# LSIデザインルールチェックにおける機能分割法と領域分割法の相補的複合並列化手法の優位範囲の測定

上坂達生[†] 松木俊寿[††,☆] 田丸啓吉[††]

LSIデザインルールチェック（以下，DRC）は設計ルールとレイアウトパターンをつき合わせて検証するものである．DRCを並列化する基本的な方法は2種類が知られており，それぞれ機能分割法，領域分割法と呼ばれている．LSIはトランジスタ等の素子と配線の集合体でレイアウトパターンデータはこれを多数の図形で記述している．これを検証する手段として設計ルールを記述したものはルールズファイルと呼ばれる．DRCのために実用されているルールズファイルの中には素子と配線に関してレイアウトパターンデータと相補的直交関係にあるものがある（以下，これを直交型ルールズファイルと呼ぶ）．また特定の考え方により意識的に直交型ルールズファイルをつくることができる．この論文は直交型ルールズファイルを用いたDRCの並列化において，直交性と上述の2種類の分割法を組み合わせた並列化手法（すなわち新しい分割方法および処理手順）について述べ，またDRCの高速化について本並列化手法が従来並列化手法より優位にある範囲をシミュレーション実験により測定した結果について述べたものである．

# A Dominant Scope of the Compound Method of Function Partitioning and Area Partitioning in LSI Design Rule Check

TATSUO UESAKA,[†] TOSHIHISA MATSUKI[††,☆] and KEIKICHI TAMARU[††]

In order to parallelize design rule check, the schemes, the function-partition and the area-partition, are always used. The design rules for checks are described in a what is called rule's file. Some of the current rule's files are reciprocal and orthogonal to the layout pattern data regard to elements and wirings. (We call them orthogonal rule's files.) We got an another parallelization scheme with combining orthogonality to those two schemes. This report describes about a dominant scope of the compound method of function partitioning and area partitioning in LSI design rule checks.

## 1. まえがき

LSIの設計工程においてデザインルールチェック（以下，DRC）は不可欠の工程であり，これを高速化する必要性については論を待たない．高速化するために従来から行われた方法に関係して次の3種類のDRCを定義する．

(1) incremental DRC
レイアウトパターン設計時点にDRCを重ねてレイアウトデータを新たに修正した部分のみチェックする．したがってDRC処理時間は設計時間の下に隠れてしまい，見かけ上高速になる[1),2)]．

(2) hierarchical DRC
LSIは一般に階層設計がされているのでこれを利用して，各セル（部分ブロック）をあらかじめチェックしておきDRC処理時にセルを展開せずにセル間でチェックを行う．したがって処理は高速になる[3),4)]．

(3) flat DRC
レイアウトパターンデータをまずフラット（単層）に展開し，展開されたデータに対してチェックをかける．したがって最終のマスクイメージでチェックすることになり，人手の設計にも対応できる最も基本的な検証方法であるが，処理時間は非常に大きくなる．これを高速化するには処理プログラムを改良する方法[5),6)]，並列化する方法[7)]，専用エンジンを開発する方法等が

† 熊本電波高専情報工学科
Department of Information, Kumamoto National College of Technology
†† 京都大学工学部
Faculty of Engineering, Kyoto University
☆ 現在，パイオニア株式会社
Presently with Pioneer Company Limited

研究されている[8),9)].

この論文はflat DRCの処理を並列化することによって高速化することに関するものである．flat DRCの処理を並列化によって高速化する基本的な手法は2種類あり[10),11)]，それぞれ機能分割法（Function-partitioning），領域分割法（Area-partitioning）と呼ばれている[12)]．

DRCは設計規則を記述したルールズファイルとレイアウトパターンデータとを照合する検証であるが，検証の手段であるルールズファイルを各単位検査機能ごとに分割して並列化するのが機能分割で，検査の対象であるレイアウトパターンを複数の領域に分割して並列化するのが領域分割である．

DRCはマスクパターン検証の重要な一部分であるが，マスクパターン検証全体すなわちレイアウトパターンに対する検証の全体はDRC（Design Rule Check），ERC（Electrical Rule Check），LVS（Layout Versus Schematic），LPE（Layout Parameter Extraction）の4種類のカテゴリーに分類される．これらの4種類の検証はお互いに補完関係にあり，4種類の検証の総合でマスクパターンに対する必要検証項目のすべてを網羅している必要がある．この場合検証であるから重複検証は許されるが，抜けは許されない．しかしながら重複についても処理時間の増大につながるので重複部分が極力少なくなるような特定の考え方がDRCとして存在する（本論文付録に述べる）[13),14)]．

LSIはトランジスタ等の素子とそれらを結ぶ配線の集合体であるが，レイアウトパターンデータはこれを多数の図形で記述しており，ルールズファイルにおいても素子，配線の両方について検証項目が含まれている．実用化されているルールズファイルの中には上述のような特定の考え方に基づいて作られたルールズファイルがあり，このような場合には素子群と配線群に関してルールズファイルとレイアウトパターンデータとが相補的直交関係になっている（以下，このようなルールズファイルを直交型ルールズファイルと呼ぶ）．

直交型ルールズファイルを用いたDRCにおいてはレイアウトパターンデータとルールズファイルの両方をそれぞれ素子群対応と配線群対応の2つに分割し，対応する検証をそれぞれ別々に処理することで全体の検証が可能である（以下，これを直交分割型DRCと呼ぶ）．

直交分割型DRCにおいて全体の処理を2つに分けても処理時間については分ける前とまったく変わらないが，素子群の処理と配線群の処理では性質のかなり異なった演算処理が行われている．すなわち全図形データ量に対して素子群の図形データ量はROM（Read Only Memory）で1/2程度，RAM（Random Access Memory），PLA（Programmable Logic Array），DTP（Data Path）で1/3以下で，平均して素子群のデータ量は全図形データ量の1/3程度であり，残りは配線群のデータで，配線群の方がデータ量は多い．一方ルールズファイルが示す処理の複雑さについては，後述する実用ルールズファイルの場合178の検査タスク（検査単位機能）のうち143タスクが素子群の検証処理に割り当てられているのに対し，残り35タスクが配線群の検証処理に割り当てられている．要約すると素子群はデータ量は比較的少なく，複雑な検証処理を受けるのに対し，配線群はデータ量は比較的多く，簡単な検証処理を受ける．

DRCの検証処理については機能分割法，領域分割法のそれぞれに性質の合ったアーキテクチャ，手順が従来から提案されている．並列数が多くなったときの高速化に対する有効性について要約すると，機能分割はデータの移動が少ないが設計ルールの並列化が難しいという性質を持ち，したがってデータ量は多くてもよいが，複雑な検証処理に対しては並列の効果が低下する．一方，領域分割はデータの移動が多く，検証処理の複雑さには処理時間がまったく関係しないという特徴を持つ．したがって複雑な検証処理に対しては良いが，データ量が大きいと並列の効果が低下する．

我々は直交分割型DRCにおいて，検証処理を素子群と配線群の処理に大きく2つに分け，素子群の処理を領域分割のアーキテクチャ手順で行い，配線群の処理を機能分割のアーキテクチャ手順で行うことを提案する．これは従来の機能分割法，領域分割法と直交分割型DRCを組み合わせた新しい並列化手法に相当する．

本並列化手法について，実例のルールズファイルと実例の4種類のレイアウトパターンデータを使いシミュレーションにより並列DRCの実験を行った．その結果，本並列化手法が既存の2並列化手法より優位になる範囲が存在すること，レイアウトパターンデータの種類すなわちDTP，PLA，RAM，ROM等に応じて機能分割と領域分割の最適合並列台数範囲がずれてくることが観察できた．

以下2章に複合並列機による直交分割型DRC，3章に実験方法，4章に実験結果と検討，5章にまとめを述べ，付録に直交型ルールズファイルについて述べる．

## 2. 複合並列機による直交分割型DRC

直交分割型DRCの手順を図1に示し，これにつ

いて説明する．DRC処理のために入力されたパターンデータとルールズファイルの両方を，パターンデータは各図形層ごとに選別し，ルールズファイルは各項目ごとに選別してそれぞれ素子群，配線群に分ける．素子群のパターンデータを素子群のルールズファイルで領域分割の手法で検証する．同様に配線群のパターンデータを配線群のルールズファイルで機能分割の手法で検証する．領域分割部分，機能分割部分はそれぞれ必要に応じて並列処理で高速化を計る．発見された設計エラーデータは1カ所で集計して出力する．以後直交分割という言葉でこの手法を指すものとする．**図2**にこの論文で使用した直交分割型複合並列DRCのアーキテクチャを示す．このアーキテクチャは既存の機能分割法，領域分割法のアーキテクチャを複合化したものである[7),15)]．

図中ディストリビュータは全体を管理しパターンデータ等の入出力を行うコンピュータで，ワーカは各並列処理要素となるコンピュータである．ディストリビュータの下に2本の並列バスが配置され，バス1の下に機能分割法ワーカが，バス2の下に領域分割法ワーカが接続されている．図1の手順に従いディストリビュータはパターンデータを2群に分け，配線群についてはバス1に1度送り出し後述するタスク割付けを行うことで，処理はワーカ間のみで実行される．素子群のデータについてはディストリビュータは区分領域に分割し，バス2に繰返し放送する．領域分割法の各ワーカはディストリビュータに指定された領域を処理する．バス1，バス2ともにディストリビュータがメッセージコントロール方式により管理する．以後複合並列という言葉でこのアーキテクチャを示すものとする．

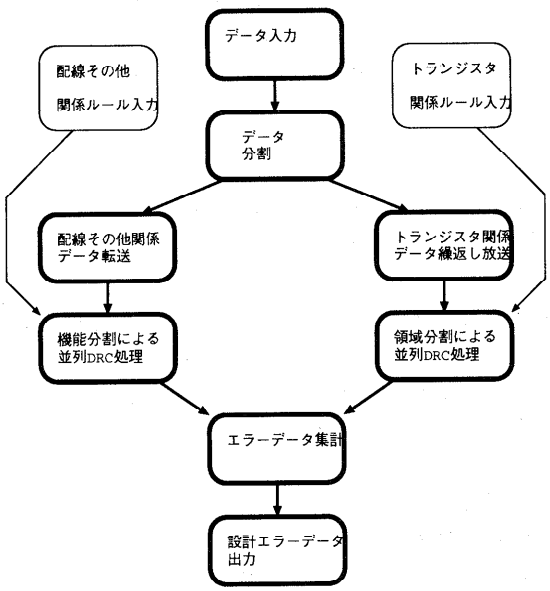



**図1** 直交分割型複合並列DRCの手順
Fig. 1 A block diagram of the orthogonal parallel DRC.

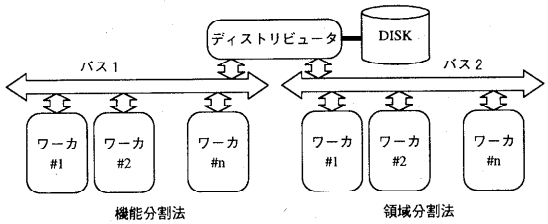



**図2** 直交分割型複合並列DRCのアーキテクチャ
Fig. 2 An architecture for the orthogonal parallel DRC.

## 3. 実験方法

実験に使用したLSIの設計データについて説明する．**表1**に示す178行から成るCMOS-LSIの実用レベルにあるルールズファイルを実験モデルとして使用し，レイアウトパターンデータとしては**図3**に示す5種類のレイアウトパターンデータを使用した．この図でDTP，PLA，ROM，RAMについては実用されているパターンデータでその外形寸法とデータ量を記した．TypicalについてはDTP，PLA，ROM，RAMの4種類のパターンをそれぞれ1/4ずつ切りとりこの実験のために合成したもので，外形は多角形のパターンである．

### 3.1 機能分割法と領域分割法の割振りについて

直交分割型DRCに合計32台のコンピュータを図2のように並列動作ができるように配置し，表1のルールズファイルと図3のパターンデータを使い図1の手順でDRC処理実験を行った．32台のうち，機能分

**表1** ルールズファイルの実例
Table 1 The rules file for the simulation.

| 検証項目 | 素子関係タスク数 | 配線関係タスク数 |
|---|---|---|
| Non45,non orthogonal checks | 9 | 11 |
| Definition of test layers | 13 | 2 |
| Fundamental check rules | 26 | 0 |
| Transister rules | 20 | 0 |
| Channel dope rules | 44 | 0 |
| Well rules | 11 | 0 |
| Gate rules | 13 | 0 |
| Contact rules | 0 | 20 |
| Proboundary check rules | 7 | 2 |
| タスク数合計 | 143 | 35 |

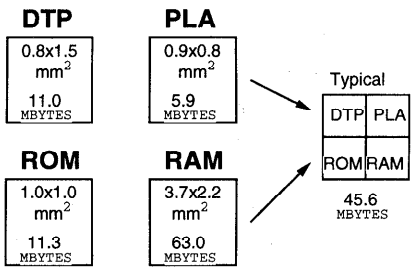



図3 実例パターンデータ
Fig. 3 The layout pattern data for the simulation.

割法の台数と領域分割法の台数の割振りを変えて5種類のパターンのDRC処理時間を調べ，各パターンに対する最適の割振りを求めた．

### 3.2 直交分割型複合並列DRCの優位範囲の測定について

パターンデータのTypicalに対して，直交分割型複合並列DRCについては最適の割振りで，合計並列台数を変えた場合と図2のデータ転送バスの速度を変えた場合について，直交分割型複合並列DRCと機能分割法DRCと領域分割法DRCの処理時間を計り比較した．さらにパターンデータTypicalで得られた直交分割型の優位範囲について合計台数8台でDTP, PLA, RAM, ROMのパターンデータに対しても測定を行った．

### 3.3 並列DRCシミュレータについて

機能分割法DRC，領域分割法DRC，直交分割型複合並列DRCの3種類のシミュレータについて説明する．これらのシミュレータはバスの転送速度を変えた場合と，並列台数を変えた場合についてDRC処理時間の算出を行うためのものである．

#### 3.3.1 機能分割法並列DRCのシミュレータ

機能分割法でDRC処理時間を算出するためには各タスクの処理時間を知る必要がある．各タスクの処理時間はそのタスクへの入力データ量から算出できる．すべてのDRCは図形演算タスクで検証対象を検出し，寸法測定タスク等でそれを検証する．図形演算タスクの入力データ量は層ごとの図形データ量から知ることができる．図3に示した5種類のパターンについては層ごとの図形データ量をあらかじめ測定してあるので各図形演算タスクの入力データ量が決められる．1つの図形演算タスクが他のタスクのための中間データを生み出す場合があるが，この中間データの量を後段タスクの入力データ量として知る必要がある．図形演算タスクの入力，出力データ量の関係はこの実験に使用したパターンデータおよびタスクについての実験結果から**表2**に示した係数を得た．

**表2** データ量を表す係数
Table 2 Mean rates of I/O data quantity of DRC tasks.

| タスクの種類 | $\frac{\text{出力データ量}}{\text{入力データ量}}$ の値 |
|---|---|
| or | 1.0 |
| and | 0.5 |
| stradding | 0.17 |
| avoiding | 0.17 |
| in | 0.17 |
| out | 0.17 |
| andnot | 1.0 |
| butting | 0.25 |
| outside | 0.5 |
| coincident | 0.5 |
| edge | 0.5 |

**表3** シミュレーションに用いたパラメータ群
Table 3 Parameters of the simulation.

| 機能分割法に関するパラメータ | | |
|---|---|---|
| DRC処理速度 | $S_{drc}$ | 100kbyte/sec |
| 並列バス通信速度 | $S_{pw}$ | 0.2, 2, 20 Mbyte/sec |
| ベクトル数偏差率 | $R_v$ | 90-110% |
| パターンデータ誤り率 | $P_e$ | 0.0001 % |
| ワーカ間制御時間 | $T_c$ | 20 msec |
| DRC処理速度偏差率 | $R_{drc}$ | 90-110% |
| 通信速度偏差率 | $R_{pw}$ | 90-110% |
| 領域分割法に関するパラメータ | | |
| DRC処理速度 | $S_{drc}$ | 100kbyte/sec |
| 並列バス通信速度 | $S_{pw}$ | 0.2, 2, 20 Mbyte/sec |
| 1ブロック転送量 | $V_b$ | 131072 byte |
| パターンデータ誤り率 | $P_e$ | 0.0001 % |
| ディストリビュータ、ワーカ間書き込み終了通知時間 | $T_{tdw}$ | 20 msec |
| 次領域割り当て時間 | $T_{con}$ | 20 msec |
| DRC処理速度偏差率 | $R_{drc}$ | 90-110% |
| ベクトル数偏差率 | $R_V$ | 90-110% |

また本論文では**表3**に示す実物実験から得たシミュレーション用パラメータ群を用いた．表中の各項目の説明をする．DRC処理速度 ($S_{drc}$) は各タスクが毎秒処理できるパターンデータ量の平均値を示す．タスクの種類によってこの値は異なるが，これはDRC処理速度偏差率 ($R_{drc}$) で吸収する．並列バス通信速度 ($S_{pw}$) は機能分割と領域分割のそれぞれの並列処理を構成しているバスの通信速度を表す．機能分割の場合にはパターンデータの転送時に待ち時間を発生する場合があるが，これは通信速度偏差率 ($R_{pw}$) で吸収する．ベクトル数偏差率 ($R_v$) は図形の濃淡にともなう図形ベクトル数の変動を表すものとする．パターン

データ誤り率 ($P_e$) は検証対象となるパターンデータに含まれる設計エラーデータの比率を表す．ワーカ間制御時間 ($T_c$) はタスクの割当て，ワーカ間の通信の制御等のシステムの制御に必要な時間である．これらのパラメータはすべて技術的に実現可能な範囲での数値である[7),15)]．

各種のタスクについて表3に示したDRC処理速度とDRC処理速度偏差率の範囲内で各タスクの処理時間 ($T_{tk}$) が入力データ量に比例する．したがって各タスクの入力データ量からそのタスクの処理時間（絶対値）を算出できる．発生した中間データを転送する必要がある場合は，その中間データの量 ($V_{bt}$) と表3の並列バス通信速度から必要な転送時間を算出できる．このようにしてすべてのタスクのDRC処理時間と各タスク間のデータ転送時間 ($T_{bt}$) を算出した．

設計エラーデータを検出するタスクでは，そのタスクへの入力データ量 ($VI$) とパターンデータ誤り率 ($P_e$) から設計エラーデータの量を算出し，さらにその転送時間 ($T_{back}$) を算出する．

ここに

$$T_{tk} = \frac{VI \cdot R_v}{S_{drc} \cdot R_{drc}}$$

$$T_{bt} = \frac{V_{bt}}{S_{pw} \cdot R_{pw}}$$

$$T_{back} = \frac{VI \cdot P_e}{S_{pw}}$$

また機能分割ではタスクの割当て方法が非常に大きな要素となるが，これにはTALSを用いた[15)]．すなわちタスクグラフを作り，これに従って各ワーカを割り当て，前述した方法で順次に各タスクの処理時間および転送時間を算出し，直列タスク系列の各タスクの処理時間と転送時間の和 ($T_{seq}$) でその直列タスク系列の処理時間とした．すなわち $h$ 番目の直列タスク系列の処理時間は，

$$T_{seq(h)} = \sum_h (T_c + T_{tk} + T_{bt} + T_{back})$$

ここに $\sum_h$ は $h$ 番目の直列タスク系列について合計するものとする．

この直列タスク系列の処理時間の最大のものを機能分割型DRCの処理時間とした．

### 3.3.2 領域分割法並列DRCのシミュレータ

領域分割法の手順については，デイストリビュータは全データを多数のブロックに分けて間欠的に繰返し放送形式で流しながら空いているワーカに次々とDRC未完了の領域を割り当ててゆき，各ワーカは全ルールズファイルを持ち，割り当てられた領域のデータを放送された全データの中から選び，担当した領域のDRCを行う，領域分割数を並列数の数十倍に設定しているので各ワーカは平均して数十回の領域割当てを受けることになり各領域の図形密度の濃淡にかかわらずに各ワーカの処理する仕事量を平均化する作用が働き，このため1つのワーカの処理を他のワーカが待つという時間をきわめて小さくできる．

処理時間は表3のパラメータ群に従って算出する．表中，DRC処理速度 ($S_{drc}$)，並列バス通信速度 ($S_{pw}$)，パターンデータ誤り率 ($P_e$)，DRC処理速度偏差率 ($R_{drc}$)，ベクトル数偏差率 ($R_v$) は機能分割の場合と同じである．

ワーカは次領域の割当てを受けると（次領域割当て時間 ($T_{con}$)），放送形式で転送されてくるデータを1ブロックずつ（1ブロック転送量 ($V_b$)）間欠的に並列バス通信速度に従って取り込み，担当領域のデータを選別する．また1ブロック転送終了ごとに通知を受ける（表3のディストリビュータ，ワーカ間書き込み終了通知時間 ($T_{tdw}$)）．

全データ（全データ量を $V_0$ とする）を一巡するとその領域のデータは確定する．担当領域のデータを取り込むのに要する時間を $T_{get}$ とする．領域分割ではルールズファイルのタスク数はあらかじめ分かっている（全タスク）から入力データ量（$V_{in}$ すなわちこれは担当領域のデータ量）とタスク数 ($N_{task}$) と表3のDRC処理速度 ($S_{drc}$) からその領域のDRC処理時間 ($T_{drc}$) は算出できる．

表3のパターンデータ誤り率 ($P_e$) と入力データ量から，この領域についての検証結果である設計エラーデータの量を算出でき，これを並列バスを使用してディストリビュータに送り返す（返送時間を $T_{back}$ とする）．これらの時間を計算して1つの領域の処理時間を求め，表3の次領域割当て時間の後，次の担当領域の指定を受ける．全ワーカがこれを繰り返してすべての領域が完成すれば終わる[7)]．

領域分割数を $n$ とすると各パラメータの関係は次のようになる．

$$T_{get} = \frac{V_0}{V_b} \cdot \left( \frac{V_b}{S_{pw}} + T_{tdw} \right)$$

$$T_{drc} = \frac{V_{in} \cdot N_{task}}{R_{drc} \cdot S_{drc}}$$

$$T_{back} = \frac{V_{in} \cdot P_e}{S_{pw}} + T_{tdw}$$

$$V_{in} = \frac{V_0 \cdot R_v}{n}$$

$k$ 番目のワーカが終了までに要する時間を $T_k$ とすると

$T_k = \sum_k (T_{con} + T_{get} + T_{drc} + T_{back})$

ここに $\sum_k$ は $k$ 番目のワーカに割り当てられる領域についての和である．

この $T_k$ をワーカ台数分算出し，その最大のものを領域分割法のDRC処理時間とした．

### 3.3.3 直交分割型複合並列DRCのシミュレータ

前述した機能分割，領域分割の両方のシミュレータを組み合わせ，さらに複合並列システム制御時間等を加え，機能分割部分にはトランジスタ関係を除くルールとパターンデータを与え，領域分割部分にはトランジスタ関係のルールとパターンを与えた．両方の部分についてそれぞれのDRC完了時間がより大きな方をとって複合並列DRCの完了時間とした．

## 4. 実験結果と検討

### 4.1 直交分割型複合並列DRCにおける機能分割法と領域分割法の割振りについて

複合並列で直交分割型DRCの場合の並列台数の合計が32台で割振りを変えた場合の各パターンにおけるDRC処理時間の変化を図4に示す．図中グラフ横軸の並列数は並列台数を表している．この図からパターンの内容によってDRC処理時間が最小になる割振りが変化すること，また最小を示す範囲が種類によっては広いものもあることが分かる．この最適点の変化と範囲をパターンの性質を示すデータ量の比率のグラフ上に表したのが図5である．この図で背景になっているデータ量の比率と直交分割型DRCの最適点の間には明らかに強い相関が見られ，たとえばROMのように素子群の比率の高いものは領域分割の並列台数の多いところに，またPLAのように配線群の比率の高いものは機能分割の並列台数の多いところに最適点があることが分かる．

### 4.2 直交分割型複合並列DRCの優位範囲の測定について

前述のTypicalというパターンは4種類のパターンを合成したものであり平均的な性格を示すので，これを使い機能分割法，領域分割法および直交分割法の比較を行った．領域分割の分割数についてはシミュレーション実験により，分割数が並列数より多ければ3者のDRC処理時間にまったく影響しないことを事前に確かめた．

図6，図7，図8にバスの通信速度が200 KByte/sec，2 MByte/secおよび20 MByte/secの場合の並列台数（図中グラフ横軸では並列数で表示する）に対する処理時間の変化を示す．これらの図から通信速度が遅い場合（200 KByte/sec）には並列台数8～32台

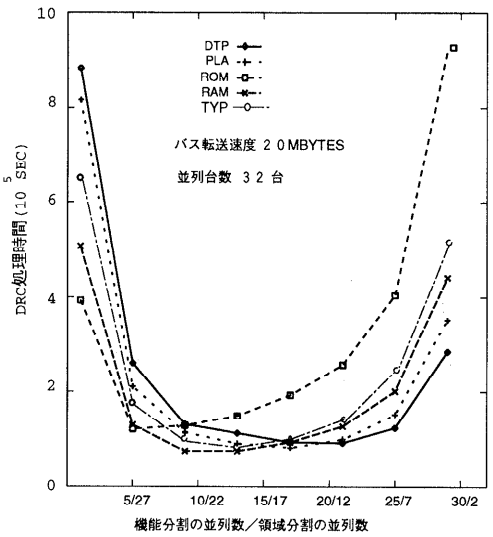



図4 パターンの種類と割振りの変化
Fig. 4 Layout pattern vs. optimum rates.

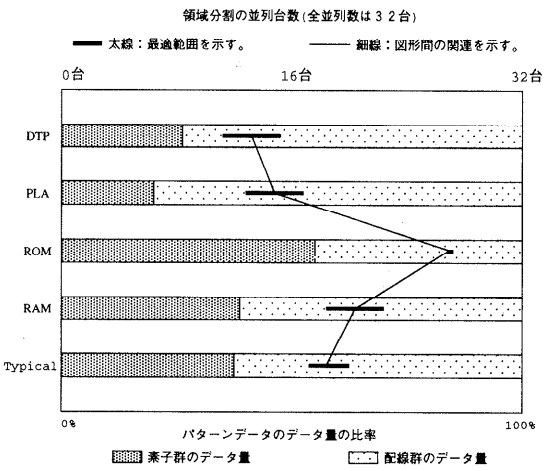



図5 データ量の比率と最適点の変化
Fig. 5 Layout pattern vs. optimum condition.

で直交分割が最優位，32台以上で領域分割が最優位，通信速度が速い場合（2 MByte/sec，20 MByte/sec）には並列台数8～11台で直交分割が最優位，11台以上で領域分割が最優位，また8台以下ではいずれの場合も機能分割が最優位という結果を得た．表4に図6，図7，図8の並列数8の部分を時間数（hour）で示す．約90時間かかるDRC処理時間について直交分割型複合並列DRCでは通信速度が遅い場合に約4時間，通信速度が速い場合には約2時間短縮されている．

図9，図10に8台並列の場合と32台並列の場合の通信速度と処理時間の関係を示す．これらの結果から並列台数が8台では通信速度の広い範囲で直交分割

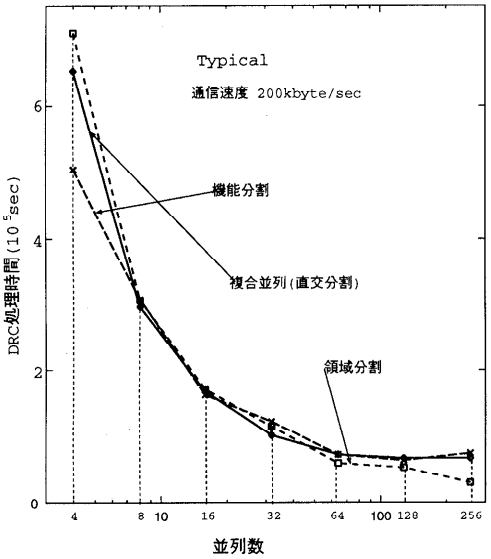



図6 並列台数と処理時間（200K）
Fig. 6 Processing time vs number of parallel (200 K).

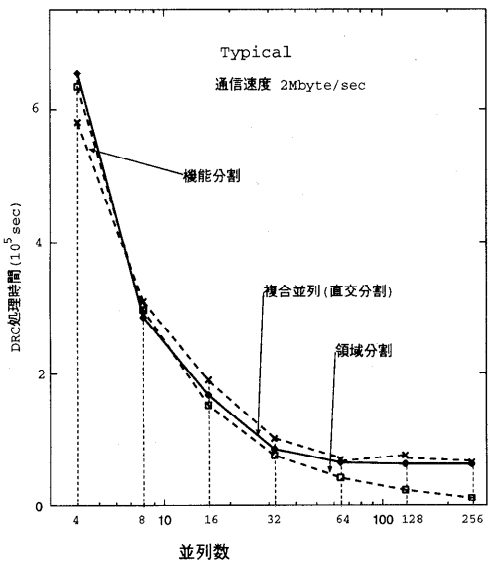



図7 並列台数と処理時間（2M）
Fig. 7 Processing time vs number of parallel (2 M).

が優位，32 台並列の場合は通信速度が 1 MByte/sec 以下の遅い範囲で直交分割が優位，それ以外では領域分割が優位であることが示された．

これらの実験結果から各並列化手法の優位範囲が**図 11** のように得られる．この図はパターンデータ Typical についての 3 手法の優位範囲を図 6，図 7，図 8，図 9，図 10 から抽出して示したものでバス通信速度を縦軸に並列数を横軸にとった平面の中で 3 手法のそれぞれの優位範囲を示したものである．

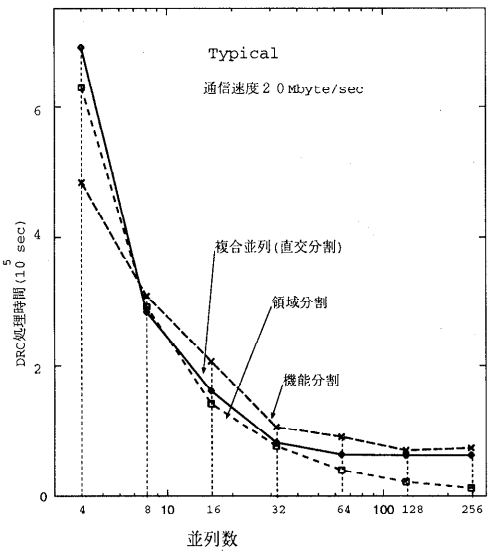



図8 並列台数と処理時間（20M）
Fig. 8 Processing time vs number of parallel (20 M).

表4 グラフの局部点の数値
Table 4 Numerical values of the graph.

| 並列数 | 8 | | |
|---|---|---|---|
| 通信速度 (kbyte/sec) | 200 | 2000 | 20000 |
| 領域分割 | 93.1 | 86.1 | 84.2 |
| 機能分割 | 90.8 | 90.8 | 86.1 |
| 複合並列 | 86.4 | 83.9 | 81.6 |

[ 時間 (hour) ]

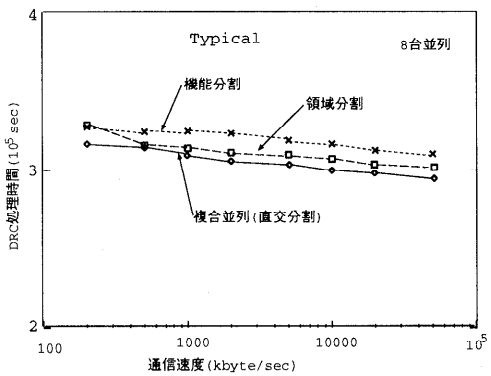



図9 バスの通信速度と処理時間（8 台）
Fig. 9 Processing time vs transfer rate (8 parallel).

直交分割型複合並列 DRC が優位にある 8 台並列での 3 手法の処理時間を図 **12**，図 **13**，図 **14** に示す．これらの図から Typical では機能分割 3 台，領域分割 5 台で，DTP では機能分割 4 台，領域分割 4 台で，ROM では機能分割 1 台，領域分割 7 台の割振り

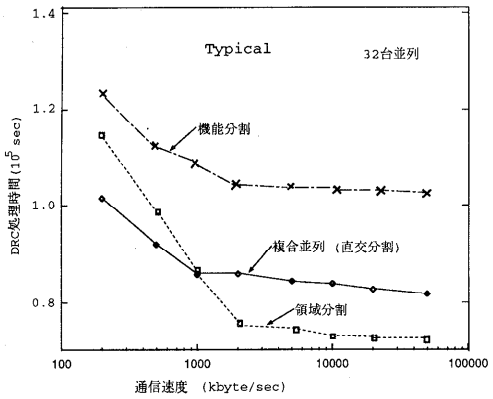



図10 バスの通信速度と処理時間（32台）
Fig. 10 Processing time vs transfer rate (32 parallel).

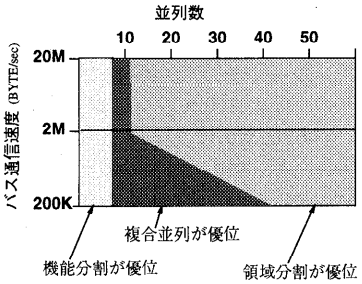



図11 優位範囲
Fig. 11 A dominant scope.

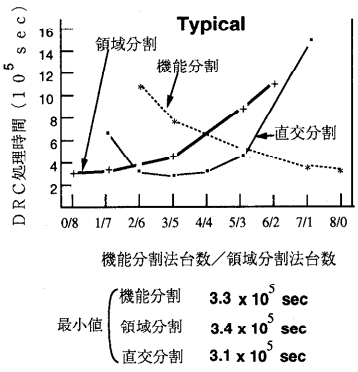



図12 3手法の比較（8台）Typical
Fig. 12 A comparison of three schemes in 8 parallel (Typical).

のときに直交分割がそれぞれの場合に最高速になることが示され，図11がTypicalに対してだけではなくDTP，ROMに対しても成立し，したがってパターンの種類によらずに図11が成立することを示している．機能分割法はバス転送速度にはあまり依存しないのでバス転送速度が速くなっても大きくは変化しない，この方法の場合並列化による高速化はワーカ台数約20

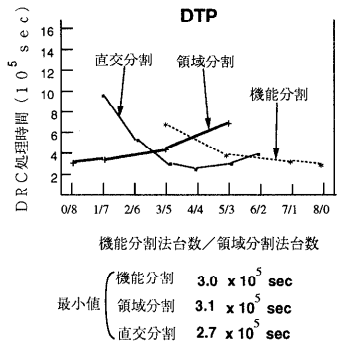



図13 3手法の比較（8台）DTP
Fig. 13 A comparison of three schemes in 8 parallel (DTP).

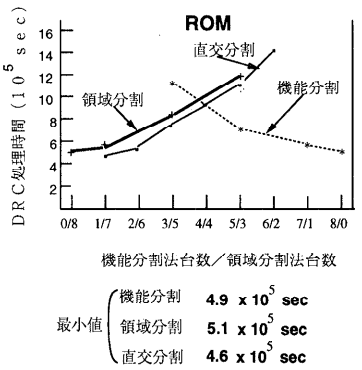



図14 3手法の比較（8台）ROM
Fig. 14 A comparison of three schemes in 8 parallel (ROM).

数台で飽和する．領域分割法はDRC速度とバス転送速度の比がある一定値以上になると全体の効率が良くなるのでバス転送速度の増加とともに並列化による高速化が進み，バス転送速度が20 MByte/secに近づくとDRC処理時間がワーカ数分の1に近づくまで減小する．

直交分割型複合並列DRCではこの場合図3のTypicalに示されるように約3分の1の量のパターンデータを領域分割部分に割り当て，残り約3分の2のパターンデータを機能分割部分に割り当てる，また表1に示されるようにルールズファイルのうちのトランジスタ関係の143のタスクが領域分割部分に適用され，配線その他関係35のタスクが機能分割部分に適用される．この状態で機能分割法3台，領域分割法5台の近辺の点が最適点となる．

直交分割型複合並列DRCにおける機能分割部分ではルールの引用関係は簡単（たかだか3処理層）で総タスク数が35と少ないので，3台のワーカで割り当

てられたデータを高速に処理する．一方，領域分割部分ではルール側の高速化の寄与はタスク数が178から143に減少するだけであるが，ワーカ数が8台から5台に減少するのに対して割り当てられたパターンデータ量が約3分の1に減少するのでやはり有効に高速化でき，直交分割型複合並列DRCが同じワーカ台数で機能分割法DRC，領域分割法DRCのいずれよりも高速になる．

## 5. まとめ

flat DRCを高速化する目的で並列化する手法について従来からある機能分割法DRC，領域分割法DRCと複合並列アーキテクチャ上における直交分割型DRCとを，実用レイアウトパターン，実用ルールズファイルを用いてシミュレーション実験によりDRC処理時間の比較を行い次のような結果を得た．

(1) 実用ルールズファイルの中に直交型ルールズファイルがあり，これを用いたDRCについては直交分割型複合並列DRCが可能でかつ有効であることを提案した．

(2) 既存の機能分割型DRC，領域分割型DRCとこの直交分割型DRCを比較した．その結果，並列数が8～11の範囲で直交分割型DRCが最も優位性があることが分かった．

(3) 直交分割型DRCではレイアウトパターンデータの種類によって機能分割と領域分割の最適合並列台数が変化する．

(4) 領域分割法DRCはデータを転送するバスが十分高速の場合には並列数で11以上で最も高速にできる並列化手法であるが，データ転送バスが低速の場合には，さらに並列数の大きな範囲まで直交分割型DRCに優位性がある（図10）．

また今後の問題として，直交分割型DRCの並列数の割振りにフレキシブルに対応する複合並列アーキテクチャ，最近のLSIの配線層数の増加にともなう直交分割型DRCの対応，さらに高速化に適したDRC並列処理手法（演算処理速度とバス通信速度の配分）の研究が必要と考える．

## 参考文献


1) Tailer, G. and Ousterhout, J.: Magic's Incremental Design Rule Checker, *Proc. 21st Design Automation Conference*, pp.160–165 (1984).
2) Suzuki, G. and Okamura, Y.: A Practical Online Design Rule Checking System, *Proc. 27th Design Automation Conference*, pp.246–252 (1990).
3) Hedenstiena, N. and Jeppson, K.: The Halo Algorithm–An Algorithm for Hierarchical Design of Rule Checking of VLSI Circuits, *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Vol.12, No.2, pp.265–272 (1993).
4) Hedenstiena, N. and Jeppson, K.: A Parallel Hierarchical Design Rule Checker, *The European Conference on Design Automation*, pp.142–146 (1992).
5) 築添 明，小沢時典，酒見淳也，三浦地平，石井建基：VLSIマスクパターンに対する論理演算と交点計算を同時に処理するパターン論理演算手法，信学論（D），Vol.J69-D, No.6, pp.975–983 (1986).
6) Nielson, R.: Algorithmically Accelerated Cad, *VLSI SYSTEM DESIGN*, Vol.7, No.2, pp.65–66 (1986).
7) 上坂達生，南 利秋，松木俊寿，田丸啓吉：LSIデザインルールチェックの並列化における領域分割法のアーキテクチュアモデル，情報処理学会論文誌，Vol.36, No.11, pp.2540–2548 (1995).
8) Carlson, E. and Rutenbar, R.: Mask Verification on the Connection Machine, *Proc. 25th Design Automation Conference*, pp.134–140 (1988).
9) Macomber, S. and Mott, G.: Hardware Acceleration for Layout Verification, *VLSI DESIGN*, Vol.6, No.7, pp.18–27 (1985).
10) Marantz, J.: Exploiting Parallelism in VLSI CAD, *Proc. IEEE ICCD'86*, pp.442–445 (1986).
11) Bier, G. and Pleszkun, A.: An Algorithm for Design Rule Checking on a Multiprocessor, *Proc.22nd Design Automation Conference*, pp.299–304 (1985).
12) Banerjee, P.: *Parallel Algorithms for VLSI Computer-Aided Design*, PTR Prentice Hall (1994).
13) 飯塚哲哉，森祥次郎ほか：CMOS超LSIの設計，培風館，p.266 (1989).
14) 市川 浩，山本博文，藤本 豊，碇 逸男，沢田透：LSIアドバンストレイアウト検証ツールの開発と実用化，MSC技報（三菱電機セミコンダクタソフトウエア技報），No.6, pp.115–119 (1993).
15) 河野一郎，小野寺秀俊，田丸啓吉：デザインルールチェック並列処理化の一手法—並列スケジューリングによる手順割り当て，信学技報，No.FTS93-32,VLD93-56, pp.39–46 (1993).
16) Ohtsuki, T.: *Layout Design and Verification*, pp.251–263, North-Holland (1986).

## 付　　録

### 直交型ルールズファイルについて

LSIはトランジスタ，コンデンサ等の回路素子（以下，素子群）と接続線，コンタクト，スルーホール等からなる配線（以下，配線群）から構成されている．レイアウト設計完了段階では，これらの素子群，配線群を表す図形はそれぞれの属している層に描かれている．たとえばトランジスタは拡散層やwell層等に描かれ，配線はメタル層やコンタクト層に描かれている．

一方このレイアウト設計を検証するルールズファイルにも素子群，配線群の両方の検査内容が含まれている．その内容は本文の表1に例が示されているように約1対4程度の比率で素子群に対する検査項目の方が項目数が多い．この例を項目別に見るとDRCでは，トランジスタが正しくトランジスタの構造を表す図形群になっているか，配線群が正しい形状で他の図形との間隔が保たれているかに最大の重点が置かれているのがよく分かる．

この素子群と配線群の接点にあるのがトランジスタから出ているソース，ゲート，ドレイン等の電極と配線群の交点にあるコンタクトポイント（接点）で，レイアウトパターンの中では素子の電極は拡散層，ポリシリコン層等に図形が描かれており，コンタクトポイントはコンタクト層に描かれている．

表1のルールズファイルの実例のようにコンタクト層の検査を配線群の中で閉じてしまい素子群との間の検査はDRCでは行わないようにすると本文に述べた直交型ルールズファイルになり，これを使用したDRCでは直交分割型DRCが可能になる．実際にこのようなルールズファイルでは，たとえば拡散層とメタル層から新しい検査層を作り，その検査層とコンタクト層の検証はERCまたはLVSの検査に含まれているのである．

一般にIC設計後のレイアウトパターンに対するチェックについてはDRC，ERC，LVS，LPEの4種類の検査が機械で行われている．このうちDRCについては各設計要素を構成する図形の間隔，幅等の検証が割り当てられており，素子間の接続についてはERCまたはLVSに割り当てている[13),14),16)]．このように割り付けるとDRCではトランジスタ関係の設計規則はトランジスタ関係の層を検査し，それ以外の配線その他の関係の設計規則はトランジスタ関係以外の層すなわち配線その他の関係の層を検査するようになる．すなわち素子群と配線群でルールズファイルとパターンデータとが相補的直交関係になる．こうなると非直交成分は検査項目から省くことが可能になるから直交分割型DRCが可能になる．

次に直交型ルールズファイルが存在する理由になっている自動的検証項目の概念の定義の一例を示す．

(1) DRC（Design Rule Checker: 設計基準検証）
ウエーハプロセス製造技術にかかわるマスクパターン作成上の設計基準（マスクパターンの間隔，幅等の基準）の検証．

(2) ERC（Electrical Rule Checker: 電気的素子接続検査と回路接続抽出）
正しい接続の検証，すなわち電気的に正常でない配線，トランジスタ（配線ショート，動作しないトランジスタ等）の検査．

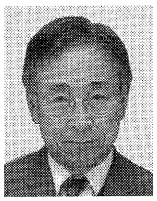
(3) LVS（Layout Versus Schematic: 回路接続比較検証）
レイアウトパターンが論理回路図と等価であるかを比較する検証．

(4) LPE（Layout Parameter Extraction: 回路定数算出）
レイアウトパターンから寄生容量，抵抗等のパラメータや，それらを含む等価回路を抽出し，シミュレータと並用して行う特性の検証．

このように分類してそれぞれの検査ファイル（たとえばルールズファイル）を作る方法はレイアウト検証全体をカテゴリー別により整理された考え方に基づいたものである．上記4項目の検査では，レイアウトパターンを直接検証する部分については使用する検査タスク（本文表2）も共通して使用される．この場合上記4種類の検証が補完しあって必要な検証項目をもれなく網羅しているのであり，このような考え方に基づいたルールズファイルは，たとえば本文表1のように市場に存在する．

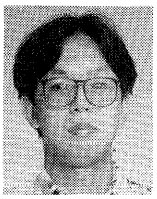
コンタクトの設計を標準規格化した場合，その標準寸法は製造プロセスの変動等でしばしば変更される場合が多いので，設計段階ではこの部分を抽象的に表現しておく方が機械的に寸法の変更ができて有利である．抽象的表現になったとき，この部分を検証する項目はDRCとLVSで共通にすることができる．設計段階でマスクパターンの実寸設計を行う場合は，これの検証に直交型ルールズファイルを使うことはできない．直交型ルールズファイルは今後増加の傾向にある存在である．

(平成9年 4 月3日受付)
(平成9年10月1日採録)



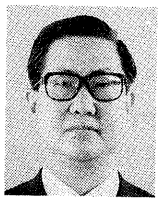
**上坂　達生**（正会員）

1937年生．1960年大阪大学理学部物理学科卒業．同年三菱電機（株）に入社．1984年三菱電機セミコンダクタソフトウエア株式会社に出向．1989年熊本電波高専情報工学科教授．この間コアメモリ，ASIC，LSI用CAD等の研究に従事．電子情報通信学会会員．



**松木　俊寿**

1971年生．1994年京都大学工学部電子工学科卒業．1996年京都大学大学院工学研究科修士課程（電子工学専攻）修了．同年パイオニア（株）に入社．現在パイオニア（株）モーバイルエンタテインメントカンパニー技術統括部第三商品開発部開発一課にてカーナビゲーション用ASICの開発に従事．



**田丸　啓吉**（正会員）

1936年生．1958年京都大学工学部電子工学科卒業．1960年京都大学大学院修士課程（電子工学専攻）修了．工学博士．1960年（株）東芝に入社．1979年京都大学工学部教授．この間マイクロコンピュータのアーキテクチャ，LSIの設計手法，LSI用CAD等の研究に従事．「ハードウェア技術」（オーム社），「論理回路の基礎」（工学図書），「マイクロコンピュータ入門」（日刊工業）等を著作．電気学会，電子情報通信学会，ACM，IEEE各会員．