SPROUT—Smart Power Routing Tool for Board-Level Exploration and Prototyping

Rassul Bairamkulov, *Graduate Student Member, IEEE*, Abinash Roy, *Senior Member, IEEE*, Mahalingam Nagarajan, *Member, IEEE*, Vaishnav Srinivas, *Senior Member, IEEE*, and Eby G. Friedman, *Fellow, IEEE*

***Abstract*—The board-level power network design process is governed by system-level parameters, such as the number of layers and the ball grid array (BGA) pattern. These parameters influence the characteristics of the resulting system, such as power, speed, and cost. Evaluating the impact of these parameters is however challenging. To estimate the reduction in impedance if, for example, additional BGA balls are dedicated to the power delivery system, adjustments to the board layout, and an additional impedance extraction process are required. These processes are poorly automated, requiring significant time and labor. Automating power network exploration and prototyping can greatly enhance the board-level power delivery design process by increasing the number of possible design options. With power network exploration and prototyping, the effects of the system parameters on the electrical characteristics can be better understood, providing valuable insight into the early stages of the design process. SPROUT—an automated algorithm for prototyping printed circuit board (PCB) power networks—is presented here. This tool includes the first fully automated algorithm for board-level power network layout synthesis. Two board-level industrial power networks are synthesized using SPROUT. The impedance of the resulting layouts exhibits good agreement with manual PCB layouts while significantly reducing the design time. The tool is used to explore area/impedance tradeoffs in a three-rail system, providing useful data to enhance the PCB design process.**

***Index Terms*—Cost benefit analysis, design optimization, design tools, high level synthesis, iterative algorithms, power distribution, rapid prototyping.**

I. Introduction

MODERN high-performance VLSI systems require stable power [1]. Voltage scaling combined with shrinking interconnect dimensions and increasing current consumption result in significant power noise, degrading power integrity [2]. Fast transition times significantly broaden the spectrum of the power noise. Different strategies are employed at the die, package, and board levels to mitigate this power noise. The board-level power delivery network is a crucial component of the power delivery system, connecting the power management integrated circuit (PMIC) with the die or package. Careful design of the board-level power delivery system is crucial for connecting the power management IC with the package or die as well as the onboard decoupling capacitors.

The flow of the power delivery design process for printed circuit boards (PCBs) is illustrated in Fig. 1. The quality and cost of the PCB are governed by a set of system-level parameters, such as the location and model of the components, and the number and thickness of the metal layers. These parameters affect the floorplan and placement of the components. After the location of the components is known, the power management IC is connected to the target ball grid array (BGA) and decoupling capacitors. If the impedance profile of the resulting layout does not satisfy the target requirements, the layout is iteratively adjusted. These adjustments range from minor changes to the routed shape to altering the entire floorplan. Several iterations are often necessary to comply with the target impedance requirements [3].

The influence of the system parameters on power integrity and cost is qualitatively well understood. For example, adding decoupling capacitors would likely reduce the inductive noise while adding cost. Quantifying these effects prior to floorplanning and routing is however difficult. Due to the lack of information during early stages of the system design process, the system-level parameters are often arbitrarily chosen. These power delivery systems may fail to satisfy target impedance requirements, leading to a costly redesign process. Early exploration of the design space may eliminate or decrease the number of layout adjustments at later stages of the design process.

The objective of the proposed smart power routing algorithm for PCBs (SPROUT) is to produce a prototype of the power network based on a target set of design parameters (see Fig. 2). The resulting layout is suitable for impedance extraction. Therefore, the impedance of the layout based on the target set of design parameters may be efficiently and automatically evaluated. This capability supports a more rigorous evaluation of the design space and better exploration of design tradeoffs, such as performance and cost. An informed choice of design parameters early in the development process reduces

Manuscript received December 31, 2020; revised April 12, 2021; accepted July 11, 2021. Date of publication July 30, 2021; date of current version June 20, 2022. This work was supported in part by the National Science Foundation under Grant CCF-1716091; in part by the Intelligence Advanced Research Projects Activity under Grant W911NF-17-9-0001; in part by Qualcomm; and in part by Synopsys. This article was recommended by Associate Editor L. Behjat. *(Corresponding author: Rassul Bairamkulov.)*

Rassul Bairamkulov and Eby G. Friedman are with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627 USA (e-mail: rbairamk@ur.rochester.edu; friedman@ece.rochester.edu).

Abinash Roy, Mahalingam Nagarajan, and Vaishnav Srinivas are with QCT, Qualcomm Technologies, Inc., San Diego, CA 92121 USA (e-mail: abinashr@qti.qualcomm.com; mahaling@qti.qualcomm.com; vaishna@qti.qualcomm.com).

Digital Object Identifier 10.1109/TCAD.2021.3101411

1937-4151 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

TABLE I
COMPARISON BETWEEN SIGNAL AND BOARD-LEVEL POWER ROUTING

Feature	Signal Routing	Board-level Power Routing
Goal	Connect signal source and destination locations	Connect power management IC with ball grid array
Resulting geometry	Rectilinear metal tracks	Arbitrary shaped metal segment
Typical number of nets	More than 50	Less than ten
Typical constraints	Crosstalk noise, timing	Current density, temperature, metal resources

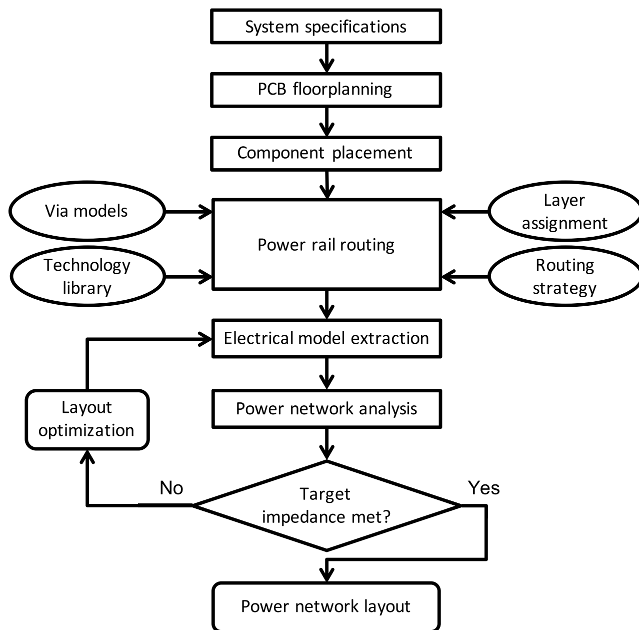


Fig. 1. Conventional design flow for PCB-based power delivery networks.

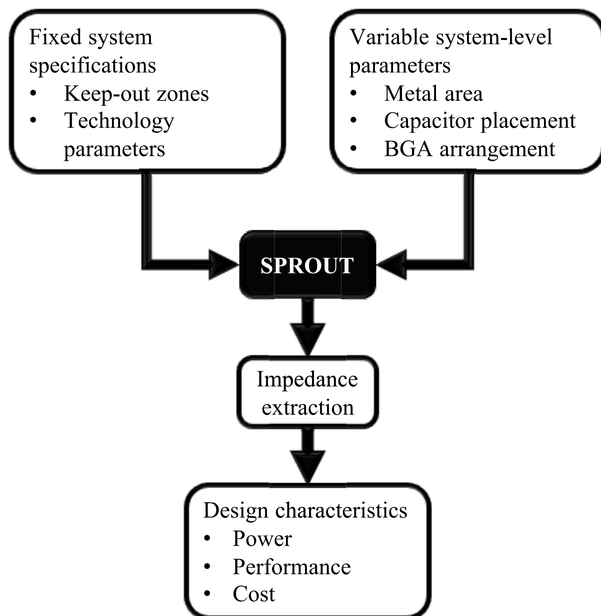


Fig. 2. Proposed prototyping flow for PCBs using SPROUT. The PCB layout parameters are the inputs to SPROUT that produce a prototype of the power network. The parasitic impedance of the prototype is estimated. This process is repeated for different sets of system-level parameters. The power, performance, and cost of each prototype are evaluated and compared to other prototypes to determine the most favorable system parameters.

the likelihood of not satisfying the target impedance. In addition, the layout prototype may guide the final layout, further accelerating the development process.

On-chip signal routing is a well-established subject in the research community [4]–[8]. Signal routing is, however, significantly different from the problem discussed in this article, as described in Table I. The number of on-chip signal terminals exceeds thousands, whereas fewer than 20 voltage domains are supported by a board-level power network. On-chip signal nets are typically routed via rectilinear metal tracks, whereas board-level power shapes can have an arbitrary form. Design priorities and constraints for signal and power routing are also different. Characteristics prioritized in signal routing include crosstalk, attenuation, impedance matching, intersymbol interference, timing, and mode conversion. Different metrics are prioritized in power routing, such as current density, thermal profile, and resistive and inductive noise.

Unlike automated signal routing, which is extensively studied in the literature, the automated *synthesis* of board-level power nets has received minimal attention. Most works in the literature focus on the analysis of existing power delivery networks. For example, in [9]–[12], fast methods for estimating the impedance of board-level power networks are described. In [13], a simplified circuit model is presented to evaluate inductive power noise. An accurate PCB analysis methodology is proposed in [14] where the finite difference model is integrated with SPICE. Methods for enhancing electromagnetic compatibility and power integrity are discussed in [4]–[8]. IC power network synthesis has been discussed in the literature. Decoupling capacitor selection and placement is a common topic [4], [15], [16]. In [17], the top layers of an IC power network with minimal congestion are synthesized. Using a neural network, a nonuniform power mesh that minimizes routing congestion is generated while satisfying electromigration and voltage drop constraints.

Several features distinguish on-chip and board-level power network synthesis. The top layers of the IC power network are often structured as a mesh [1], [18], [19]. The current is supplied to the load using vias and orthogonal wires. The PCB power network does not typically span the entire layout. Instead, specific areas of the PCB, namely, the PMIC output, BGA, and, optionally, decoupling capacitors, are connected using an arbitrarily shaped metal segment.

SPROUT is the first automated power network prototyping tool for PCBs, initially presented in [20]. Major contributions of this article include the application of graph-based optimization to the synthesis of board-level power network layout and a multilayer power network routing algorithm. The remaining portion of this article is organized as follows. In Section II, the power routing algorithm is described. The algorithm is validated using industrial case studies in Section III. Some conclusions are provided in Section IV. A modification of SPROUT to support multilayer routing is described in the Appendix.

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

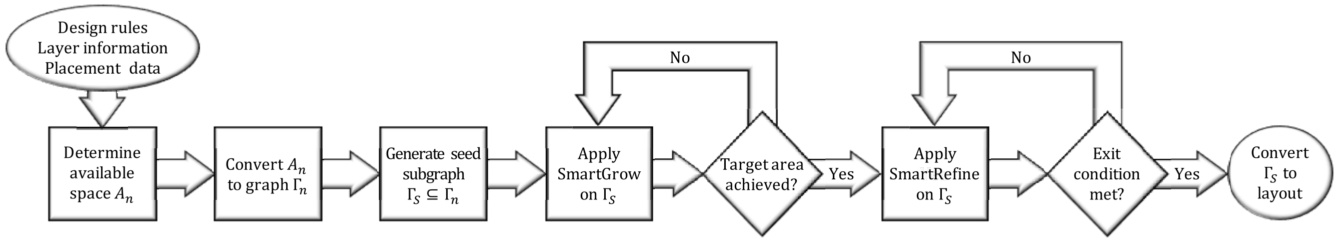


Fig. 3. Overview of the SPROUT algorithm. The available space A_n is converted into an equivalent graph Γ_n. The subgraph seed Γ_n^s is generated by SPROUT and expanded using the SmartGrow algorithm described in Section II-D. After achieving the target area, the nodes in Γ_n^s are rearranged using the SmartRefine algorithm to enhance the electrical characteristics. The final subgraph is converted into a physical layout.

II. SPROUT Algorithm

A typical board-level layout consists of several metal layers, each separated by a dielectric layer. The connections between the layers are provided by vias. SPROUT uses layer information, design rules, and placement data to produce an initial layout. The objective of the algorithm is to generate a shape connecting the power management IC with the target BGA balls and decoupling capacitors while complying with the design rules and minimizing the impedance between the terminals. Note that the resulting design is not the final topology but a prototype used to estimate the effects of the design parameters on system performance.

Similar to many signal routing algorithms, SPROUT works in the graph domain, permitting the exploitation of powerful graph-based algorithms. An overview of the proposed algorithm is shown in Fig. 3. The space available for routing is initially determined from the input layout, as described in Section II-A. This layout is converted into a graph, and the initial seed connection is established between the terminals, as described in, respectively, Sections II-B and II-C. SmartGrow and SmartRefine algorithms are introduced in, respectively, Sections II-D and II-E. Based on these algorithms, the impedance between the terminals is iteratively reduced by adding and rearranging the nodes. A subgraph reheating technique, inspired by simulated annealing, is proposed in Section II-F where the size of the graph is temporarily increased to reduce the probability of a suboptimal impedance. In Section II-G, the placement of the resulting graph into the original layout is described. The complexity of SPROUT is discussed in Section II-H.

A. Available Routing Space

An assessment of the available space commences with processing the input information. Each element of the layout is converted into a polygon with four parameters: 1) layer; 2) net; 3) geometry; and 4) buffer. To understand each component, consider three vias placed on the top layer of a PCB [see Fig. 4(a)]. The via pads are converted into polygons. Assuming the vias are placed on layer 1, the layer parameter of the corresponding polygons is 1. Each capacitor pad is assigned a net, namely, V_{DD} and V_{SS}. The geometry of each pad is expressed as an ordered set of coordinates. To decrease the likelihood or minimize the effects of manufacturing defects, such as unintended shorts, spurs, underetches, and electromagnetic interference [21], each geometry is assigned a buffer.

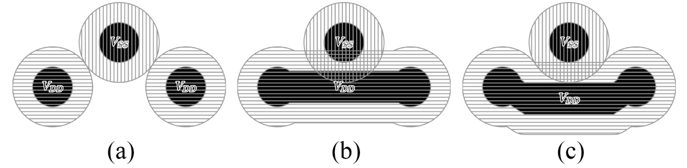


Fig. 4. One V_{SS} (vertical hatch), two V_{DD} (horizontal hatch) via pads (dark), and buffers (light). (a) Initial layout. (b) Connection to the V_{DD} vias is invalid since the buffer around the V_{DD} connection overlaps the V_{SS} via, and the V_{DD} connection overlaps the V_{SS} via buffer. (c) Example of valid routing. Neither the V_{DD} nor the V_{SS} buffer intersects the vias or connections to a different net. Note that the V_{DD} connection can be placed in the buffer around the V_{DD} vias because both the via and connection belong to the same net.

This buffer ensures polygons from different nets are properly spaced. To illustrate the buffering process, consider the example shown in Fig. 4(b). Contact between the two V_{DD} vias is not possible using a straight interconnect segment because this segment intersects the buffer of the V_{SS} via, and the via intersects the buffer of the interconnect. The bent interconnect segment shown in Fig. 4(c) produces a valid connection since the geometries do not intersect the buffers of the other nets. Note that it is legal for a V_{DD} polygon to cross a V_{DD} buffer because these polygons belong to the same net.

The entire design space U is initially viewed as available for routing. The available space A_n for a particular net n is determined by removing buffers of the other nets from the design space

$$A_n = U \setminus \bigcup_{n_j \neq n} b_j. \tag{1}$$

Polygon removal is achieved by utilizing efficient polygon clipping algorithms [22], [23] that require negligible time, as discussed in Section II-H. After removal, the available space on each layer may become disjoint, leaving no valid path between terminals on the same layer, as illustrated in Fig. 5. In this case, routing is accomplished using multiple layers. Based on the algorithm described in the Appendix, the multilayer routing problem is decomposed into several single-layer routing problems.

B. Equivalent Graph

Once the available space of the layout is determined, it is converted into an equivalent graph Γ_n, as described in Algorithm 1. The available space A_n is divided into tiles a_n.

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

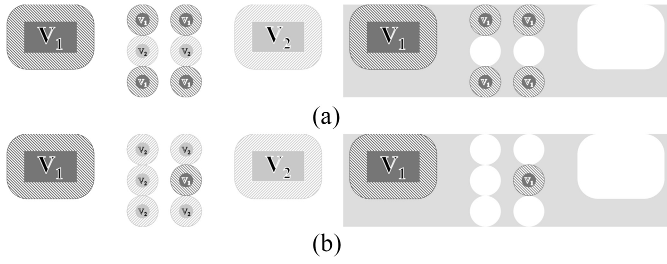


Fig. 5. Available space (shaded) for V_1 in two layouts. (a) Layout (left) where routing from the pad on the left to four vias is possible, as evident from the connected available space (right). (b) Layout (left) where connecting a pad with a via is not possible within a single layer due to the disjoint available space (right).

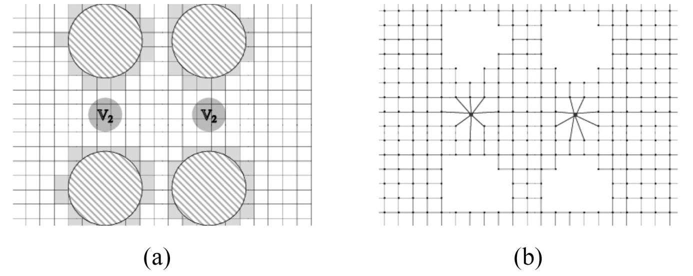


Fig. 6. Conversion of the available space for net V_2 into an equivalent graph. (a) Available space is split into unit cells. Cells with irregular shapes are shaded. (b) Equivalent graph. The tiles overlapping vias are treated as a single node. Nodes are not generated in prohibited areas.

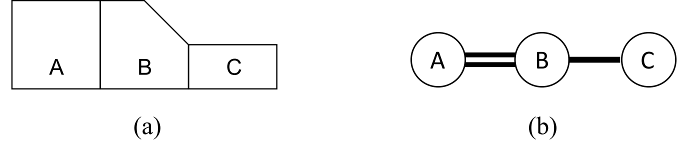


Fig. 7. Conversion of irregularly shaped tiles into an equivalent graph. (a) Tiles A and B have a twice wider contact than tiles B and C. (b) Nodes A and B have double conductance as compared to nodes B and C.

Algorithm 1 Convert Available Space A_n into Equivalent Graph Γ_n Using Tiles of Size $(\Delta x, \Delta y)$

1: **procedure** SPACETOGRAPH(A_n, Δx, Δy)
2: $V_n \leftarrow \varnothing$
3: $E_n \leftarrow \varnothing$
4: $[x_{min}, x_{max}, y_{min}, y_{max}] \leftarrow bounds(A_n)$
5: $n_x \leftarrow \left\lfloor \frac{x_{max}-x_{min}}{\Delta x} \right\rfloor$
6: $n_y \leftarrow \left\lfloor \frac{y_{max}-y_{min}}{\Delta y} \right\rfloor$
7: **for** $i = 0, 1, 2, \ldots, n_x$
8: $x^i_{min} \leftarrow x + i\Delta x$, $x^i_{max} \leftarrow x + (i+1)\Delta x$
9: **for** $j = 0, 1, 2, \ldots, n_y$
10: $y^j_{min} \leftarrow y + j\Delta y$, $y^j_{max} \leftarrow y + (j+1)\Delta y$
11: $box_{i,j} \leftarrow rectangle(\{x^i_{min}, y^j_{min}\}, \{x^i_{max}, y^j_{max}\})$
12: $cell_{i,j} \leftarrow box_{i,j} \cap A_n$
13: **if** $cell_{i,j} \neq \varnothing$
14: Add $cell_{i,j}$ to V_n
15: $overlap_y = cell_{i,j} \cap cell_{i,j-1}$
16: **if** $overlap_y \neq \varnothing$
17: Add $\{cell_{i,j}, cell_{i,j-1}, \frac{length(overlap_y)}{\Delta x}\}$ to E_n
18: $overlap_x = cell_{i,j} \cap cell_{i-1,j}$
19: **if** $cell_{i,j} \cap cell_{i-1,j} \neq \varnothing$
20: Add $\{cell_{i,j}, cell_{i-1,j}, \frac{length(overlap_x)}{\Delta y}\}$ to E_n
21: **return** $\Gamma_n = (V_n, E_n)$

Using a bijective map

$$f : A_n \leftrightarrow \Gamma_n \tag{2}$$

each tile a_n becomes a node γ_n within the graph. This mapping is recorded and used in the last stage of the algorithm to convert each node back into a tile. The dimensions Δ_x and Δ_y of the tiles are set in advance and affect the performance of the algorithm, as described in Section II-H. Finer tiling produces smoother shapes and a smaller resistance at the cost of additional runtime. Due to the irregular shape of the available space, tiles near the boundaries may be irregular in shape, as shown in Fig. 6.

The adjacent vertices in the equivalent graph are connected with edges. To mimic the electrical behavior of the rail, the weight of the edges is proportional to the conductance between adjacent tiles. An accurate estimate of the resistance between arbitrary shapes requires computationally expensive methods, such as the finite element method [24]. For routing, however, a more efficient heuristic is proposed. The conductance of each edge is proportional to the width of the contact between two corresponding tiles. For example, the conductance between tiles A and B in Fig. 7 is twice larger than the conductance between tiles B and C due to the wider contact.

C. Seed Subgraph

Once the available space is converted into an equivalent graph Γ_n, the power routing problem is transformed into finding the subgraph $\Gamma^s_n \in \Gamma_n$ connecting the terminal nodes such that the resistance between terminals is minimized. The order of the subgraph $|V^s_n|$ is limited by the preset area constraint $A_{\max}$. In SPROUT, the routing process commences with determining the initial connection between the source and target terminals, as described in Algorithm 2. The location of the source and target terminals is supplied externally as a set $T_n = \{t^1_n, \ldots, t^k_n\}$. Efficient routing algorithms exist to determine the shortest path, such as Dijkstra [25] and Bellman–Ford [26]. This seed subgraph is iteratively improved using SmartGrow and SmartRefine algorithms, as described in, respectively, Sections II-D and II-E.

To generate the seed subgraph, the shortest path is determined for each pair of nodes, as shown in Fig. 8(a). The resulting subgraph is directly passed to the SmartGrow algorithm. To accelerate convergence, however, the nodes located within the boundary of the seed are added to the subgraph, producing a subgraph without voids, as illustrated in Fig. 8(b).

D. Growth Stage

The seed subgraph typically exhibits high resistance. The impedance of the subgraph can be lowered by increasing the order $|V^s_n|$ of the subgraph. To identify those parts of the subgraph that benefit most from reinforcement, a node current

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

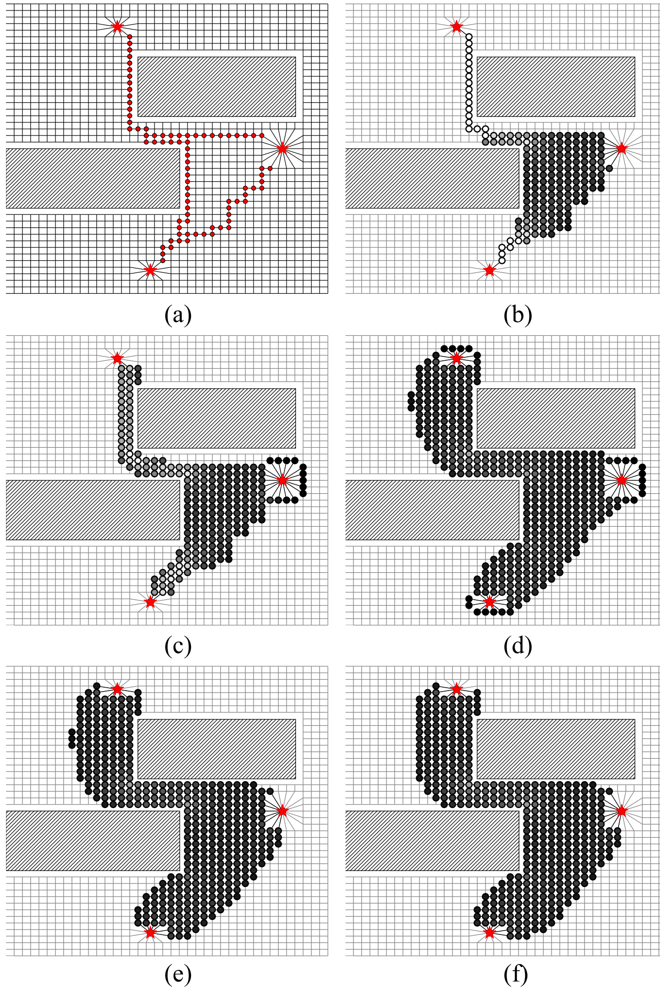


Fig. 8. Example of the graph-based routing process among three terminals. (a) Initial seed subgraph, (b) voidless subgraph after filling the internal voids, (c) initial stage of subgraph growth, and d) final stage of subgraph growth. Areas with large current are reinforced with new nodes. (e) Initial stage of the refinement process. Areas with a small current, specifically those nodes near the terminals, are replaced by nodes in the areas of current crowding, i.e., closer to the obstacles. (f) Final stage of the refinement process. The reduction in impedance is negligible, triggering termination of the algorithm.

metric is introduced here. Those regions within the subgraph with the highest node current metric indicate a high current density. Additional nodes would likely produce a significant reduction in the impedance. In contrast, those regions with a smaller node current would produce a negligible reduction in impedance, resulting in suboptimal allocation of metal resources. These low current density regions are therefore left unchanged.

The node current metric is evaluated in three stages, as illustrated in Algorithm 3. The current is initially injected into each pair of terminals. The magnitude of the current is proportional to the expected current carried by the connection. For example, those pairs of terminals with large current, e.g., between the PMIC and the BGA balls, are injected with larger current as opposed to those pairs requiring relatively smaller current, such as the connections between the BGA balls. This current injection process is expressed as a current injection matrix, $E \in R^{(|\Gamma s|-1)\times n_{\text{pairs}}}$. Each column of E corresponds to a node

Algorithm 2 Generate Voidless Seed Subgraph $\Gamma_n^s = (V_n^s \in V_n, E_n^s \in E_n)$ Such That Terminals in Set $\Theta_n \in V_n$ Are Connected

1: **procedure** SEED($\Gamma_n, \Theta_n = \{\theta_1, \ldots, \theta_k\}$)
2: $V_n^s \leftarrow \varnothing$
3: **for** each node θ_i in Θ_n
4: *paths* $\leftarrow$ SHORTESTPATH($\Gamma_n, \theta_i, \{\theta_i + 1, \ldots, \theta_k\}$)
5: Add *paths* to V_n^s and E_n^s
6: *poly* $\leftarrow$ EXTERIOR($\bigcup(V_n^s)$)
7: **for** each node v in V_n
8: **if** $v \cap poly \neq \varnothing$
9: Add v to V_n^s
10: Add edges adjacent to v to E_n^s
11: **return** $\Gamma_n^s = (V_n^s, E_n^s)$

Algorithm 3 Evaluate the Current Metric for Each Node in Subgraph Γ_n^s and Set of Terminals $\Theta_n \in V_n^s$

1: **procedure** NODECURRENT(Γ_n^s, Θ)
2: $N = |\Gamma_n^s|$
3: $[\Theta]^2 = \{\theta' \subseteq \Theta \mid |\theta'| = 2\}$
4: $N_{pairs} \leftarrow |[\Theta]^2|$
5: $L \leftarrow$ Laplacian matrix of Γ_n^s
6: $E \in \mathbb{R}^{(N-1)\times N_{pairs}}$
7: **for** each pair (s, t) in $[\Theta]^2$, $i = 1, 2, \ldots, N_{pairs}$
8: $E_{s,i} \leftarrow 1$
9: $E_{t,i} \leftarrow -1$
10: $V \leftarrow L^{-1}E$
11: $I \in \mathbb{R}^N$
12: **for** $p \in \Gamma_n^s$
13: $I_p \leftarrow \sum_{i=1}^{N_{pairs}} \sum_{j\in N(\Gamma_n^s,i)} g_p j|V_i - V_j|$
14: **return** I

within the subgraph. All entries in E are zero except the two nodes where current i is injected. The value of these currents is, respectively, $+i$ and $-i$. The voltage distribution for each current injection is determined using nodal analysis

$$V = L^{-1}E \tag{3}$$

where L is a grounded Laplacian matrix. The current within each edge is determined by multiplying the voltage matrix V by the weighted directed incidence matrix B of subgraph Γ_n^s

$$I = BV = BL^{-1}E. \tag{4}$$

The total current carried by an edge is the sum of the absolute value of the current for each pair of terminals. The node current is the sum of the total current in the adjacent edges. Thus, those nodes adjacent to the edges carrying large current exhibit a large node current.

The subgraph growth procedure is described in Algorithm 4. The boundary of subgraph Γ_n^s is defined as C, a set of nodes in Γ_n adjacent but not belonging to Γ_n^s. The nodes in C adjacent to the nodes in Γ_n^s with the highest current are added to the subgraph along with the corresponding edges. This process is iteratively repeated until the area limit $A_{\max}$ is reached. Therefore, regions with high current are reinforced whereas

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

Algorithm 4 Given Available Space Graph Γ_n, Seed Subgraph Γ_n^s, and Set of Terminals $\Theta \in V_n^s$, Add k Nodes From Γ_n to Γ_n^s to Reduce the Impedance of the Subgraph

1: **procedure** SMARTGROW(Γ_n, Γ_n^s, Θ, k)
2: $V_n^c \leftarrow V_n \setminus V_n^s$
3: $[\Theta]^2 = \{\theta' \subseteq \Theta \mid |\theta'| = 2\}$
4: $N_{pairs} \leftarrow |[\Theta]^2|$
5: $I \leftarrow$ NODECURRENT(Γ_n^s, Θ)
6: $I^c \in \mathbb{R}^{|V_n^c|}$
7: **for** $p \in V_n^c$
8: $I_p^c \leftarrow \sum_{j \in N(\Gamma_n, p), j \in \Gamma_n^s} I_j|$
9: **for** $i = 1, 2, \ldots, k$
10: $m \leftarrow \{c \mid I_c = max(I)\}$
11: $V_n^s \leftarrow V_n^s \cup m$
12: $I \leftarrow I \setminus m$
13: $\Gamma_n^s \leftarrow G_n[V_n^s]$
14: **return** Γ_n^s

Algorithm 5 Given Available Space Graph Γ_n, Subgraph Γ_n^s, and Set of Terminals $\Theta \in V_n^s$, Replace k Nodes in Γ_n^s by k Nodes From Γ_n to Reduce the Impedance of the Subgraph

1: **procedure** SMARTREFINE(Γ_n, Γ_n^s, Θ, k)
2: $I \leftarrow$ NODECURRENT(Γ_n^s, Θ)
3: **for** $i = 1, 2, \ldots, k$
4: $m \leftarrow \{c \mid I_c = min(I)\}$
5: $V_n^s \leftarrow V_n^s \setminus m$
6: $I \leftarrow I \setminus m$
7: $\Gamma_n^s \leftarrow$ SMARTGROW(Γ_n, $\Gamma_n[V_n^s]$, Θ, k)
8: **return** Γ_n^s

those areas with smaller current are left unchanged, maximizing the reduction in resistance per unit of added metal. To illustrate this process, an example seed subgraph is shown in Fig. 8(b). Brighter nodes correspond to nodes with high current, whereas the darker nodes represent nodes with a small current. In the next iteration, the brighter zones are reinforced, leading to a reduction in the impedance in that region [see Fig. 8(c)]. Further iterations reinforce the brightest zones, increasing the conductance until the target area is reached [see Fig. 8(d)].

E. Refinement Stage

Due to the area constraint, the growth process cannot continue indefinitely. Further lowering of the subgraph impedance is however possible without increasing the area using the SmartRefine procedure described in Algorithm 5. The areas with the largest and smallest current are identified using the node current metric described in the previous section. Those nodes conducting the smallest current are removed without exhibiting a significant effect on the impedance. Using the vacated metal, those regions carrying large current are reinforced, further reducing the subgraph impedance. This process is illustrated in Figs. 8(d)–(f). The nodes behind the terminals in Fig. 8(d) carry a smaller current than the rest of the subgraph. These nodes are removed and replaced by the nodes near the blockages with greater node current.

The SmartRefine process can be viewed as moving nodes from quiescent zones to hot spots. The number of nodes removed per iteration is a design variable. Removing additional nodes during each iteration would initially converge faster. At later stages of the refinement process, however, the subgraph is close to being locally optimal; excessive movement would possibly increase the impedance. Moving fewer nodes at later stages of the refinement process would therefore yield a lower impedance.

F. Subgraph Reheating

The graph-based power routing problem can be viewed as an optimization problem

$$\textbf{Minimize} : R\left(\Gamma_n^s, \Theta_n\right) \textbf{ s.t.} : A(\Gamma_n) \leq A_{\max}. \quad (5)$$

From an optimization perspective, the SmartGrow and SmartRefine procedures are a form of gradient descent. The resistance of the subgraph is the objective function, and the node current metric is a proxy metric for the gradient of the objective function. These algorithms are, therefore, a form of local optimization where the result is not guaranteed to be a global minimum. To mitigate this issue, the subgraph reheating technique is presented in this section, inspired by the simulated annealing algorithm [27] where the objective function can temporarily increase to explore the design space.

The reheating process consists of two operations: 1) dilation and 2) erosion, inspired by image processing operations. Initially, the subgraph is dilated beyond the area constraint by adding nodes adjacent to the subgraph. After completing the dilation operation, the erosion process commences. Using the node current metric, those nodes with the smallest current are removed from the subgraph, eliminating any redundant nodes while reinforcing the hot spots. The number of dilation iterations determines the extent to which the search space is explored. Additional iterations would explore a wider space while requiring greater runtime for the subsequent erosion process.

G. Back Conversion

Once the reheating process is complete, the resulting subgraph is converted back into a polygon. Recall that each node within the graph Γ_n is associated with a tile within the available space. The subgraph Γ_n^s, therefore, corresponds to a polygon comprised of multiple merged tiles. A typical PCB consists of several nets. Thus, it is crucial to remove the routed polygon from the available space of other nets.

H. Algorithm Runtime Analysis

The runtime of the algorithm depends upon a multitude of parameters, including the number of terminals, grain size, and the size of the available physical space. The first stage of the algorithm is the available space. Modern polygon clipping algorithms exhibit linear complexity with the number of vertices [28]. The PCB layout may contain more than many hundred thousands of vertices [29]. An early PCB prototype, however, contains much fewer vertices due to the fewer polygons and simpler geometry. In the case studies presented in

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

Section III, fewer than 10,000 vertices are processed, requiring up to 50 s for six power rails.

The complexity of the Dijkstra shortest path algorithm is $O((|V_n| + |E_n|)\log|V_n|)$, where V_n and E_n are sets of, respectively, nodes and edges of Γ_n. Due to the rectangular tiling of the available space, the number of edges is approximately twice larger than the number of vertices, yielding

$$O((|V_n| + 2|V_n|)\log|V_n|) = O(|V_n|\log|V_n|). \tag{6}$$

The complexity can be improved by employing alternative algorithms such as A-star [30], which utilizes the location of the nodes to accelerate the search process. The complexity of the Dijkstra algorithm, however, is smaller than the complexity of subsequent stages, namely, SmartGrow and SmartRefine. In the case studies, finding the shortest path between all pairs of nodes requires negligible time. Thus, accelerating the shortest path algorithm yields only a marginal improvement in computational performance.

The SmartGrow and SmartRefine algorithms both require computation of the voltages within the graph. These processes require the node current metric to be iteratively computed, requiring a solution of the matrix equation. This step is the main bottleneck of the algorithm, requiring up to 90% of the total runtime. Using sparse linear equation solvers, the complexity of solving a linear equation is $O(|V|^q)$, where $q \in [1.5, 3]$ is the scaling exponent which equals 1.5 in the best case and 3.0 in the worst case [31]. Both SmartGrow and SmartRefine solve a single linear equation per iteration. Thus, the runtime for SmartGrow stage T_g is

$$T_g = c_g \sum_{i=0}^{k_g-1} \left(|V_n^s| - i\Delta V\right)^q \tag{7}$$

where k_g is the number of growth iterations, ΔV is the number of nodes added per iteration, and c_g is the proportionality coefficient. The number of iterations k_g during the growth stage is approximately

$$k_g \approx \frac{A_{\max}}{\Delta A} \tag{8}$$

where $A_{\max}$ is the area of the resulting polygon, and ΔA is the area added to the subgraph during each iteration of SmartGrow. Similarly, the runtime for SmartRefine stage T_r is

$$T_r = c_r k_r \left|V_n^s\right|^q \tag{9}$$

where c_r is the proportionality coefficient.

The reheating process exhibits a complexity similar to SmartGrow and SmartRefine. The dilation process requires negligible time as compared to the erosion process which requires the node current metric to be evaluated. The runtime T_e required to apply erosion to a dilated subgraph is

$$T_e = c_e \sum_{i=0}^{k_e-1} \left(c_d\left|V_n^s\right| - i\Delta V\right)^q \tag{10}$$

where $c_d|V_n^s|$ is the number of nodes after the dilation process, c_e is the proportionality coefficient, ΔV is the reduction in order of the subgraph per iteration, and k_e is the number of erosion iterations

$$k_e = \left\lceil \frac{|V|_d - \left|V_n^s\right|}{\Delta V} \right\rceil. \tag{11}$$

The back conversion process reconstructs a set of polygons from the resulting subgraph. The polygons corresponding to each node are iteratively merged using the union operation, exhibiting $O(N\log(N))$ complexity for N vertices. In the worst case, the number of vertices grows linearly with each converted node, yielding a worst case complexity $O(|V_n^s|(|V_n^s| - 1)) = O(|V_n^s|^2)$. Practically, however, the union of multiple tiles often yields the same number of vertices. For example, the union of tiles A and B, shown in Fig. 7, has the same number of vertices as tile B. The complexity of the back conversion process is therefore between $O(|V_n^s|)$ and $O(|V_n^s|^2)$.

Greater complexity occurs when the node current metric is evaluated, namely, during the SmartGrow, SmartRefine, and erosion procedures. Combining (7), (9), and (10) yields a complexity of approximately

$$O\left(\left(\frac{A_{\max}}{\Delta A} + k_r + k_e\right)|V_n^s|^q\right). \tag{12}$$

The number of nodes $|V_n^s|$ is approximately

$$\left|V_n^s\right| \approx \frac{A_{\max}}{\Delta x \Delta y}. \tag{13}$$

The complexity is

$$O\left(\frac{A_{\max}}{\Delta A} + k_r + k_e\right)\left(\frac{A_{\max}}{\Delta x \Delta y}\right)^q. \tag{14}$$

Therefore, to reduce the computational time, the tile size and incremental increase in the area during the growth stage should be increased, while the number of refinement and erosion iterations should be reduced.

III. Validation of Case Study

Three practical case studies are presented in this section to demonstrate the validity of the proposed tool. In the first case, as described in Section III-A, a layout of a portion of the PCB between the PMIC and the two groups of vias is synthesized. In the second case, as described in Section III-B, the connections among the PMIC, capacitor, and a congested group of vias are established for the six nets. An example of PCB resource planning using SPROUT is described in Section III-C.

A. Two Rail System

A part of an eight-layer PCB for an industrial wireless application is shown in Fig. 9(a). The PMIC is placed at the bottom layer and provides power to the two power rails, V_{DD1} and V_{DD2}, and the corresponding BGA balls at the top layer. The power rails connect the PMIC inductor to the group of BGA vias on the penultimate (seventh) layer. Dedicated ground planes are placed in layers two, six, and eight.

The manually generated layout is shown in Fig. 9(b), and the synthesized layout using SPROUT is shown in Fig. 9(c).

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

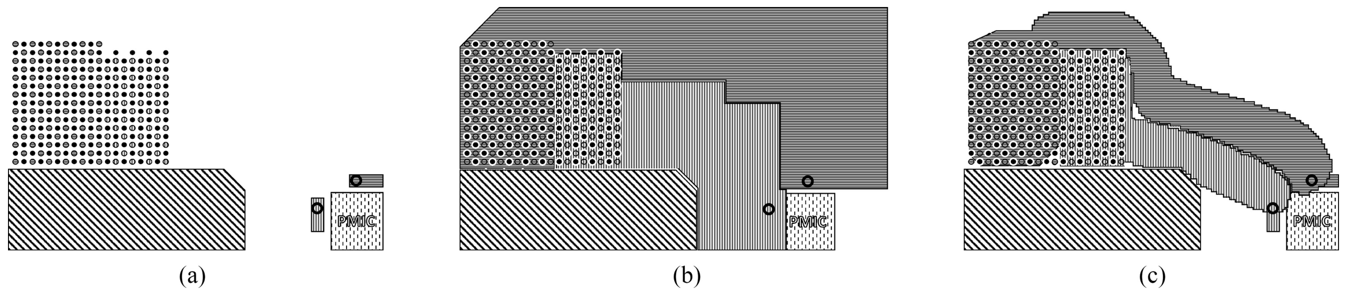


Fig. 9. Automated power routing using SPROUT and manual routing. (a) Initial layout with blockage (diagonal hatch), and two rails, V_{DD1} (dark horizontal hatch) and V_{DD2} (light vertical hatch). A single PMIC supplies power to the rails using two inductors at bottom layer 8. The inductors are connected to routing layer 7 using a via. Any blockage is shaded with a diagonal pattern. (b) Manually routed layout. (c) Layout synthesized using SPROUT.

TABLE II
COMPARISON OF NORMALIZED IMPEDANCE BETWEEN SPROUT AND MANUAL ROUTING FOR THE TWO RAIL SYSTEM SHOWN IN FIG. 9

	Net	Manual	SPROUT
Normalized inductance @ 25 MHz (picohenrys)	V_{DD1}	100	87.5
	V_{DD2}	136	138
Normalized DC resistance (milliohms)	V_{DD1}	10.0	10.1
	V_{DD2}	12.7	13.1

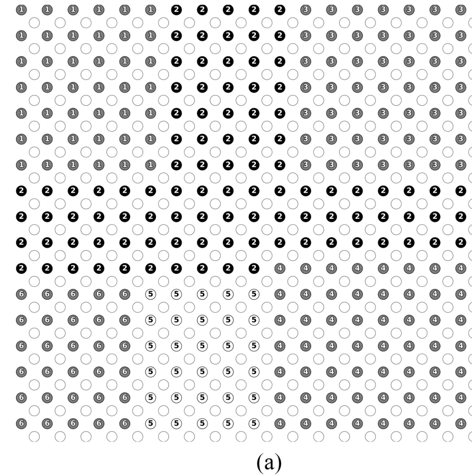
Note that regular geometries are utilized primarily in the manual layout whereas the automatically generated layout exhibits greater diversity in the shape of the geometries. The impedance of the layouts is extracted using a commercial parasitic extraction tool and compared in Table II. The two layouts (manual and synthesized) exhibit similar impedance characteristics. The difference in resistance does not exceed 3.1%. The inductance of rail V_{DD1} is reduced by 12% by using SPROUT, whereas the inductance of rail V_{DD2} is increased by 1.47%.

B. Six Rail System

In this case study, SPROUT is applied to a congested BGA arrangement, as shown in Fig. 10(a). 612 BGA (306 BGA for six power supply nets and 306 BGA for ground) are located at the top layer, and two PMICs are located in the bottom layer of a ten-layer PCB. Each PMIC regulates the current for the three voltage domains. Layers four, six, and eight are used for ground routing, and the power rails are routed on the ninth layer.

The power supply rails are routed and compared to the manual layout. The resulting topologies are shown in Figs. 10(b) and (c). Note the visual similarity between the layouts. The dc resistance and loop inductance of each rail are listed in Table III. The loop inductance of the rails generated by SPROUT is 1%–4% smaller than the manual layout while the difference in dc resistance is below 11%.

The six rail PCB layout is synthesized in approximately 11 min using an Intel Core i7-67003.40-GHz eight-core computer. Although the manual layout time varies with expertise and software, the typical time for manual layout is significantly greater than the time required by SPROUT. Furthermore, after setup, SPROUT does not require active human involvement, providing additional reductions in time and labor.



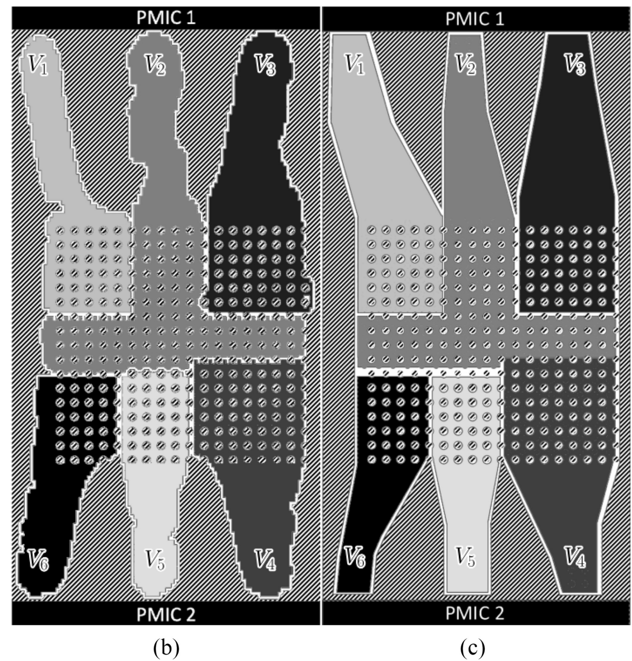


Fig. 10. Comparison between the automated power routed layout using SPROUT and manually routed layout. (a) BGA placement. The numbers indicate the net of the vias; vias without number are ground vias. (b) Layout synthesized using SPROUT and (c) manual layout. The routing layer is filled with ground metal shown with diagonal hatch.

C. Area/Impedance Tradeoff

The quality of the power network can directly influence the performance of an integrated circuit. Noise in power networks produces fluctuations in the load voltage. An

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

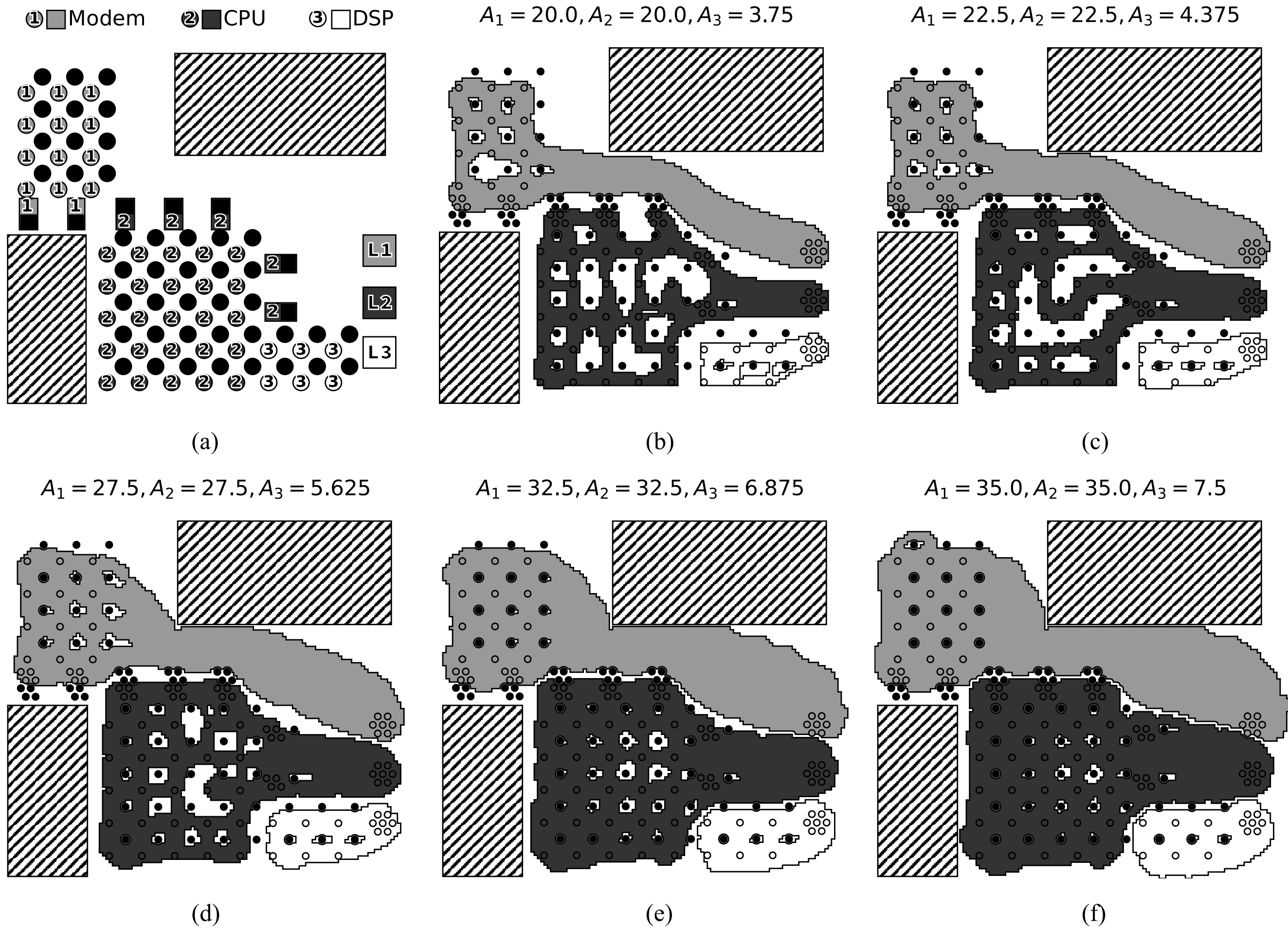


Fig. 11. Layout generated using SPROUT for three rails, modem (top left), CPU (center), and DSP module (bottom right), for varying metal area. (a) Initial BGA arrangement. The numbers within the circles indicate the nets. The vias for the ground net are solid black. The hatched rectangles represent the blockages. The size of the vias is intentionally exaggerated to show the nets, (b) $a_{\text{modem}} = 20.0$, $a_{\text{CPU}} = 20.0$, and $a_{\text{DSP}} = 3.75$, (c) $a_{\text{modem}} = 22.5$, $a_{\text{CPU}} = 22.5$, and $a_{\text{DSP}} = 4.38$, (d) $a_{\text{modem}} = 27.5$, $a_{\text{CPU}} = 27.5$, and $a_{\text{DSP}} = 5.62$, (e) $a_{\text{modem}} = 32.5$, $a_{\text{CPU}} = 32.5$, and $a_{\text{DSP}} = 6.88$, and (f) $a_{\text{modem}} = 35.0$, $a_{\text{CPU}} = 35.0$, and $a_{\text{DSP}} = 7.50$. Area is normalized.

TABLE III
Comparison of Normalized Impedance Between SPROUT and Manual Routing for the Six Rail System Shown in Fig. 10

	Net	Manual	SPROUT
Normalized inductance @ 25 MHz (picohenrys)	V_{DD1}	133	131
	V_2	103	99
	V_3	131	127
	V_4	161	155
	V_5	152	150
	V_6	116	114
Normalized DC resistance (milliohms)	V_{DD1}	15.0	16.8
	V_2	8.4	9.1
	V_3	13.0	14.2
	V_4	18.4	18.2
	V_5	18.5	18.9
	V_6	9.2	9.2

excessive reduction in the load voltage increases the delay of the transistors, degrading system performance. Variations in the load voltage are typically mitigated with voltage guard bands [32], [33] or with timing guard bands [34]. Additional voltage produces an unnecessary increase in power consumption [33]. The timing guard directly degrades the performance of the system by reducing the maximum clock frequency. Improving the power network helps reduce voltage fluctuations and, consequently, the power and performance of the system. Additional resources are however required, such as metal and layout area, as well as decoupling capacitors and voltage regulators. A complex tradeoff, therefore, exists among the system performance, power consumption, and cost.

With the ability to efficiently prototype and evaluate a power network, design tradeoffs can be extensively explored. In this case study, the relationship among the area, impedance, and load voltage is investigated in an industrial PCB. Modem, CPU, and DSP power supply nets are routed within a ten-layer board containing 86 BGA, as illustrated in Fig. 11(a). Two and five decoupling capacitors are placed at the bottom layer of, respectively, the modem and CPU rails. To determine the effects of the additional metal area on the parasitic impedance, nine PCB layout prototypes are generated using SPROUT. The area of the power rails in each prototype is summarized in Table IV. The current demand of each rail is uniformly distributed within the BGA. The modem and CPU are provided with, respectively, two and five decoupling capacitors.

With greater area, the impedance is reduced while increasing the cost of the PCB. To explore this tradeoff, nine layouts with different area for the power rails are generated using SPROUT.

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

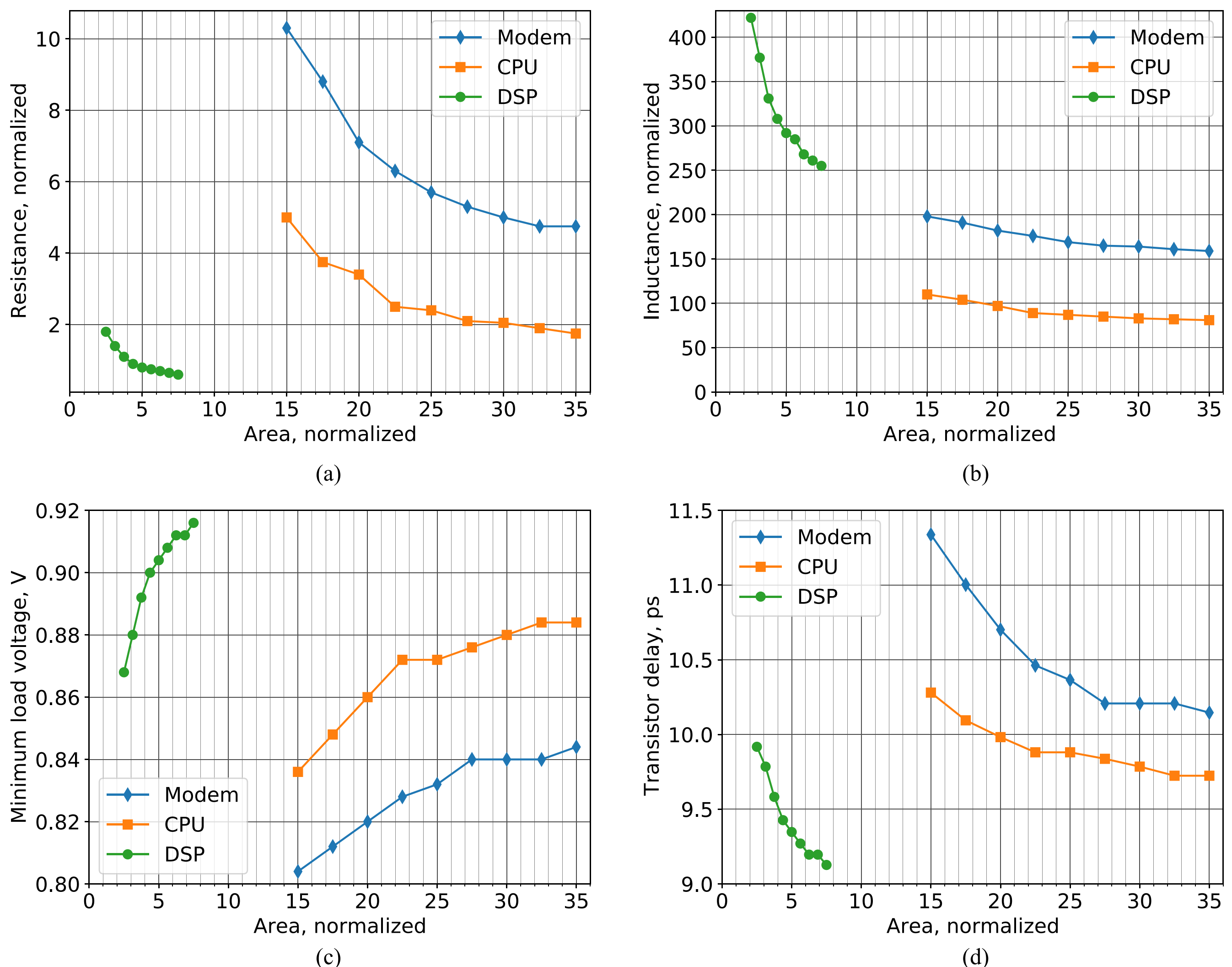


Fig. 12. Parasitic impedance of PCB rails as a function of area. (a) Effective resistance. (b) Effective inductance. (c) Minimum load voltage. (d) Transistor propagation delay.

TABLE IV
TARGET AREA OF THE TEST LAYOUTS FOR EXPLORING AREA IMPEDANCE TRADEOFFS

Layout #	Modem	CPU	DSP
1	15	15	2.5
2	17.5	17.5	3.125
3	20	20	3.75
4	22.5	22.5	4.375
5	25	25	5
6	27.5	27.5	5.625
7	30	30	6.25
8	32.5	32.5	6.875
9	35	35	7.5

Examples of these layouts are shown in Figs. 11(b)–(f). Note that with a smaller area, the BGA is connected while leaving large voids to satisfy the target area. In contrast, the larger area produces congestion due to a lack of space. The relationship between the area allocated to each rail and the impedance is shown in Figs. 12(a) and (b). The resistance of the rails is significantly reduced with the additional area. The rate of reduction, however, diminishes with a larger area. The inductance of the DSP rail exhibits similar behavior. The inductance of the modem and CPU rails is, however, not significantly reduced due to the decoupling capacitors.

The supply voltage for each rail is 1 V. The minimum load voltage is shown in Fig. 12(c). Despite the greater inductance, the voltage drop in the DSP power rail is significantly less due to the smaller load current. In contrast, the voltage drop in the modem and CPU rails is significantly larger due to the greater load current and current slew rate. Note that the voltage drop in the modem with an area of 27.5 units does not significantly decrease. The blockages likely prevent adding metal to those regions with a high current density, impeding any increase in load voltage. A similar trend is observed in the CPU rail. Beyond 22.5 units, the linear reduction in the voltage drop with area significantly slows, requiring additional metal to produce a similar gain in conductance.

The effect of the reduction in power delivery noise on performance depends upon several factors, such as the quality of the package and PCB power networks and the system and device temperature. The gain in performance can be approximated using guidelines characterizing a specific technology. Guidelines for a 32-nm FinFET technology [35] are used in this case study. The estimated propagation delay is shown in

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

Algorithm 6 Given Available Space for Net n, $A_n = \{A_n^1, A_n^2, \ldots, A_n^L\}$, on Layers 1 to L, Routing Terminals $T_n = \{t_1^{l_1}, \ldots, t_k^{l_k}\}$, and via Pitch p_{via}, Determine Least Expensive Multilayer Path Between Terminals

1: **procedure** MULTILAYER(A_n,T_n,r_{via},w_{via})
2: $\Gamma_n^{3D} = (V_n^{3D} = \varnothing, E_n^{3D} = \varnothing)$
3: **for** $l = 1, 2, \ldots, L$
4: $\Gamma_n^l = (V_n^l, E_n^l) \leftarrow$ SPACETOGRAPH(A_n^l,$\Delta x = \Delta y = r_{via}$)
5: **FOR** EACH TERMINAL $t_i^{l_i}$ IN T_n
6: **IF** $l_i = l$
7: Γ_n^l, Θ_l ←IDENTIFYTERMINALS (Γ_n^l, $t_i^{l_i}$)
8: $V_n^{3D} \leftarrow V_n^{3D} \cup V_n^l$
9: $E_n^{3D} \leftarrow E_n^{3D} \cup E_n^l$
10: **FOR** EACH VERTEX v IN Γ_n^l
11: **IF** NODE v^{l-1} EXISTS
12: $E_n^{3D} \leftarrow E_n^{3D} \cup \{v^l, v^{l-1}, w = w_{via}\}$
paths ← SHORTESTPATH (Γ_n, θ_i, $\{\theta_i + 1, \ldots, \theta_k\}$)
13: **FOR** $e = \{v_i, v_j\}$ IN *paths*
14: $\Theta_i \leftarrow v_i$
15: $\Theta_j \leftarrow v_j$
16: **RETURN** $\Theta = \Theta_1, \Theta_2, \ldots, \Theta_L$

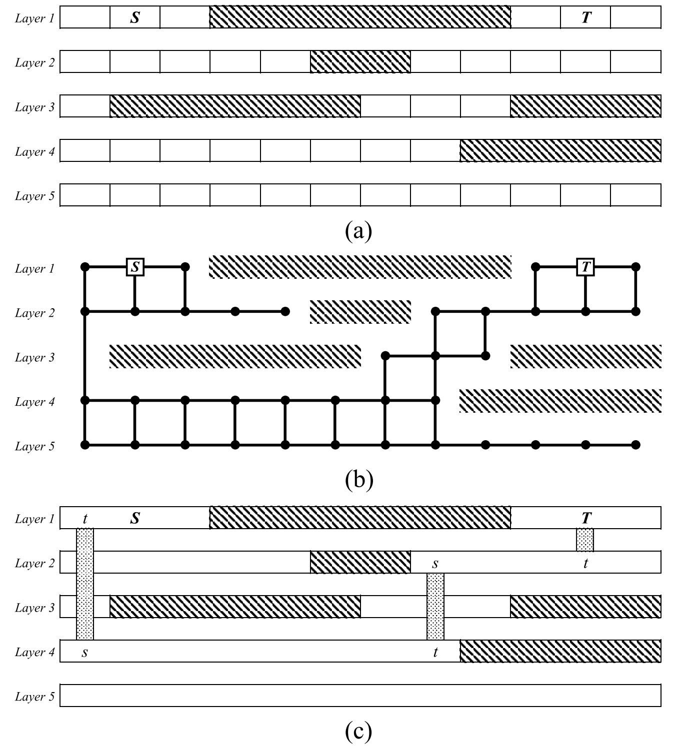


Fig. 13. Cross-sectional view of the multilayer routing process. Prohibited areas are filled with a diagonal pattern. (a) Available space is determined at each layer. Routing between the source (S) and target (T) is not possible within a single layer. (b) Equivalent graph showing potential via locations. (c) Via placement. The routing process is decomposed into three single-layer routing steps between the local source s and target t.

Fig. 12(d). Increasing the DSP rail area from 3.75 units to 7.5 units produces a 36-mV increase in the minimum voltage. A higher load voltage translates into a 7% reduction in the propagation delay of the transistors. Alternatively, a 36-mV reduction in the power supply voltage produces a 7% reduction in dynamic power.

IV. Conclusion

The power network design process at the board level is highly influenced by system-level parameters, such as the BGA pattern, layer specifications, and placement of the individual components. Changing a floorplan if a target impedance is not satisfied significantly degrades the speed of the development process. To increase the likelihood of satisfying target design objectives, system-level parameters are evaluated to determine appropriate tradeoffs among power, performance, and design time. To accelerate this evaluation process, SPROUT, an automated routing algorithm for power network exploration and prototyping, is introduced here. Based on the node current metric also introduced in this article, a layout of a power network suitable for impedance extraction is automatically synthesized.

The primary contribution of SPROUT is the automation of layout prototypes, enhancing the exploration of the design space. As compared to manual layouts, automated synthesis requires similar time for PCB prototyping without human involvement, providing significant savings in both time and labor. The impedance of the generated layout is similar to a manual layout, achieving less than a 4% difference in the two case studies. Due to automation, a large number of layout prototypes can be analyzed. By providing greater insight into the layout during early stages of the design process, system parameters can be accurately determined, reducing the likelihood of not satisfying target impedance objectives. The tool is demonstrated on two industrial applications. In addition, area/impedance tradeoffs are explored for a three rail PCB layout. The trends revealed in this case study illustrate the potential of automated exploration. SPROUT enables fast PCB prototyping and provides valuable information on design tradeoffs. For example, increasing the area of a modem rail beyond 27.5 units is not likely to yield a lower impedance.

SPROUT is the first board-level automated layout prototyping tool. Further development of the tool is possible using novel techniques, such as neural networks and evolutionary optimization, enabling faster synthesis of power network layouts with superior impedance characteristics.

Appendix
Multilayer Routing Algorithm

If a routing path between terminals is not possible in a single layer due to the space being disjoint, a routing path can be allocated utilizing vias to connect the different layers. The routing process is decomposed into two parts. The layers through which a routing path is possible are initially determined. Due to the relatively high cost of the vias [36], the number of interlayer connections is also minimized. After the placement of the vias, the routing process is decomposed into several single-layer routing steps.

To determine the layers connecting the terminals, the routing process, described in Algorithm 6, is utilized. The available space for each layer is determined using Algorithm 1 [see Fig. 13(a)]. The available space within each layer is converted into an equivalent 2-D graph. The vertical edges connect the

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

vertices within the adjacent layers through a via. This process produces a 3-D graph Γ_n^{3D}, as shown in Fig. 13(b). The vertical edges are assigned a higher cost, as compared to those edges within the same layer, to model the higher cost of the via.

Once a 3-D graph Γ_n^{3D} is generated, the shortest path between nodes in Θ_n is determined using a shortest path algorithm, such as Dijkstra [25] or Bellman–Ford [26]. After placing vias, the routing process is separately performed on each layer, from source to via, between vias, and from via to target. Those vias utilized during the routing process between nodes in Θ_n become a terminal on the respective layer [see Fig. 13(c)]. The multilayer routing process is thereby split into several 2-D routing steps.

References

[1] I. Partin-Vaisband, R. Jakushokas, M. Popovich, A. V. Mezhiba, S. Köse, and E. G. Friedman, *On-Chip Power Delivery and Management*, 4th ed. Cham, Switzerland: Springer, 2016.

[2] K. Shringarpure *et al.*, "On finding the optimal number of decoupling capacitors by minimizing the equivalent inductance of the PCB PDN," in *Proc. IEEE Int. Symp. Electromagn. Compatibility*, Sep. 2014, pp. 218–223.

[3] C. M. Smutzer, C. K. White, C. R. Haider, and B. K. Gilbert, "Power delivery network pre-layout design planning and analysis through automated scripting," in *Proc. IEEE Workshop Signal Power Integr.*, Jun. 2019, pp. 1–4.

[4] M. Popovich, M. Sotman, A. Kolodny, and E. G. Friedman, "Effective radii of on-chip decoupling capacitors," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 7, pp. 894–907, Jul. 2008.

[5] B. Archambeault *et al.*, "Design methodology for PDN synthesis on multilayer PCBs," in *Proc. IEEE Int. Symp. Electromagn. Compat.*, Aug. 2008, pp. 1–6.

[6] J. Fan *et al.*, "Quantifying SMT decoupling capacitor placement in DC power-bus design for multilayer PCBs," *IEEE Trans. Electromagn. Compat.*, vol. 43, no. 4, pp. 588–599, Nov. 2001.

[7] T.-L. Wu, H.-H. Chuang, and T.-K. Wang, "Overview of power integrity solutions on package and PCB: Decoupling and EBG isolation," *IEEE Trans. Electromagn. Compat.*, vol. 52, no. 2, pp. 346–356, May 2010.

[8] T. Hubing, "PCB EMC design guidelines: A brief annotated list," in *Proc. IEEE Symp. Electromagn. Compat.*, Aug. 2003, pp. 34–36.

[9] J. Mohamed, T. Michalka, S. Ozbayat, and G. R. Luevano, "PDN design and sensitivity analysis using synthesized models in DDR SI/PI co-simulations," in *Proc. IEEE Electr. Design Adv. Packag. Syst. Symp.*, Dec. 2018, pp. 1–3.

[10] S. Yang *et al.*, "PCB PDN prelayout library for top-layer inductance and the equivalent model for decoupling capacitors," *IEEE Trans. Electromagn. Compat.*, vol. 60, no. 6, pp. 1898–1906, Dec. 2018.

[11] Y. S. Cao *et al.*, "Inductance extraction for PCB prelayout power integrity using PMSR method," *IEEE Trans. Electromagn. Compat.*, vol. 59, no. 4, pp. 1339–1346, Aug. 2017.

[12] A. V. Mezhiba and E. G. Friedman, "Inductive properties of high-performance power distribution grids," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 10, no. 6, pp. 762–776, Dec. 2002.

[13] B. Zhao *et al.*, "Analytical PDN voltage ripple calculation using simplified equivalent circuit model of PCB PDN," in *Proc. IEEE Symp. Electromagn. Compat. Signal Integr.*, Mar. 2015, pp. 133–138.

[14] S. Sun, D. Pommerenke, J. L. Drewniak, K. Xiao, S.-T. Chen, and T.-L. Wu, "Characterizing package/PCB PDN interactions from a full-wave finite-difference formulation," in *Proc. IEEE Int. Symp. Electromagn. Compat.*, vol. 2, Aug. 2006, pp. 550–555.

[15] M. Popovich, E. G. Friedman, R. M. Secareanu, and O. L. Hartin, "Efficient placement of distributed on-chip decoupling capacitors in nanoscale ICs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2007, pp. 811–816.

[16] H. Su, S. S. Sapatnekar, and S. R. Nassif, "Optimal decoupling capacitor sizing and placement for standard-cell layout designs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 4, pp. 428–436, Apr. 2003.

[17] V. A. Chhabria, A. B. Kahng, M. Kim, U. Mallappa, S. S. Sapatnekar, and B. Xu, "Template-based PDN synthesis in floorplan and placement using classifier and CNN techniques," in *Proc. IEEE/ACM/IEEK Asia South Pac. Design Autom. Conf.*, Jan. 2020, pp. 44–49.

[18] S. Köse and E. G. Friedman, "Effective resistance of a two layer mesh," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 58, no. 11, pp. 739–743, Nov. 2011.

[19] R. Bairamkulov and E. G. Friedman, "Effective resistance of finite two-dimensional grids based on infinity mirror technique," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 9, pp. 3224–3233, Sep. 2020.

[20] R. Bairamkulov, A. Roy, M. Nagarajan, V. Srinivas, and E. G. Friedman, "SPROUT—Smart power routing tool for board-level exploration and prototyping," in *Proc. IEEE/ACM Design Autom. Conf.*, Dec. 2021.

[21] M. Moganti, F. Ercal, C. H. Dagli, and S. Tsunekawa, "Automatic PCB inspection algorithms: A survey," *Comput. Vis. Image Understand.*, vol. 63, no. 2, pp. 287–313, Mar. 1996.

[22] G. Greiner and K. Hormann, "Efficient clipping of arbitrary polygons," *ACM Trans. Graph.*, vol. 17, no. 2, pp. 71–83, Apr. 1998.

[23] B. R. Vatti, "A generic solution to polygon clipping," *Commun. ACM*, vol. 35, no. 7, pp. 56–63, Jul. 1992.

[24] E. L. Wilson and R. E. Nickell, "Application of the finite element method to heat conduction analysis," *Nucl. Eng. Design*, vol. 4, no. 3, pp. 276–286, Oct. 1966.

[25] S. Peyer, D. Rautenbach, and J. Vygen, "A generalization of Dijkstra's shortest path algorithm with applications to VLSI routing," *J. Discr. Algorithms*, vol. 7, no. 4, pp. 377–390, Dec. 2009.

[26] S. H. Gerez, *Algorithms for VLSI Design Automation*, vol. 8. Chichester, U.K.: Wiley, 1999.

[27] P. J. M. Van Laarhoven and E. H. L. Aarts, "Simulated annealing," *Simulated Annealing: Theory and Applications*. Dordrecht, The Netherlands: Springer, 1987, pp. 7–15.

[28] Y. K. Liu, X. Q. Wang, S. Z. Bao, M. Gomboši, and B. Žalik, "An algorithm for polygon clipping, and for determining polygon intersections and unions," *Comput. Geosci.*, vol. 33, no. 5, pp. 589–598, May 2007.

[29] Z. Tang, J. Zhu, F. He, L. Feng, G. Yang, and G. Han, "Adaptive polygon simplification basing on delaunay triangulation and its application in high speed PCBs and IC packages simulation," in *Proc. IEEE Int. Conf. Microw. Technol. Comput. Electromagn.*, May 2011, pp. 253–256.

[30] W. Zeng and R. L. Church, "Finding shortest paths on real road networks: The case for A-star," *Int. J. Geograph. Inf. Sci.*, vol. 23, no. 4, pp. 531–543, Jun. 2009.

[31] A. George and E. Ng, "On the complexity of sparse QR and LU factorization of finite-element matrices," *SIAM J. Sci. Stat. Comput.*, vol. 9, no. 5, pp. 849–861, Sep. 1988.

[32] M. Cho *et al.*, "Postsilicon voltage guard-band reduction in a 22 nm graphics execution core using adaptive voltage scaling and dynamic power gating," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 50–63, Jan. 2017.

[33] J. Leng, Y. Zu, M. Rhu, M. Gupta, and V. J. Reddi, "GPUVolt: Modeling and characterizing voltage noise in GPU architectures," in *Proc. ACM/IEEE Int. Symp. Low Power Electron. Design*, Aug. 2014, pp. 141–146.

[34] H. Amrouch, S. Salamin, G. Pahwa, A. D. Gaidhane, J. Henkel, and Y. S. Chauhan, "Unveiling the impact of IR-drop on performance gain in NCFET-based processors," *IEEE Trans. Electron Devices*, vol. 66, no. 7, pp. 3215–3223, Jul. 2019.

[35] S. A. Tawfik and V. Kursun, "FinFET technology development guidelines for higher performance, lower power, and stronger resilience to parameter variations," in *Proc. IEEE Int. Midwest Symp. Circuits Syst.*, Aug. 2009, pp. 431–434.

[36] Y. Xu, Y. Zhang, and C. Chu, "FastRoute 4.0: Global router with efficient via minimization," in *Proc. IEEE Asia South Pac. Design Autom. Conf.*, Jan. 2009, pp. 576–581.

Rassul Bairamkulov (Graduate Student Member, IEEE) received the B.Eng. degree in electrical and electronic engineering from Nazarbayev University, Astana, Kazakhstan, in 2016, and the M.S. degree in electrical engineering from the University of Rochester, Rochester, NY, USA, in 2018, where he is currently pursuing the Ph.D. degree under the supervision of Prof. E. G. Friedman.

He was an Intern with Qualcomm Technologies, Inc., San Diego, CA, USA, in 2018 and 2020. His current research interests include power delivery network design, electronic design automation, and optimization algorithms in very large-scale integration.

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.

Abinash Roy (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in November 2004, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Chicago, Chicago, IL, USA, in May 2011.

He has been working as a Signal and Power Integrity Engineer with Qualcomm Technologies Inc., San Diego, CA, USA, since May 2011. He has published over 25 research articles in leading journals and conferences and holds several U.S. patents. His research interests include low-power ASIC, signal and power integrity, chipset power distribution network design, architecture and pathfinding, high-speed DDR, SERDES and RFIC channel design and modeling, IC package design and architecture, advanced technology development, and electronic design automation.

Mahalingam Nagarajan (Member, IEEE) received the B.S. degree in electrical engineering from the National Institute of Technology Tiruchirappalli, Tiruchirappalli, India, in 1995, and the M.S. degree from the University of Florida, Gainesville, FL, USA, in 1997.

He is a Principal Engineer with Qualcomm Corporation, San Diego, CA, USA, involved in the design of high-speed interface architecture, mixed-signal circuit design, power delivery, and signal integrity. Prior to that, he worked with Digital Equipment Corporation, Maynard, MA, USA, and Intel Corporation, Santa Clara, CA, USA, where he was involved in high-speed mixed-signal design. His current research interests include low-power and high-speed IO architectures, high-speed digital circuit, and system-technology co-optimization.

Vaishnav Srinivas (Senior Member, IEEE) received the B.Tech. degree from the Indian Institute of Technology Madras, Chennai, India, in 2000, the M.S. degree from the University of California at Los Angeles, Los Angeles, CA, USA, in 2002, and the Ph.D. degree in electrical engineering from the University of California at San Diego, La Jolla, CA, USA, in 2019.

He is a Senior Director of Engineering with Qualcomm Corporation, San Diego, CA, USA, leading a team working on low-power high-speed interface architecture, PDN, power and signal integrity, and circuit-system co-design. His current research interests include roadmapping interface and PDN technology and design, advanced PPA modeling techniques, die/PKG/PCB co-design, and hardware/firmware co-design and validation.

Eby G. Friedman (Fellow, IEEE) received the B.S. degree in electrical engineering from Lafayette College, Easton, PA, USA, in 1979, and the M.S. and Ph.D. degrees in electrical engineering from the University of California at Irvine, Irvine, CA, USA, in 1981 and 1989, respectively.

He was with Hughes Aircraft Company, Newport Beach and Carlsbad, CA, USA, from 1979 to 1991, rising to a Manager of the Signal Processing Design and Test Department, where he was responsible for the design and test of high-performance digital and analog ICs. He has been with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY, USA, since 1991, where he is a Distinguished Professor and the Director of the High-Performance VLSI/IC Design and Analysis Laboratory. He is also a Visiting Professor with the Technion—Israel Institute of Technology, Haifa, Israel. He has authored over 500 articles and book chapters and authored or edited 19 books in the fields of high-speed and low-power CMOS design techniques, 3-D design methodologies, high-speed interconnect, superconductive circuits, and the theory and application of synchronous clock and power distribution networks. He holds 23 patents. His current research and teaching interests include high-performance synchronous digital and mixed-signal circuit design and analysis with application to high-speed portable processors, low-power wireless communications, and server farms.

Dr. Friedman was a recipient of the IEEE Circuits and Systems Mac Van Valkenburg Award, the IEEE Circuits and Systems Charles A. Desoer Technical Achievement Award, the University of Rochester Graduate Teaching Award, and the College of Engineering Teaching Excellence Award. He was the Editor-in-Chief and the Chair of the Steering Committee of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS and the *Microelectronics Journal*, a Regional Editor of the *Journal of Circuits, Systems and Computers*, an editorial board member of numerous journals, and a program and technical chair of several IEEE conferences. He is a Senior Fulbright Fellow, a National Sun Yat-sen University Honorary Chair Professor, and an Inaugural Member of the UC Irvine Engineering Hall of Fame.

Authorized licensed use limited to: UNIVERSITY OF ROCHESTER. Downloaded on October 06,2022 at 18:00:30 UTC from IEEE Xplore. Restrictions apply.