A Partition-based Algorithm for Power Grid Design using Locality

Jaskirat Singh Sachin S. Sapatnekar

Department of Electrical & Computer Engineering

University of Minnesota

Minneapolis, MN 55455

{jsingh,sachin}@ece.umn.edu

Abstract

This paper presents an efficient heuristic algorithm, which employs a successive partitioning and grid refinement scheme, for designing the power distribution network of a chip. In our iterative procedure, the chip area is recursively bipartitioned, and the wire pitches and the wire widths of the power grid in the partitions are repeatedly adjusted to meet the voltage drop and current density specifications. By using the macromodels of the power grid constructed in the previous levels of partitioning, the scheme ensures that a small global power grid system is simulated in each iteration. The idea is based on the notion that due to the locality properties of the power grid, the effects of distant nodes and sources can be modeled more coarsely than nearby elements, and includes practical methods that enhance the convergence of the iterative conjugate-gradient based solution engine that is used in each step. Finally, a post-processing step at the end of the optimization is employed to maximize the alignment of wires in adjacent partitions. The effectiveness of the scheme is demonstrated by designing various power grids with real circuit parameters and realistic input current values. The proposed algorithm is able to design power grids comprising thousands of wires and more than a million electrical nodes in about 6 to 13 minutes. When compared to a multigrid-based power grid design scheme, it is found to save about 7% to 12% of wire area, and on an average is 14% faster.

I. Introduction

Increases in the operating frequency and power density, and reductions in the supply voltages, have made the design of a high performance power distribution network a challenging problem. While reductions in the widths of the metal wires have led to increased wire resistances, the currents through these wires have also drastically increased due to the increase in chip densities. As a consequence, it has become essential for the power delivery network to be carefully designed to prevent the loss of circuit performance due to IR drop and failures due to electromigration (EM).

In view of this, various schemes have been proposed for power grid design. Typically, the choices available to a supply net designer are to (i) appropriately size the supply net wires [1–4], (ii) perform topology optimization, i.e., to assign suitable pitches to the power grid wires and/or determine the optimal assignment of the pins to the pads and placement of pads on the power grid [5–9], and (iii) add decoupling capacitors [10], [11].

Schemes for automated power distribution network design can be divided into the following two categories, based on tradeoffs between the accuracy of the embedded power grid simulator and the level of sophistication of the optimizer:

(A) Heuristic iterative optimization methods, employing an explicit and exact circuit analysis step in the main optimization loop to determine constraint violations in the power grid [6], [10].

(B) Mathematical optimization schemes, formulating the problem as a nonlinear program by approximating the circuit equations, and solved with the aid of nonlinear optimization techniques [1–4], [7].

The desirable characteristic of the supply network design methods based on scheme (A) is the guaranteed accuracy of the final solution, ensured by the process of performing an explicit circuit simulation step in each iteration, to detect and fix the IR drop and EM violations. However, these methods typically have large run times as each simulation of a power network, comprising hundreds of thousands of electrical nodes, is extremely time consuming. The methods built on scheme (B), solve the supply net design problem by formulating it as an optimization problem of minimizing a nonlinear function subject to nonlinear constraints. In this scheme, typically, the circuit analysis is implicitly carried out by explicitly or implicitly listing the circuit constraints, i.e., the Kirchoff's current and voltage laws and the device equations, as a part of the constraints set. In the original form, the solution to such a nonlinear problem formulation is known to be computationally intensive which makes it prohibitive for problems involving millions of design variables. Hence, to achieve efficiency these methods typically either employ some constraint relaxations to transform a general nonlinear optimization problem to special forms of nonlinear programs such as the convex program, which can be efficiently solved, or introduce some approximations to reduce the problem size. Although, these methods provide a solution that is more efficient than those from scheme (A), the final solution is inherently subject to inaccuracies due to the relaxations and approximations introduced in the original nonlinear problem formulation.

In this paper, we propose a novel and fast, yet accurate, algorithm to design the power distribution network in the form of a non-uniform power grid. We use a hierarchical design approach, based on successive partitioning of the chip area, to design the supply network. Our procedure achieves efficiency by using the notion of *locality*, similar to that proposed in [12]. This concept is based on the observation that nearby elements have the greatest influence on the voltage at any node. Therefore, while constructing the power grid locally in a specific region of the chip, it suffices to use fine-grained and accurate models only in or near the that region. The regions of chip that are *far away* from the specific region are not likely to affect the local grid design in the specific region, and can be abstracted using coarse models.

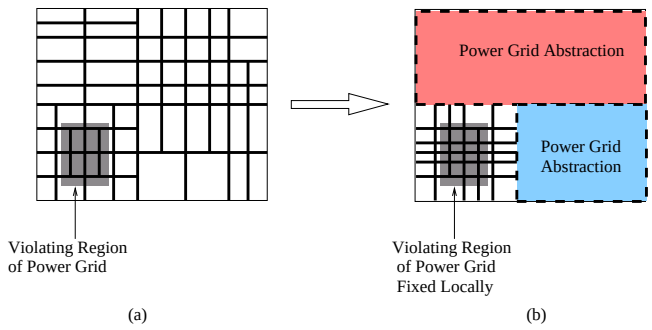


Fig. 1. The concept of locality in power grid design. (a) A detailed power grid with a violating region shown with the shaded rectangle. (b) Details of regions of power grid far from the violating region abstracted away and the violations fixed locally.

This concept of locality is illustrated in Figure 1, where a violating grid region, i.e., a region that violates the constraints, is shown by the shaded rectangle. Generally speaking, these violations can be fixed by adding more power grid wires locally in and around the violating region. Due to locality, as we make these local changes, it is reasonable for the details of parts of power grid in the regions far away from the violating region to be abstracted away, as shown in Figure 1(b). The use of these abstractions, or *macromodels*, for parts of the power grid circuit has two main advantages. Firstly, they improve the speed of the analysis, since they reduce the size of the system to be solved. Secondly, by focusing the grid design effort on the details of the local region only, the search space for choosing the optimal design parameters, e.g., the wire pitches and the wire widths, is greatly reduced.

The idea of working with detailed local models and abstractions of far away regions is especially useful in the case of flip-chip packages where the power pads are distributed throughout the chip area by using C4 bumps. For a flip-chip package, most violations of power grid in a specific regions can be fixed by locally modifying the power grid just in and around the violating regions. Due to the availability of a sufficient number of pads around the violating region, the power grid wires in the local region contain the path of least resistance for the current to flow from the nearest V_{DD} pads to the violating nodes. The same idea may not be true of chips with wire-bond packages, since the power pads are on the periphery of the chip, and the strategy of local modification of the grid may not work because of the concentration of power pads on the chip periphery. However, the hierarchical design method mitigates the effect of weak locality in grid design, as may be the case for wire-bond chips. By the process of successive bipartitioning, the hierarchical grid design procedure ensures the presence of low-resitivity conduction paths in various regions of the chip. Due to the availability of these high-conductance paths, it is reasonable to expect that a violating region of the power grid, which may or may not exhibit strong locality property, would be appropriately fixed by employing a top-down hierarchical partitioning scheme.

Based on the hierarchical design approach and the notion of locality in power grid design, we propose an efficient and accurate grid design procedure. The method employs an iterative scheme of recursively partitioning the chip area and constructing the power grid locally in the partitions by adjusting the wire pitches and the widths. The power grids constructed by our design procedure have a locally regular, globally irregular structure [6]. The grids within a partition are constructed to be uniform, i.e., they have the same wire width and wire pitch, but power grid wires in different partitions may have different widths and pitches, as determined by the current density requirements of the underlying functional blocks. Such a locally uniform, globally non-uniform power grid structure has the desirable properties of efficient wire area utilization, ease of power grid circuit modeling and optimization. The outline of our method is as follows:

- We present a heuristic algorithm to design a supply grid that meets the IR drop and the EM constraints. The optimization is carried out under DC conditions using an iterative refinement scheme. In our implementation, the power grid is designed for the top two metal layers of the chip, but the same procedure can be easily extended to design a power network spanning multiple layers of metal.
- We commence by dividing the chip area into two partitions. The power grid in the two partitions is then constructed by placing thick or very wide wires at an initial pitch. The pitches in the two partitions are then repeatedly reduced until the initial grid meets the constraints.

- In each of the subsequent iterations, a previous partition is further divided into two smaller partitions and a refined power grid is reconstructed *locally* in these smaller partitions. The solution of the grid designed in the previous iterations is used to guide the design of the power grid in the current iteration.
- We use a previously proposed macromodeling technique [13] to construct abstractions of partition of the power grid. Employing a hierarchical circuit analysis method in each iteration, to determine the nodes and branches which violate the reliability constraints, ensures the accuracy of grid design. Since, in each iteration we construct the macromodels of only two partitions, and reuse the macromodels formed in previous iterations, the analysis step is very fast.
- We improve the run time of our algorithm, proposed in our previous work [14], by further speeding up the circuit analysis step. In order to reduce the simulation time for the global matrix system, we use a preconditioned conjugate gradient based iterative linear solver, with an initial guess vector whose components are derived from the power grid solution of the previous iteration. Thus, reusing the grid solution of the previous iteration to drive the grid design in the current iteration aids in making the procedure faster.
- At the end of the optimization, a post-processing step is used to maximize the alignment of wires in adjacent partitions.

The rest of the paper is organized as follows. Section II explains the power grid circuit model used and other preliminaries for the power supply network design problem. The details of the grid design procedure is contained in Section III. Section IV contains the pseudo-code and the description of the complete power grid design algorithm. The extension of this work for power grid design spanning multiple metal layers is discussed in Section V. Sections VI and VII describe the experimental results and the conclusion, respectively.

II. Preliminaries

A power grid comprises metal wires running in the orthogonal directions and spanning multiple layers (typically, 5 to 8 for current microprocessor designs). The wires in two consecutive layers of metal are electrically connected to each other by using vias. The wires in the top-most metal layers are electrically connected to the V_{DD} pads that are located either on the peripheral power ring, as in the case for a chip with a wire-bond package, or are distributed over the entire chip area, using C4 bumps, as in the case of flip-chip package. This system of pad connections and network of metal wires carrying currents from the V_{DD} pads to the underlying gates in the functional blocks, can be modeled as an equivalent electrical circuit comprising possibly millions of nodes. Under DC conditions, as illustrated in Figure 2, the power grid can be modeled as a resistive mesh, with the pads replaced by voltage sources. As seen in the figure, the wires are replaced by their equivalent resistances and the worst-case switching activities of the gates in the underlying functional blocks determines the loading currents.

Using the circuit model as shown in Figure 1, the branch resistance r_i of branch i can be expressed as:

$$r_i = \rho_s \frac{p}{w_i} \tag{1}$$

where ρ_s is the sheet resistivity, w_i is the width of the wire segment corresponding to the branch i and p is the pitch of the power grid wires in the orthogonal direction, which is the same as the length of the wire segment. The pitch of the power grid wires could be different for different layers and in fact, may not even be constant for a given layer. After converting the

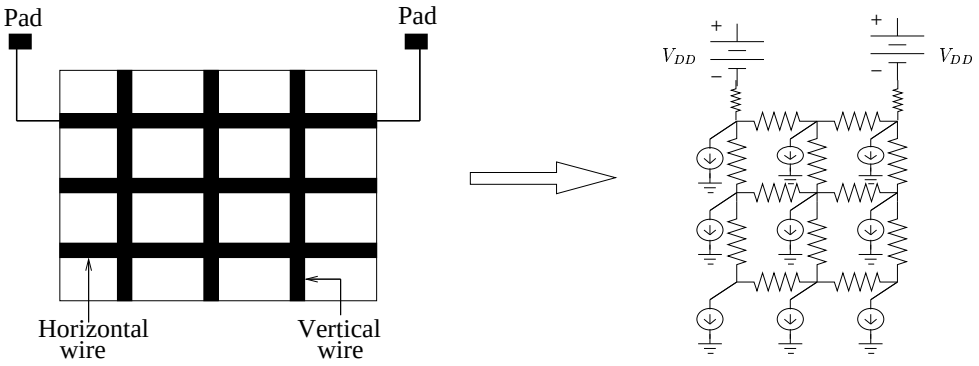


Fig. 2. A power grid and its equivalent circuit model.

voltage sources in the grid model of Figure 2 to their Norton equivalents, the solution to the node voltages of the circuit is given by the following system of Modified Nodal Analysis (MNA) equations:

$$G \cdot \mathbf{V} = \mathbf{J}, \quad \mathbf{J}, \mathbf{V} \in R^n, \quad G \in R^{n \times n} \tag{2}$$

where n is the number of electrical nodes in the power grid circuit, G is the conductance matrix which comprises stamps of all branch resistances, **V** is the vector of node voltages and **J** is a vector of load currents and the Norton currents of voltage sources.

For all nodes *j* and all branches *i* in the power grid circuit, the following constraints must be satisfied:

1) **The IR drop constraint**: $V_j > V_{spec}$
2) **The current density constraints (EM)**: $|I_i| < \sigma w_i$

The voltage of node *j* is denoted as V_j, the branch current of branch *i* is denoted as I_i and σ is the specified current density for a fixed thickness (height) of the metal layer.

In the following sections of the paper, we describe the design procedure for constructing a power grid that meets the two above constraints. The inputs to our power grid design problem are the number of power pads and their precise connection locations to the grid wires in the top layer, the placed functional blocks and an estimate of worst case currents drawn by the gates in the functional blocks. The amount of current draw by each of the functional blocks can be determined by estimation techniques such as the ones proposed in [15] and [16]. Although, these methods extract time varying current waveforms, appropriate simplifications can be made to these techniques to estimate worst case steady state currents drawn by the functional blocks. We use the power grid circuit model as shown in Figure 2. Each node j in the power grid circuit is loaded by a constant current source c_j. Given the current estimate I_{f_k} drawn by a functional block k, and the physical coordinates of the placed functional block k, the values of constant current sources loading the power grid are chosen in such a way that the sum of all current sources at node locations lying over the functional block k add up to the functional block current I_{f_k}. The values of constant current sources and the functional block currents are expressed by the following relation:

$$\sum_{j \in \{\text{Nodes over block k}\}} c_j = I_{f_k} \tag{3}$$

III. Power grid design procedure

A. Grid refinement

A power grid covering the entire chip area comprises thousands of wires, and its equivalent circuit, using the mapping shown in Figure 2, contains millions of electrical nodes, making it prohibitive to simulate. To achieve efficiency in the circuit analysis step, it thus becomes essential to work with a coarse initial power grid representation, which yields a power grid circuit of manageable size, and then iteratively refine the coarse model. In our power grid design procedure, we construct the coarse grid by using unrealistically thick power grid wires initially, and then subsequently improve the coarse grid model by a *grid refinement* operation, in which a power grid containing a smaller number of thick wires is replaced by a grid comprising of a larger number of thinner wires. The advantage of using a grid with thick wires is that it greatly reduces the system size, as there are fewer electrical nodes in the equivalent circuit model of Figure 2. Moreover, since we employ a hierarchical approach for power grid design, the coarse grid representation is adequate for the abstractions of parts of power grid circuit. However, such a grid may result in over-utilization of the chip wiring resources. Hence in our optimization procedure, we begin with a grid with very thick power grid wires to gain speed up in the circuit analysis step, and then subsequently perform grid refinement in later iterations to achieve savings in the wire area.

The effective resistance of a given region of a power grid circuit depends on both the number of wires in the region, and the resistances of the individual wires. Having more wires reduces the effective resistance of the grid by increasing the number of paths that current can take from a V_{DD} pad to any node. Assuming a constant wire width for the given region, as required for a locally regular structure of the power grid in our work, the number of wires is determined by the wire pitch of the region. On the other hand, the resistance of the individual wires depend on the widths of the wires. Figure 3 depicts the grid refinement operation. As seen in the figure, by a grid refinement operation, a coarse grid with thick wires and a large pitch, is converted to a grid of thinner wires and smaller pitch. It is reasonable to expect that the impact of an increase in the resistances of the individual wire segments of the refined power grid, due to reducing the wire widths of the coarse grid, would be offset by having more wires in the refined grid. Moreover, since the magnitude of currents carried by more thinner wires of the refined grid reduces, the wires in the new and the old grid are expected to have similar current densities. This observation that the effective resistance of a power grid circuit depends both on the wire width, and the wire pitch, forms the basis of employing grid refinement in our optimization procedure.

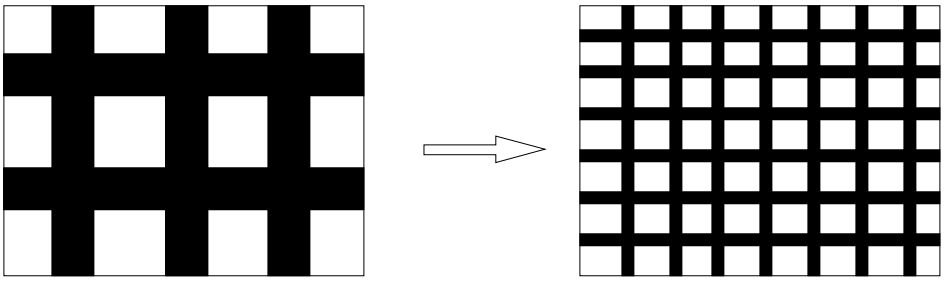


Fig. 3. A power grid with thick wires and large pitch refined to a grid with thinner wires and smaller pitch.

The grid refinement idea is similar to the multigrid based method of coarsening the power grid system [4]. However, the

method in [4] deals with a fixed topology of the supply network and optimizes the parameters of a pre-constructed power grid. As will be explained in the following sections, in our method, the optimization procedure decides the topology of the power grid by repeatedly adjusting the wire pitches and wire widths, while maintaining the locally regular, globally irregular structure of the supply network.

B. Power grid design by recursive bipartitioning

We apply the *divide and conquer* approach to our power grid design procedure by successively dividing the chip area into smaller regions or partitions and iteratively constructing power grids in the smaller partitions, using the notion of locality, introduced in Section I. The recursive bipartitioning idea employs the strategy of solving a small local power grid design problem in each step, and involves the selection of optimal pitches for two partitions, such that (i) the new grids constructed in the two partitions each meet all of their specifications, and (ii) the previously constructed grids in the other partitions maintain their correctness in terms of meeting the IR drop and EM constraints.

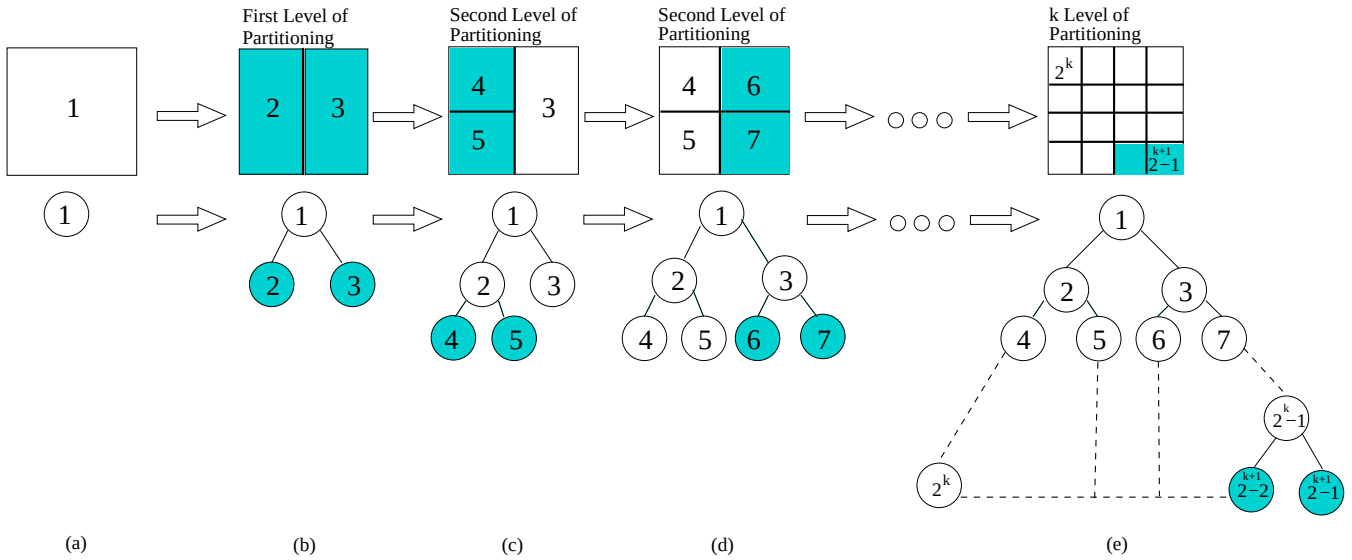


Fig. 4. The recursive bipartitioning process to design the power grid. Each partition cut is equivalent to adding two elements in the binary *partition_tree*. The height of the tree represents the level of partitioning. The two shaded elements refer to the two partitions where the new power grid is designed in the current iteration.

Figure 4 illustrates the recursive bipartitioning process. As shown in the figure, the process of partitioning the chip area to iteratively construct the power grid is equivalent to growing an *almost complete* binary tree[1] referred to as the *partition tree*. Partitions that have already been processed are represented as the non-shaded tree elements, and the two partitions being processed in the current iteration, referred to as the *active partitions*, are shown as the shaded tree elements. Each element in the partition tree contains information about the wire widths and the pitches of the power grid constructed in the corresponding partition. The height of the partition tree determines the current level of partitioning. A new partition is constructed by making a vertical (horizontal) cut, by introducing a vertical (horizontal) *partition wire* halfway across the parent partition as a part of the power grid. Consecutive levels of partitioning alternate the cut directions between vertical cuts and horizontal cuts. The

[1]As the smaller partitions are produced by the splitting the larger partition in sequence from left to right, the resulting partition tree has children added from left to right, which makes the tree an almost complete binary tree.

process of adding two children to a parent node in the tree is equivalent to the grid refinement operation described in Section III-A. In other words, when two child nodes are added to a parent node in the tree, the coarse power grid in the partition corresponding to the parent node is replaced by finer grids of the child nodes.

In the following sections we explain the idea of recursive bipartitioning for power grid design in detail.

1) First level of partitioning: As seen in Figure 4(b), the first level of partitioning is equivalent to adding two child nodes to the root of the partition tree. In this step, we begin with the full chip area and divide it into two parts by adding a vertical power grid partition wire across the chip going through its middle. We define this as a *vertical cut* across the *first partitioning level*, referred to as $Part_1$.

Figure 5 depicts the division of the chip into two partitions. Next, we select an initial wire width and pitch for the partitions such that the worst case voltage drop meets its specifications. These choices are made by constructing an initial grid in the two partitions, and then iteratively decreasing the pitches of power grid in the two partitions until there are no IR drop and EM violations. This choice of starting point for the initial wire widths and the pitches is determined empirically, or based on designer input, so that we begin not too far away from the final solution point in the search space.

The circuit analysis step, which detects constraint violations in the power grid, is performed using the macromodeling approached proposed in [13], coupled with a preconditioned conjugate gradient based iterative solver to speed up the power grid circuit simulation. A macromodel is an abstraction of a linear network and we use one that has the following transfer characteristic:

$$\mathbf{I} = A \cdot \mathbf{V_p} + \mathbf{S}, \quad \mathbf{I}, \mathbf{V_p}, \mathbf{S} \in R^m, \quad A \in R^{m \times m} \tag{4}$$

where m is the number of ports in the system, **I** is a vector of currents flowing into the system through the port nodes which are the nodes that are located on the partition boundaries and form the connection between the various partitions, V_p is the vector of voltages at the port nodes , A is the admittance matrix and **S** is a vector of currents from each port to the reference node.

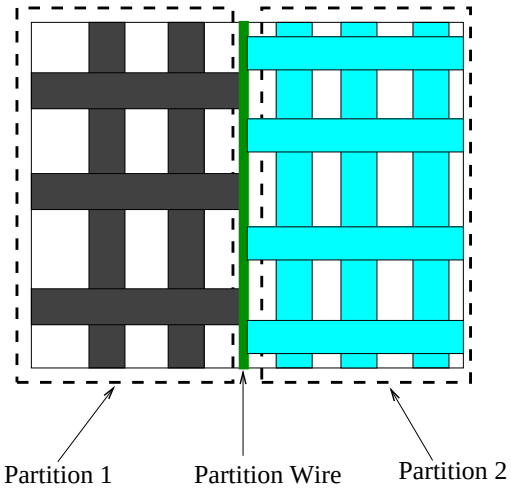


Fig. 5. Chip divided into two partitions and the power grid constructed in the two partitions.

Using the macromodeling idea, all nodes in the two partitions, except the port nodes, are abstracted away. The port nodes lie on the partition wire through which the two partitions connect to each other. Figure 6 depicts this situation, where the left and the right partitions are abstracted as macromodels connected to each other through the port nodes located on the partition wire. As seen in the figure, the power grid constructed in the two partitions is reduced to a system of two macromodels that connect to each other through the port nodes. Using the matrix algebra described in [13], we construct the macromodels given by the parameters $(A, \mathbf{S})$ for the two partitions. Following the hierarchical circuit analysis approach, the macromodel parameters are stamped in the MNA equations of the global system:

$$M \cdot \mathbf{X} \quad = \quad \mathbf{b} \tag{5}$$

where

- M is the matrix containing the conductance links between the pads and the wires in all of the partitions and the stamps of macromodel parameter A for each partition,
- **X** is the vector of voltages at the port nodes of the partitions, and
- **b** is vector containing the stamps of macromodel parameter **S** for each partition.

The global system is solved, and subsequently, the node voltages within the two partitions are determined to check for any IR drop and EM violations. As an extension to our work of [14], we improve the run time of our procedure by solving the global system given by equation (5) using an iterative solver: specifically, a conjugate gradient method with diagonal preconditioning. As will be explained in the following sections, we can reuse the solution of the power grid designed in the previous iteration to speed up the current iteration of the grid design procedure by employing an iterative linear solver to solve the global matrix of (5). In case a partition does not meet the voltage drop and current density specifications, the wire pitch of the partition is reduced by a factor β_1, and the process of creating the macromodels and solving the power grid system using the iterative linear solver is repeated. As an implementational detail, it is worth emphasizing that at this first level of partitioning, we use unrealistically thick wires, e.g., wires having widths between 80 and 150 μm, and these will later (after further partitioning) be replaced by thinner wires through the grid refinement operation. Hence, the system size of each partition is fairly small, e.g., 1000 to 3000 nodes. As a consequence, the iterative process of constructing new macromodel parameters $(A, \mathbf{S})$ after decreasing the wire pitch and simulating the grid repeatedly is extremely fast. At the end of this step, we obtain a coarse power grid constructed in the two partitions that span the entire chip area.

2) Second level of partitioning: Recall that $Part_1$ used a vertical wire that divided the chip area in the left and right halves. We use the power grid constructed at the first partitioning level to guide the grid design at the next level, referred to as $Part_2$. The second level of partitioning has two steps, as depicted in Figure 4(c) and (d), referred to as $Part_{2_1}$ and $Part_{2_2}$, respectively. First, the power grid constructed in the left partition is ripped up. Next, a horizontal cut is made in the left partition by introducing a horizontal power grid partition wire in the middle of this region. As a result, the left partition is further divided into top-left and bottom-left partitions, as shown in Figure 7. As seen in Figure 4(c), $Part_{2_1}$ begins by adding child nodes to the parent nodes 2 and 3 in the partition_tree, or equivalently, growing the almost complete binary tree to the next level. Leaving the previously constructed grid in the right partition intact, a finer grid is designed in each of the two new

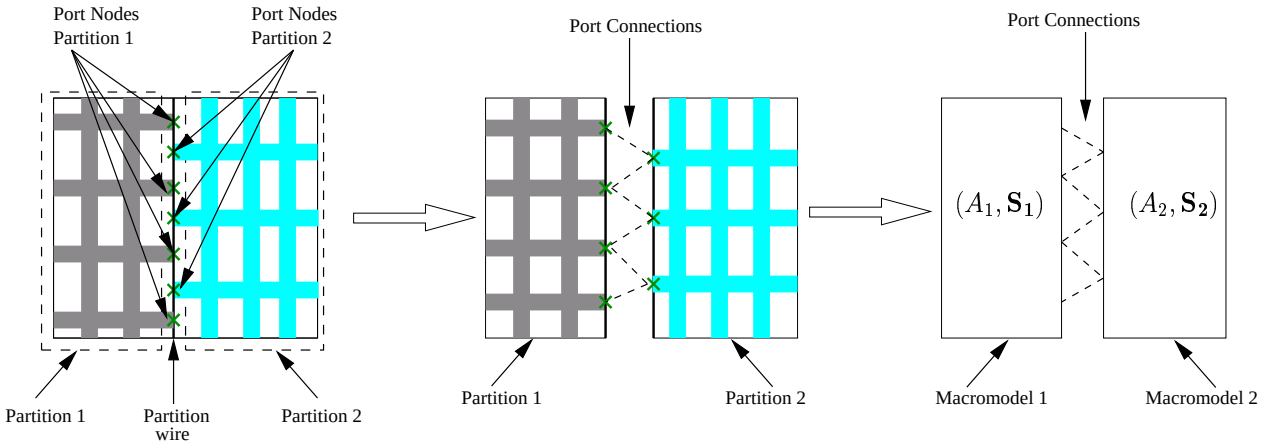


Fig. 6. Power grid constructed in the two partitions changed to a system of macromodels. The macromodels connect with each other through the port nodes on the partition wire.

partitions, i.e., the top-left and the bottom-left partitions.

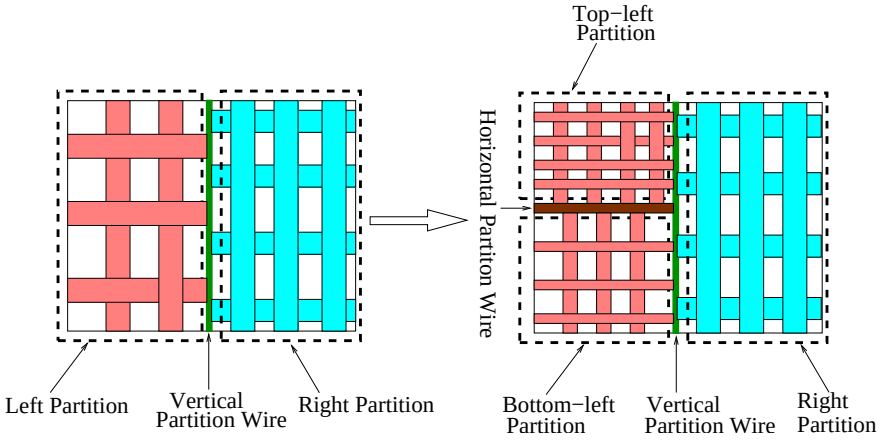


Fig. 7. The second partitioning level in the grid design procedure

The grid design procedure for these two partitions is similar to that employed in $Part_1$, where grids were constructed in the left and the right partitions. We begin the grid construction in the top-left and bottom-left partitions by using the notion of grid refinement to construct an initial grid whose wire width is chosen using a multiplicative factor, $\gamma_2 < 1$, on the wire width at the parent node, i.e., the wire width used for the grid in the left partition at the first level. In general we set, $w_k = \gamma_k \cdot w_{k-1}$, where w_k is the wire width for the k^{th} level of partitioning and γ_k is the multiplicative factor used when refining from the $(k-1)^{st}$ level of partitioning to the k^{th} level. This is precisely the grid refinement technique explained in Section III-A and shown in Figure 3, with the difference that we now also maintain a coarse macromodel for the right partition. The grid of the left partition, which had thicker wires and a larger pitch, is therefore replaced with the grids in partitions top-left and bottom-left that each have thinner wires but a smaller pitch. The macromodel parameters $(A, \mathbf{S})$ are calculated for the top-left and bottom-left partitions and they are stamped in the global system of equation (5) along with the macromodel parameters of the right partition which were previously computed in $Part_1$.

To solve the global matrix system of (5), we use a preconditioned conjugate gradient iterative solver to speed up the solution

of the linear system of equations. The simulation speed up is obtained by the observation that since the result of the grid refinement operation is designed to offer a similar effective resistance, it is reasonable to expect that the perturbation between the old system (i.e., the power grid design obtained at the end of the first level of partitioning), and the new system (i.e., the refined power grid for the left partition at the second partitioning level), would be small. Hence, we can use the voltages of port nodes located on the vertical cut wire, solved by the old global system of (5), as the components of the initial guess vector X^0 corresponding to the same ports in the new system. The other components of the guess vectorX^0, which correspond to the voltages of the newly introduced port nodes by addition of a partition wire, are set to be equal to V_{spec}. Thus, each component i of the initial guess vector used for the conjugate gradient method, after introducing each new horizontal or vertical partition wire is given by:

$$\mathbf{X_i^0} = \begin{cases} V_{port_{old}} : \text{i} \in \text{Port node of the old system} \\ V_{spec} : \text{i} \notin \text{Port node of the old system} \end{cases} \tag{6}$$

We find that reusing the solution of power grid design of the previous iteration for solving the new global matrix system improves the run time of our supply net design procedure of [14]. The guess vector as formed by equation(6) provides a fairly good starting point for the conjugate gradient method, as a result it converges to the final solution in only a few steps. To improve the conditioning of the problem, we use a diagonal or a Jacobi preconditioner for the conjugate method. We pre-multiply the global system of (5) by a diagonal matrix $\sum$ consisting of the reciprocals of the corresponding diagonal entries of the global MNA matrix M. Alternatively, we could have chosen Cholesky preconditioning, which is known to have better conditioning properties [17]. However, as the step of solving the global system is a part of the inner loop in the grid design procedure, it is much cheaper to from a diagonal preconditioner as compared to the more expensive operation of finding the Cholesky factors of the global matrix M. After solving the global matrix system and determining the port voltages from the solution vector $\mathbf{X}$, using the hierarchical analysis technique, the IR drop and EM violations are detected for the power grid in the top-left and bottom-left partitions. The violations are corrected by inserting more wires in the partitions, by reducing the pitch by a factor β_2.

In addition to the voltage drop and the current density specifications, there is an additional requirement that the grids of the top-left and bottom-left partitions must meet. It is essential that the process of ripping up the original grid of the left partition and replacing it with the grids constructed in the top-left and the bottom-left partitions does not render the grid of the right partition ineffective in terms of meeting the specifications. To inspect whether the correctness of the grid in the right partition is maintained, we could completely solve the right partition power grid again. However, this would be costly in terms of run time as we would need to check the voltages of all of the nodes of the grid within the right partition each time the pitches of the top-left or bottom-left partitions are decreased. To avoid this high simulation cost, we make use of the abstraction of the power grid of the right partition effectively. It is reasonable to expect that a very small change in port voltages of right partition would result only in a small change in the voltages of the internal nodes of the right partition. Thus, the voltages at only the port nodes of the right partition grid are evaluated. These voltages are compared with the port voltages of the right partition power gird obtained at the end of $Part_1$ and a grid violation, referred to as *previous_grid_violated*, is flagged if the

maximum change in the port voltages is greater than a specified value, MAX_{spec}. In the event of such a violation, the pitches of the top-left and bottom-left are further decreased, thereby increasing the number of wires in the power grid in order to maintain the correctness of the previously designed grid in the right partition. This process ensures that the port voltages of the previously correct grid design are not significantly disturbed by the new power gird which replaces the previous grid.

The process of power grid design in $Part_{2_2}$ is similar to that of grid design in $Part_{2_1}$. In this case, as shown in Figure 4 (d), the active partitions are the top-right and the bottom-right partitions. For $Part_{2_2}$, the old power grid system becomes the one designed in $Part_{2_1}$. The rest of the procedure of power grid design is essentially the same and can be derived from the explanation for constructing the power grid in $Part_{2_1}$.

3) Grid refinement by recursive bipartitioning: In the process of growing the partition tree of Figure 4(e), when each time two child nodes are added to a parent node in the partition tree, the coarser power grid in the partition corresponding to the parent node is converted to finer grids in the two active partitions corresponding to the child nodes. As described in the previous section, the wire widths of the child nodes are some factor, $\gamma_k < 1$, of the widths of the parent nodes.

With the growth of the partition tree, the number of partitions in the current partitioning level increases exponentially with the number of levels.As a result, the number of port nodes and thus, the size of the global system of equation (5), grows rapidly. This adversely affects the run time of the design procedure as in each iteration, following a pitch decrease in any one of the active partitions, it is necessary to construct the global matrix M, whose dimensions are rapidly increasing which leads to increase in number of conjugate gradient steps to solve the global system. To overcome this problem, we employ the *port approximation* technique suggested in [6] to reduce the macromodel sizes. By this approach, some of the port nodes located on the partition wires are collapsed. The port approximation scheme helps in checking the fast increase of the global system of equation (5) at the cost of reasonable simulation errors. However, the accuracy of the final solution is not compromised since the port approximation technique is switched off during the final few iterations of the design procedure.

The addition of two new child nodes in the partition tree can only take place if the following are satisfied:

1) IR drop and EM constraints, for the new power grid being constructed in the two active partitions, are met. These violations are detected by the hierarchical circuit analysis step.
2) There were no previous_grid_violated flags set as described in Section III-B.2. These violations are detected by checking the port voltages of all of the neighboring partitions of the two active partitions. If the maximum change, between the new port voltages of the neighboring partitions and the port voltages of the neighboring partitions before constructing the new grid in the active partitions, exceeds a specified value, MAX_{spec}, the previous_grid_violation flags are set to true.
3) The wire pitches in the two active partitions are greater than the minimum wire pitch, p_{min}.

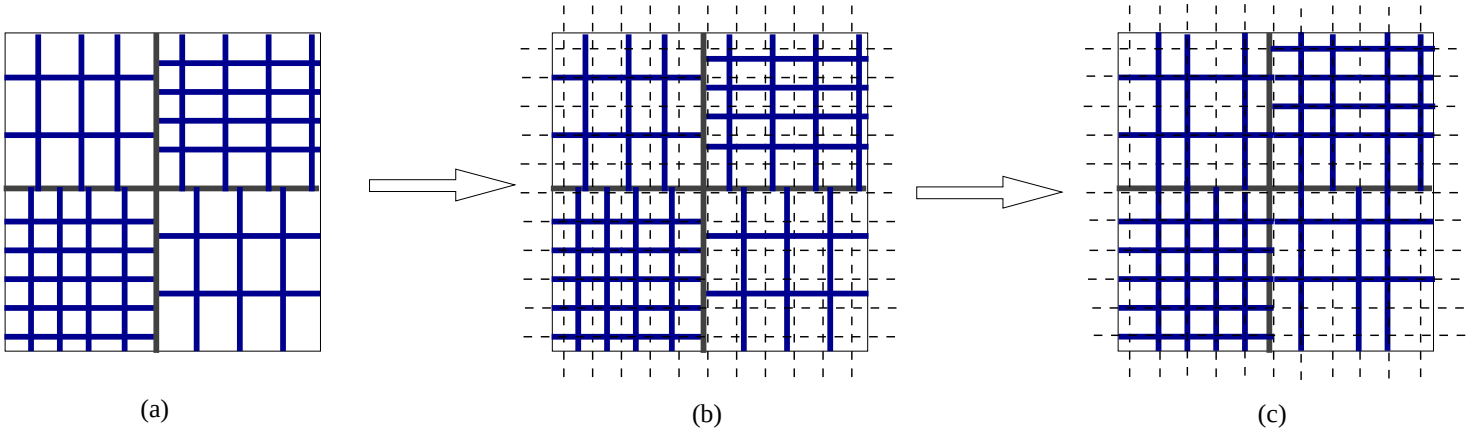
In order to fix the violations 1 and 2, the pitch of one or both the active partitions is decreased, depending on which of the two active partitions exhibits these violations. The minimum pitch violation is an undesirable breakdown in our power grid design procedure, and occurs when the grid refinement operation does not work, i.e., the increase in the wire resistance by replacing a power grid having thicker wires with a power grid having thinner wires cannot be compensated by adding more wires in the replacement grid. These minimum pitch violations cannot be fixed locally by modifying the grid in the active partitions. In this

case, we need to traverse the partition tree to the other tree elements that neighbor the active partitions, and add more wires in these neighboring partitions by reducing the wire pitches within them. Fixing the minimum pitch violations thus adversely affects the run time of the grid design procedure. However, we found out empirically that if the width reduction factor at the k^{th} partitioning level, $\gamma_k < 1$, is chosen such that it is not too small, (empirical values correspond to $\gamma_k \in [0.65, 1)$), such breakdowns are very rare events. We conclude the optimization procedure at the end of k levels of partitioning. The value of k is determined by the specifying the minimum size of the partition by the following relation:

$$k = 2 \cdot \min\left(\left\lceil \log_2 \frac{CHIP_w}{PART_w} \right\rceil, \left\lceil \log_2 \frac{CHIP_l}{PART_l} \right\rceil\right) \tag{7}$$

where $CHIP_w$ and $CHIP_l$ are the chip dimensions, and $PART_w$ and $PART_l$ are the specified dimensions of the minimum partition size.

C. Post processing for wire alignment



(a) (b) (c)

Fig. 8. The post-processing step to align the power grid wires in different partitions. (a) Power grid wires in adjacent partitions are misaligned. (b) A minimum pitch virtual grid, shown with dashed lines is constructed over the entire layout area. (c) The power grid wires are moved to the nearest position on the virtual grid. The wires in adjacent partitions are better aligned now.

At the end of k-levels of partitioning, we have designed a power grid in each of the 2^k partitions of the chip, with potentially different wire pitches in each partition. As a result, the wires in the adjacent partitions maybe offset with respect to each other, as illustrated by the simple example in Figure 8(a), where a power grid with four partitions has misaligned wires in the adjacent partitions. This misalignment may lead to use of extra vias to maintain the electrical connectivity of the power grid wires in different metal layers.

To alleviate the misalignment, we introduce a post-processing step in our grid design procedure, illustrated through the example in Figure 8. To rectify the misalignment in Figure 8(a), a uniform *virtual grid* is superimposed over the entire layout area, whose pitch is chosen to be the minimum pitch of wires in all of the partitions. The virtual grid is represented by dashed lines in Figure 8(b). Next, the power grid wires in all partitions are moved to the nearest location on the virtual grid. The virtual grid thus acts as a place holder for the real power grid wires. As seen in Figure 8(c), the wires in the adjacent partitions are better aligned at the end of the post-processing step. To ensure that the small movement of power grid wires does not affect the correctness of the grid, we perform a complete simulation using the hierarchical analysis technique. Our experiments

showed that the post-processing step hardly ever introduced any violations in the power grid circuit. In the rare cases where this was not true, the violations were easily fixed by adding more wires in the violating partitions. The extra wires are also added in such a way that they fall on the virtual grid, to remove any possibility of misalignment.

Algorithm 1 Power Grid Design

```
1:  Power_Grid_Design(func_block_currents,power_pads_pos)
2:  /*Initialize the root node of the partition tree*/
3:  tree[1].wire_width=W_init;
4:  tree[1].wire_pitch=P_init;
5:  tree[1].width=chip_width;
6:  tree[1].length=chip_length;
7:  i=2; k=⌊log2(i)⌋;
8:  cut_dir=0; /* cut_dir = 0/1 for a ver/horz cut*/
9:  /*Begin Outer While Loop*/
10: while (i < MAX_NUM_PARTITIONS) do
11:    parent_index=i/2;
12:    [child1, child2]=Divide(tree[parent_index],cut_dir);
13:    /* Decrease the width, γ_k ∈ (0, 1) */
14:    child1.wire_width = γ_k * tree[parent_index].wire_width ;
15:    child2.wire_width = γ_k * tree[parent_index].wire_width;
16:    spec_met_flag_this_part_level = 0;
17:    /*Begin Inner While Loop*/
18:    while (spec_met_flag_this_part_level ==0) do
19:       child1.wire_pitch= β_k1 * tree[i].wire_pitch; β_k1 ∈ [0, 1)
20:       child2.wire_pitch= β_k2 * tree[i].wire_pitch; β_k2 ∈ [0, 1)
21:       [(A1, S1), (A2, S2)]=make_pow_grid(child1, child2);
22:       [spec_met_flag1, spec_met_flag2]=solv_grid((A1, S1), (A2, S2))
23:       if (spec_met_flag1 AND spec_met_flag2) then
24:          spec_met_flag_this_part_level =1;
25:       end if
26:       if (!spec_met_flag1) then
27:          β_k1=δ_k * β_k1; δ_k ∈ (0, 1) /*Decrease the pitch*/
28:       end if
29:       if (!spec_met_flag2) then
30:          β_k2=δ_k * β_k2; δ_k ∈ (0, 1) /*Decrease the pitch*/
31:       end if
32:    end while
33:    /*End Inner While Loop*/
34:    prev_grid_viol_flag=1;
35:    while (prev_grid_viol_flag ==1) do
36:       [prev_grid_viol_flag]=chk_other_parts(child1, child2);
37:       if (prev_grid_viol_flag == 0) then
38:          β_k1=δ_k * β_k1; δ_k ∈ (0, 1) /*Decrease the pitch*/
39:          β_k2=δ_k * β_k2; δ_k ∈ (0, 1) /*Decrease the pitch*/
40:       end if
41:    end while
42:    /* Add two child nodes to the tree*/
43:    tree[i]=child1;
44:    tree[i+1]=child2;
45:    i=i+2;
46:    cut_dir=!cut_dir;
47: end while
48: /*End Outer While Loop*/
49: post_processing_of_grid_to_align_wires();
```

IV. The complete algorithm

The pseudo-code of the main loop of the power grid design algorithm is presented in Algorithm 1. We use the binary tree data structure to model the successive partitioning of the chip area. This is represented by the *tree* array in Algorithm 1. Each element of the tree array represents a node in the partition_tree of Figure 4, and contains the information about the partition dimensions, and the wire width and pitch of the power grid constructed in the partition corresponding to the tree node. In

lines 2 to 8 of the pseudo-code, the root node of the tree, which represents the full chip area, is initialized. The outer while loop, extending from lines 10 to 47, performs the recursive bipartitioning and grid construction process. Inside the outer loop, a parent partition is divided into two child partitions in lines 11 to 15. The **Divide** sub-routine, shown on line 12, implements the task of dividing the chip area of the parent partition into two smaller sized partitions. The first inner while loop, in lines 18 to 32, performs the task of constructing the local power grid in the two active partitions, subject to reliability constraints. Within this loop, the steps of constructing the macromodel parameters $(A, \mathbf{S})$ for the two active partitions, as explained in Section III-B.1, are contained in the routine **make_pow_grid**, listed on line 21. The steps of solution of the global system by preconditioned conjugate gradient method and the solution of the partitions by the hierarchical approach, as described in Section III-B.2 are performed by the routine **solv_grid**, shown on line 22. In case the local grid, constructed in the two active partitions, does not meet the reliability constraints, the pitches of the partitions are reduced by a factor δ_k, as shown in lines 26 to 31. The second inner while loop, shown in lines 35 to 41, ensures that the grid constructed in the active partition does not render the previously constructed grids in other partitions ineffective, as explained in Section III-B.2. This is ensured by the routine **chk_other_parts**, shown on line 36, by checking the port voltages of neighboring partitions of the two active partitions. Lines 43 to 46, which are a part of the outer while loop, add the two child nodes to the parent node, after a satisfactory local grid has been designed in the two active partitions. At the end of the design procedure, the post-processing step, as described in Section III-C is employed to improve the alignment of power grid wires in the adjacent partitions. This task is performed by the sub-routine **post_processing_of_grid_to_align_wires**.

In summary, the main features of the power grid design algorithm are:

1) The design procedure is based on a recursive bipartitioning scheme. At each step a simple bipartitioning problem of figuring out the appropriate wire pitches of the two active partitions is solved.
2) Using the concept of grid refinement, the power grid is iteratively refined. The abstraction of grid constructed in previous levels of partitioning is used to guide the power grid design in the active partitions.
3) Using the circuit analysis step to detect the IR drop and EM violations in the inner loop ensures the accuracy of the design scheme. The speed up of this method over other power grid design schemes, relying on the explicit circuit analysis, is obtained by making the analysis step extremely fast. This is achieved by:
    - Controlling the circuit size by using very thick wires in the initial levels of partitioning. The wire width is successively reduced in each partitioning level.
    - Using the macromodeling technique for abstraction of power grid in different partitions and focusing on local analysis of the two active partitions.
    - Solving the global matrix system by a preconditioned conjugate gradient based iterative solver and using the starting guess vector of the voltages at the port nodes, as determined by the solution of the power grid system designed in the previous iteration.
    - Employing the port approximation technique during the intermediate partition levels.
4) A post-processing step at the end of the design procedure is employed to improve the wire alignment in adjacent partitions.

V. Extension to Multiple Layers

We have used the proposed power grid design scheme to construct a power grid in the top two metal layers. The wires added to the grid, in each iteration, are in both horizontal and vertical directions in the top two metal layers.

The same approach can be easily extended to design a power grid in multiple layers of metal. In the case of multiple metal layers, the partitioning step would involve the introduction of partition wires in each of the multiple layers, and would result in partitions that would span multiple metal layers. In this case, the number of active partitions would be more than two and the wires added in the partitions would be added in each layer separately. The wires in different layers can be of different sizes as typically, the wires in top two metal layers are much wider than the ones in the intermediate metal layers. The step of checking for previous_grid_violated flags would now entail evaluation of port voltages of the neighbors of the active partitions in each layer. Other steps in the proposed design procedure remain the same while designing a multi-layered power grid.

VI. Experimental Results

The proposed power grid design scheme was implemented in C using a sparse matrix library [18] for both computing the macromodels and solution to the global system by preconditioned conjugate gradient method, and design of several power networks were tested. The input to our power grid design procedure is a floorplan with functional block current estimates and the locations and number of the power pads on the chip. The output is a non-uniform power grid that meets the IR drop, EM and minimum pitch constraints. We could find only two real benchmark floorplans [19], [20] for a microprocessor chip in which the functional block currents could be determined. These are the floorplans of ALPHA 21364 microprocessor chip. The block currents of the functional blocks in these floorplans were estimated from the given power consumption estimates of each functional block, in 130nm technology, using a V_{DD} of 1.2V. The functional block current I_{f_k}, of a block k is computed as $I_{f_k} = \text{Power}_k / V_{DD}$, where Power_k is the total power consumption of block k. Due to the paucity of real full chip level benchmark floorplans, with functional block current estimates, we randomly generated floorplans and assigned realistic block currents to various functional blocks in the floorplans. The block currents were assigned by assuming the total power consumption of the chips to be between 40 to 80 Watts and distributing the total power consumed randomly between the various functional blocks. For each of our experiments, we assume an availability of 400 to 600 power pads, distributed either throughout the chip, as in the case of a flip-chip package, or 200 to 300 pads located on the chip periphery, as in the case of a wire-bond package. The power pads are assumed to be connected to the top metal layer which has wires running only in the horizontal direction.

The circuit parameter values, sheet resistivity (ρ_s), current density (σ) and minimum wire pitches (p_{min}), were taken from [21] and [22] for power delivery to a 2cm×2cm chip in 130nm technology with V_{DD} = 1.2V. The voltage constraints for the power grids, i.e., V_{spec} was 1.08V, i.e., 90% of V_{DD}. The experiments were performed on P-4 processor, Linux machines with a speed of 2.4 GHz and 2GB RAM.

At the beginning of our optimization procedure, in the first level of partitioning, $Part_1$, the initial wire widths and wire pitches for the two partitions are chosen such that the worst case voltage drop is about twice the specified drop, e.g., 20%-25% of V_{DD}. This choice of starting point for the initial wire widths and the pitches selection has been empirically determined so

that we begin not too far away from the final solution point in the search space. Choosing an initial pitch and width assignment corresponding to a much worse initial voltage drop, e.g., 50% of V_{DD}, would mean that the design procedure has to spend much more time to reach the feasibility region and find a point that meets the reliability constraints. On the other hand, choosing a starting point which is very near to the feasibility region, e.g., 10%-12% of V_{DD}, may lead to over utilization of wiring resources as the design heuristic may not have enough iterations to explore the search space, before it finds a feasible solution that has a wire width and pitch assignment which may be sub-optimal in terms of wire area used. We assume equal pitches of the wires in horizontal and vertical directions within a partition, but, clearly this is not a restriction in the proposed scheme.

We construct the power grid by the proposed scheme for a set of eight benchmark floorplans, both for a flip-chip (FC) and a wire-bond (WB) case. Table I shows the results for these power grid constructions. For each experiment for both the FC and the WB case, corresponding to one row in Table I, the initial power grid in the first partitioning level comprises very thick wires in the range of 60 to 100 μm. In subsequent partitioning levels, the width reduction factor, γ is assigned an appropriate value in the interval $[0.65, 1)$ so that at the end of k levels of partitioning, the final value of wire width for the power grid in 2^k partitions is between 2 to 6 μm. The design procedure is terminated at the end of $k = 7$ levels of partitioning. The value of MAX_{spec} parameter, to flag the previous_grid_violations, was chosen to be 15mV.

Ckt	# of Blocks	# of Nodes		Wire Area (cm^2)		Run Time (sec)	
		Flip-Chip	Wire-Bond	Flip-Chip	Wire-Bond	Flip-Chip	Wire-Bond
pg-1	17	1557504	1635841	0.0812	0.0852	351	592
pg-2	17	1185921	1216609	0.0783	0.0816	425	693
pg-3	12	1261129	1375929	0.0721	0.0754	537	753
pg-4	16	1050625	1207801	0.0688	0.0738	491	747
pg-5	20	1216609	1343281	0.0704	0.0806	475	727
pg-6	24	1136356	1199025	0.0722	0.0786	577	793
pg-7	20	1640961	1703025	0.0852	0.1022	358	603
pg-8	22	1292769	1364224	0.0840	0.0992	381	584

TABLE I

RESULTS OF POWER GRIDS DESIGNED BY THE PROPOSED SCHEME FOR BOTH FLIP-CHIP AND WIRE-BOND CASES

The first two rows in Table I represent the power grid constructed for the two real benchmark floorplans of ALPHA 21364 chip. The other rows correspond to the power grid designed for the randomly generated floorplans. The second column in the table shows the number of blocks in the floorplan. The next two columns indicate the number of electrical nodes in the final optimized power grid circuit. For each circuit there are more than a million electrical nodes in the final circuit. The optimization is terminated when the worst voltage of all nodes in the final power grid circuit is greater than V_{spec} and all branches meet the current density specifications at the end of $k = 7$ levels of partitioning and the post-processing step to align the wires. The worst voltage measured by performing an accurate simulation, at the end of the design procedure without the port approximation technique, verifies the accuracy of the final solution. The wire area consumed by the final power grid is listed in the fifth and the sixth column, for a flip-chip and a wire-bond case, respectively. The last two columns report the run time for constructing the power grid by the proposed design procedure. The run times of the proposed power grid design

procedure are in the range of about 6 to 10 mins for the grids designed for the flip-chip case and about 10 to 13 mins for the wire-bond case. The order of the run times obtained underscores the efficiency of the design algorithm, considering the fact that for each of the test cases in Table I, the final power grid for both the flip-chip and the wire-bond case, spanning the entire chip area in two layers of metal, comprises more than a million electrical nodes.

As seen in Table I, the proposed scheme performs better for the flip-chip case than the wire-bond case, both in terms of utilizing lower wire area and in its faster run times. This can be ascribed to the fact that the notion of locality in power grid design, which is one of the motivating factors of the proposed algorithm, is more pronounced in the case of a flip-chip package, where there are sufficient number of pads near the violating regions. For a wire-bond chip, the fact that the pads around the chip periphery are located far way from the violating regions that may be located at the center of the chip, could make local grid correction step for fixing the violations, a sub-optimal choice. Hence, the procedure has to expend a larger amount of computational time and wiring resources to meet the reliability constraints for a wire-bond chip.

Ckt	# of Wires Multigrid Scheme	Run Time (sec)		Wire Area (cm^2)		% Saving in Wire Area	% Saving in Run Time
		Proposed Method	Multigrid Scheme	Proposed Method	Multigrid Scheme		
Ckt-1	1000 × 1000	550	586	0.0701	0.0751	7.2%	6.1%
Ckt-2	1100 × 1100	555	641	0.0741	0.0801	8.2%	13.4%
Ckt-3	1150 × 1150	561	681	0.0766	0.0843	10.1%	17.6%
Ckt-4	1200 × 1200	569	705	0.0759	0.0849	11.9%	19.3%
Ckt-5	1250 × 1250	605	733	0.0793	0.0866	9.2%	17.5%
Ckt-6	1300 × 1300	679	775	0.0840	0.0935	12.3%	12.4%

TABLE II

RESULTS OF POWER GRIDS DESIGNED FOR FLIP-CHIP CIRCUITS BY THE PROPOSED METHOD AND THE MULTIGRID-BASED SCHEME OF [4]

In another set of experiments, we compare the proposed power grid design algorithm with a previous grid design scheme [4]. We implemented a simple version of the multigrid-based power grid optimization scheme of [4] in C++ to compare the results of our proposed power grid design algorithm with this method. Table II shows a comparison of the performance of the proposed power grid design algorithm with the multigrid-based technique of [4]. The two schemes are used to design power grids for six randomly generated floorplans for a flip-chip case. The floorplans comprise 20 to 40 functional blocks with currents assigned randomly to each block so that the total power consumption of the chip is between 40 to 80 Watts. Some functional blocks are assigned about 3 to 4 times more power than the other blocks so that there are distinct high and low current density regions on the chip. The assignment of block currents, to model the high and low current density regions, follows from the observation that most full-chip microprocessor floorplans have about 30%-50% of chip area dedicated to caches which consume much less power than the other functional blocks, e.g., arithmetic and logic units [20]. In our implementation of the multigrid-based method, only the wire widths are optimized by setting the weights, corresponding to the decoupling capacitor cost and the congestion term, to zero in the objective function of [4]. A uniform power grid is constructed for the six cases using the multigrid-based design scheme, with an initial constant wire width of 1.5μ for all wires. Next, by enumeration, the numbers of wires in the uniform grid topology is determined so that the initial voltage drop is about two times the specified drop. This initial starting point, in terms of the initial voltage drop, is the same as chosen for selecting the initial pitches in $Part_1$ of

our proposed design procedure. The second column in the table II lists the number of horizontal and vertical wires used for the uniform grid construction. Following a series of network reductions, about 8 to 14 levels of reduction for each circuit, the top level power grid is reduced to a much smaller grid so that the problem size is sufficiently small. The wire sizing solution for reduced network is then obtained by solving a constrained non-linear optimization problem by using a sequential quadratic programing software [23]. The back-mapping to the original network is performed by solving a series of linear programs as formulated in [4].

The fourth and the fifth columns in the table show a comparison of the run times of the two schemes. For all of the six circuit examples, the total time taken by the multigrid-based scheme to perform the network reduction, solve the non-linear optimization problem and solve a series of linear programs for back-mapping is greater than the proposed power grid design algorithm. On an average for the six example cases the proposed method is about 14% faster than the multigrid-based design algorithm. Columns six and seven show the wire area utilized for each of the example circuits by the two design methodologies. The wire area utilized by the proposed heuristic is about 7%-12% less than the grid design method of [4]. In the multigrid-based design method, each column (row) of vertical (horizontal) wire is constrained to have the same wire width in order to reduce the number of design variables for efficiently solving the resulting non-linear program. Since the width of all of the wire segments on a column (row) of the wire is determined by the highest current density blocks, the power grid has to be over-designed in the low current density regions of the chip. The proposed design algorithm is run for $k = 10$ levels of partitioning so that the granularity of each partition is of the order of the block size. By designing local power grids in each partition, it is ensured that the wiring resources are utilized in a judicious manner as per the current density requirements.

In another set of experiments we study the effect of choosing different levels of partitioning for the power grids designed for the same input floorplan. Table III represents these experiments for constructing power grids for the pg-6 floorplan and

SNo.	# of Partition Levels	Wire Area (cm^2)	Run Time (sec)
1	6	0.0790	498
2	7	0.0722	577
3	8	0.0710	639
4	9	0.0702	674

TABLE III

POWER GRIDS FOR PG-6 FLOORPLAN BY CHOOSING DIFFERENT PARTITIONING LEVELS

assuming the flip-chip case. As seen in the table, the algorithm runs faster for smaller number of partitioning levels. This can be ascribed to the fact that by increasing the height of partition_tree of Figure 4(e), the number of partitions and consequently the size of the global system of equation (5), even with the port approximation technique increases, which results in greater simulation time in each iteration. The cost of using fewer partitioning levels is over-utilization of wiring resources. The number of wires required in the partition is determined by the the maximum of all current demands over the entire partition area. Since the partition size is larger for a smaller value of k, more wiring resources are wasted in the region within the large partitions where the current requirements are less than the maximum. By splitting the partitions into smaller sizes, the regions for different current demands can be isolated, and power grids with better area utilization can be constructed separately in

these regions. However, beyond a point, the run time penalty for increasing the granularity of partitions or the number of partitioning level, outweighs the savings in the wire area.

VII. Conclusions

In this paper, we have proposed a novel and an efficient power grid design procedure. Our method is based on an iterative grid refinement scheme by recursive bipartitioning of the chip area. We use the concept of locality in power grid design to abstract away the details of some parts of power grid by the macromodeling technique. Using the grid abstractions, along with the strategy of constructing an initial coarse grid followed by a successive refinement of the grid and reusing the solution of grid designed in the previous iteration as a starting guess point for the conjugate gradient linear solver, speeds up the circuit analysis step. Experimental results on real and randomly generated realistic test cases show that the proposed power grid design algorithm is considerably fast and has efficient utilization of the wiring resources.

References

[1] X. D. S. Tan, C. J. R. Shi, and J. C. Lee. Reliability-Constrained Area Optimization of VLSI Power/Ground Networks Via Sequence of Linear Programmings. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 22:1678–1684, Dec 2003.

[2] X. Wu, X. Hon, Y. Ca, C. K. Cheng, J. Gu, and W. Dai. Area Minimization of Power Distribution Network Using Efficient Nonlinear Programming Techniques. In *Proceedings of the IEEE/ACM International Conference on Computer Aided Design*, pages 153–157, 2001.

[3] T. Wang and C. C. Chen. Optimization of the Power/Ground Network Wire-Sizing and Spacing Based on Sequential Network Simplex Algorithm. In *Proceedings of the IEEE International Symposium on Quality Electronic Design*, pages 157–162, 2002.

[4] K. Wang and M. M-Sadowska. On-chip Power Supply Network Optimization using Multigrid-based Technique. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24:407–417, Mar 2005.

[5] M. Zhao, Y. Fu, V. Zoltov, S. Sundareswaran, and R. Panda. Optimal Placement of Power Supply Pads and Pins. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 165–170, 2004.

[6] J. Singh and S. S. Sapatnekar. Congestion-Aware Topology Optimization of Structured Power/Ground Networks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24:683–695, May 2005.

[7] T. Mitsuhashi and E. S. Kuh. Power and Ground Network Topology Optimization for Cell Based VLSIs. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 524–529, 1992.

[8] H. Cai. Multi-pads Single Layer Power Net Routing in VLSI Circuit. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 183–188, 1988.

[9] J. Oh and M. Pedram. Multi-pad Power/Ground Network Design for Uniform Distribution of Ground Bounce. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 287–290, 1998.

[10] H. Su, K. H. Gala, and S. S. Sapatnekar. Fast Analysis and Optimization of Power/Ground Networks. In *Proceedings of the IEEE/ACM International Conference on Computer Aided Design*, pages 477–480, 2000.

[11] H. H. Chen and D. D. Ling. Power Supply Noise Analysis Methodology for Deep-Sub-micron VLSI Chip Design. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 638–643, 1997.

[12] E. Chiprout. Fast Flip-Chip Power Grid Analysis via Locality and Grid Shells. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 485–488, 2004.

[13] M. Zhao, R. V. Panda, S. S. Sapatnekar, T. Edwards, R. Chaudhry, and D. Blaauw. Hierarchical Analysis of Power Distribution Networks. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 482–486, 2000.

[14] J. Singh and S. S. Sapatnekar. A Fast Algorithm for Power Grid Design. In *Proceedings of the ACM International Symposium on Physical Design*, pages 70–77, 2005.

[15] H. Kriplani, F. Najm, and I. Hajj. Pattern Independent Maximum Current Estimation in Power and Ground Buses of CMOS VLSI Circuits : Algorithms, Signal Correlations, and Their Resolution. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 998–1012, 1995.

[16] Z. Wang, R. Murgai, and J. Roychowdhury. Automated Accurate Macromodelling of Digital Agressors for Power/Ground/Substrate Noise Prediction. In *Proceedings of the IEEE Design, Automation and Test in Europe*, pages 824–829, 2004.

[17] Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts, 1999.

[18] http://www.netlib.org/c/meschach.

[19] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-Aware Microarchitecture. In *Proceedings of the International Symposium on Computer Architecture*, 2003.

[20] W. Liao, L. He, and K. Lepak. Temperature-Aware Performance and Power Modeling. In *Technical Report 04-250, UCLA Engr, Los Angeles, California*.

[21] Semiconductor Industry Association. International Technology Roadmap for Semiconductors, 2001. Available at `http://public.itrs.net`.

[22] J. Cong. An Interconnect-Centric Design Flow for Nanometer Technologies. In *Proceedings of the IEEE*, volume 89, pages 477–480, 2000.

[23] C. Lawrence, J. Zhou, and A. Tits. User's gude for cfsqp version 2.4. Institute for Systems Research, University of Maryland, College Park, MD, Tech. Rep. TR-94-16rl, 1996.