A Practical and Reliable Methodology for Hierarchical Signal Planning

Paper presented at DesignCon 2012
Santa Clara, CA
Introduction

Advanced custom IC designs at leading-edge process nodes (45/40nm and below) are rapidly outgrowing traditional manual IC routing techniques that have been the norm for over 25 years. Large custom IC’s are often produced by multiple design and layout teams that use schematics, hand routing and semi-automated scripts to generate custom block level designs. In parallel, or even after block implementation, a top level design team has to work across multiple levels of hierarchy to integrate the custom blocks and to close the top level floor plan and routing using schematics, in conjunction with manual routing and semi-automated scripts.

Implementation and optimization of signals and buses is primarily an iterative process that involves floor plan, routing and congestion trade-offs between the top level, lower level blocks and sub-blocks. The lack of comprehensive custom design tools which support a common design database that provides visibility up and down the design hierarchy makes it even more difficult to optimize routing in custom designs. As we move to more advanced process nodes with more complex design rule requirements, resolving timing and parasitic issues while meeting die size targets and managing congestion has become even more difficult and costly to achieve.

The EDA industry has provided powerful suites of digital design tools that have been optimized for use in the digital design ecosystem, which includes standardized digital libraries and models plus deep metal stacks providing ample routing resources. However, digital design tools often lack the features and flexibility needed to efficiently support routing of advanced custom IC designs with their complex, specialized topologies, limited routing resources, non-standardized libraries, large numbers of analog IP's and DFM rules (such as via redundancy, via orientation, via enclosures, dense line end rules, etc.). Historically there has been limited automation for custom design tasks due to a combination of relatively small market size, large variance in design automation needs, non-standardized IP’s and cell libraries, plus a slow market acceptance rate for design automation software.

This paper gives an overview of a proven methodology that when coupled with a set of flexible planning tools, such as those found in the Pulsic Planning Solution, which is designed for hierarchical custom IC’s, provides a reliable and comprehensive solution to common hierarchical signal planning issues. In order to provide a baseline starting point, a review of the current custom design approach along with associated limitations will be provided. Next, the following key topics, presented in the order of a typical design flow, will be covered.

- Floor planning
- Power planning
- Bus, long net and datapath planning
- Signal planning
- Signal and bus routing
- ECO Considerations
The Current Approach

Current custom and AMS design efforts often begin by leveraging design information from earlier design projects to help provide a baseline for both top down and bottom up floor planning signal/bus planning and routing. Designs are often implemented with traditional point tools, schematics and spice level netlists plus a combination of manual layout tools and hand routing, along with custom scripts to improve design throughput.

For example, from a top down perspective block sizes and shapes can be estimated to help drive a ‘first cut’ bottom up block implementation. Block level teams would then use largely manual floor planning, power planning, and placement techniques to complete a first pass of their block level designs. If the block cannot be routed in the initial footprint or if the block has extra space, the block’s footprint would normally be reworked. Blocks, block locations and routing channels often need to be adjusted as nearby blocks change in size and shape. The top level team typically performs manual top level routing while assembling the top level design to help drive block level pin placement optimization. Various blocks often need to be reworked as pin placements are shifted to help resolve congestion problems. Many design iterations occur at both the block level and the top level to resolve various floor plan, routing, and congestion issues as the project converges towards final implementation. In parallel with the above bottom up and top down design processes, design feature changes and verification related fixes drive additional schematic updates that also impact the design implementation effort.

Current Approach Limitations

Market requirements are driving custom and AMS IC providers to develop IC’s with ever increasing complexity and performance. Current high end custom and AMS IC’s are outgrowing the capabilities of traditional custom and AMS design techniques. This is especially true in the areas of: hierarchical floor planning, signal/bus planning and routing, design change incorporation, and project coordination and communications.

Hierarchical floor planning and signal routing often leverages relative block size, block position, and pin location information from previous design efforts. This approach has several inherent disadvantages. Designs that are similar in functionality to existing designs may start with a reasonable baseline, but for new designs, that are significantly different in terms of functionality, architecture or process node, the use of previous design information is far more problematic. In any event layout designers will have two main problems to resolve: routing congestion within blocks, and routing congestion between blocks. These are often concurrent problems, where block level optimizations impact the top level and top level optimizations impact the block level. Furthermore, design schematics are changed as new functionality is added or design flaws are corrected. These changes may impact the top level routing topologies, signal/bus ordering and top level channel sizes etc. Additionally the design hierarchy may need to change as lower
level blocks are added, shifted, flattened etc. This in turn changes the block level pin interfaces, and creates a ripple effect of block resizing, pin location changes and cross-chip routing changes. In some cases design updates introduce new congestion problems into a mature design database that can only be resolved through major rework at the block and top levels. The above scenario is compounded when one complete pass of both the block level and top level placement and routing loops often takes several weeks to complete.

IC verification is usually performed throughout the design process, and this can drive schematic updates, routing changes and associated rework until just before tape out. Furthermore, schedules and/or market driven requirements may force design managers to add, remove or change design features during the design effort. In most cases the resulting effect is to have a moving design target throughout the design cycle. This is especially troublesome as the design matures because the costs of design rework increase.

As custom design complexity increases, so does the design team size and the tendency for team members to specialize, and this impacts project coordination and communication. Large custom chips typically have deep design hierarchies. Design managers often assign the top level and block level floor planning and routing tasks to different layout resources that may be located at different sites. The typical custom design communication and coordination method is to address pin planning and routing issues by using the schematics and trial layout information to help refine relative pin positions and perform signal planning. Information is often exchanged during project meetings or by email. Communication issues are more acute for IC development in newer process technologies where design rules are still being stabilized and library cells are still being updated. Design rule and library changes can also impact routing and introduce design rule (DRC) violations that need to be fixed before the design can be completed.
A New Custom Design Methodology

A new design methodology that addresses the above design flow limitations by providing the following capabilities will now be presented. This methodology:

- automates the design processes that have been undertaken manually or with traditional planning and routing tools
- allows design and layout teams to view the entire design hierarchy and understand the implications of potential design changes throughout the design
- provides repeatable and predictable design automation that can accommodate design changes at every stage of the design process.
- facilitates rapid design prototyping to produce several hierarchical floor plans that can be used to help optimize die size, routability, reliability and manufacturability.

Floor Planning

The main starting point for optimizing hierarchical signal planning and routing involves hierarchical floor plan optimization. Floor planning for custom designs requires a great deal of flexibility, so support for both top down and bottom up approaches is needed to help optimize routing paths, soft macro (block) floor plans and soft macro (block) pin placements. From a top down perspective top level pins that have fixed positions, such as I/O related signal pins, should be visible to the rest of the design in order to help optimize associated routing and soft macro pin positions. From a bottom up perspective, designs often contain hard macros or IP's with fixed pin positions that should be placed in specific optimal locations. These ‘bottom up’ hard macro locations and pin positions should also be used to help optimize associated routing and soft pin positions.

Floor planning is an iterative process, involving many possibilities and permutations, and using an automated tool that supports rapid hierarchical floor plan generation for custom designs will hasten the design process. Early floor planning steps usually involve trying to fix the positions of the parts of the design that are well defined, since these will impact more flexible parts of the design. The next steps involve sizing soft macros and performing trial and error block and pin placement to obtain a decent first cut floor plan.
Figure 1) Top level of a design showing 8 sets of hard macros that are placed along the left and right sides. The center area has 8 smaller macros plus large routing channels.

Regarding the design example in Figure 1, the soft macro sizes and the channel sizes will probably need to be refined in order to improve the floor plan. As the floor plan matures, the soft macro floor plans are often optimized, their sub-blocks are updated, and their pin positions further optimized and legalized. Key floor plan factors that impact signal planning include: channel width, block count, block size, signal count, signal ordering, and signal/bus topologies such as: “L”, “C”, “J”, or “H” shaped patterns, and so forth.
An automated design tool that can span the design hierarchy, automate many of the key floor planning steps and automate the routing of the different bus/signal topologies will improve the pin placements and signal/bus planning as well as the overall floor plan quality.

Figure 2) Block level view of the highlighted soft macro near the top center of Figure 1. An automatic block placer has placed the macros. The two pink macros (light gray in B&W) are hard macros. The nine green macros (dark gray in B&W) are soft macros. The signal fly lines are largely randomized, since this is before automatic pin placement.
Figure 3) This figure is the same soft macro shown in Figure 2, but after running automatic soft macro pin placement. The yellow fly lines show the updated pin positions that were derived by the automatic pin placer. Inputs to help guide the automatic pin placer can also be used to further optimize and finalize the soft macro pin placements.

**Power Planning**

After determining the basic floor plan and block sizes, the initial power plan should be added to the design. Aside from hard macros that contain internal power and ground routing, the power plan is normally added in a top down manner. For custom designs where the power is typically on a different layer than the signal routing, it could be added before or after the block level pins are placed. However, the power and ground mesh should always be added before signal and bus routing, otherwise resources needed for power routing may be occupied (blocked) by the signal routing. For custom designs with extremely limited metal layers where the power routing is on the same layer as the signal routing, the power structure/mesh needs to be added before defining the block level pin locations, otherwise the structure/mesh pin positions may be blocked by signal pins.

Power planning parameters can be easily defined using a GUI-based power planning tool that supports working at a high level of abstraction by creating power guides for all
power domains. These power guides can include attributes such as (absolute/relative) location and shape, power supply names, routing layers, routing widths and so forth. Aside from power guidance, via topologies can also be specified. Once defined, the power nets can then be pushed down into the soft macros for use by their design teams. When the design is reassembled, the soft macros are brought back into the top level and connected to corresponding top level supply routes.

![Diagram of power planning guides](image)

*Figure 4) Power planning guides for the top center area of the design.* Notice the power and ground guides are placed in the open routing channels between the macros. In contrast mesh guides are placed over macros, such as the two soft macros in the center area. The power and ground guides in the GUI show the recommended routing topology, plus it has attributes for routing width and layers, which can be displayed before and after completion of the power and ground routing.
Figure 5) The same view as Figure 4, but the power and ground mesh routing based on the mesh guides from Figure 4 have been added. The horizontal guides that were placed over the two center blocks provide mesh inputs to generate one or more parallel straps. In this case 6 thin power and ground routes have been generated. The guides used in the routing channels, are for single power or ground routes. These guides can be connected together into more complex intersecting topologies.

**Bus, Long Net and Datapath Planning**

The next major step after soft macro pin placements have been completed involves adding top level buses. This is best achieved using a Bus Planner tool (e.g. the Pulsic Unity Bus Planner) that will allow users to quickly input top level GUI based bus guides, which provide information used to define and optimize the top level routing paths of corresponding buses. All signal routing for guided buses follows the same topology, and uses the same layer and vias, so by design the bus routes will have similar timing. Designers should focus on creating bus guides for their most critical buses before moving on to less critical buses. Those buses that don't require routing guides could be routed as ordinary signals.
Inputs for bus guidance include: physical topology, signal ordering, optional shielding, optional signal interleaving with other buses, metal layer selection, via selection, and via topology selection for forward or backward via sets. Various generic controls can be provided by using an attribute editor which can define buses by combining signals together, set signal routing widths and set signal route spacing, select X, Y routing layers, and so forth. Also note, bus bits can be twisted, that is reordered, to arrange adjacent signals in such a way so as to help reduce cross talk effects.

Once the bus guides are in place the buses can be routed. Afterwards, the topology based repeater planner (a feature of the bus planner tool) can be used to specify locations for bus repeater insertion, which is often used to speed up signal propagation and minimize cross talk effects. The next step is to automatically place and connect the repeaters to the existing bus routing. Overall, propagation delays for signals on the same bus will be very close since their routing and buffering will be similar.

Figure 6A) Two example bus routing guides that are between two neighboring macros. Notice the vertical bus guides are overlapping, this causes these routed buses to be interleaved. If the vertical guides were separate the bus routing would remain separate rather than be interleaved.
Figure 6B) Bus routing based on the two bus routing guides in 6A. The routing of the two vertical buses is interleaved. These two buses have different bit widths, and settings can be used to specify the bits to be interleaved. The via orientation can be specified as forward or backward, and is usually chosen to minimize signal crossing.
Figure 7) A close up view of the lower vias and routed buses shown in Figure 6B. The placement of the upper left set of vias slopes downwards from left to right (that is it slopes backwards). In contrast, the placement of the lower right set of vias slopes upwards from left to right (or it slopes forwards). Bus via patterns that slope backwards and forwards is user definable, which can be used to minimize signal crossing.
Figure 8A) An example of a bus repeater guide added to a routed bus. The repeater guide also specifies the buffer cell macro.
Figure 8B) An example of the automatically placed and routed bus repeaters that have been inserted into the bus routes, based on the repeater guides in figure 8A. The net names of the bus signals have been automatically updated to reflect the repeater insertion.

**Signal Planning**

Signal planning can be the most time consuming process and is the crux of meeting design schedules and area requirements for hierarchical custom design. This process would benefit from the use of a comprehensive signal planning tool such as the Pulsic Unity Signal Planner that supports two modes: strictly biased routing and multiple-bias routing that employs a jumper layer. Strictly biased mode limits signal routing to user specified X and Y routing layers without exceptions, and is a capability provided by many routing tools. The multiple-bias routing mode, is an advanced capability that supports signal routing on the same metal layer for both X and Y directions. Meanwhile a
jumper layer can be defined and used to cross other signals that are blocked by potential X,Y multi-biased routing. For example, signals could use multi-biased metal 3 routing for both X and Y directions, while using metal 2 as a jumper layer. The metal 2 jumper layer is employed when multi-biased metal 3 signal routing is blocked by other metal 3 routes in the perpendicular direction. These biased routing capabilities can enable designers to employ multiple complex space saving routing topologies such as “L”, “C”, “J”, “H” and so forth. Additionally, signal routing can be stacked into complex patterns for processes with deeper metal stacks. Processes with limited metal stacks that include a high resistance routing layer, could leverage the multi-biased routing capability to use a high resistance layer as a jumper layer, thereby increasing available routing resources without seriously degrading timing. Different biased signal planning attributes can be applied to specified sets of signals, so the routing can be customized as needed within different parts of the design. All routeable signals have unique user defined attributes and constraints such as: metal width, metal spacing, routing layer, net shielding, via selection and usage, path length, path delay, path capacitance, path resistance and so forth.

Figure 9) Multi-biased signal routing with metal 3, shown as light blue routes (light grey for B&W), crossing other metal 3 routes by using a metal 2 jumper layer, shown as dark blue routes (dark gray for B&W).

**Signal and Bus Routing**

An automatic shape based router that supports Manhattan routing and angled routing is employed to avoid unnecessary jogs, minimize via counts and wire length. All nets are sorted and ordered to further minimize wire crossing and via usage, which supports more efficient utilization of limited routing resources. Because the underlying router is shape based, it can view the entire routing database at one time. It does not use routing ‘bins’,
as a result, signals aren't shunted unnecessarily through out of the way routing bins in order to help minimize routing congestion hot spots. Due to this unique approach, signal routing takes straight direct routes rather than longer scenic routes that are typical of bin based routers. Critical signals and buses can also be selected and routed before less critical signals and buses, so that they have earlier access to limited routing resources.

The aforementioned specialized bus planner and router is also needed to support guidance based bus routing and repeater insertion. The routed bus signals will follow matching routing topologies, use the same routing widths, via sizes and so forth.

Furthermore, a global router can be used for rapid congestion analysis feedback to help optimize the floor plan's block placement and channel sizes and thus remove routing bottlenecks before employing the Manhattan shape based router.

A spine and stitch routing capability has been developed to support routing channels and or designs that have long thin aspect ratios. The spine and stitch router can optimize use of very limited routing resources in one direction. The spine route typically connects many loads, but only uses one track, called the spine, in the direction that has limited routing resources, to the perpendicular routes from the spine route called stitch routes. These stitch routes are routed orthogonally from the spine to destination pins.

![Image](image.png)

*Figure 10) An example net, highlighted in white, that was routed using the spine and stitch routing capability. This net connects to 21 pins, but only one horizontal routing track, called the spine route, is required. The 21 vertical stitch routes all use perpendicular connections to the spine route.*

Manual routing capabilities have historically been very important to the custom design community. An intuitive GUI interface should provide a wide variety of interactive routing features and these features include short cut keys that can set attributes that specify the metal width, metal spacing, via topologies and so forth. Additional capabilities can be used to automatically move and spread selected signals to free up local routing space. Furthermore, an online DRC capability is automatically run to check manual routing edits and provide instant feedback needed to maintain design rule correctness. Routing could also be completed by using semi-automatic routing methods.
that involve a mixture of manual routing plus automatic routing. For example, manual routing features could be used to squeeze a route through a highly congested area, and then automatic routing could be used to complete a connection once the route reaches a less congested area.

**Route Fixing and Optimization**

After the design is routed, some post route optimization and error fixing is normally required. A smooth routing command removes unnecessary jogs, corners, segments and vias, while evening out the spacing between adjacent routes. This streamlines routing, improves timing and boosts production yields. The design database is automatically checked by an internal DRC tool that ensures the routing in the final database is DRC correct and will pass DRC checks run at verification. Automated checks and fixes for layer density should also be supported. Automated signal integrity checks for antenna violations followed by layer change fixes (preferred) or by diode insertion fixes can also be implemented.

**ECO Considerations**

Complex custom designs often require last moment updates that reflect changes to the design functionality. Consequently, having an automated ECO flow integrated with the physical design tools that can rapidly integrate design changes into the routed design database is very useful. ECO's can be brought into the design database in netlist format. The design software automatically checks for differences between the ECO netlist and the current database netlist and implements the required changes. For example, an additional cell may need to be added to the design database. The ECO tool would flag the designer regarding the presence of the additional cell in the ECO netlist, and then add the cell to the design database while making the necessary netlist connectivity changes. Afterwards, placement and routing of the new cell will be handled automatically with minimal impact on the rest of the design.

**Guided design flows**

Projects in the custom design world have inherently diverse design tool requirements. By extension, custom design automation tools must provide a wide variety of features and capabilities that can meet the needs of widely differing custom design projects. It follows that custom design tools must provide extensive sets of commands, options settings and configuration settings. However, only a limited subset of these commands, options and settings would be applicable to any particular design project.

Guided design flows, such as those in the Unity Planning Solution, are essential for
making custom design automation software tools easier to use. These guided flows can provide the necessary step by step design automation commands needed to work on a targeted custom design, while bypassing design tool features, capabilities, and or flow steps that aren't needed for a particular design project. From a GUI perspective, a lot of unneeded menus and options can be hidden from users that don't require them, thus simplifying the overall look and feel of the design tools while reducing the designer's learning curve. For example, a design with an extreme aspect ratio would largely rely on a channel router plus spine and stitch technology to handle the routing of the connections. Multi-biased signal planning capabilities wouldn't be appropriate for this design effort and these capabilities could be hidden by the guided design flow. If required, designers could update their guided design flow to bring in additional feature(s) or they can also use the full solution flow to access the needed feature(s) in conjunction with the guided design flow.

Summary

In summary this paper has presented a diverse set of hierarchical signal routing and optimization features and capabilities provided by a proven suite of custom design tools. This suite of design tools relies on a design database that supports visibility throughout the design hierarchy so that floor planning, bus planning, signal planning, and pin placement can be efficiently optimized using both bottom up and top down approaches. Once the hierarchical signal planning effort is complete, a wide variety of routing capabilities are available to route hierarchical signals while using topology guidance along with routing attributes such as routing layer, route width, via size, via orientation and so forth. This approach optimizes the hierarchical routing connections while minimizing routing lengths so that scarce routing resources are used more efficiently.

A top down and bottom up approach should be utilized for more reliable signal planning and routing of complex hierarchical custom designs. Designers will benefit from using signal planning and routing tools that provide the following capabilities to hierarchically plan and route signals and buses in complex custom IC’s.

1. Spine and stitch router for extreme aspect ratio routing
2. Shape based router that provides straight-direct connections
3. Multi-topology router for compact bus and signal connections
4. Multiple bias routing for compact XY routing on one layer
5. Seamless integration of macro/block placement and pin placement
6. Fast hierarchical prototyping
7. Net sorting and ordering for routing
8. Constraint driven routing
9. Automated ECO implementation
10. Post route jog and via optimization to improve performance and yield
11. Automated DRC, density and antenna checking and fixing
12. User friendly guided design flows
Author Biography

Bob Eisenstadt is a Senior Technologist for Pulsic, an EDA company specializing in physical design solutions for custom and structured IC designs. Before joining Pulsic, Bob was a Principal Engineer at Rambus and Alchip. Bob has also worked at Qthink, Silicon Image, 3dfx, and SGI.

Bob holds a BSEE from Cornell University, plus MSEE and MBA degrees from Santa Clara University.