Under-the-Cell Routing to Improve Manufacturability

Alex Vidal-Obiols
avidal@cs.upc.edu

Jordi Cortadella
jordi.cortadella@upc.edu

Jordi Petit
jpetit@cs.upc.edu

Department of Computer Science, Polytechnic University of Catalonia, 08034 Barcelona, Catalonia

ABSTRACT

The progressive miniaturization of technology and the unequal scalability of the BEOL and FEOL layers aggravate the routing congestion problem and have a negative impact on manufacturability. Standard cells are designed in a way that they can be treated as black boxes during physical design. However, this abstraction often prevents an efficient use of its internal free resources.

This paper proposes an effective approach for using internal routing resources without sacrificing modularity. By using cell generation tools for regular layouts, libraries are enriched with cell instances that have lateral pins and allow under-the-cell connections between adjacent cells, thus reducing pin count, via count and routing congestion.

An approach to generate cells with regular layouts and lateral pins is proposed. Additionally, algorithms to maximize the impact of under-the-cell routing are presented. The proposed techniques are integrated in an industrial design flow. Experimental results show a significant reduction of design rule check violations with negligible impact on timing.

1. INTRODUCTION

Routing is becoming an increasingly challenging task as on-chip component density grows [6]. Given the unequal scaling between BEOL and FEOL, pin accessibility in new technology nodes is a critical problem. The impact of routing congestion on metrics such as performance, area and yield poses a major obstacle that must be addressed.

Moreover, the chip design community is presented with the challenging task of finding new ways to improve manufacturability in the latest technology nodes. Some lithography processes have added coloring requirements for wires [5], and printed layout quality is becoming increasingly dependent on details such as line-end distribution [9]. The profusion of new and complex design rules has posed additional constraints to pin placement and accessibility, creating even more congested zones in the most pin-dense areas.

Such issues are already being addressed in a per-technology basis. One example is self-aligned double patterning (SADP),

GLSVLSI '17, May 10-12, 2017, Banff, AB, Canada

DOI: http://dx.doi.org/10.1145/3060403.3060428

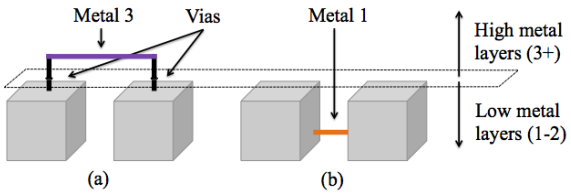

Figure 1: Symbolic cell interconnection: (a) over-the-cell and (b) under-the-cell routing.

one of the technologies used for multi-patterning at 10nm nodes [5]. The concern for pin accessibility, ultimately resulting in routing congestion, can be seen in several proposals that consider constraints imposed by this fabrication process to the synthesis flow. The work presented in [7] considers the impact of SADP, as well as other technology constraints, to decide how to access I/O pins and offer maximized flexibility to the routing stage. Another interesting example is presented in [8], with a router which explicitly considers pin accessibility constraints induced by the SADP process. However, these methods are only valid for specific processes, and additional technology-independent solutions would be desirable.

The current paper proposes under-the-cell routing, that is, to systematically exploit the connection of adjacent circuit components using lateral connections in the internal routing metal layers. White space in those layers can be used for routing in current design flows, but to a fraction of the potential use that could be achieved by exploiting it systematically. Relying on the regularity imposed by technology constraints in modern nodes, standard cell libraries can incorporate several similar cell layouts for each logical cell. These layouts are enriched with lateral pins that can be abutted against neighboring cells for direct connection without the need of accessing the high level metal layers. Cell modularity allows the design tools to pick the most suitable cell layout after the placement stage.

This method decreases both pin count and congestion on the upper metal layers, resulting in a reduction of the number of design rule check violations with negligible impact on timing. The approach is technology independent and can be used in any fixed-height standard-cell technology node.

Challenges and contributions

Whereas standard cells in traditional synthesis flows only offer I/O pins on their top, the aim of our method is to take advantage of cells whose I/O pins can also be on their sides, reducing pin count and enabling better manufacturability.

Figure 1 conveys this idea by representing two pairs of adjacent standard cells that must be connected assuming that metals 1 and 2 are used for internal cell connections. The cells in Figure 1(a) show the traditional connection scheme: Higher metal layers (at least metal 3 and 4, depending on metal direction) must be used to connect the two pins of the cells. The cells in Figure 1(b) illustrate the new connection scheme: They can be connected directly using the lower level metal layers by abutting one cell to the other without reaching higher metal layers via I/O pins. We refer to a net that can be connected via lateral pins without accessing the high metal layers as a *buried net*.

More specifically, Figure 2 shows two views of a small circuit consisting of four NAND gates. The dotted wires in the schematic view represent connections that are routed using under-the-cell routing. The lower part shows the layout implementation of the circuit using gates with lateral pins: in particular, four different layouts for the NAND cell. Each routing presents a different lateral pin interface: C_1 has no pin, C_2 has an output lateral pin to the right, C_3 has an input lateral pin to the left and an output lateral pin to the right, and C_4 has an input lateral pin to the left. To simplify the design flow, the lateral pins are enforced to be in the same track, in this case track number 5, marked in red in the example. Moreover, each cell is allowed to have at most one input and one output lateral pin in its boundaries. The rationale for these decisions is discussed in Section 2.

In the example, the output of C_2 is routed to one of the inputs of C_3 by a straight metal-1 wire extension, and the same happens with the output of C_3 being connected with one of the inputs in C_4. The remaining connection between C_2 and C_1 is done via high-metal layers. This circuit would normally require five I/O pins for its internal connections, but given that we have routed two of the three nets using under-the-cell routing, only two I/O pins are needed. Notice that the connection between C_2 and C_3 does not allow us to also connect C_2 and C_1, given that each cell can have at most one output lateral pin. The process of mapping cells to layouts becomes an interesting optimization problem.

To fully take advantage of this novel technique, two major challenges appear:

- The library of standard cells must be extended with new layouts for each cell that has lateral pins. In particular, re-routing the cells is necessary to use their versions with lateral pins.
- Traditional EDA flows must be enhanced with a lateral pin-aware placement flow in order to take maximum advantage of the possibilities offered by under-the-cell routing. In particular, one wishes to maximize the number of laterally connected adjacent cells.

The collaboration of both cell library providers and EDA vendors is required for the proposed flow to be applied at industrial level. This paper proposes an approach to address the two previous challenges, along with experimental results to show the validity of the approach. The main contributions are:

1. The use of lateral connections to reduce the complexity of higher metal layer routing, improving circuit manufacturability.
2. A systematic approach that exploits regularity to generate standard cell layouts enriched with lateral pins.

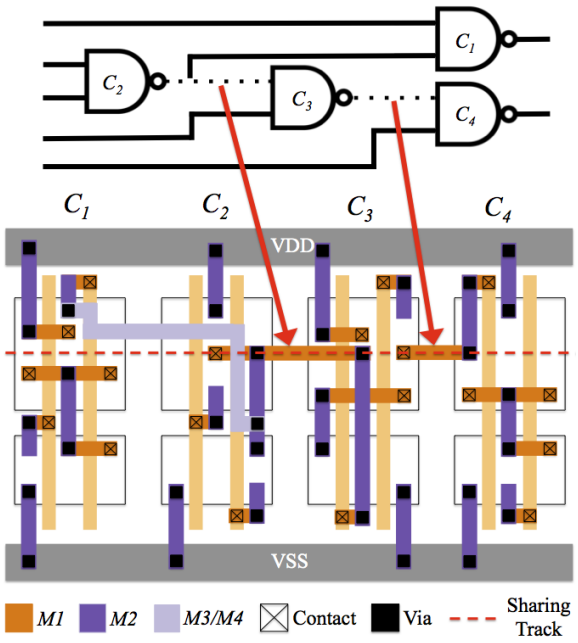

Figure 2: Small circuit consisting of four NAND gates

3. A dynamic programming and a graph-based algorithm to maximize the number of under-the-cell connections in a given circuit while preserving timing.
4. Experimental results showing a reduction in the number of design rule check violations, pins and vias with negligible timing degradation.

2. STANDARD CELLS WITH LATERAL PINS

Cell library providers must enrich their libraries incorporating cell layouts with lateral pins. Given an original standard cell library *Lib*, an extended version *eLib* is created. In *Lib* each logical cell is associated with a layout without lateral pins, whereas in *eLib*, each logical cell is mapped to the corresponding layout on *Lib* and to several other layouts with a set of lateral pins. The interface of these lateral pins is different in each one of the additional layouts, but all of them have the same area footprint.

As the number of cells in *eLib* is significantly larger than *Lib*, the first steps of the chip synthesis flow (e.g. technology mapping) will use only *Lib* to avoid an increase in algorithm complexity. The use of *eLib* is postponed until placement has been completed, at which point the final cell layout for each cell instance is determined.

Creating the extended library *eLib* requires some important decisions to be taken:

- Which pins must be provided as lateral connections?
- Are traditional pins needed in cells with lateral pins?
- At which cell tracks must the lateral pins be accessible?
- How to generate the cells with such features?

2.1 Defining the I/O interface

Potentially, any pin could be made accessible at any track if the routability of the cell would allow it. However, generating cell instances with all possibles subsets of pins accessible

at any track would make the exploration of solutions unaffordable and the size of the library unmanageable. Based on empirical experimentation, a pragmatical approach has been taken by imposing the following two constraints:

One-Input/One-output sharing. Cells are only allowed to share (at most) one input and one output lateral pin, each one at a different side of the cell. The rationale behind this decision lies on the fact that most buried nets have 2 pins, thus requiring a balanced number of input and output pins. Moreover, most cells only have one output pin. In the case of sequential elements, the lateral pins are reserved for the `D` and `Q` (or $\bar{\texttt{Q}}$) pins.

Unique sharing track. One of the tracks is chosen to be the sharing track *for all cells*, thus unifying the position for lateral pins. This greatly simplifies routing as lateral pins are always connected via abutment, or with a straight metal segment if there is white space between the cells.

The fact that the final cell layout is decided for every cell after placement allows to synthesize cells without the full set of traditional I/O pins. In general, cell layouts with an input lateral pin can skip its corresponding traditional pin, given that they have a unique driver. However, cells with an output lateral pin might need to keep the corresponding traditional pin depending on whether they are driving only their neighbor cell or also other cells of the circuit. The top I/O pins removed from layouts with lateral pins are called *buried pins*.

Given the constraints mentioned above, three kinds of cells with lateral pins exist: cells with an input lateral pin, an output lateral pin, or both. For every cell in the last two groups we can have two versions, one keeping the I/O pin for the output and one without it. Since a large majority of cells have only one output, it is easy to see that the number of layouts in *eLib* will be about three times the number of input pins in *Lib*.

2.2 Generating the cells

When synthesizing different instances of the same standard cell, transistor placement can be kept identical in all the layouts. However, the internal routing of each cell instance must be recomputed to adjust it to its interface, as shown in Figure 2.

In order to automatically generate the routing for each extended cell, we advocate for the use of a Boolean rule-based approach such as the one described in [4]. This approach is efficient, technology-independent and parameterizable for different fabrics and design rules, including support for multiple-patterning lithography. Given the gridded transistor placement of a cell, such router generates a Boolean formula that encodes the routing problem and feeds it to a SAT solver. The returned model is translated into a valid routing.

The underlaying representation for a cell is a discrete 3D grid, whose edges correspond to potential metal segments; see Figure 3. For each edge in the grid, a set of Boolean variables describe whether certain nets of the cell use or not that edge and, therefore, ultimately define whether or not a segment of wire must be laid out on that edge. Using these variables, the routing problem is formulated through two kinds of clauses:

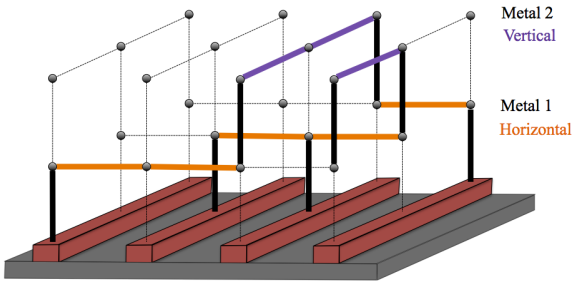

Figure 3: Grid representation for internal cell routing

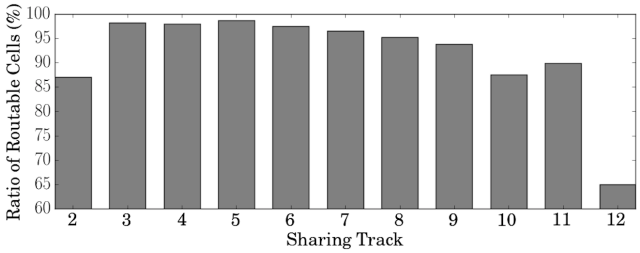

Figure 4: Routable cells per position of sharing track.

- Clauses that impose design-rule constraints.
- Clauses that represent routability constraints.

In particular, routability constraints guarantee that all pins of each net are connected. Such formulation can handle end-points whose position is either fixed at some grid point or selected from a specified set of grid points. In our context, lateral pins must be aligned with the sharing track at one side of the cell, whereas conventional pins can be located at any grid point of the top layer of the cell.

2.3 Selecting the sharing track

In order to find the sharing track that yielded more routable cells, we have undertaken the task of routing all cells in the Nangate 45nm cell library [2] using the routing approach discussed in Section 2.2 and honoring a commercial set of design rules (including rules for double patterning) on placements with minimal area. The template uses 13 tracks as shown in Figure 2: track 1 is reserved for VDD, track 13 for VSS, and tracks 2 to 12 for signal routing. Using the criteria for external pins described in the previous subsection, the extended library demands a total of 1576 layouts for the 125 cells.

The exploration began by obtaining routings with lateral pins on a reduced set of hard cells. The results of this exploration are shown in Figure 4. The vertical axis represents the percentage of cells that have been routed using the track indicated in the horizontal axis as sharing track. On the basis of this experiment, we have chosen track 5 (with a success ratio of 98.7%) to be the sharing track of the extended cell library. Then, using this sharing track, a valid routing was found for all cell instances except nine of them, leading to a routing ratio of 99.4% our *eLib*. In later experiments it was observed that virtually the 100% of the demanded layouts with lateral pins were among those successfully routed. Example of layouts obtained by the regular routing using these techniques are shown in Figure 2.

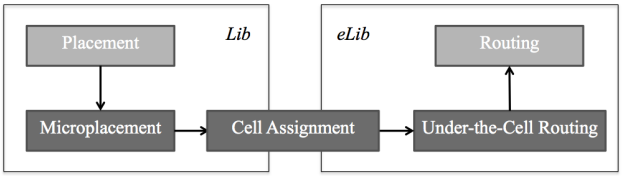

Figure 5: Proposed design flow modifications.

3. PLACEMENT AND ROUTING

The physical synthesis tools must be enhanced to take advantage of under-the-cell routing, in particular by mapping cells from *Lib* to *eLib* and modifying the placement to increase the opportunities for lateral connections.

3.1 Synthesis flow

Under-the-cell routing requires new steps to be added between the placement and routing stages of the physical design flow, as shown in Figure 5. The light boxes represent the usual steps in physical synthesis, whereas the dark boxes represent the newly introduced steps. The proposed process is transparent to the physical design flow before the cell assignment step, after which *eLib* is used.

The first new step is a *microplacement stage* which is performed after the tool has completed the placement. One of the conditions of lateral connections is that both cells to be connected must be placed side by side. Current placement tools optimize very complex functions that do not pay particular attention to having cells that need to be connected to be immediate neighbors: they try to place them close, but not necessarily adjacent. The microplacement stage presents the challenging problem of introducing small modifications to the original placement where opportunities to exploit adjacent connections are detected while preserving timing.

The second step is *cell assignment*. Each placed cell from *Lib* can be potentially substituted by one of the cell layouts in *eLib*. The enriched library provides several viable candidates, and the assignment of one cell to a layout with lateral pins affects the assignment of its neighboring cells. Cell assignment also impacts the amount of I/O pins that can be saved by using lateral pins. Finally, the *under-the-cell routing stage* adds horizontal wires for buried nets of cells which are not immediately adjacent, ie. are not connected by simple abutment.

3.2 Microplacement

The microplacement algorithm takes a valid placement and detects cells that could be connected using lateral pins. The entire flow is shown in Algorithm 1. The input is a placement P of cells from *Lib* generated by an generic placer, that is first partitioned into several regions r. Then, a greedy method is used to maximize the number of lateral connections. It generates relative placement constraints, which impose that some group of cells must be adjacent, for every region of the circuit. Finally, the generic tool performs an incremental placement on P with the newly added relative placement constraints, trying to improve the number of buried nets and pins.

Two mechanisms are applied to preserve the quality of the original placement.

Region partition: Lateral connections maximization is applied to exclusive regions of a fixed size, ensuring the global placement is preserved.

Algorithm 1 Microplacement Stage

1: **Input:** Placement P and slack tolerance ε
2: **Output:** Placement P'
3: $RPC \leftarrow \emptyset$
4: $R \leftarrow partition\_in\_regions(P)$
5: **for** each region $r \in R$ **do**
6: $\quad G \leftarrow build\_graph(r)$
7: $\quad V' \leftarrow maximum\_independent\_set(G, \varepsilon)$
8: $\quad RPC \leftarrow RPC \cup relative\_place\_constraints(V')$
9: $P' \leftarrow incremental\_placement(P, RPC)$

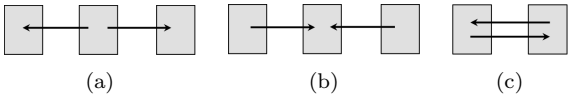

Figure 6: Microplacement incompatibilities

Slack control: Relative placement constraints are forbidden on cells with slack smaller than a tolerance ε to avoid significant timing alterations.

Maximizing lateral connections. Consider a set of placed cells $C = \{c_1, ..., c_n\}$. Each cell c_i has a set of input and output signals, $I(c_i)$ and $O(c_i)$, respectively. Each pair of cells, c_i and c_j defines a set of potential connections $S_{i,j} = \{s_{i,j,k} \mid k \in O(c_i) \cap I(c_j)\}$, where $s_{i,j,k}$ means that signal k is an output of c_i and an input of c_j. The union of all potential connections is

$$S = \bigcup_{\substack{1 \le i,j \le n \\ i \ne j}} S_{i,j}$$

Because of the restrictions explained in Section 2, not all potential lateral connections can coexist in the final placement. The method considers such restrictions only for every pair of potential lateral connections. Figure 6 shows the form of these incompatibilities for a given pair $s_{i,j,k}, s_{i',j',k'}$:

- The case $i = i'$ implies a cell with two output lateral pins (Figure 6a).
- The case $j = j'$ implies a cell with two input lateral pins (Figure 6b).
- The case $i = j' \wedge j = i'$ implies lateral connections in both directions between two cells (Figure 6c).

We can now define a graph $G = (S, E)$ as the graph where each vertex represents a potential connection and each edge an incompatibility:

$$E = \{(s_{i,j,k}, s_{i',j',k'}) \mid i = i' \vee j = j' \vee (i = j' \wedge j = i')\}.$$

By construction, the problem of maximizing the number of lateral connections of the cells C corresponds to finding a maximum independent set (MIS) of G, that is, the largest set of vertices such that no pair of them is adjacent. Each vertex can then be translated to a relative placement constraint that imposes both of the related cells to be placed one besides the other.

As the maximum independent set is a well-known NP-hard problem, a greedy approach is used as an approximation. A score is assigned to each potential lateral connection

(represented by a node in our graph) and the algorithm iteratively picks the one with the highest score, incrementally propagating the incompatibilities on the graph. This score is obtained by performing a normal routing of the circuit and comparing the estimated wirelength and the final route of the nets, prioritizing nets that have taken a long detour in order to be routed.

3.3 Cell assignment

Cell assignment substitutes each cell in *Lib* by its version in *eLib* maximizing the amount of lateral connections and buried pins.

The entire cell assignment flow is shown in Algorithm 2. The input is a placement for all the cells and the output is an equivalent placement in which some of the cells from *Lib* have been substituted by one of their layouts with lateral pins in *eLib*. For each row, *lateral_connections* applies the dynamic programming technique explained below to obtain the set of lateral connections that maximizes the number of buried pins. For each lateral connection, the original cells in P and the proposed layouts with lateral pins are obtained via *original_cells* and *enhanced_cells*. The *substitution* finally changes them in the placement. As the mapping to lateral pins has already been decided, the routing stages that take place after cell assignment use *eLib*.

Algorithm 2 Cell Assignment Stage

```
1: Input: Placement P
2: Output: Placement P'
3: P' ← P
4: R ← partition_in_rows(P)
5: for each standard cell row r ∈ R do
6:     LC ← maximize_lateral_connections(R)
7:     for each lateral connection lc ∈ LC do
8:         Lib_cells ← original_cells(lc)
9:         eLib_cells ← enhanced_cells(lc)
10:        substitution(P', Lib_cells, eLib_cells)
```

A dynamic programming solution. The goal of the function *maximize_lateral_connections* is to determine the lateral connection to be used between each pair of cells of a row, $\{c_1, ..., c_n\}$, while maximizing the number of buried pins. The problem can be solved for each row independently, given that there are no incompatibilities among lateral connections between different rows. Interestingly, the problem for cells $\{c_1, ..., c_{i-1}\}$ is nested in the problem for $\{c_1, ..., c_i\}$, and thus solving the problem until c_{i-1} is helpful to solve it for c_i. We propose a dynamic programming approach exploiting the optimal substructure of the problem.

The following nomenclature is illustrated in Figure 7. Assume a set of the best solutions has been computed for the darker cells $\{c_1, ..., c_{i-1}\}$, and now these solutions must be extended up to c_i. Let k_i be the number of possible lateral connections between c_i and c_{i+1}. At each step between two cells, either one among k_i possible lateral connections is fixed, or none is. Given $i \in \{1..n\}$ and $j \in \{0..k_i\}$, let *buried_pins*(i, j) be the number of pins the local connection will bury. It takes value 0 for $j = 0$, representing that no lateral connection is taken, and either 1 or 2 for any other valid connection $1 \leq j \leq k_i$. Let *compatibles*(i, j) be all the indices of connections from c_{i-i} to c_i such that they are compatible with connection j from c_i to c_{i+1}. The function f that counts the optimal number of buried pins from c_1 to c_i for a given connection from c_i to c_{i+1} (indexed for $1 \leq i \leq n$ and $0 \leq j \leq k_i$) is recursively defined by

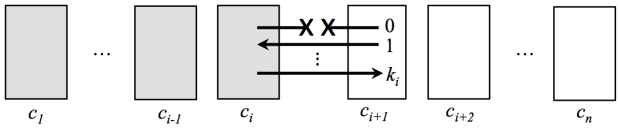

Figure 7: Dynamic programming nomenclature

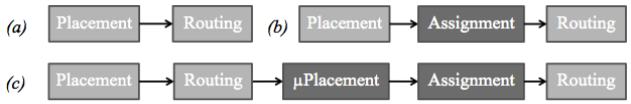

Figure 8: Synthesis flows analyzed in the experiments

$$f(i,j) = \begin{cases} \textit{buried}(i,j) & \text{if } i = 1, \\ \max\{\, f(i-1,j') + \textit{buried}(i,j) \mid & \\ \quad j' \in \textit{compatibles}(i,j)\,\} & \text{otherwise.} \end{cases}$$

The case $i = 1$ represents the first cell, which has no lateral connection to its left side. In the general case, the optimal number of pins buried up to cell i using connection j depends on the pins j can bury at cell c_i and the best solutions of compatible lateral connections in the previous step. The value $f(n, 0)$ provides the optimal lateral connection assignment that maximizes the number of lateral pins for the whole row of standard cells.

Using a bottom-up approach, the algorithm sweeps the standard cell row from left to right and at each cell c_i the best solution for any lateral connection that can be taken to the right is kept. When the best solution for the final cell c_n is fixed, it suffices to trace back the cell row to fix the solution for each of the lateral connections. The complexity is $O(kn)$, where n is the number of cells and $k = \max\{k_i \mid i = 1..n\}$. As k is low, the algorithm is essentially linear in time and space with respect to the number of cells in the row.

4. EXPERIMENTAL RESULTS

We carried out the experiments using the 6 largest circuits of the itc99 benchmark [1], with a period of about 2ns. They were placed and routed with Synopsys IC Compiler [3] using the Nangate 45nm standard cell library [2]. The three synthesis flows that were used are depicted in Figure 8:

(*a*) Original: Placement and routing with DRC reduction. Gives the base number of DRC violations.

(*b*) Assignment: Placement, cell mapping into *eLib* and routing of the unburied nets with DRC reduction. This flow exploits lateral pins and shows the basic benefits of the approach.

(*c*) μPlacement: μPlacement is added before assignment to maximize the number of lateral connections.

The original Nangate library was used for physical synthesis (placement and routing) under the assumption that the *eLib* instances with lateral pins were available with identical timing characteristics. This assumption was made to avoid the complete timing characterization of *eLib*, which is an arduous task out of scope of the paper. We believe this is

Table 1: DRC violations and congestion

		DRC violations			ΔWNS	ΔVias	ΔPins
	Util.	65%	70%	75%		70%	
b17	Orig.	53	21	140			
M4	Assign.	26	10	37	–0.6%	–4.5%	–7.0%
18k	μPla.	4	0	50	+4.9%	–2.5%	–7.7%
b18	Orig.	105	53	184			
M5	Assign.	129	49	179	–1.1%	–4.9%	–7.2%
36k	μPla.	34	31	132	+2.1%	–4.9%	–10.3%
b19	Orig.	43	79	263			
M5	Assign.	19	53	180	–1.5%	–5.5%	–7.6%
79k	μPla.	12	59	245	–1.0%	–8.0%	–15.7%
b20	Orig.	0	42	147			
M4	Assign.	0	34	105	–0.7%	–6.4%	–8.7%
18k	μPla.	5	0	80	+1.2%	–8.1%	–11.9%
b21	Orig.	44	109	137			
M4	Assign.	5	15	47	–0.4%	–6.3%	–8.7%
20k	μPla.	2	0	23	+0.5%	–9.9%	–13.0%
b22	Orig.	58	107	82			
M4	Assign.	16	49	46	–3.5%	–5.3%	–8.1%
12k	μPla.	0	10	46	+0.1%	–5.6%	–10.4%

Table 2: Summary of results (Average)

Flow	# lat. wires	Buried 2-pin nets	ΔPins	ΔVias	ΔWL
Assignment	1x	18.1%	–7.7%	–5.4%	–1.1%
μPlacement	2.1x	33.1%	–11.5%	–6.7%	+0.3%

lat. wires reports the number of adjacent cells that can be connected using lateral pins. Buried 2-pin nets reports the percentage of 2-pin nets that are connected using lateral pins. ΔPins reports the percentage of cell I/O pins that are removed when using their lateral pin counterpart. ΔVias reports the reduction of vias when the circuit is routed with lateral pins. ΔWL represents the wirelength variation.

a conservative approach since timing with lateral pins could slightly improve, as routing of the I/O pins to the upper metal levels could be avoided in many cases.

Table 1 shows the reported number of DRC violations and variations on worst negative slack (in percentage of circuit period, positive means better WNS), vias and pins across the selected circuits obtained after using the three routing flows. The different area utilization and metal layers (indicated using M under the circuit name) are chosen to show the point at which a circuit begins to present DRC violations. Approximate cell count is also shown under the benchmark's name.

Applying the assignment synthesis flow obtains fewer DRC violations than the original flow in almost all circuits. The results show that the timing of the circuit can be preserved when using these techniques. Reduction for vias and pins is obtained in all cases. When comparing the two flows using lateral pins, μPlacement tends to perform better on close to all of the circuits in terms of DRC violations. Timing is improved except in one case (−1.0% in b19). Additionally, μPlacement achieves a better reduction on vias in almost all cases and performs specially well with I/O pins, achieving a reduction of up to a 15.7%.

Table 2 shows more statistics comparing lateral pin metrics between the assignment and μPlacement flows. One immediate result of the μPlacement stage is the increase on the number of lateral connections, which roughly doubles in all of the cases. However, the wirelength increases slightly after using the μPlacement flow, canceling part of its benefits and leading to examples in which the final DRC violation reduction is lower than when only assignment is used.

It is important to note that the experiments were obtained on a 45nm technology. The impact in buried pins would probably be more noticeable in modern technology nodes, where the different scaling between transistors and wires has increased the criticality of cell pins. At these nodes, the direct reduction on the number of pins would have a heavier impact on routing congestion and could overcome the placement degradation induced by the μPlacement stage.

5. CONCLUSIONS

This paper presents under-the-cell routing, a novel and effective way of exploiting internal routing resources and routing adjacent standard cells using lateral pins. The systematic extension of a standard cell library to support lateral pins is discussed in detail. Algorithms that apply and maximize the lateral connections are proposed and integrated in an industrial circuit design flow. The experiments show the viability of the approach and its potential to improve manufacturability by obtaining fewer design rule violations with negligible timing degradation. The presented techniques require the collaboration of cell library providers to enhance libraries with lateral pins and EDA companies to implement assignment and μPlacement in the circuit synthesis tools. Future lines of work include the refinement of μPlacement algorithms to maximize the number of lateral connections while preserving all quality metrics of the circuit.

6. ACKNOWLEDGMENTS

This work has been partially supported by funds from the Spanish Ministry for Economy and Competitiveness and the European Union (FEDER funds) under grants TIN2013-46181-C2-1-R and FPI 2015, and the Generalitat de Catalunya (2014 SGR 1034 and FI-DGR 2015).

7. REFERENCES

[1] itc99 Benchmark Homepage. www.cad.polito.it/downloads/tools/itc99.html.

[2] Nangate 45nm Open Cell Library. www.nangate.com/?page_id=2325.

[3] Synopsys IC Compiler Homepage. www.synopsys.com/Tools/Implementation/PhysicalImplementation/Pages/ICCompiler.aspx.

[4] J. Cortadella, J. Petit, S. Gómez, and F. Moll. A Boolean Rule-Based Approach for Manufacturability-Aware Cell Routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(3):409–422, Mar. 2014.

[5] D. Pan, B. Yu, and J.-R. Gao. Design for Manufacturing With Emerging Nanolithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(10):1453–1472, Oct. 2013.

[6] P. Saxena, R. S. Shelar, and S. S. Sapatnekar. *Routing Congestion in VLSI Circuits - Estimation and Optimization.* Series on Integrated Circuits and Systems. Springer, 2007.

[7] X. Xu, B. Cline, G. Yeric, B. Yu, and D. Pan. Self-Aligned Double Patterning Aware Pin Access and Standard Cell Layout Co-Optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(5):699–712, May 2015.

[8] X. Xu, B. Yu, J.-R. Gao, C.-L. Hsu, and D. Pan. PARR: Pin access planning and regular routing for self-aligned double patterning. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, June 2015.

[9] H. Zhang, M. Wong, and K.-Y. Chao. On process-aware 1-D standard cell design. In *Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific*, pages 838–842, Jan. 2010.