Floorplanning: concept, challenges, and closure

Kushagra Khorwal, Naveen Kumar, and Sonal Ahuja, Freescale Semiconductor - September 19, 2012

In today’s world, there is an ever-increasing demand for SOC speed, performance, and features. To cater to all those needs, the industry is moving toward lower technology nodes. The current market has become more and more demanding, in turn forcing complex architectures and reduced time to market. The complex integrations and smaller design cycle emphasize the importance of floorplanning, i.e., the first step in netlist-to-GDSII design flow. Floorplanning not only captures designer’s intent, but also presents the challenges and opportunities that affect the entire design flow, from design to implementation and chip assembly.

A typical SOC can include many hard- and soft-IP macros, memories, analog blocks, and multiple power domains. Because of the increases in gate count, power domains, power modes, and special architectural requirements, most SOCs these days are hierarchical designs. The SOC interacts with the outside world through sensors, antennas, displays, and other elements, which introduce a lot of analog component in the chip. All of these limitations directly result in various challenges in floorplanning.

Floorplanning includes macro/block placement, design partitioning, pin placement, power planning, and power grid design. What make the job more important is that the decisions taken for macro/block placement, partitioning, I/O-pad placement, and power planning directly or indirectly impact the overall implementation cycle.

Lots of iterations happen to get an optimum floorplan. The designer takes care of the design parameters, such as power, area, timing, and performance during floorplanning. These estimations are repeatedly reviewed, based on the feedback of other stakeholders such as the implementation team, IP owners, and RTL designers. The outcome of floorplanning is a proper arrangement of macros/blocks, power grid, pin placement, and partitioned blocks that can be implemented in parallel.

In hierarchical designs, the quality of the floorplan is analyzed after the blocks are integrated at the top level. That can results in unnecessary iterative work, wasted resource hours, and longer cycle times, which could mean missed market opportunities. This underscores the importance of floorplanning.

In this paper, we will discuss some of the good practices, techniques, and complex cases that arise while floorplanning in an SOC.

The first rule of thumb for floorplanning is to arrange the hard macros and memories in such a manner that you end up with a core area (to be used for SOG placement) square in shape. This is always not possible, however, because of the large number of analog-IP blocks, memories, and
Before going into the details of floorplanning, here are few general terms that the designer has to understand:

1. **Track**: Track is a virtual guideline/path for the tool at which the signal routing happens in an SOC design. Tracks are defined for each metal layer in both preferred and non-preferred directions, which are used by the router. The router routes the signal assuming the track to be at the center of metal piece.

2. **Row**: This is the area defined for standard-cell placement in the design. A row height is based on the height of the standard cells used in design. There can be rows of various sites/heights in the design based on the type of standard cells used.

3. **Guide**: A module guide is the guided placement of a logical module structure in the design. The guide is a soft constraint. Some of the module guide logic can get placed outside the guide, and other logical module logic can be placed in the guide region.

4. **Region**: The region is a hard constraint in the design, and the design for the module is self-contained inside the physical boundary of region. However, it is possible for outside modules to have some logic placed inside the region boundary.

5. **Fence**: This is a hard constraint specifying that only the design module can be placed inside the physical boundary of fence. No outside module logic can be placed inside the fence boundary.

6. **Halo**: The halo/obstruction is the placement blockage defined for the standard cells across the boundary of macros.

7. **Routing blockage**: Routing blockage is the obstruction for metal routing over the defined area.

8. **Partial blockage**: This is the porous obstruction guideline for standard-cell placement. It is very helpful in keeping a check on placement density to avoid congestion issues at later stages of design. For example, if the designer has put a partial placement blockage of 40% over an area, then the placement density is restricted to a maximum value of 60% in the area.

9. **Buffer blockage/soft blockage**: This is a type of placement obstruction in which only buffer cells can be placed while optimization or legalization. No other standard-cell placement is allowed in the specified area during placement, but while legalization and optimization some cells can be placed in this region.

Let us now explore the various considerations and special scenarios in an SOC design one by one.

Plan your partitions depending upon architectural requirements and different power modes in an SOC.

Today, the design approach is shifting toward hierarchical closure. The hierarchical approach is also derived by the architectural requirements, such as safety; tool limitations due to higher gate counts; late IP deliverables, and different power modes in an SOC. The hierarchically partitioned blocks are implemented independently in terms of placement, routing, timing, and noise closure.

When partitions are merged at the top level, there are cases of many top-level nets detouring across the boundary of these partitions, resulting in unnecessary timing violations, wasted routing resources due to large net lengths, and buffer insertion to avoid DRVs (design-rule violations) on these nets. All these, in turn, result in increased power consumption and increased placement density. It may also include some critical nets like clock nets which need to meet particular latency targets. These issues often result in reopening the timing and routing closed partitions, finally hitting in terms of the design cycle and design resources.

So the floorplanner has to plan at the beginning only for providing metal channels for top-level routes. The routing resources used inside the partition block can’t be used for routing at the top level. One has to define certain routing resource chunks inside the partition block that are not used inside the block and hence can be used for top-level routing. The early decision for providing routing
chunks avoids iteration at later stages of design. Figure 1 explains the scenarios.

![Figure 1](image1)

**Figure 1** Top-level routing/timing issues and channel implementation in the partition block.

**Page 2**

There are partitions derived from various power modes, such as “always on,” “sleep,” and “standby.” In “always on” mode, there is small logic in the SOC that remains “live” when the rest of the chip is off to save power. This small logic is contained within a fence and has a different power supply than the rest of the SOC.

![Figure 2](image2)

**Figure 2** Routing channels defined in the partition block for top-level routing.

There are some I/O pads that are required to be an always-on supply for the chip to have interface with external world. These I/O pads are termed A-ON (always-on) pads. These A-ON pads can be placed anywhere in the pad ring (although a cut cell is required to cut the supply rail from other I/O pads). In most cases, because the always-ON logic is quite small, these I/O pads are at a greater
There are nets traversing throughout the SOC from these pads to the logic. Because the nets have to travel large distances, they may have transition or max-load violations. These violations are generally observed at very late stages of the design cycle. Buffers on the always-on supply need to be inserted to fix these DRVs.

Because the supply for rest of SOC is different, a special domain needs to be created for the always-on buffer islands. This may result in a lot of iterations in terms of floorplan changes including SOG-placement changes, incremental block movement, and more, hitting the design cycle. The floorplanner needs to be proactive in the approach and understanding of connectivity. The fence for the always-on logic should be extended throughout the SOC boundary until it reaches the A-ON pads. Figure 3 explains this scenario.

**Figure 3** Issues and implementation of buffer channels for the always-on power domain.

**Decision about the orientation and placement of analog/hard blocks**

Today’s SOCs contain a lot of analog IP that may be either in-house designs or third-party deliverables. The analog IP contains some critical analog component along with a digital portion. In most of cases, to prohibit the chip top routing from interfering and producing noise with the analog block, there are routing obstructions defined over the analog portion inside the analog IP.

These routing obstructions break the uniformity of the top-level power routing, and this may cause serious issues in IR drop closure. As such, it is important for the floorplanner to understand how the hard macro should be oriented so that top-layer metal resources an be used over the analog block for power routing. Figure 4 explains the scenario.
The analog-IP elements also have special routing requirements in terms of resistance and capacitance for the analog supplies to block, along with other current/voltage references. In order to meet the routing requirements, lot of dedicated metal resources are used, which can cause signal-routing issues at the SOC level.

Placement of the I/O pads plays a critical role. The true impact of I/O-pad locations is usually obscure until more detailed placement and routing is complete. At this point, designers generally iterate back to floorplanning to fix the analog-block placement or I/O-pad locations and then perform placement and routing again. It is important to work out proper I/O- and supply-pad placement and analog-IP placement at early stages of floorplanning. **Figure 5** explains this scenario.

**Decisions on memory stacking, channels, and orientation**

SOCs have a lot of memory content, such as RAM and ROM, for various interfaces. The placement of
these memories holds the key not only for the SOG area, but also for timing closure. There are timing paths having no wait states or one wait state to memory pins, posing memory-placement challenges to the implementation team. The timing criticality of an interface plays an important role in memory placement.

The orientation of memory cells in the design is another consideration. Depending on the technology node, there are requirements to have a particular percentage of the poly in one direction for memory cells in a design. These requirements are checked at a very late stage of design during physical verification, i.e., DRC-LVS (design-rule check and layout versus schematic). The floorplanner must be aware of these technology-specific guidelines so as to avoid last-minute design changes.

Another important decision is how many memory cells are to be stacked and how much of a routing channel is required between the memory cells. Early analysis regarding number of memory cells that can be stacked and the channel estimation helps reduce iterations with the implementation team and early closure of the floorplan.

Let’s assume a case where no signal routing is allowed over the memories, for simplicity’s sake, and do a sample calculation for the channel requirements between memories:

a) The number of pins for each memory cell is 100 (one pin will require one routing track);
b) The number of signal-routing layers in the design is four (two horizontal and two vertical);
c) The metal pitch is 0.4 µm. So the number of tracks available for one metal in 2 µm is five.

All of the pins for the memory need to be routed through the channel. These pins use the horizontal metal resources (nondefault metal can also be used, if allowed). So in order to route the memory pins through the channel, we will need 100 horizontal metal tracks for one metal layer. For 100 horizontal tracks, we will need a 40-µm (400×2/5 µm) space between two memories. Because we have two horizontal metal resources, the channel required is halved, to 20 µm. Figure 6 explains this scenario.

Floorplanners need to keep a minimum of a 10-µm channel between two memory cells, so that there
is no routing-resource crunch for memory pins and hence iterations are saved. There may cases in which the upper one/two metals are allowed for routing over the memory cells, depending upon the technology node, and the floorplanner can include these available metal resources over the memory in the channel calculation.

Also, in cases where there is lot of stacking done, DRVs may also occur on memory pins or on the nets traversing the memories. In this case, the floorplanner needs to preplan and leave one or two standard-cell rows for buffer placement to fix the DRVs. This helps avoid unnecessary iterations for memory placement. Page 3

**Obstruction placement around the corners of hard macros**

There are obstruction requirements (halos) around the hard macros and analog blocks, such as:

a) The noise-critical analog portion inside the hard block needs to be placed at a certain distance from the standard cells.

b) The well potential of the logic inside the hard macro is generally different from rest of SOC. To maintain the well spacing rules, the floorplanner has to prevent the placement of cells within certain spacing from the hard macro.

c) Signal-routing congestion issues must be avoided near macro corners.

Because of large analog-IP blocks, flash, and memory cells in the design, channels are formed between the hard macros. Based on logic connectivity, the standard-cell logic is pulled inside these channels and also in the placeable area across the boundary of hard macros. This results in the formation of a wall of standard cells in the channels and across the corners of the hard-macro blocks. Because of this, there are many routes that need to go around these hard-macro corners, resulting in routing congestion in these areas. All of these results come at routing stage in which the floorplan is in quite a solid state and any change can result in huge rework.

The floorplanner must take care of these issues up front. The designer needs to put halos around the macro corners. But inserting a rectangular halo may waste placeable area. Having a staircase-type obstruction around the corners of macros limits the placement of standard cells and leads to efficient use of the placement area. **Figure 7** explains this.
Pin placement for the partitions

In hierarchical designs, the floorplan is partitioned into functional groups based on such factors as different power supplies and architectural requirements. Pin placement for the partitions plays a critical role in the top-level design closure. The pin placement must satisfy all the design criteria, most importantly the timing requirements between different partitions at all corners and modes in the design. Poor pin placement can result in huge timing violations in interpartition paths and can cause ECOs between the block level and chip level at late design stages.

The floorplanner has to understand the top-level connectivity and architecture of the design. With input from the timing team on pin placement, the floorplanner must work out an initial pin placement for partitions based on the timing criticality of the logic and paths formed. This minimizes repetitive iterations and saves precious cycle time.

Early SOC power analysis and power-grid design

Based on the standard-cell logic and hard macros in the design, early power estimation can be worked out on paper or with the help of an EDA tool. This estimated current value is critical in determining the metal resources to be used for the power-grid design. The power grid should be strong enough to have worst voltage drop across the chip within the specifications of the design. Based on the technology options, there are one or two upper metal layers that have low resistivity. The uppermost layer (aluminum layer) has the least resistance and is mostly used for power-grid design only.

The designer has to work out optimum specifications for the metal layers to be used for the power grid for supporting the IR-drop limits and signal-routing resources in the design. The floorplanner has to avoid the lower metal layers for power because they have high resistance numbers and don’t help much in IR drop.
Also, being thin layers, these lower layers are generally used for signal resources. Generally, the designer will opt for a mesh-based power grid in the design. Figure 8 shows a conceptual diagram for mesh-type power grid.

![Figure 8 A conceptual power grid mesh design][Reference 1]

**Conclusion**

Floorplanning is the foundation step for quality implementation of an SOC. The decisions made regarding the partition blocks, pin placement, memory stacking and placement, hard-IP placement and orientation, I/O-pad placement, and power planning ripple through the entire design flow.

Most of the issues that result from poor floorplanning are generally seen at late routing stages, causing last-minute changes in the floorplan and affecting the entire design cycle. The floorplanner has to work out macro placement in sync with I/O-pad placement, achieve a symmetric block placement to have an almost square-shaped core area, plan partitions and buffer channels, group memory placement per the controller logic, avoid placing small memory cells to avoid higher pin density, and have an optimum power-mesh design. All of these recommendations, if followed, should result in efficient die area utilization, and quick design closure.

**Authors' biographies**

Kushagra Khorwal is working with Freescale Semiconductor, Noida, India, as a lead design engineer in the field of SOC physical design.

Naveen Kumar is working as a senior design engineer at Freescale Semiconductor, India Pvt Ltd with more than four years of experience in SOC physical design activities.

Sonal Ahuja is working as a lead design engineer at Freescale Semiconductor, India Pvt Ltd with more than six years of experience in SOC backend activities.

**Reference**