Vol.2011-SLDM-148 No.17
2011/1/17

Via 数削減による大規模 LSI レイアウトの高速 DRC 手法

亀井 智紀[†] 安部 拓哉[†] 本垰 秀昭[†] 渡邊 孝博[††]

GDS Ⅱなどのレイアウトデータでは，多層配線の配線間の接続箇所に Via が使用されているが，近年，DFM（Design for Manufacturability）技術の浸透により，配置される Via の数が爆発的に増加している．Via 図形のデータが増加すると，DRC（Design Rule Check）を行う EDA ツールにとっては，計算コストが増大し，解析に多くの時間とメモリ空間を必要とする．そこで本研究では，解析データから Via を必要最小限まで擬似的に削減し，その上で DRC の一つである配線幅チェックを行った．Via を削減しない場合と比較して，数倍～数百倍解析時間が短縮された．

The High-speed DRC Technique for VLSI Layout by Reducing Vias

Tomoki Kamei[†] Takuya Anbe[†] Hideaki Hontao[†] and Takahiro Watanabe[†]

In layout data such as GDS II, Vias are used for connection points between multilayer wiring. In recent years, the number of Vias to be arranged has been increasing explosively in accordance with the spread of DFM (Design for Manufacturability) technology. Increase of Via graphic data would rise computation costs for an EDA tool that performs DRC (Design Rule Check) and require long time and memory space for an analysis. Therefore, in this study, Vias were spuriously reduced to minimum from analysis data and the wire width was checked, which was one of the DRCs. Comparing with the case that Vias are not reduced, analysis time has been shortened by several times - several hundred times.

1. はじめに

近年，GDS Ⅱに代表される LSI レイアウトデータの総サイズは，さらに増加傾向にあり，ギガバイト超のデータも珍しくない．レイアウトデータの中でも電源線の箇所は，十分な配線幅が確保されていないと，過剰な電流に対して溶断事故が発生するため，DRC（Design Rule Check）の重要性は極めて大きい．しかし，ギガバイト超のレイアウトデータの電源線ネットは長大かつ，それを構成している図形数も数百万のオーダーになるため，DRC 処理時間が深刻な問題となっている．

一般的に大電流が流れるネットでは層間接続に非常に多くの Via が使用され[1]，さらに DFM（Design for Manufacturability）や DFY（Design for Yield）の一つとして，Redundant-Via を付加する手法も提案されている[2][3]．これは，製造ばらつき対策や信頼性確保のために有効であるが，結果として多層配線において配線間を接続する Via の数が著しく増加し，レイアウトデータのサイズをさらに肥大化させている．例えば，実際の LSI のチップで，全ポリゴン数が約 2570 万に対して，Via 数が約 2560 万であり，約 99.6%が Via となっている例もある．

Via 図形の増加はレイアウトの解析，検証を行う EDA ツールにとっては，極めて大きな問題となる．EDA ツールは，配線幅チェック等の DRC を実行する際，層間の接続を追跡するために Via 1 個で接続される箇所が一つの回路となる．ゆえに Via 個数分の並列回路の経路を解析することになり，Via が大量に存在すると計算コストが増大する．このため，EDA ツールを使用した通常の手法の DRC では処理に時間がかかる上に，大量のメモリ空間を必要とするため，LSI 検証工程の大きな問題になってきている．

一方で，例えば電源ネットの配線幅チェックを行うのであれば，先ず多層に渡る電源ネットを抽出するために，電気的な配線間の接続を示すための情報を持つ最低限の Via 情報が必要である．しかし，そのためには一接続箇所当たり一個の Via が存在すればよく，層の接続箇所での大量の Via 一個一個を処理する必要はない．そこで，本研究では先ず DRC 処理には冗長な Via を電源ネットから取り除き，得られたネットに対して，配線幅チェックを行う手法を提案する．また，本手法を EDA ツールに実装し，効果を確認する．

実験の結果，電源一ネットあたりの総図形数は大幅に減少し，ネット全体に対しての配線幅チェックは瞬時に結果を得られるようになった．さらに，ネット中の 2 点を始点，終点として指定し，指定経路に沿った配線幅チェックを行う処理においても処

[†] TOOL 株式会社 EDA 製品事業部
Technical Department EDA Product Division, TOOL Corporation
[††]早稲田大学大学院情報生産システム研究科
The Graduate School of Information, Production and System, Waseda University

理時間が数倍～数百倍短縮された.

2. 方法

2.1 概要

図 1 に配線図形と Via の概念を示す．Via は全て配線図形内に包含されているものとする．今回提案する手法は，配線幅チェックを実行する前段階で配線層の重なっている領域を抽出し，その領域内に一個以上の Via が配置されていたら，その領域全体を一個の Via とした上で，元々存在している Via 情報を DRC 対象データからすべて破棄する．本手法により，一接続箇所あたりの Via 数は 1 になるため，大幅に図形数を削減できる．

2.2 処理フロー詳細

DRC を行うツールは商用の TOOL 株式会社製 LAVIS[4]を使用する．図 2 に本処理の全体のフローを示す．大きく，等電位追跡処理と配線幅チェック処理に分かれる．本研究の対象は GDS Ⅱ データなので，図形情報のみでネットリストは存在しない．そこで先ず，等電位追跡を行い，検証を行う電源ネットを抽出する．この際，どのレイヤとどのレイヤがどの Via で接続される，というテクノロジー定義情報をあらかじめ利用者が作成し，その情報を元に抽出する．電源ネットは長大であり，抽出自体に相当の時間を要するため，あらかじめ全ネットのデータベースを作成する作業を行う．一度，データベースを作成しておくと，その後は GDS Ⅱ データに存在する任意の等電位ネットを瞬時に抽出することができる．このデータベース作成段階で，以下のような Via の削減処理を行う．

例えば，Metal1 配線と Metal2 配線が Via12 で接続されている例を考える．先ず，Metal1 配線の図形と Metal2 配線の図形を抽出し，双方が重なっている AND 領域を取り出す．その AND 領域内に一個以上の Via が存在すれば，領域全体を一個の Via として配置，元の Via は破棄する．この手法を式(1)，(2)および図 3 に示す．ここで，配線レイヤ m，n の配線図形の領域を Wm，Wn とし，Via 図形の領域を Vmn とする．また，∩，⊃はそれぞれ図形演算の AND，包含を表す．テクノロジー定義情報に定義されている全ての配線レイヤと Via レイヤについて，同様の処理を行う．

$$(W_m \cap W_n) \supset V_{mn} \qquad (1)$$

を満たす領域があるとき

$$V_{mn} \Leftarrow W_m \cap W_n \qquad (2)$$

とおく．

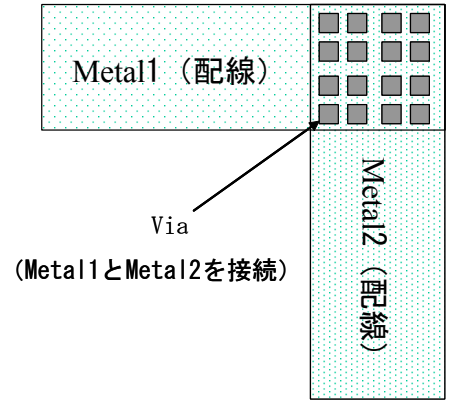

図 1．配線図形と Via の関係

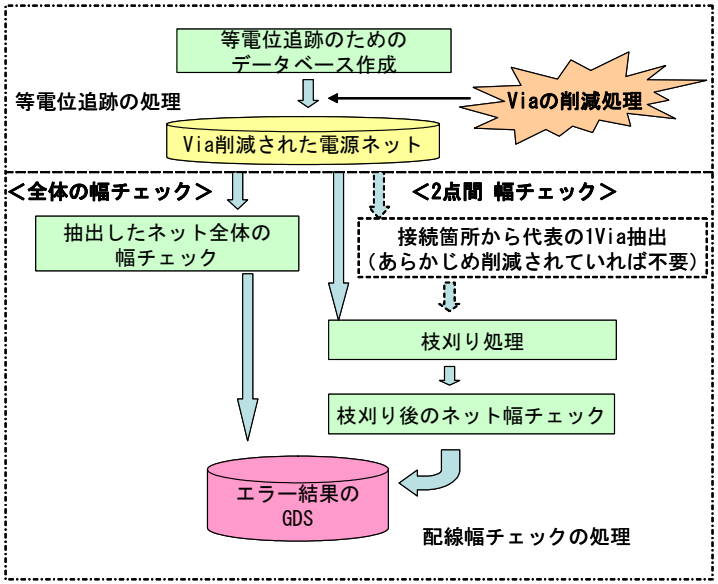

図 2．配線幅チェック全体の処理フロー

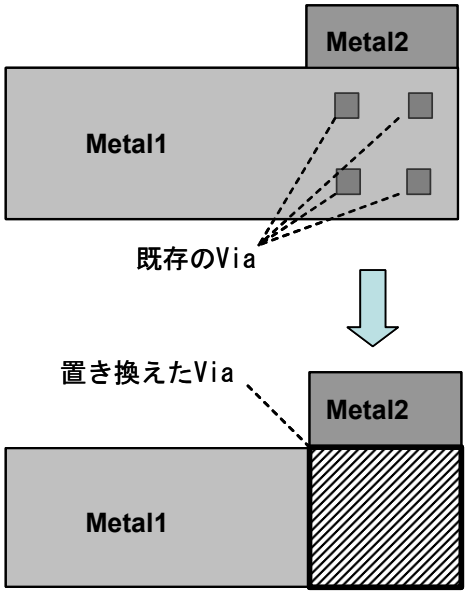

図 3. Via の削減手法

Via が削減された電源ネットが抽出されたら，そのネットに対し，配線幅チェックを行う．今回は 2 種類の配線幅チェックを実施する．一つは，抽出された電源ネット全体に対して行うもの，もう一つは開始点，終了点の 2 点の座標を指定して，その 2 点間の経路上をチェックするものである（図 4）．2 点指定のものは，配線幅チェック対象が 2 点の経路上に限定されるため，利用者にとってチェック不要の配線を除外することができ，効率よく処理できる．経路上，並列回路になる場合は，並列部分も全てチェック対象となる．2 点間の経路のみを抽出するために，経路から枝分かれしている配線部分を枝刈り処理によって取り除く必要があるが，詳細なアルゴリズムは次節で説明する．最終的に，配線エラーが見つかった場合は，エラー部分の図形が別 GDS II ファイルに出力される．

2.3 枝刈りのアルゴリズム

2 点を指定して配線幅チェックを実行する際，配線間の接続箇所に複数の Via が存在している場合は，処理上，最初に見つかった一個の Via を採用し，残りの Via は捨てる．次に配線をエッジ，Via をノードとしてグラフを作成し，さらに始点 S，終点 E の 2 点のノードを置く（図 5）．以下に S,E を含む連結グラフを前提に枝刈りのアルゴリズムを記す．目的は始点－終点間の経路（並列を含む）のみを取り出し，支線の枝となっている部分を削除することである．

（1） S,E 以外の次数 1 のノードとそのエッジを削除し，グラフを更新する．更新したグラフに新たに次数 1 のノードがあれば，同様に削除しグラフを更新する．
（2） S,E 以外に次数 1 のノードが無くなるまで，(1)を繰り返す．
（3） S,E 以外の次数 3 以上のノードから S および E までの経路を探索する．ただし，S,E までは別のエッジを経由するものでなければならない．
（4） (3) の条件に該当しないノードを削除する．
（5） 残ったグラフが，始点，終点間の経路となる．

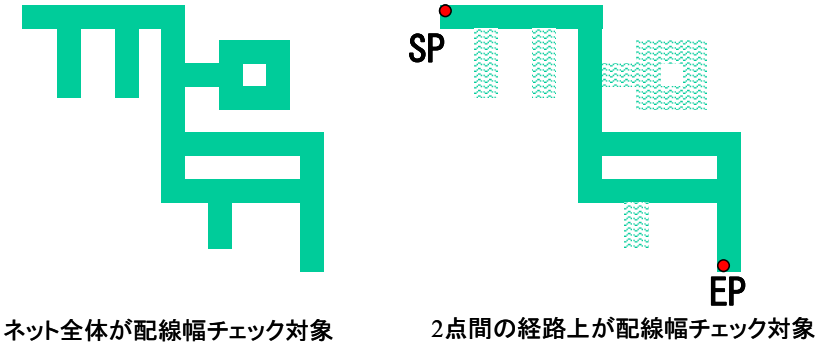

図 4. 電源ネットの 2 種類の配線幅チェック方法

3. 実験

3.1 実験データと環境

今回実験に使用したデータの詳細を表 1 に，実験環境を表 2 に示す．ネットごとの図形数は電源ネット抽出終了時に計算されている．配線幅チェックの所要時間は Linux の top コマンドを使用して計測した．

3.2 データベース作成時間

電源ネットを抽出しながら，Via の削減処理を実施したデータベースの作成時間を表 3

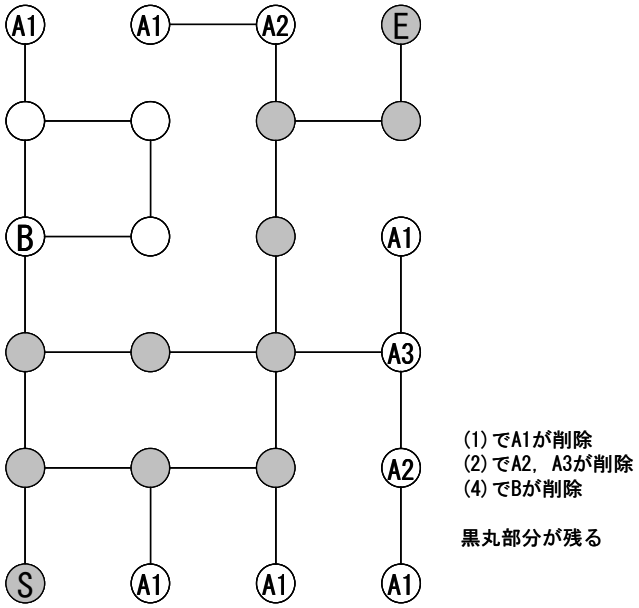

図 5. Via と配線をグラフ化して枝刈り
（S：始点，E：終点）

に示す．データベース作成の際には，CPU 分散が可能である．今回，実験に使用したマシンは 4CPU を搭載しているので，4CPU 分散処理でデータベースを作成した．Via の削減処理を実施したものは，しないものと比較して 2 分半ほど時間がかかった．ただし，データベース作成は最初に 1 回行えばよいだけなので，後続の DRC 処理を繰り返す場合に効率が向上する．

また，データベースのファイルサイズ自体も，Via 図形が減少したことにより小さくなる．そのため，データベース読込時間は短縮され，読込時に使用するメモリも少なくなる．

表 1．実験に使用したデータ

ファイルフォーマット	GDSⅡ
ファイルサイズ[MB]	1700
処理領域（設計値）[um]	6000 x 6000
配線層数	メタル4層＋Via3層＋Poly
プロセス（設計値）[um]	0.2
チップ面積[um2]	36602550
総図形数	25876841

表 2．実験に使用したマシン環境

CPU	4 x AMD Opteron 850 (2.4GHz)
Memory	64GB
OS	CentOS4 (x86_64)
HDD	750GB + 2000GB + 1000GB SATA

表 3．Via 削除有無によっての DB 作成および読み込み

	Via削減なし	Via削減あり
DB作成時間(4CPU分散) [sec]	1055	1217
DB読込時間 [sec]	45	0.003
DBファイルサイズ [MB]	559	5.6
DB読込使用メモリ [MB]	454	23

3.3 図形数と幅チェック時間

1 ネットあたりの図形数は，Via の削減処理を実施したことで，大幅に少なくなる．ネットごとの図形数の変化を表 4 に示す．

また，ネット全体の接続箇所数を N，i 番目の接続箇所での Via 数を ci，ネット全体の配線図形ポリゴン数を W とすると，ネットの総図形数 Fcount は以下の式で表すことができる．
Via の削減処理をしない場合

$$Fcount = \sum W + N \qquad (3)$$

Via の削減処理をした場合

$$Fcount = \sum W + \sum_{i=1}^{N} ci \qquad (4)$$

配線幅チェック時間の計測は Linux の Top コマンドの表示結果を参照したが，結果が瞬時に得られたものは実測値測定が不可能なので，表 4 では NA とした．ネット全体に対して行ったものは，Via の削減処理実施後のネットでは，瞬時に処理が終了している．一方，2 点間に対して行ったものも Via 削減処理をしていないものと比較して，大幅に処理時間が削減されていることが判る．

4. 考察

4.1 配線幅チェック（全体）における Via 削減効果

配線幅チェックの前段階で Via を削減したものは，瞬時に実行結果を得ることができた．接続箇所あたりの Via 図形が一つにまとまったことで，全体の図形数が大幅に減ったためと考えられる．ところで，Via 削減処理の有無で幅チェック時間を比較する場合は，データベース作成と読み込みに要する時間の差分を考慮しなければならない．今回，データベース作成において，Via 削減処理を実施すると 2 分半ほど余分に時間がかかる．逆に読み込みにおいては，45 秒短縮される．Via 削減処理なしでデータベースを作成すると，配線幅チェックにかかる時間は最も長いもので 1 分半ほどであるため，複数箇所を検証する場合は，Via 削減を実施したほうが効率良いといえる．

4.2 配線幅チェック（2 点間）における Via 削減効果

2 点間のチェックでも大幅に時間が短縮されたが，全体チェックと比べるとかなりの時間を要している．これは，枝刈り処理の中のグラフ作成部分および枝刈り実行部分がオーバーヘッドになっていると考えられる．枝刈り処理は，アルゴリズム上，多分岐をしているノード数に依存して指数関数的に処理時間が増えていく．しかし，Via を削減しても配線の分岐数自体は変わらないため，この部分の処理時間は従来のままで改善されない．一方，配線幅チェックの前段階で Via の削減処理が実施されれば，配線幅チェック処理の中で，配線交差部分の AND 領域から代表の 1 個の Via を抽出する作業が省略でき，時間の短縮となる．

表 4．図形数と幅チェック時間（時間は CPU 時間）

	Via図形数		処理時間（2点）[sec]		処理時間（全体）[sec]	
ネット名	Original	Via削減	Original	Via削減	Original	Via削減
A	2176634	208	354	NA	36	NA
B	2377162	2204	340	68	36	NA
C	6142784	2122	633	70	94	NA
D	9599292	2001	1286	44	90	NA

2 点間の幅チェックの場合は，Via を削減しないと非常に時間がかかるため，データベース作成のオーバーヘッドを考慮しても，Via を削減しておいた方が，処理が速く終わるという結果になった．

4.3 2 点間チェックの従来手法との比較

今回の Via の削減処理と削減処理を行わない従来の手法とを比較してみる．従来の手法は，幅チェック処理の中で，先ず，配線の交差している AND 領域を取得し，その中に存在している Via の中から，最初に見つかる Via をその接続箇所の唯一の Via として採用し，残りの Via を捨てるというものである（図 5）．枝刈りのためのグラフを作成する段階で，各接続箇所の Via は一つにしておく必要があるためである．このように従来手法でも Via を一つに絞り込むが，接続箇所ごとに代表の 1 Via を選ぶ処理が発生する．

一方，前段階で Via を一つにした場合には，データベース作成時に処理が増えるものの，以降は Via を一つに減らしたデータベースを使用して，配線幅チェックが行える．そのため，検証全体の TAT としては大幅に短縮される．

5. 結論

我々は，今回，配線間の接続箇所に大量に存在する Via を一つにまとめ，配線幅チェックに要する時間を検証した．配線間の接続箇所に大量の Via が存在するとネットの配線幅チェック等の DRC に多くの時間がかかっていたためである．大量 Via を一つにまとめる手法として，配線層の交差する AND 領域内に Via が一つ以上存在すれば，

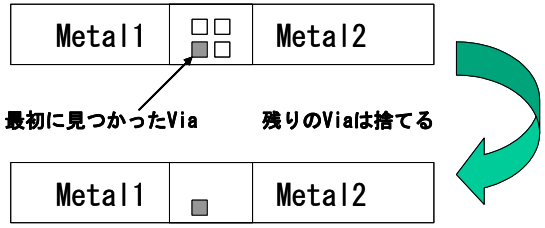

図 5　最初に見つかった Via のみ採用する手法

その AND 領域全体を新たな Via とみなす方法を採用した．

Via の削減処理は配線幅チェックを実行する前段階でデータベース化しておき，以降の配線幅チェックは，そのデータベースを用いて行う．

先ず，Via の削減処理を行うことで，このデータベースのサイズが減少し，読み込みに要する時間短縮と使用メモリの減少が確認された．

次に，配線幅チェックはネット全体に対して行う方法と 2 点間を指定して，経路上の枝刈りを行う方法の 2 種類がある．そのうち，ネット全体に対して行う方法に関しては，ネットの Via 図形数が大幅に減少したため，非常に高速に処理が行えるようになった．今回実験したネットでは，いずれも瞬時に処理が終了している．

一方, 2 点間の配線幅チェックでは, まだ処理に数分を要するネットがあるものの, 従来に比べて数倍～数百倍の処理時間短縮が得られた．

今後は，さらに，今回の Via の削減手法を配線幅チェックのみならず，抵抗値計算等への適用を検討していく．

6. 謝辞

本研究を行うにあたり、種々のご助言を賜りました早稲田大学大学院情報生産システム研究科渡邊研究室の皆様に心より感謝の意を表します。

7. 参考文献

1) Dan Clein, “CMOS IC Layout Concepts, Methodologies and Tools“, Newnes, Elsevier,2000.
2) Yuji.Takashima, Kazuyuki Ooya, and Atsushi Kurokawa, “Practical Redundant-Via Insertion Method Considering Manufacturing Variability and Reliability,” IEICE Trans. Fundamentals, vol.E92-A, no.12, pp.2962-2970, Dec.2009.
3) Cheok-Kei Lei, Po-Yi Chiang, Yu-Min Lee, “Post-Routing Redundant Via Insertion with Wire Spreading Capability,” Asia South Pacific Design Automation Conference (ASPDAC), pp. 468-473, Jan. 2009.
4) LAVIS.Users-Guide,TOOL Corp., 2010.