Guiding Template-aware Routing Considering Redundant Via Insertion for Directed Self-Assembly

Kun-Lin Lin and Shao-Yun Fang

Jan 17, 2017

The Electronic Design Automation Laboratory
Department of Electrical Engineering
National Taiwan University of Science and Technology
Taipei 106, Taiwan
Block copolymer directed self-assembly (DSA) is one of Next Generation Lithography (NGL) technologies.

- Common material: PS-b-PMMA (polystyrene-block-polymethyl methacrylate)

![Diagram of block copolymer directed self-assembly](image)
Different DSA Morphologies

- Different proportions of block components form different DSA morphologies
  - (a) Spherical (A << B)
  - (b) Cylinders (A < B)
  - (c) Lamellae (A ≈ B)
Guiding patterns are used to provide additional driving forces to turn random fingerprints into highly orientated and aligned patterns.
(1) Surround each via with a guiding template
   - Require higher lithographic resolution for template fabrication

(2) Design larger templates for closely positioned vias
   - Relax the resolution requirement but the pitch of DSA holes must match the contact pitch in the layout

An axis showing the range of template sizes for generating DSA patterns from 1 hole to 4 holes
X: template width, Y: template length (nm) [Yi et al., SPIE’13]
Not every via pattern can be well generated with guiding templates.

We consider six types of DSA feasible via patterns, which are highly oriented and aligned.
Redundant Via (RV) Insertion

- RV insertion has become a necessary step
  - Improve yield and circuit reliability
  - Serve as the back up of an original via
- Inserting an RV has to avoid any short circuit problem and design rule violation
- An original via has at most 4 RV candidates
Routing by only considering either DSA or RV is not sufficient. A DSA- and RV-aware router is desirable.
Contributions

- The first work of detailed routing considering DSA via manufacturability and RV insertion
  - Propose a sophisticated routing graph model to avoid generating undesired wiring patterns
  - Propose a trunk assignment method for better via planning
  - Propose several rip-up and rerouting techniques for via layout optimization
  - Experimental results demonstrate the effectiveness and the practicality of our method
Via Graph

- One-dimensional routing is assumed
  - A via is required to connect a horizontal wire segment and a vertical wire segment in adjacent layers
- Four directions of a wire segment connection: top-left, top-right, bottom-left, bottom-right
- A via graph is used to indicate different via statuses

![Diagram of Via Graph]

- Metal 1
- Metal 2
- Via

- Inserted via
- Non-preferred via grid
- Conditional via grid
- On-track redundant via grid
- Off-track redundant via grid
- Empty via grid
Non-Preferred Via Grid

- Non-preferred via grid: a new via inserted at a non-preferred via grid results in two dead vias, regardless of connection directions.
Conditional via grid: whether a new via should be inserted depends on the direction of a wire connection.
On-/Off-Track Via Grid

- On-track via grid: no via of other connections can be inserted
- Off-track via grid: all the connection directions are favorable
- Additional cost should be added to wire routing upon/below redundant via grids

![Diagram showing On- and Off-track via grids with different types of vias and connections.](image-url)
Via Graph Update

- The via graph is updated after routing each net
- Forbidden via grid: no via can be inserted to guarantee via manufacturability
Overall Flow

A Routing Instance

Trial Routing with Trunk Assignment

Unrouted Nets?

Yes

Routing Graph Model Construction

Pick One Net

DSA and RV-aware A*-Search

Self-Conflict or Via Reservation Violation?

Yes

Rip Up and Update Via Costs

No

Update Via Graph and Routing Graph

Unrouted Nets?

Yes

Optimized Layout

Dead Via Detection and Reroute

Post-Layout Redundant Via Insertion [7]
Overall Flow

A Routing Instance

Trial Routing with Trunk Assignment

Unrouted Nets?

Yes

Unrouted Nets?

No

Dead Via Detection and Reroute

Post-Layout Redundant Via Insertion [7]

Optimized Layout

Routing Graph Model Construction

Pick One Net

DSA and RV-aware A*-Search

Self-Conflict or Via Reservation Violation?

Yes

Rip Up and Update Via Costs

No

Update Via Graph and Routing Graph

Unrouted Nets?

Yes

No
Assume each net has the highest probability to be in L-shaped such that wirelength and vias are minimized

⇒ Via and RV positions can be predicted

Try to route each net in L-shape and Z-shape

- Only consider a Z-shaped route composed of one vertical (trunk) and two horizontal wire segments

![Diagram showing trunk assignment with pins, predicted vias, and redundant via candidates.]

- Predicted via
- Predicted redundant via candidates
Trunk Assignment (cont’d)

- Scan the columns from S/T to the right/left and stop as a predicted via is encountered.
- Some feasible trunk columns exist if two scan lines are vertically overlapped.
- A route sacrificing the least predicted RV candidates is chosen.
Overall Flow

A Routing Instance

Trial Routing with Trunk Assignment

Unrouted Nets?

Yes

No

Dead Via Detection and Reroute

Post-Layout Redundant Via Insertion [7]

Optimized Layout

Routing Graph Model Construction

Pick One Net

DSA and RV-aware A*-Search

Self-Conflict or Via Reservation Violation?

Yes

No

Update Via Graph and Routing Graph

Rip Up and Update Via Costs

Yes

No

Unrouted Nets?

Yes

No
A via graph only indicates via statuses but cannot distinguish different wire connection directions.
A novel routing graph model is proposed

The model for each routing grid
- Two in-vertices and two out-vertices
- Directed edges pointed from in-vertices to out-vertices
The grid models are connected across different layers.

Four types of cross-layer connections represent four connection directions.

- Top-left
- Bottom-left
- Top-right
- Bottom-right
Propagation Example

- A propagation example for a top-left connection
A propagation example for a top-right connection
Large costs can be directly assigned on the cross-layer edges to discourage non-preferred connection directions.
Extended Graph Model

- Stack vias can be simply modeled
- Source and target pin models are used
DSA- and RV-aware A*-Search

- The routing cost of a routing path $p$:
  
  $cost(p) = \alpha \times WL + \beta \times NCC + \hat{\beta} \times CC$
  
  $+ \gamma \times NPC + \delta \times V$

  - $WL$: wirelength
  - $NCC$: # non-critical RV candidates eliminated by $p$
  - $CC$: # critical RV candidates eliminated by $p$
  - $NPC$: # non-preferred connection edges
  - $V$: # vias
  - $\hat{\beta} \gg \beta, \gamma \gg \delta$, to maximize RV insertion rates
Overall Flow

A Routing Instance

Trial Routing with Trunk Assignment

Unrouted Nets?

Yes

Routing Graph Model Construction

Pick One Net

DSA and RV-aware A*-Search

Self-Conflict or Via Reservation Violation?

Yes

Rip Up and Update Via Costs

No

Update Via Graph and Routing Graph

Optimized Layout

Dead Via Detection and Reroute

Post-Layout Redundant Via Insertion [7]

Unrouted Nets?

No
Self-Conflict Removal

- Infeasible via patterns caused by multiple vias of a net being routed may not be detected
- Increase via costs at specific grids and reroute
A dummy via is required to make an infeasible L-shaped 3-via pattern become a feasible squared 4-via pattern.

Short circuits or infeasible via patterns could be generated.
Overall Flow

A Routing Instance

Trial Routing with Trunk Assignment

Unrouted Nets?

Yes

Routing Graph Model Construction

Pick One Net

DSA and RV-aware A*-Search

Self-Conflict or Via Reservation Violation?

Yes

Rip Up and Update Via Costs

No

Update Via Graph and Routing Graph

No

Unrouted Nets?

Yes

Post-Layout Redundant Via Insertion [7]

Optimized Layout
Experimental Settings

- **Platform**
  - 2.0 GHz Linux machine with 56 GB memory
  - C++ programming language

- **Three routers are implemented based on A*-search**
  - Conventional router
  - DSA-aware router [Du et al, SPIE’14]
  - Our DSA- and RV-aware router

- **An optimal ILP-based DSA-aware post-routing RV insertion algorithm is implemented** [Fang et al, ICCAD’15]
Experimental results

- Compared to the conventional router
  - 47% RV insertion rate improvement
  - The convention router generates many unmanufacturable vias

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Conventional A*-Search</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WL</td>
<td># V</td>
</tr>
<tr>
<td>Test0.5k</td>
<td>103</td>
<td>1036</td>
</tr>
<tr>
<td>Test1k</td>
<td>219</td>
<td>2092</td>
</tr>
<tr>
<td>Test3k</td>
<td>736</td>
<td>6512</td>
</tr>
<tr>
<td>Test5k</td>
<td>1107</td>
<td>10592</td>
</tr>
<tr>
<td>Test7k</td>
<td>1792</td>
<td>15268</td>
</tr>
<tr>
<td>Test10k</td>
<td>2412</td>
<td>21538</td>
</tr>
<tr>
<td>Comp.</td>
<td>0.96</td>
<td>0.99</td>
</tr>
</tbody>
</table>

- WL: wirelength in um
- # V: # vias
- RV: RV insertion rate
- # UN: # unmanufacturable vias
- CPU: runtime
Experimental results (cont’d)

- Compared to the DSA-aware router
  - 39% RV insertion rate improvement
  - 5% WL overhead and 1% increase in # vias

<table>
<thead>
<tr>
<th>Circuit</th>
<th>DSA-aware Router [Du et al.]</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WL</td>
<td># V</td>
</tr>
<tr>
<td>Test0.5k</td>
<td>102</td>
<td>1040</td>
</tr>
<tr>
<td>Test1k</td>
<td>218</td>
<td>2118</td>
</tr>
<tr>
<td>Test3k</td>
<td>727</td>
<td>6496</td>
</tr>
<tr>
<td>Test5k</td>
<td>1102</td>
<td>10658</td>
</tr>
<tr>
<td>Test7k</td>
<td>1783</td>
<td>15374</td>
</tr>
<tr>
<td>Test10k</td>
<td>2396</td>
<td>21646</td>
</tr>
<tr>
<td>Comp.</td>
<td>0.95</td>
<td>0.99</td>
</tr>
</tbody>
</table>

- WL: wirelength in um
- # V: # vias
- RV: RV insertion rate
- # UN: # unmanufacturable vias
- CPU: runtime
Example Layout

- Blue: Manufacturable via
- Red: Unmanufacturable via
- Purple: Redundant via

- Metal 1
- Metal 2

Conventional router  DSA-aware router  Ours
Introduction

Via Graph and Wiring Pattern Enumeration

DSA- and RV-aware Detailed Routing

Experimental Results

Conclusions
Conclusions

- We present the first work on detailed routing considering DSA via manufacturability and redundant via insertion.

- A sophisticated routing graph model is proposed that is able to distinguish different wire connection directions.

- Trunk assignment and rip-up/rerouting techniques are proposed for better via planning.

- Experimental results validate the effectiveness of the proposed algorithms.