1

AN INTRODUCTION TO ROUTING CONGESTION

A traditional standard cell design contains wires that implement the power supply network, clocks, and signal nets. All these wires share the same set of routing resources. With the number of cells in a typical design growing exponentially and the electrical properties of metal wires scaling poorly, the competition for preferred routing resources between the various interconnects that must be routed is becoming more severe. As a consequence, not only is routing congestion increasing, but it is also becoming more damaging to the quality of the designs.

Most conventional design flows synthesize the power supply and clock networks prior to the signal routing stage. The power supply and clock nets do not perform any logical operation, but provide crucial logistical support to the circuits that actually implement the desired logical functionality. The power supply network is designed accounting for several factors such as the current requirements of the design, acceptable bounds on the noise in the supply voltage, and electromigration constraints. This network is designed in the form of a grid which may or may not be regular. Typically, the power supply network is created first and has all the routing resources to choose from. The clock nets are routed next and still have relative freedom, since only the power supply grid has used up some of the routing resources when the clocks are being routed. The clocks, which synchronize the sequential elements in the design, have strict signal integrity and skew requirements. Although they are usually designed as trees in mainstream designs, high-end designs often use more sophisticated clocking schemes such as grids in order to meet their stricter delay and skew requirements (even though such schemes can consume significantly more routing resources). Furthermore, the clock wires are typically shielded or spaced so that the signals on the neighboring wires do not distort the clock waveform; the shielding and spacing also consume some routing resources. The signal nets are routed last and can only use the routing resources that have not been occupied by the power supply and clock wires. Therefore, these are the nets that face the problem of routing congestion most acutely.

In this chapter, we will first introduce the terminology used in the context of routing congestion in Section 1.1, reviewing the basic routing model along the way. Then, we will motivate the need for congestion awareness through a discussion of the harmful effects of congestion in Section 1.2. Next, in Section 1.3, we will try to understand why the problem of routing congestion is getting worse with time. Finally, we will lay out a roadmap for the rest of this book in Sections 1.4 and 1.5 by overviewing the metrics and the optimization schemes, respectively, that are used for congestion.

1.1 The Nature of Congestion

A design is said to exhibit routing congestion when the demand for the routing resources in some region within the design exceeds their supply. However, although this simple intuitive definition suffices to determine whether some design is congested or not, one has to rely on the underlying routing model in order to quantify the congestion and compare its severity in two different implementations of a design.

1.1.1 Basic Routing Model

The routing of standard cell designs follows the placement stage, which fixes the locations of all the cells in rows of uniform height(s) as shown in Fig. 1.1(a). In today's standard cell designs, there is usually no explicit routing space between adjacent rows, since the wires can be routed over the cells because of the availability of multiple metal layers. The entire routing space is tessellated into a grid array as shown in Fig. 1.1(b). The small subregions created by the tessellation of the routing region have variously been referred to as *grid cells*, *global routing cells*, *global routing tiles*, or *bins* in the literature. The dual graph of the tessellation is the *routing graph* $G(V, E)$, an example of which is shown in Fig. 1.2(a). In this graph, each vertex $v \in V$ represents a bin, and the edge $e(u, v) \in E$ represents the boundary between the bins u and v (for $u, v \in V$).

In the routing graph shown in Fig. 1.2(a), the vias and layers are not modeled explicitly. On the other hand, the graph in Fig. 1.2(b) explicitly models bins on two horizontal and two vertical layers, as well as the vias between the different routing layers. The horizontal line segments in this figure represent the boundaries of bins on the same horizontal routing layer, the vertical line segments correspond to vias between adjacent horizontal and vertical routing layers, and the remaining line segments denote the boundaries between the bins on the same vertical routing layer. The process of routing a net on such a graph, therefore, implicitly determines its layer assignment as well.

A routing graph that models each layer explicitly consumes considerably more memory than one that bundles all the layers together. It is possible to

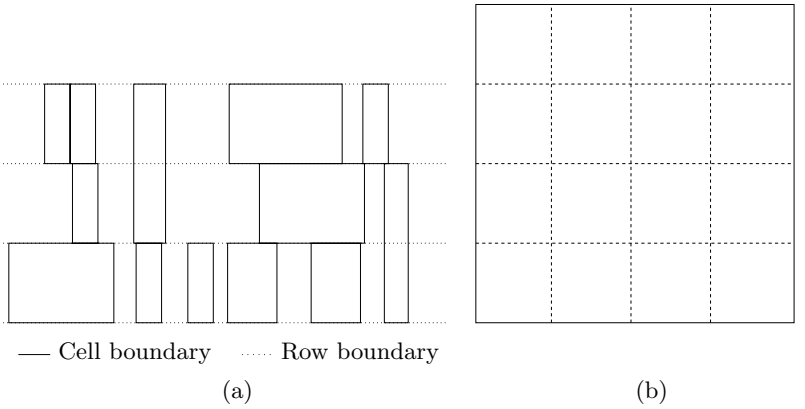

Fig. 1.1. (a) The placement of standard cells in rows of uniform height. (b) Tessellation of the routing area into bins.

retain most of the benefits of the layer-specific routing graph and yet reduce its memory footprint by grouping together layers that have similar electrical properties and wire widths. Today's process technologies offer up to nine metal layers [TSM04]. The lowermost one or two layers typically have the smallest wire widths and heights and are therefore the most resistive (although they can accommodate a larger number of tracks in each bin); these layers are appropriate for short, local wires. Minimum width wires on the middle routing layers are somewhat wider (and therefore, somewhat less resistive); these layers are used for the bulk of the global routing. The uppermost one or two layers are often reserved for very wide and tall wires that can provide low resistance paths for extremely critical signal nets and the global clock and power supply distributions.

Typically, most of the wires in a given layer are routed in the same direction (namely, either horizontal or vertical)[1], since this orthogonality of the layers simplifies the routing problem and allows for the use of the routing resources in a more effective manner. However, the lowermost layer is often allowed to be non-directional, since the flexibility to use both horizontal and vertical wires without needing a via between them facilitates pin hookups. Adjacent layers are usually oriented orthogonally to each other.

The bins are usually gridded using horizontal and vertical gridlines, referred to as *routing tracks*, along which wires can be created. The routers that use such grids are called *gridded routers*, whereas those that do not are said to be *gridless*. Although the use of the grid may appear to reduce the design freedom during routing, it allows for a simpler representation of the rout-

[1] Some process technologies do support diagonal wires in addition to the horizontal and vertical ones [XT03], but such routing architectures are not yet common in mainstream designs.

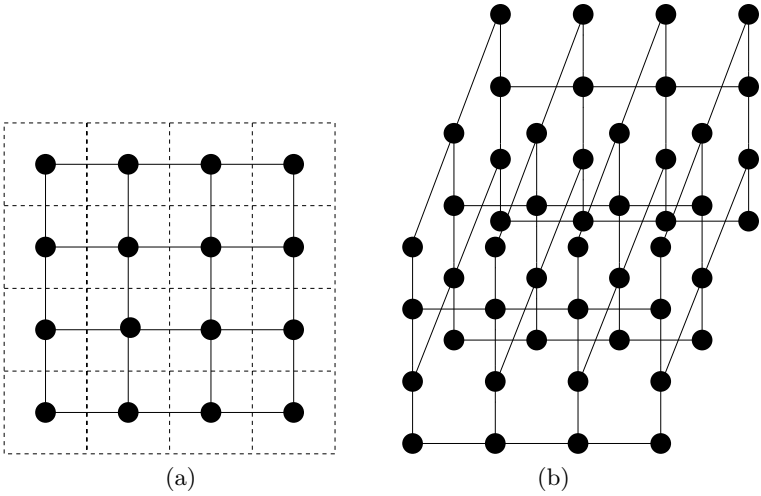

Fig. 1.2. (a) A routing graph that does not model vias and layers explicitly. (b) A routing graph for a four-layer routing architecture with explicit via and layer modeling.

ing configuration that can permit a more extensive exploration of the search space than would otherwise be possible. In particular, no special handling is required to handle via stacks across different layers, since the tracks in the grid automatically line up across the layers.

A bin can accommodate only a finite number of *routing tracks*, which may be contributed by several different layers if all the layers and vias have not been modeled explicitly in the routing graph. The number of tracks available in a bin denotes the *supply* of routing resources for that bin; this number is also known as the *capacity* of the bin. Similarly, the number of tracks crossing a bin boundary is referred to as the supply or the capacity of the routing graph edge corresponding to that boundary. A route passing through a bin or crossing a bin boundary requires a track in either the horizontal or the vertical direction. Thus, each such route contributes to the routing *demand* for that bin and edge.

Because of the complexity and the level of details involved in the routing, it is usually divided into two major stages, namely, *global routing* and *detailed routing*. The responsibility of the global routing stage is to generate routing topologies and embeddings for all the nets at the granularity of the bins. The layers in which a net is routed is determined by the layer assignment step. In today's routers, this step is usually performed simultaneously with the global routing. In other words, the regions through which a net is routed and the layers that it uses are determined concurrently. The subsequent detailed routing stage refines the global routing by assigning specific locations to all the wires within the bins, and legalizes the routing solution by eliminating

routing design rule violations. Some routers use an explicit *track assignment* stage between the global and detailed routing stages [BSN+02], even though this step has traditionally been merged with detailed routing. Global and detailed routing algorithms are discussed in more depth in Sections 4.1 and 4.2, respectively, in Chapter 4.

A net N_i denotes the logical connectivity between its *pins* (also referred to as its *terminals*) that are located in some of the bins, and may be represented as $\{v_{i,1}, v_{i,2}, \cdots, v_{i,k_i}\} \subseteq V$, where k_i is the number of terminals of N_i. Usually, one[2] of the pins of the net represents its *driver* or *source*, and the remaining pins represent its *receivers* or *sinks*. For every net N_i, the objective of the global routing stage is to find an additional subset of vertices $V_{S_i} \subset V$ (referred to as its *Steiner* nodes) and a set of edges $E_i = \{e_{i,1}, e_{i,2}, \cdots\} \subset E$ in the routing graph so as to form a rectilinear spanning tree $T_i = (V_i, E_i)$, where $V_i = N_i \cup V_{S_i}$, that minimizes some cost metric (such as the total wirelength of the net, the delay to its most critical sink, or the maximum congestion along the route of the net). When the global routing of all the nets has been completed, the demand for routing tracks is known for each bin as well as for each bin boundary. One of the objectives of the global routing stage is to route all the nets such that the demand for tracks in any bin does not exceed the supply of the tracks in that bin.

The global routing stage is followed by the detailed routing. The detailed router typically handles small regions consisting of a few bins at a time, and focuses on generating a clean routing that does not violate any design rules. This is in contrast to global routing that operates on the entire routing area, and can abstract away many of the detailed design rules. The success of detailed routing depends heavily on the quality of the results obtained during the preceding global routing. For instance, if the global routing has assigned more nets to a bin than the number of available tracks, then the successful detailed routing of all the nets in that bin may not be possible. When more wires than can be accommodated on the tracks in a bin compete to pass through that bin (even after attempting to find alternative global routes for the nets routed through that bin through uncongested bins), the routing may remain incomplete with either opens or shorts on the wires, or some of these wires may be detoured. The occurrence of such a scenario is referred to as *routing congestion*; it hints at the unavailability of sufficient routing resources in particular regions to successfully route the wires assigned to those regions.

Figure 1.3 depicts some of the steps involved in the routing of a net connecting terminals pin_1 and pin_2. The selection of the bins for the global routing of the net is shown in Fig. 1.3(b). Most detailed routers perform track

[2] A few nets can have multiple parallel drivers if the total capacitive load of the net and its sinks is too large for a single driver. Moreover, some nets (such as bidirectional buses) can have multiple drivers, at most one of which may be active at any time, with the remaining drivers being cut off using the high impedance state in tristate logic.

assignment either independently or along with the creation of the wire segments and vias in the final layout of the net. As shown in Fig. 1.3(c), the track assignment step chooses the tracks for a net, typically optimizing cost functions such as delay, number of vias, crosstalk noise, etc.. Finally, as shown in Fig. 1.3(d), the detailed router completes the layout by creating wires of appropriate length and by generating vias where required, while taking into account all the design rules.

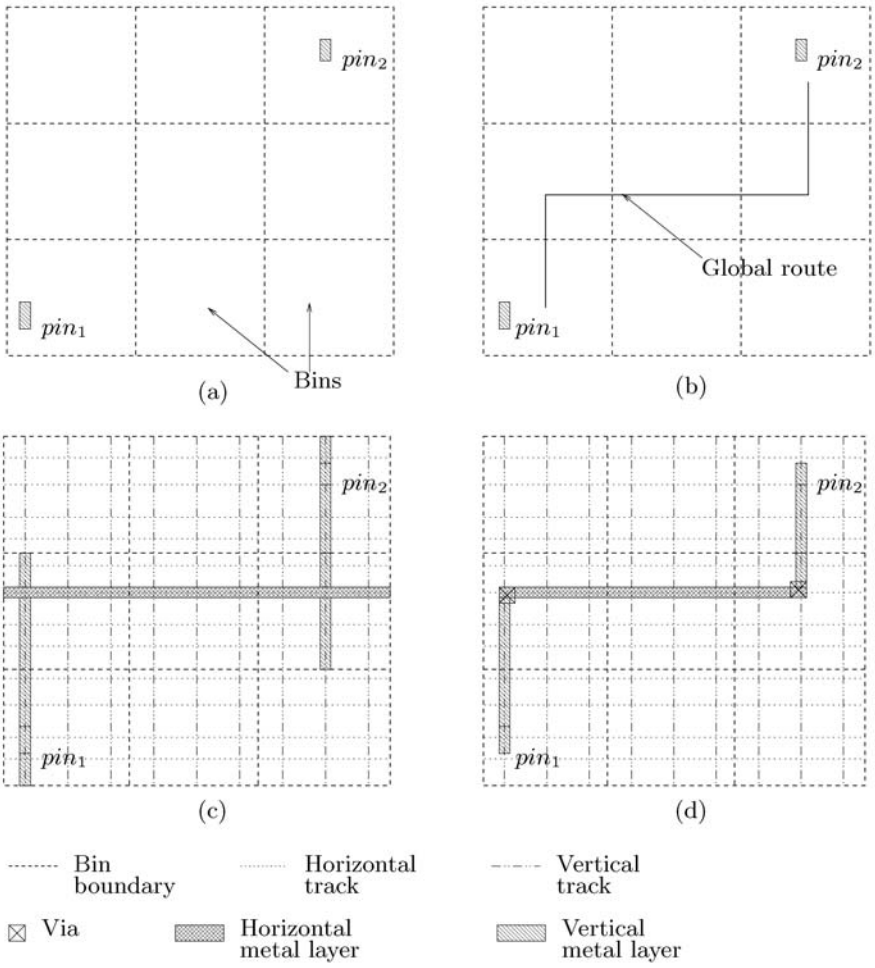

Fig. 1.3. (a) Tessellation of a routing area into bins for global routing. (b) The global routing of a net connecting pin_1 and pin_2. (c) Selection of horizontal and vertical tracks during track assignment. (d) Creation of the final routing that obeys all design rules.

1.1.2 Routing Congestion Terminology

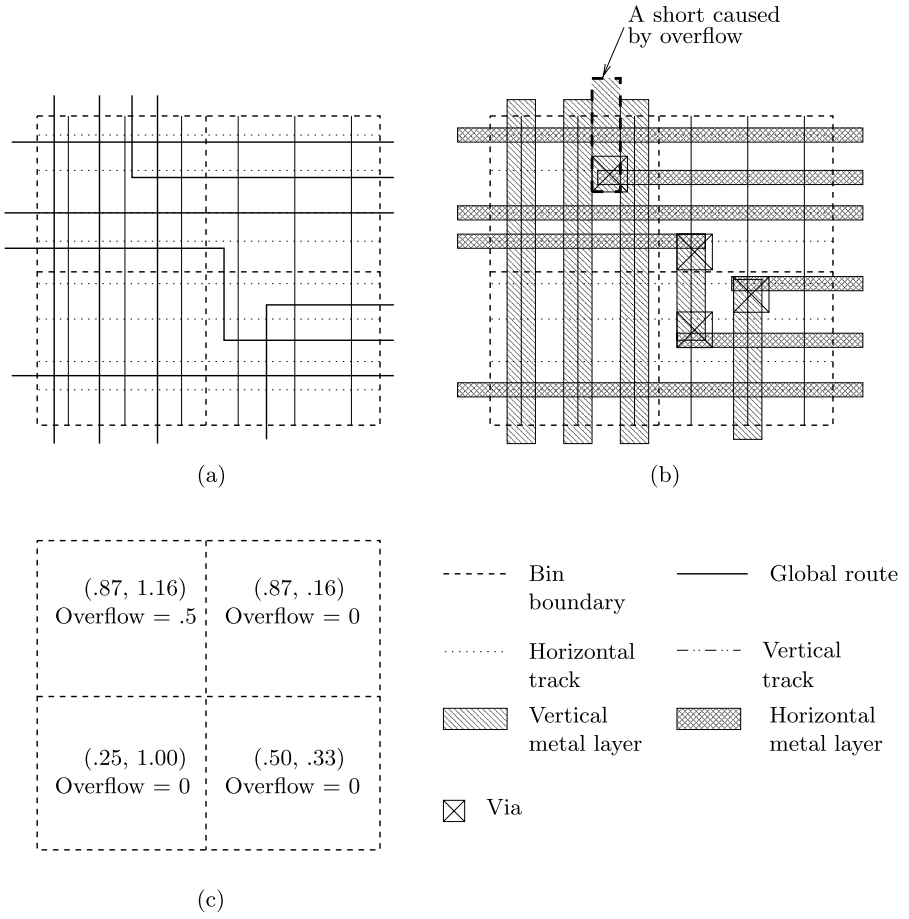

Fig. 1.4. (a) An example of a global routing. (b) A corresponding detailed routing showing a short in an overflowing bin. (c) The congestion and track overflow in each bin.

The existence of routing congestion is often manifested as detoured wires, poor layer assignment, or incomplete routes containing opens and shorts. As an example, consider Fig. 1.4, which depicts nine nets routed through four adjacent bins. Let us assume that each bin in the figure accommodates three vertical tracks and four horizontal tracks. The global routing of the nets is shown in Fig. 1.4(a), where one can observe that four nets are assigned to vertical tracks in the top left bin. Since this bin can accommodate only three

vertical tracks, one of the nets there cannot be detailed routed successfully unless it is rerouted through some other bin. If no rerouting can be found, it may create a short, as shown in Fig. 1.4(b). The remaining bins do not show any routability problems, since the demand for horizontal or vertical routing tracks never exceeds their supply in any of those bins. In other words, all the bins in the figure except for the top left one are uncongested.

One of the metrics commonly used to gauge the severity of routing congestion is the *track overflow* that measures the number of excess tracks required to route the wires in a bin. It can be defined formally as follows:

Definition 1.1. *The horizontal (vertical) track overflow T_x^v (T_y^v) for a given bin v is defined as the difference between the number of horizontal (vertical) tracks required to route the nets through the bin and the available number of horizontal (vertical) tracks when this difference is positive, and zero otherwise.*

In other words,

$$T^v = \begin{cases} \text{demand}(v) - \text{supply}(v), & \text{demand}(v) > \text{supply}(v), \\ 0 & \text{otherwise.} \end{cases}$$

Throughout this book, whenever the routing direction is left unspecified in some equation or discussion, it is implied that the equation or discussion is equally applicable to both the horizontal and the vertical directions. Thus, for instance, usage of the notation T^v (for the track overflow) in a statement implies that the statement is equally applicable to both T_x^v and T_y^v. In the same vein, if the bin to which a congestion metric pertains is clear from the context, it may be dropped from the notation (as is the case with T_x and T_y in the following paragraph).

If we assume that a route segment that enters a bin and then terminates inside that bin consumes half a routing track within that bin, it is easy to verify that $T_y = 0.5$, $T_x = 0$ for the top left bin, and $T_y = T_x = 0$ for all other bins in Fig. 1.4.

The formal definition of the *congestion* metric is as follows:

Definition 1.2. *The horizontal (vertical) congestion C_x^v (C_y^v) for a given bin v is the ratio of the number of horizontal (vertical) tracks required to route the nets assigned to that bin to the number of horizontal (vertical) tracks available.*

Thus, the congestion in a given bin is simply the ratio of demand for the tracks to their supply in that bin, and can be written as:

$$C^v = \frac{\text{demand}(v)}{\text{supply}(v)}.$$

Figure 1.4(c) also shows the horizontal and vertical routing congestion in each bin. The first element in each congestion 2-tuple associated with a bin denotes the horizontal routing congestion C_x, whereas the second represents

the vertical congestion C_y. For instance, the bottom left bin has a congestion of $(0.25, 1.0)$, since the horizontal demand and supply for that bin are one and four routing tracks, respectively, whereas the vertical demand and supply are both three tracks each. One can observe that the top left bin has a congestion of 1.16 in the vertical direction, indicating that the demand for vertical routing tracks in that bin exceeds their available supply.

The overflow and congestion metrics can be defined similarly for the bin boundaries (or equivalently, for the routing graph edges). These definitions can also be further extended to consider each routing layer individually.

The notion of a *congestion map* is often used to obtain the complete picture of routing congestion over the entire routing area. The congestion map is a three-dimensional array of congestion 2-tuples indexed by bin locations and can be visualized by plotting congestion on z-axis while denoting bins on x- and y-axes. Such a visualization helps designers easily identify densely congested areas (that correspond to peaks in the congestion map).

Some other commonly used metrics that capture overall routability of the design rely on scalar values (in contrast to three-dimensional congestion map vectors). These metrics include the total track overflow, maximum congestion, and the number of congested bins, and are defined as follows:

Definition 1.3. *The total track overflow (OF) is defined as the sum of the individual track overflows in all of the bins in the block.*

In other words,

$$OF = \sum_{\forall v \in V} T^v.$$

Definition 1.4. *The maximum congestion (MC) is defined as the maximum of the congestion values over all of the bins in the block.*

In other words,

$$MC = \max\{C^v : \forall v \in V\}.$$

Definition 1.5. *The number of congested bins (NC) is defined as the number of bins in the block whose congestion is greater than some specified threshold* $C_{threshold}$.

It can be written as:

$$NC = |\{v : v \in V \text{ and } C^v > C_{threshold}\}|$$

Note that the number of congested bins in a design is a function of the value of the selected threshold congestion. The designer may choose $C_{threshold}$ to be 1.0 to find the number of bins where the demand exceeds the supply, or may select some slightly smaller value (such as 0.9) to identify the bins where the nets are barely routable.

1.2 The Undesirability of Congestion

Designers attempt to minimize the routing congestion in a design because congestion can lead to several serious problems.

- It may worsen the performance of the design.
- It may add more uncertainty to the design closure process.
- It may result in degraded functional and parametric yield during the manufacturing of the integrated circuits for the design.

We discuss each of these issues in detail in the following subsections.

1.2.1 Impact on Circuit Performance

With wire delays no longer being insignificant in modern process technologies, an unexpected increase in the delay of a net that lies on a critical path can cause a design to miss its frequency target. The most common reason for such an unexpected increase in the delay of a net is routing congestion. Congestion can affect the delay of a net in several ways (that are listed next and then elaborated upon in the remainder of this section):

- The routing of the net may be forced to use the more resistive metal layers, resulting in an increase in the delay of the net.
- The routing of the net may involve a detour created to avoid passing through congested regions. This detour will increase the delay of the net as well as that of its driver.
- The routing may include a large number of vias generated when the router attempts to find a shortest path route through (or complete the detailed routing in) a congested region containing numerous obstructions corresponding to the nets routed earlier, leading to an increase in the delay of the net.
- Wires routed in a congested region may be more susceptible to interconnect crosstalk, leading to a greater variation in the delays of the nets.

A good timing-driven global router will attempt to route long or timing-critical nets on the less resistive upper layers, where the improved wire delays can amortize the via stack penalties involved in accessing those layers. However, if those preferred layers have already been occupied by other nets (that are presumably also critical), then the lower layers that are usually more resistive may also have to be used for some of the critical nets that are routed later. The resulting increase in the delays of the critical nets routed on the lower layers can cause timing violations on the paths passing through those nets.

If a net is detoured to avoid a congested region, the detour can increase the delay not only of the net but also of its driver. Even if we use a very simple (lumped parasitic) delay model, it is easy to show that the delay of

an unbuffered net increases quadratically with its length. Under this delay model, the delay $D_w(l)$ of a wire of length l can be written as:

$$D_w(l) = (rl)(cl) = rcl^2, \tag{1.1}$$

where r and c are, respectively, the per unit length resistance and capacitance of the wire. In other words, an increase in the wirelength of a net caused due to a detour results in its delay growing quadratically with that increase. (A similar relationship can also be shown using more sophisticated delay models). Furthermore, the increased wirelength also raises the total capacitive load seen by the driver of the net, increasing its switching time. This additional capacitance also results in an increase in the dynamic power dissipation in the net. If the driver of the net needs to be sized up to drive the increased wire load, or if the detour is large enough to require the insertion of buffers, the leakage power may also increase.

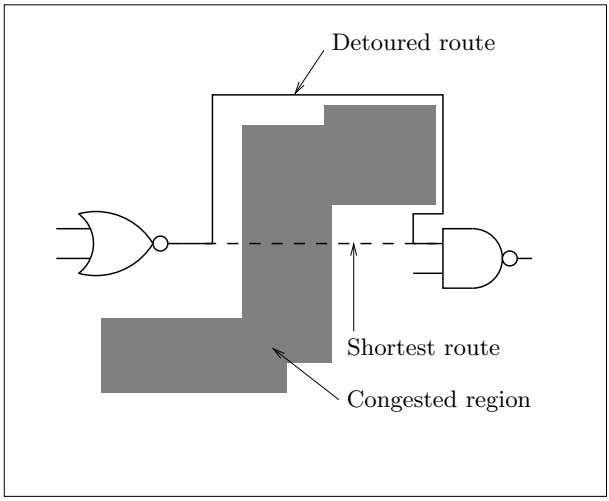

Fig. 1.5. A wire detoured because of congestion.

For example, Fig. 1.5 illustrates a scenario in which a congested region forces a net to detour significantly. Let us assume that the length of the shortest possible route for the net is 300μ, whereas that of the actual route is 700μ. Using representative values of $1.6\Omega/\mu$ for the resistance and $0.2fF/\mu$ for the capacitance of the wire, the delays of the wire based on the shortest possible and actual routes (as per Equation (1.1)) are $300 \times 1.6 \times 300 \times 0.2$ $ps = 28.8$ ps and $700 \times 1.6 \times 700 \times 0.2$ $ps = 156.8$ ps, respectively. Thus, in this case, even if we ignore the resistance of the larger number of vias that the detoured routing is likely to require, the delay of the net increases by a factor of more than five because of its detour.

The resistance of vias is scaling across process generations even more poorly than the resistance of wires. Vias in modern process generations can be significantly resistive, often being equivalent in resistance to as much as several tens of microns of minimum width wiring on one of the middle layers. Each via inserted into the routing of a net adds a significant resistance to that net, thus increasing its delay. Furthermore, most process technologies use *landed* vias for purposes of manufacturability; these vias are wider than the corresponding minimum width routing tracks. Therefore, these vias present additional blockages to the router, worsening the congestion even further.

In all the above cases, a secondary effect that can further aggravate the critical paths passing through nets that obtain a poor routing due to congestion is the worsened delays of the logic stages downstream from these nets. The increased resistance of these nets causes a degradation in the transition times for the rising and falling signal edges at their sinks. Consequently, the cells at the sinks of these nets also slow down[3].

Another potential problem that is aggravated in congested regions is that of interconnect crosstalk. A signal switching in a net driven by a strong driver can affect neighboring victim nets significantly. This interaction may result in a functional failure (if the coupled noise causes the logic value stored in a sequential element or at the output of some non-restoring logic element such as a domino gate to flip), or in a widening of the switching windows in the neighboring nets. The latter effect leads to an increased variation in the delays of the victim nets, because their effective capacitance varies depending on the switching state of their neighboring aggressor nets. Although gate sizing and buffering can ameliorate some of the noise problems, other instances of these problems are best fixed through the insertion of shields between the aggressor and victim nets, or by spacing the victim nets farther away from their aggressors. However, these techniques are difficult to apply in congested layouts because of a shortage of routing resources.

1.2.2 Impact on Design Convergence

Routing congestion adds unpredictability to the design cycle. This unpredictability of design convergence can manifest itself in two ways (that are discussed in the remainder of this section):

- Congestion-oblivious net delay estimates may mislead the design optimization trajectory by failing to correctly identify the truly critical paths.
- If a block cannot be successfully routed within its assigned area in a hierarchical design flow, the block designer may need to negotiate with the designers of neighboring blocks for more space, thus possibly necessitating a redesign of those blocks also.

[3] Note, however, that this dependence of the delay of a cell on its input slews is not captured by first-order delay models.

If the effects of congestion such as detours, unroutability and the selective delay degradation of some nets are not adequately modeled during logic synthesis or physical synthesis, then the optimizations applied at these stages can be easily misled by erroneous estimates for the design metrics. This may lead the optimization trajectory to poor configurations in the design space, recovering from which may require design iterations that are not only time-consuming but also may not guarantee convergence.

As an illustration, if the placement is oblivious to the routing congestion, the placer will not be able to position the cells so that the critical nets avoid congested regions. This also has impact on the timing, as can be seen from the example in Fig. 1.5. In this example, the placer may no longer try to position the two depicted cells any closer if the net delay of 28.8 *ps* (computed in Section 1.2.1) that it has estimated using the shortest route assumption is acceptable, even though the actual delay of 156.8 *ps* results in a timing violation.

As another example, consider the case of two nets N_1 and N_2 such that the former lies in a very densely congested region that causes its actual delay to be several times larger than its estimated delay, whereas the latter has full access to preferred routing resources. Furthermore, let us assume that N_2 is slightly more critical than N_1 on the basis of the estimated net delays. In this scenario, a circuit optimization engine that does not comprehend routing congestion will select a path through N_2 for optimization, even though the actual criticality (*i.e.*, the criticality based on achievable net delays rather than estimated ones) of some path through N_1 may be considerably higher.

While computing net parasitics and net delays, several of today's placement engines use a length-layer table that attempts to mimic how a designer or a good performance-driven router would ideally assign the layers to the nets based on their lengths. For instance, long and timing-critical nets would be assigned to upper layers, whereas short and non-critical nets would be allocated to the lower layers. However, the wire delays based on such a table quickly become invalid in congested designs, when the router is unable to route nets on the layers to which they have been assigned by the placer because of congestion. Again, this mismatch between the assumed and actual layers on which a net is routed can invalidate many of the optimizations applied to the net during physical synthesis.

Although the layer assignment and detour assumptions for a given net can be enforced using the rip-up and reroute of other nets in its vicinity or by changing the net ordering, this procedure may merely cause some other nets to become critical because of their poor routes. Indeed, if the unexpected detours are large, the nets may require buffering or significant driver upsizing. Buffer insertion can aggravate the congestion because of the extra via stacks required to access the buffers and the reduced flexibility in rerouting buffered nets. Furthermore, both newly inserted buffers and upsized drivers can create cell overlaps, whose resolution through placement legalization can cause more nets to be rerouted, often invalidating their assumed delays in the process.

If a design is unroutable or fails to converge timing because of unexpected net delays, the design flow may need to revisit the global placement stage in the hope of generating a more routable placement. If the placement stage is not able to resolve the routability issue, it may become necessary to remap or resynthesize either some or all of the logic. These additional design iterations are not only expensive from a runtime perspective, but, more significantly, may result in layouts whose congestion profile is no better than that obtained in the original iteration, unless the design flow is congestion-aware.

The unroutability of a block can prove particularly problematic in the hierarchical design methodologies that are used for large, complex designs. In these designs, different blocks are independently designed in parallel by different designers. This ability to design the blocks in parallel crucially depends on the designers obeying mutually negotiated physical, temporal, and logical interfaces for all the blocks. In high performance designs that use today's process technologies, blocks containing even a few tens of thousands of cells may require more than 30% white space in order to ensure routability. In such a scenario, the area required to route all the wires internal to a block is very difficult to predict accurately without actually implementing the block[4]. Therefore, late changes to the floorplan may be inevitable. If a block is found to be unroutable within the area allocated to it in the floorplan for the design, the need to increase its area or change its aspect ratio in order to accommodate the routing of its nets breaks the clean interface between the blocks, possibly requiring the redesign of its neighboring blocks that may already be in an advanced stage of implementation.

An example of this phenomenon is illustrated in Fig. 1.6, in which the area for block P is expanded to include the shaded area. This expansion occurs at the expense of the areas for its neighboring blocks Q and R, which now have to ensure that they remain routable in their newly reduced areas while still meeting all their design constraints.

When floorplan changes occur, all the affected blocks have to be resynthesized and laid out with new pin locations and modified block areas. These blocks are typically still required to meet the same delay constraints as they did earlier, although some power goals for individual blocks may need to be recomputed in the process of redistributing the overall power budget among the blocks. Converging the design using the new floorplan presents a variety of challenges for both types of blocks – the ones whose areas have grown, as well as the others whose areas have shrunk. For example, the block P in the floorplan in Fig. 1.6(b) may have longer wires, on the average, than those for the same block placed in the smaller area shown in the original floorplan in Fig. 1.6(a). The additional capacitance of the longer wires results in larger

[4] Overestimating the white space for a block is undesirable, because it leads to an unnecessary increase in die area, which in turn increases the manufacturing cost and reduces the yield for the integrated circuits implementing the design. This will be expanded upon in Section 1.2.3.

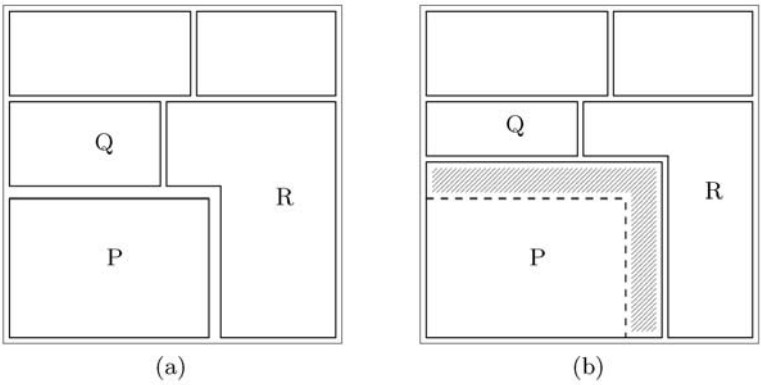

Fig. 1.6. (a) Original floorplan. (b) Altered floorplan to alleviate routing congestion in P by creating more space for the block P at the expense of the areas reserved for the blocks Q and R.

net delays, which in turn leads to increased power as drivers are upsized or extra buffers are added to the design. The blocks Q and R in the new floorplan face the challenge of placing and routing their logic in areas that are smaller than those allocated to them in the original floorplan. If either of the blocks Q or R cannot be converged in the new floorplan, more changes to the areas and shapes of neighboring blocks may be required, the success of which will not be known until those blocks have also been taken through entire synthesis-to-layout flow.

Thus, ensuring the routability of the entire design through floorplan changes involves a large cost, since it may involve many time-consuming iterations. While some of these iterations may be required anyway for delay and power budgeting across the blocks, routability adds one more factor that can necessitate additional iterations during the process of design convergence.

1.2.3 Impact on Yield

A densely congested design is likely to result in a lower manufacturing yield than a similar uncongested design. The yield of the integrated circuits implementing a design is affected by the congestion of the design in three ways:

- Congestion typically results in an increased number of vias in the routes, which can affect the yield.
- Congested layouts tend to have larger critical areas for the creation of shorts and opens due to random defects.
- Any increase in the area of a congested design in order to accommodate the routings of all its nets typically leads to some yield loss.

As mentioned in Section 1.2.1, routes in congested regions typically contain a larger number of vias than similar routes in uncongested regions. When a

router tries to find a shortest path route for a net passing through a congested region, it may create numerous bends when it maneuvers around the obstructions created by other routings. Each of these bends results in additional vias. Nets whose routes have been detoured in order to avoid congested regions also tend to have more vias than nets that have more direct routes passing through uncongested regions. Furthermore, during the process of detailed routing, if a local region is heavily congested, the reduced flexibility of the router often results in the routes having to change layers frequently in the process of being maze routed to their pins (because direct routes with few layer changes may not be possible). This too leads to an increase in the number of vias required for the routing.

The existence of a large number of vias is problematic for two reasons. Firstly, since vias are often wider than the minimum widths of the routing tracks on the corresponding layers, a large number of vias may create routing blockages that may further aggravate the congestion (and consequently result in the generation of yet more vias). Secondly, having a large number of vias can lead to yield loss during manufacturing. Vias have traditionally been undesirable from the manufacturability point of view because of the mask alignment problem, which occurs because masks for two different metal layers are required to align perfectly in order to create the vias as desired. Although advances in manufacturing technologies have reduced the severity of this problem, vias are still harder to manufacture than wire segments. Furthermore, with the shrinking geometries of modern process technologies, vias are often a factor in parametric yield loss (*i.e.*, a reduction in the maximum frequency at which the integrated circuit can operate). Ensuring perfect electrical connectivity through a via requires the metal deposition to go all the way down to the lower metal layer, without the creation of any void. However, this is not very easy to enforce in practice. A void within a via can increase its resistance dramatically, causing the delay of the net containing the via to also grow appreciably. If this net lies on a critical path, it may lead to a parametric yield loss for the integrated circuits implementing the design.

Many industrial detailed routers try to minimize the potential parametric yield impact of vias by automatically inserting redundant vias in the routes wherever possible. (This also has the collateral benefit of reducing the effective resistance of the vias, leading to better net delays). However, this technique is not very applicable in the congested regions of the layout, where there may be little or no space available to insert additional vias.

The *critical area* of a layout is a metric that indicates the likelihood of a random defect particle of a given size to cause an open or short in the layout [Fer85, MD83]. Most of the wires in a congested layout are forced to be routed with no more than minimum spacing between them. This increases the critical area of the layout with respect to shorts, because a small deposition of extra metal between two neighboring wires can cause a short, leading to circuit failure and yield loss. In a sparsely congested layout, the wires can have larger spacings between them, reducing the possibility of such shorts. The

critical area with respect to opens depends primarily on the total wirelength of the design (under the simplifying assumption that most of the signal nets are routed using minimum width wires, as is usually the case in practice). We have seen that routing congestion can result in an increase in wirelength because of detoured routes. Therefore, the probability of a random dust particle leading to an open on a wire also increases in such layouts.

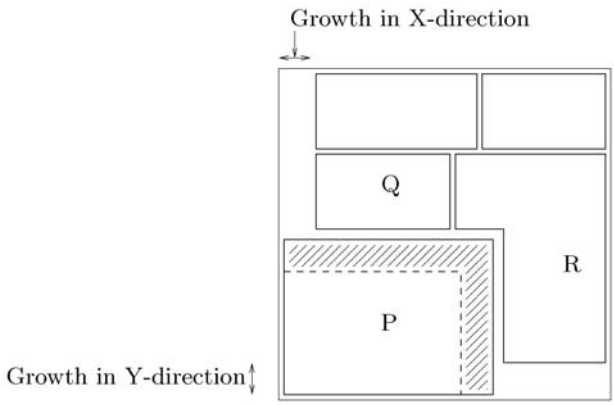

Fig. 1.7. Die size increase due to routing congestion in the block P.

Another way in which the yield can be affected due to congestion is through growth in the die size. As mentioned earlier, congestion can lead to a design being unroutable within its assigned area, if its nets cannot be routed completely even after the application of routing, placement, and synthesis optimizations targeted towards congestion minimization. In such a case, the die size may have to be increased to spread the cells out and make a larger number of routing tracks available, as depicted in the example in Fig. 1.7. In this figure, let us assume that the additional routing space required for the block P cannot be obtained at the cost of the block areas for Q and R. As a result, the die area for the chip is increased in both the horizontal and vertical directions to accommodate the growth of the block area for P. This increase in die area may, however, affect the yield adversely.

Many studies in the past have shown that the yield decreases with any increase in the die area, since the probability of random defects affecting the functionality of the circuit increases with the area. Several empirical models have been proposed to capture the relationship between the yield and die area, a typical example (based on a Poisson distribution for the occurrence of defects) [War74, Ber78] being:

$$Y = Y_0 e^{-AD},$$

where Y is the yield for chips whose die size is A, D is the defect density that depends on the clean room standards and the manufacturing process, and Y_0 is a constant. The above equation emphasizes the fact that any linear growth in area may affect the yield exponentially. Similar conclusions may also be drawn from most other yield models proposed in the literature. Thus, any increase in the die area due to routing congestion usually decreases the yield. A reduced yield translates into a rise in the per unit cost of the manufactured integrated circuits.

1.3 The Scaling of Congestion

We have seen that the existence of routing congestion in a design can lead to several serious problems. Unfortunately, an implication of today's design and process technology scaling trends is that the routing congestion problem will become even more acute in the coming years. In this section, we will build some intuition for this poor expected scaling of the routing congestion problem.

1.3.1 Effect of Design Complexity Scaling

The primary reason for the increase in congestion with successive process generations is design size scaling. With transistors getting smaller and cheaper with each successive process generation, it becomes feasible to pack more of them in a single design. Moore's Law, whose commonly accepted version states that the number of on-chip transistors is doubling every eighteen[5] months, is likely to continue to hold at least for the next decade (even if the rate of doubling slows down) [Moo03]. The semiconductor industry has managed to obey this law for the last four decades, enabling designers to integrate yet more functionality in their designs at each successive process node.

However, with an increase in the number of transistors and cells in a design comes a corresponding increase in the number of interconnections between them. Furthermore, the routing complexity of the designs also increases with an increase in the sizes of the designs. This can be illustrated by a simple thought experiment. Consider an optimized design in some process generation that is then shrunk to the next process generation without any change in logic or layout. This shrink involves the reduction of each of the geometric features of the original design by some scaling factor (that is typically 0.7×); these features include the sizes of all its gates as well as the widths, spacings, and lengths of all its wires. If one ignores the additional buffers that will be required to re-optimize the design at the new process node, one can argue that the

[5] Moore's original observation and prediction in 1965 was that the number of components in an electronic design was doubling every year [Moo65]. He later updated his predicted rate of doubling to once every two years in 1975 [Moo75].

routing complexity of the design has remained unchanged across the shrink; the insertion of additional buffers can only worsen the routing complexity. This is illustrated in Fig. 1.8, where the design shown in Fig. 1.8(a) is shrunk to the block placed in the lower left corner of the design depicted in Fig. 1.8(b).

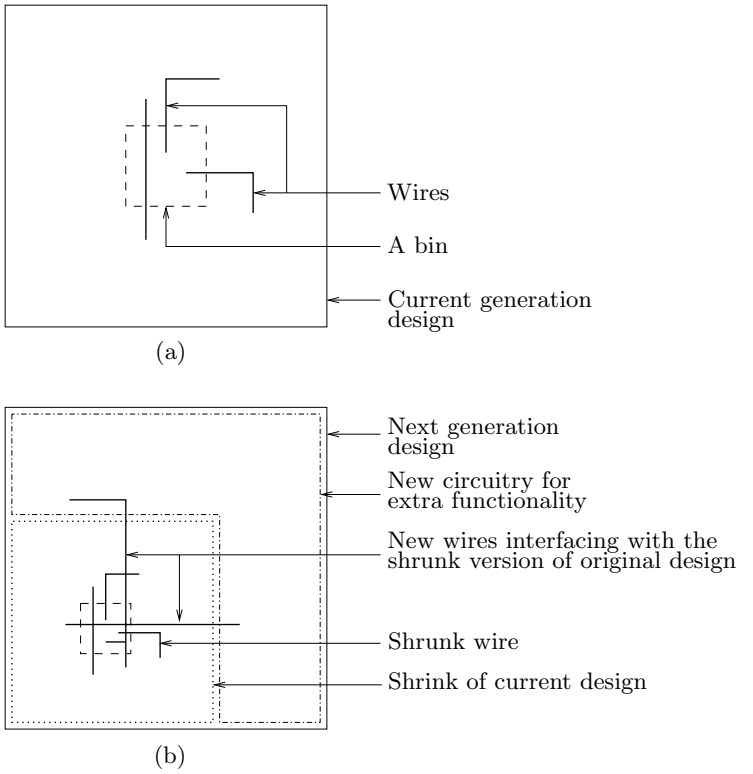

Fig. 1.8. (a) A bin with wires causing routing demand for a design in a given process technology. (b) The corresponding routing demand in a scaled version of the design that includes the original design as a shrunk block.

However, the new process node allows for the use of many more transistors, in accordance with Moore's Law. For the sake of simplicity, let us assume that the die size of the design and its cell density have remained constant[6] across the shrink. Therefore, the new design can accommodate twice as many transistors as in the old design; these additional transistors can be

[6] In practice, the die area usually grows slightly or remains unchanged across successive process generations, while the cell density decreases slightly in order to permit the successful routing of the wires in the shrunk design. As an example, the die sizes for recent Intel microprocessors have grown at the rate of 14% every two

used to integrate additional functionality, as shown in Fig. 1.8(b). However, the communication between this added functionality and the original block, which had earlier been implemented through off-chip interconnections (or not implemented at all), must now be through on-chip wires in the integrated design. These wires are in addition to the original on-chip routing of the shrunk block, and add to the routing complexity and congestion of that block. This increase in routing complexity is illustrated in Fig. 1.8 using the example of the wires passing through or connecting to a typical bin in the original design block.

This simple example illustrates the increase in routing complexity resulting from design size growth. One approach to handling this increased routing complexity could be through the introduction of new metal layers to accommodate the additional wires. However, each new metal layer involves significant additional mask generation costs. Furthermore, as will be discussed in Section 1.3.2, the introduction of additional metal layers is a strategy of diminishing returns in terms of easing routing congestion, in spite of the apparent extra routability afforded by these new layers. Consequently, the introduction of new metal layers has not been keeping pace with the rate at which the routing demands have been growing. Indeed, the number of routing layers has grown at the average rate of one new layer every three years over the last three decades, even though the number of transistors (and nets) in a design has been doubling every two years during this period.

Thus, worsening routing congestion is one of the costs that designers must pay in order to benefit from the increased integration made possible by process scaling. As design sizes increase, the routing congestion in those designs also becomes more severe. Since the introduction of additional routing layers does not help much in alleviating this congestion, designers tend to decrease the cell density and introduce more white space into their designs with each successive process generation in order to accommodate the increased routing demands.

1.3.2 Effect of Process Scaling

Although design size scaling is the prime reason behind the worsening of the routing congestion across successive process generations, the poor scaling of wires also plays a significant role in aggravating this problem. *Ideal* technology scaling [DGY+74] [Bak90] refers to the reduction of each dimension of the wires and the devices in a design by a constant shrink factor s (that has traditionally been $0.7\times$) while migrating a design from one process generation to the next. It is illustrated for wires in Fig. 1.9, where each dimension of the wire, *i.e.*, its width W, height H, distance D between the wires in the same layer, and interlayer dielectric thickness T shrinks by the constant scaling factor of s.

years [Bor00], whereas the number of transistors in the processors have doubled every two years during the same period.

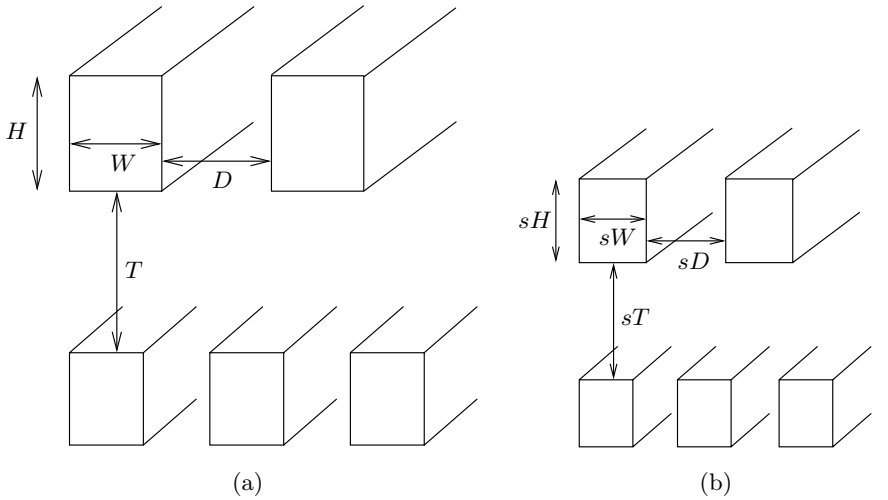

Fig. 1.9. (a) Interconnects in a given process technology. (b) Ideally scaled interconnects in the next process generation.

The resistance r, line-to-ground capacitance c, and the coupling capacitance c_c for a unit length of the wire in the current process technology generation are given by:

$$r = \frac{\rho}{W \times H},$$
$$c = \frac{\epsilon \times W}{T},$$
$$c_c = \frac{\epsilon \times H}{D},$$

where ρ is the resistivity of the metal (that was historically aluminum, but is usually copper in modern processes) and ϵ is the permittivity of the insulator, which is typically silicon dioxide. The corresponding quantities for the next process generation, where the dimensions of the wires are scaled as shown in the figure, are given by:

$$r^{next} = \frac{\rho}{sW \times sH},$$
$$c^{next} = \frac{\epsilon \times sW}{sT},$$
$$c_c^{next} = \frac{\epsilon \times sH}{sD}.$$

The above equations imply that the per unit length resistance of the wire doubles in each process generation, whereas the per unit length capacitances of the wire remain unchanged, as shown below:

$$\begin{aligned} r^{next} &= \frac{r}{s^2} = \frac{r}{0.7 \times 0.7} \approx 2r, \\ c^{next} &= c, \\ c_c^{next} &= c_c. \end{aligned}$$

As a result, assuming no activity on the neighboring wires, the delay $D_w^{next}(sl)$ for a wire of length sl obtained by shrinking a wire of length l (whose delay is denoted by $D_s(l)$) is given by the following equation:

$$\begin{aligned} D_w^{next}(sl) &= r^{next}(sl) \times (c^{next} + c_c^{next})(sl) \\ &= \frac{r}{s^2}(c + c_c)(sl)^2 \\ &= D_w(l). \end{aligned}$$

The above equation corresponds to the scaled delay of a local interconnect whose length shrinks by the usual shrink factor s. It indicates that in spite of the reduction in length, the delay of the wire does not decrease. This is in sharp contrast to the delays through the transistors, that typically speed up by a factor of s with every process generation.

The situation for global nets, whose length does not shrink with scaling (because the die size does not shrink), is even more dire. The delay of a global interconnect of length l is given by:

$$\begin{aligned} D_w^{next}(l) &= r^{next} l \times (c^{next} + c_c^{next}) l \\ &= \frac{r}{s^2}(c + c_c) l^2 \\ &= \frac{D_w(l)}{s^2}. \end{aligned}$$

In other words, the delay of a global net doubles from one process generation to the next. Even with optimal buffering, it can be shown that the delay of these nets degrades by a factor of $\sqrt{s}$. Furthermore, the inter-buffer separation in an optimally buffered wire shrinks much faster than the geometric shrink rate (shrinking instead at the rate of $s\sqrt{s}$ [Bak90]), resulting in a rapid increase in the number of buffers inserted into the nets [SMC+04] (along with its ramifications on congestion).

Consequently, wire delays become increasingly dominant with every process generation. Furthermore, since these delays do not scale well as shown above, much of the expected benefit of obtaining faster circuits on scaled process nodes is lost. Therefore, process designers often use non-ideal scaling on the wires in order to make them less resistive and improve their delay. This is done by making them wider or taller than would be indicated by the ideal scaling recipe; this is referred to as the *reverse scaling* of wires [SK99]. However, this has other undesirable side effects:

- When the wires are made wide, the number of tracks available in a given area decreases in a proportionate manner.

- When the wires are made relatively tall, the coupling component of the capacitance increases, which in turn results in increased crosstalk noise on the interconnects, causing high uncertainty in timing and possible functional failures.

Furthermore, tall or wide wires are also power hungry because of their increased capacitance; not only do they result in increased switching power, but they also require larger drivers.

The widening of the wires directly affects the supply of routing resources, which in turn increases the routing congestion. In contrast, tall wires are susceptible to interconnect crosstalk. As discussed in Section 1.2.1, interconnect crosstalk not only widens the switching windows in the nets, but can also result in functional failures. Many of the techniques used to counter this problem, such as the insertion of shields or an increase in the spacing between the signal nets, also consume routing resources that may be in short supply in congested regions. On the other hand, noise optimization techniques such as buffer insertion create additional routing blockages because of via stacks. Furthermore, very tall wires with highly skewed aspect ratios are difficult to manufacture. Thus, making the wires either tall or wide in order to counter the poor scaling of the wires affects the supply of routing resources and worsens the routing congestion.

One could hope that the increase in routing congestion due to design size or process technology scaling may be countered by adding extra routing resources in the form of new metal layers. However, there are several reasons why the introduction of new routing layers is not a panacea for the routing congestion problem. The via stacks required to access the top few layers create significant blockages on each of the underlying layers. This can become an especially severe problem on the bottommost few layers, since the via stacks from all the layers lying above them create blockages on these layers. Furthermore, we have seen that the resistance of wires increases rapidly with each successive process technology generation, causing the delay of global wires to degrade severely even as the gates speed up with scaling. Although buffer insertion can help reduce the severity of this imbalance, these buffers, when inserted in nets routed on the upper layers, result in yet more via stacks and their consequent routing blockages.

Another consequence of the worsening resistance of the wires is that the metal usage by the power grid and the global clock distribution is growing rapidly in order to avoid excessive voltage droop and poor clock slews, delays and skews. Indeed, most of the tracks on the topmost one or two layers are often reserved largely for the global clock and power grid distributions along with a handful of the most critical global signal nets.

The rapid reduction in the feature sizes with each successive process technology generation makes it increasingly difficult to obtain high yields during integrated circuit manufacturing. As a result, there has been much work on developing the so-called *design for manufacturing* techniques in recent years.

However, even these techniques have their limitations. It is feared that they will no longer be sufficient to ensure adequate yields at the 32 nm technology node. Therefore, many semiconductor manufacturers have started developing *restrictive design rules* (RDRs) as a way of tackling this manufacturing challenge. The RDRs impose many restrictions on the layout configurations permissible for the devices and their local interconnections. These restrictions, in turn, may reduce the flexibility available to detailed routers, making it harder for them to achieve successful route completion in congested designs. Therefore, in the presence of the RDRs, it will become even more important to address any expected congestion problems up front before handing the design over to the layout tools.

Thus, as designs get larger and more complex and process technologies descend yet deeper into the nanometer realm, routing congestion will become even more severe a problem than it is today. Therefore, it will be important for design flows to be able to predict the existence of routing congestion in some region of the design as early as possible, and take meaningful optimization steps to alleviate it with minimal impact to the primary design metrics such as performance, power and area.

1.4 The Estimation of Congestion

Routing congestion can be measured accurately only after the routing has been completed. However, if the design exhibits congestion problems at that stage, mere rerouting of the nets may not be able to resolve these problems. This may necessitate a new design iteration with changes being made to the placement or the netlist. For those changes to be effective, the designer must be able to judge whether the modified design is likely to have an improved congestion profile after it has been fully routed. It is in order to make this judgment that several congestion estimation metrics and schemes applicable to different stages of the design flow have been developed over the years.

The congestion metrics, therefore, serve two purposes. They allow the designer to predict the final routability of a point in the design space at a given design stage without actually going through the entire downstream flow. Secondly, they can guide the optimization techniques at that stage to move the design point towards a more routing-friendly implementation. The expectations from the metrics for these two purposes are slightly different from each other. For the former goal, accuracy is paramount and long computation times may be tolerated. However, for the latter purpose, good fidelity may be sufficient, but the metric must be fast to compute, since it will be repeatedly used to choose between different implementation choices during the course of the design optimization.

Several metrics that serve these purposes, at different stages in the design flow, have evolved over the last few years. At the routing stage, a number of such metrics are defined on the congestion map. As we saw in Section 1.1, these

metrics include the track overflow, the maximum congestion, and the number of congested bins. The goal of most of the congestion metrics developed for earlier stages in the design flow is to predict these routing-level congestion metrics as accurately as possible. All these metrics are the subject of Part II of this book.

At the placement stage, fast but relatively inaccurate congestion predictors such as the Rent's exponent, pin density, perimeter degree, and wirelength can guide the optimization in early iterations, whereas accurate and expensive techniques such as probabilistic congestion maps and fast global routers can be invoked once the placement has stabilized. These metrics and congestion estimation techniques are discussed in detail in Chapter 2. Some of these metrics and techniques have also been extended to be applicable at the preceding technology mapping stage, especially when that stage incorporates some placement information (as is the case with most modern physical synthesis flows). Other proxies for congestion that are targeted for use during technology mapping are independent of the placement and rely solely on the structural, graph theoretic properties of the netlist. Congestion metrics for the technology-independent logic synthesis stage rely almost exclusively on the structural properties of the netlist. Congestion estimation metrics applicable during technology mapping and logic synthesis are the subject of Chapter 3.

1.5 The Optimization of Congestion

The elimination of routing congestion in a typical design flow has traditionally been the responsibility of the routing stage. However, with the severity of the congestion problem increasing over the years, industrial tools have been forced to build congestion awareness in upstream design stages also. Modern congestion-aware physical synthesis flows usually model design routability at the placement stage, and use various heuristics to improve the estimated congestion profile of the design at that stage.

Indeed, routing congestion can be considered for optimization at various stages in the design flow, as each stage offers different flexibilities. For example, nets can be routed differently to avoid congested regions during the routing stage. While performing placement, cells can be placed so that the corresponding design has fewer and less severe congestion hot spots. The technology-independent logic synthesis and technology mapping stages determine the structural properties of the underlying network and the individual nets in the design, which are the sources of the routing demand. In general, as the level of the abstraction of the design increases, so does the design freedom. At any design stage, the designer has access not only to the flexibility at the current stage but also to those at subsequent stages. Unfortunately, the accuracy of the congestion metrics decreases with the increasing level of design abstraction. This affects the overall effectiveness of this approach of fixing potential congestion problems as early as possible, since this approach depends

not only on the design flexibilities but also on the accuracy and fidelity of the congestion metrics. Part III of this book is devoted to the optimization techniques available at various stages in the design flow and their effectiveness in improving the routability.

In the past, routing techniques such as rip-up and reroute or route spreading using congestion-based cost functions often sufficed for route completion. These are the topic of Chapter 4, which is dedicated to routing techniques aimed at relieving congestion. This chapter also discusses recent developments in the congestion-aware optimization of critical nets, as well as the interaction between signal routing and the power grid. Recent years have seen a significant emphasis on congestion alleviation during the placement stage, since relying solely on routing techniques for this purpose has often proven time-consuming and unpredictable for many modern designs. Chapter 5 describes the most important of these placement techniques in detail. Recent physical synthesis offerings from commercial vendors permit limited logic transformations during the placement optimizations. These capabilities point partly to the limitations of the placement-only techniques while optimizing the layout of a design and partly to the effectiveness of the logic transformations when they are guided by accurate placement information. Congestion-aware technology mapping is one such logic transformation that has seen much research during the last few years. This research has resulted in a few promising techniques to alleviate routing congestion. Technology-independent logic synthesis targeting routability or wirelength has also been pursued, typically by employing graph theoretic metrics. Although the effectiveness of such a transformation is limited by the difficulty of predicting downstream congestion accurately at this stage, this continues to feed an active area of research. Chapter 6 covers the current state-of-the-art in congestion-aware technology mapping and logic synthesis optimizations. Finally, Chapter 7 briefly describes the impact of behavioral and architectural choices on the final congestion in a design.

1.6 Final Remarks

Although routing congestion manifests itself only at the very end of the typical synthesis-to-layout flow, it can lead to unacceptable design quality and lack of design closure. The surest way to avoid such unpleasant last minute surprises is to improve the predictability of the design flow. With placement already having been integrated with circuit optimization in modern physical synthesis flows, one of the biggest obstacles to improving this predictability is the behavior of the router on congested designs, that can lead to unexpectedly large wire delays for some of the nets. Therefore, it is certainly desirable to build congestion awareness into the optimization of a design.

In this chapter, we defined several metrics to capture various aspects of routing congestion. We then looked at the impact of the routing congestion on the performance, convergence, manufacturability, and yield of modern designs.

We observed that the routing congestion in future circuits is expected to be even more severe because of growing design complexity and continuing technology scaling. Finally, we motivated the need for congestion metrics at different stages in the design flow, along with optimization techniques that can utilize these metrics to help mitigate the congestion.

References

[Bak90] Bakoglu, H. B., *Circuits, Interconnections, and Packaging for VLSI*, New York, NY: Addison-Wesley, 1990.

[BSN+02] Batterywala, S., Shenoy, N., Nicholls, W., and Zhou, H., Track assignment: a desirable intermediate step between global routing and detailed routing, *Proceedings of the International Conference on Computer-Aided Design*, pp. 59–66, 2002.

[Ber78] Bernard, J., The IC yield problem: A tentative analysis for MOS/SOS circuits, *IEEE Transactions on Electron Devices* 25(8), pp. 939–944, Aug. 1978.

[Bor00] Borkar, S., Obeying Moore's law beyond 0.18 micron, *Proceedings of the International Conference on ASIC/SOC*, pp. 26–31, 2000.

[DGY+74] Dennard, R. H., Gaensslen, F. H., Yu, H.-N., Rideout, V. L., Bassous, E., and LeBlanc, A. R., Design of ion-implanted MOSFETs with very small physical dimensions, *IEEE Journal of Solid-State Circuits* 9(5), pp. 256–268, Oct. 1974.

[Fer85] Ferris-Prabhu, A. V., Modeling of Critical Area in Yield Forecasts, *IEEE Journal of Solid-State Circuits* SC-20(4), pp. 878–880, Aug. 1985.

[ITR05] International Technology Roadmap for Semiconductors, 2005 ed., available at `http://www.itrs.net/Links/2005ITRS/Interconnect2005.pdf`.

[MD83] Maly, W., and Deszczka, J., Yield Estimation Model for VLSI Artwork Evaluation, *Electronics Letters* 19(6), pp. 226–227, March 1983.

[Moo65] Moore, G. E., Cramming more components onto integrated circuits, *Electronics Magazine*, pp. 114–117, April 1965.

[Moo75] Moore, G. E., Progress in digital integrated electronics, *Proceedings of the International Electron Devices Meeting*, vol. 21, pp. 11–13, 1975.

[Moo03] Moore, G. E., No exponential is forever: but "forever" can be delayed!, *Proceedings of the International Solid-State Circuits Conference*, vol. 1, pp. 20–23, 2003.

[SMC+04] Saxena, P., Menezes, N., Cocchini, P., and Kirkpatrick, D. A., Repeater scaling and its impact on CAD, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 23(4), pp. 451–463, April 2004.

[SK99] Sylvester, D., and Keutzer, K., Getting to the bottom of deep submicron II: A global wiring paradigm, *Proceedings of the International Symposium on Physical Design*, pp. 193–200, 1999.

[TSM04] TSMC unveils Nexsys^{SM} 90-nanometer process technology, 2004, available at `http://www.tsmc.com/english/b_technology/b01_platform/b010101_90nm.htm`.

[War74] Warner, R. M., Applying a composite model to the IC yield problem, *IEEE Journal of Solid-State Circuits* 9(3), pp. 86–95, June 1974.

[XT03] X Initiative, First 90 nm functional silicon using X architecture, 2003, available at `http://www.xinitiative.org/img/Toshiba_100803_Yoshimori.pdf`.