates that the "design is routable."

All of this data abstraction and flow planning has led us to a potential solution. What we need to do is now run the route engine and let it validate our proposed solution. This means the route engine will take all of our input (bundles/flow), and then route them as close as possible to our desires. It will also have to route in any remaining routes—loose logic—or things that can't be flowed easily and display the results to the user. Remember that loose logic takes up space in the design as well and may impact our flow plan negatively.

So the route engine takes all of this into account and determines that the design has congestion issues or not. This could be displayed in with visual markers in the design or any other desirable error reporting method that makes sense, e.g. a route status dialogue. The point is users get feedback and have a "comfort level" provided by the route engine that their design is routable and their intent is validated. Once they are sure the design is routable, they can then instruct the route engine to implement the plan as real interconnect for storage in the Cadence® Allegro® PCB database.

SUMMARY

The main benefit and the prime target of this Global Route Environment technology is the reduction in cycle time through increased efficiency and productivity. This is accomplished by allowing the user to create abstract interconnect data and quickly converge on a solution and validate it with the route engine.

Through the use of interconnect abstraction there is a reduced number of elements the system has to deal with. Reducing the number of elements from potentially tens of thousands of elements down to hundreds results in a significant reduction in the amount of manual interaction required. Also, the number of visual elements the user sees in the interconnect flow planner drops by the same orders of magnitude in the number of elements they must physically manage. This represents a significant simplification over current design tools allowing users to get their designs completed faster and more efficiently.

Through the interconnect flow planner and the global route engine, users can for the first time put their experience and design intent into a tool that understands what they want—natively. Users can now converge on a successful interconnect solution far faster and more easily than ever before. They can strategically plan and automatically route complex PCB interconnects that until now could not be routed automatically.

These features and technologies finally give designers what they have needed for a long time: software solutions that allow them to work smarter, not just harder and a toolset that works with them and does not make them work around its deficiencies.



For more information,
log on to www.cadence.com

cadence™

Cadence Design Systems, Inc.

CORPORATE HEADQUARTERS
2655 Seely Avenue
San Jose, CA 95134
P: +1.800.746.6223 (*within US*)
+1.408.943.1234 (*outside US*)
F: +1.408.943.5001
www.cadence.com