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

上坂達生[†] 南利秋[††,☆]
松木俊寿[††] 田丸啓吉[††]

LSIのデザインルールチェック（以下DRC）はLSIの設計検証工程の中でも最も時間を要するので高速化が強く要求されている工程である．DRCを高速化する方法の一つにこれを並列処理化する方法がある．また並列処理化する方法の一つに，レイアウトパターンデータを領域分割して，複数のコンピュータで並列にDRCをする方法（領域分割による並列処理化の方法）がある．この領域分割による並列処理化の方法では分割と割付けとDRCの処理手順について複数の組合せがあり，その方法のそれぞれについてそれを実行するのに適したハードウェアアーキテクチュアが考えられる．この論文では分割したデータを並列部分へ転送する方式に関して2種類の手順を考え，このそれぞれに対してこれを実行できるアーキテクチュアモデルを構成し，並列部分の台数と領域分割数を変数にしてシミュレーション実験により評価した．その結果それぞれの並列アーキテクチュアのより優位な範囲が解明できた．

A Study on the Two Kinds of Architecture for A Parallel Processing through Area Partitioned Design Rule Checking

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

Among the LSI designing process, LSI design rule checking (DRC) is required especially to reduce its prohibitive runtime. One of the accelerating method of DRC is parallel processing, and one of the parallel processing method is area partitioned design rule checking which is dividing the layout pattern into multi-domain. We propose two kinds of architecture, one is composed of an upper computer and lower computers (we call them a distributor and workers), the other is composed of a distributor and workers also but each worker is composed of two special purpose parts, a communication and selection part, and a DRC processing part. Those two architecture models are evaluated competitively through simulation. Consequently, we have defined superior territory of each architecture model.

1. まえがき

LSIの大規模化，高密度化によってその検証工程（デザインルールチェック以下DRC）を高速化する必要性はますます大きくなっている．従来LSI全体にわたるDRCは主として大型計算機上で行われてきた[1]．今日ではワークステーションの大幅な性能向上によってそれに代わる場合が多いが，いずれにせよレイアウトパターンデータ量が全体で数百メガバイトからギガバイト近くにもなるため，その処理時間は実時間で数日にもなり，設計工程上の障害となっている．DRCを高速化する方法は現在まで多方面から研究されており，これを要約すると次の4種の手段が挙げられる．

(1) DRCプログラムで使用しているアルゴリズムの改良により高速化する．
(2) DRCを行う計算機をより高速の計算機に変える方法．
(3) LSIの設計手法との組合せで高速化する．
(4) DRCを並列化することによって高速化する方法．

これらの方法のうち(1)については従来から多くの研究が行われたもので[2),3)]，DRCは基本的に時間計算量が$\Theta(n^2)$の演算が多く含まれているが，アルゴ

† 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，キャノン株式会社
Presently with Canon Corporation

リズム改良が行われた結果時間計算量が $\Theta(n)$ に近づいている．したがってこの方法による高速化は限界に近い．次に(2)については計算機の進歩と共に，この方法は可能性はあるが，DRCを飛躍的に高速化する特別な方法ではない．(3)はレイアウトパターンを設計中の段階で新しく設計されて加わった部分をその度にDRCを行い設計時間とDRCの時間を重ねて見かけ上のDRCの時間を小さくする方法，およびDRCずみのCELLの組合せでLSIの設計を行いCELL内のDRCを省略してDRCにかかる時間を小さくする方法などがこれに含まれる[4),5)]．このうち前者については対話形DRCまたはインクレメンタリィあるいはオンラインDRCと呼ばれるもので単純な検査項目についてのみ検査する場合が多く，ある領域内のデータについてのみ検査されるため領域境界上では不完全さが残る．このため設計の最終段階でのチップ全体に対するDRCは不可欠である．

以上のような理由でDRCを大幅に高速化するためには(4)の方法すなわちDRCを並列化することにより，またこれに合わせてDRCを実行する機械のハードウェアを根本的に改良する以外に方法は考えられない．DRCを並列処理化する方法として次の全く性質の異なる2種類の方法がある[6)]．すなわち

1. 検証の手段あるいは手法を記述したLSIの設計規則がDRCの各段階において順番に検証されるが，この検証手順を記述したルールズファイルの処理機能を各機能ごとに分割して，それぞれ別の計算機に割り当てて並列処理をする方法（機能分割による並列処理化の方法）

2. 検証の対象物であるLSIの図形を各区分領域に分割して，それぞれの区分領域を複数のコンピュータに割り当てて並列にDRC処理をする方法（領域分割による並列処理化の方法）

DRCの小規模の並列化については商用化されている例があるが，大規模（数十台以上）のものについては例がない．この論文は大規模システムを対象にした領域分割によるDRCの並列処理化に関するものである．すなわち，領域分割によってDRCを並列処理化する方法のうち高速化に適していると考えられる2種類の並列化手順と，それに対応したハードウェアアーキテクチュアを提案する．次にこの2種類のアーキテクチュアについて，分割区分領域の分割数と並列処理する部分の並列数を変数にして主としてシミュレーション実験により適応範囲を測定，比較した．その結果，原レイアウトパターンデータ量，領域分割数，並列数によって区分領域データの転送時間と区分領域のDRC処理時間の比が変わることによる各アーキテクチュアの高速化に対する適応範囲を明らかにした．以下，第2章にDRC並列化手法の要点について述べ，第3章でこの論文で提案する2種類のDRC並列処理アーキテクチュアモデルを説明する．第4章でこの実験に用いたシミュレータについて述べ，第5章で実験結果とその検討について述べる．

2. DRC並列化手法の要点

領域分割による並列化はDRCの大規模な並列化およびそれによる高速化に適した方法でその理由を次に述べる．DRCに多大の時間を要する原因は，LSIのレイアウトパターンデータの量が大きいことである．レイアウトパターンデータは一般にCIF，GDS2などの流通形式で記述されているがDRCに際してはこれを単層表現に変換する必要がある．この場合データの量はさらに数倍から数十倍に増えて1ギガバイトを超えることもある．さらにDRCの処理の途中でかなりの量の中間データを発生し，これも対象データ量を大きくする．領域分割による並列処理化の方法では，レイアウトパターンデータが局所的な性質を持っていることを利用して，これを多数の領域に分割し放送形式のデータ転送によって並列しているコンピュータに高速で受渡しする方法をとることができる[7)]．また機能分割による並列化が十数台の比較的少ない並列度で高速化が飽和するのに対し[8)]，領域分割による並列化は数百の大きな並列度まで高速化が有効である（図9参照）．

したがって高速化の目的で並列度を大きくし，1台当たりの処理データ量を小さくすることができる．またこれによって各並列コンピュータの持たなければならない記憶容量を比較的小さくできる．

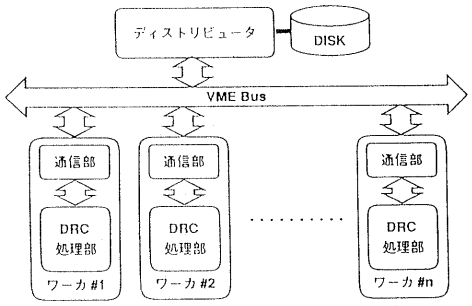
3. アーキテクチュアモデル

DRCを並列処理するためのアーキテクチュアとしては，レイアウトパターンデータやプログラムを入力し並列システム全体を制御する役目を担う1台の上位コンピュータ（以下ディストリビュータと呼ぶ）の下に並列コンピュータ（以下ワーカと呼ぶ）を配した形状が従来から提案されているが[9),10)]，ディストリビュータ，ワーカのそれぞれの役割分担の方法で数種のアーキテクチュアモデルが考えられる．レイアウトパターンデータはDRCに入力される時には一般には流通形式（CIFまたはGDS2）で記述されている．並列DRC処理を行うためにはこれを一旦単層表現形式に展開し各領域に分割し，さらにその区分領域デー

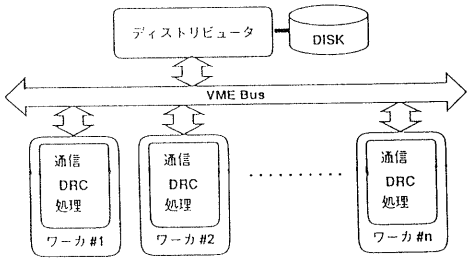
タを各ワーカに配送しなければならない．ディストリビュータが単層表現形式への変換および領域分割の役割を担うモデルも可能で，この場合ディストリビュータは区分領域データを一時に一領域分ずつ生成し順番にこれをワーカに配送することになり，ワーカ台数が多い場合にはワーカの待ち時間が大きくなるという難点がある．この論文で提案しているアーキテクチュアモデルは領域分割の役割をワーカ側に持たせた次の二つのモデル（**図1**）でいずれも領域分割の工程を並列化しており，同時に複数領域の区分データを生成する機能を持たせたものである．

第一モデル

ディストリビュータはGDS2またはCIFの形式で入力されたレイアウトパターンデータを，この機械の内部データ表現形式の一つである放送通信データ形式に変換する．このデータをディストリビュータは通信路を通じて繰り返し放送する．並列しているワーカにはあらかじめ処理すべき領域を割り当てておき，ワーカの通信部は放送されるデータの中から担当領域のデータを選別しメモリへ格納する．そして担当領域のデータが全部揃ったところで，データをDRC処理部へ送る．DRC処理部はこのデータをDRC処理に適したベクトル表現形式に変換して，その領域に対するDRC処理を始める．ワーカの中の通信部とDRC処理部は一領域分のレイアウトパターンデータを収容するに十分な容量のメモリをそれぞれ独立に持っているので通信部からDRC処理部へ一領域のデータを送ってしまえば通信部はこの領域からは独立してディストリビュータから次の担当領域の指定を受け，その領域のレイアウトパターンを選別し貯め始めることができる．ワーカの中で，ある領域についてDRC処理が終了するとディストリビュータに通知する．ディストリビュータは結果を回収し，またワーカの状況によっては次の領域を指示し，その間も放送によりレイアウトパターンデータを流し続ける．以下同様にしてすべての領域を処理する．結果（エラーデータ）をすべて収集するとディストリビュータは，このエラーデータを出力形式に変換して出力する．



(A) 第1モデル (A) Model No.1



(B) 第2モデル (B) Model No.2

図1 アーキテクチュアモデル
Fig. 1 Architecture models.

第二モデル

第一モデルでは各ワーカが通信部とDRC処理部の二つのコンピュータで構成され複雑である．これを各ワーカ一つのコンピュータにして簡単化したのが第二モデルである．ワーカが単純になっているのでワーカ台数を容易に増加できる，並列DRC機全体の制御が簡単になるなどの利点はあるが第一モデルのように次の領域のデータを用意しておくことはできない，領域分割数と並列台数等が条件にあえば第二モデルの方が有利な場合もある．

4. シミュレータ

前節に述べた2種類のモデルについて**図2**，**図3**をそれぞれの処理手順とした離散イベントシステムレベルシミュレータを作成した[11]．

このシミュレータでは表1に示すような一群のパラメータを与えることにより各イベントごとに状態が非連続的に変化し，各モデルの処理手順に従って動作をシミュレーションする．

また領域分割を行うとき隣接する区分領域と境界部分でお互いに必要幅だけ重ねるため分割数の増加と共にDRCの対象となるデータ量は増加するが，これに対してはDTP，PLA，ROM，RAMの4種の実用レイアウトパターンを組み合せてレイアウトパターンのモデルをつくり，これを領域分割の分割方法に従って分割しその時のデータ量の増加をシミュレーションによって得た．その結果を**図4**に示す．この結果をそれぞれのアーキテクチュアモデルのシミュレータに組み込んだ．

このシミュレータに与えるパラメータ群の範囲を求

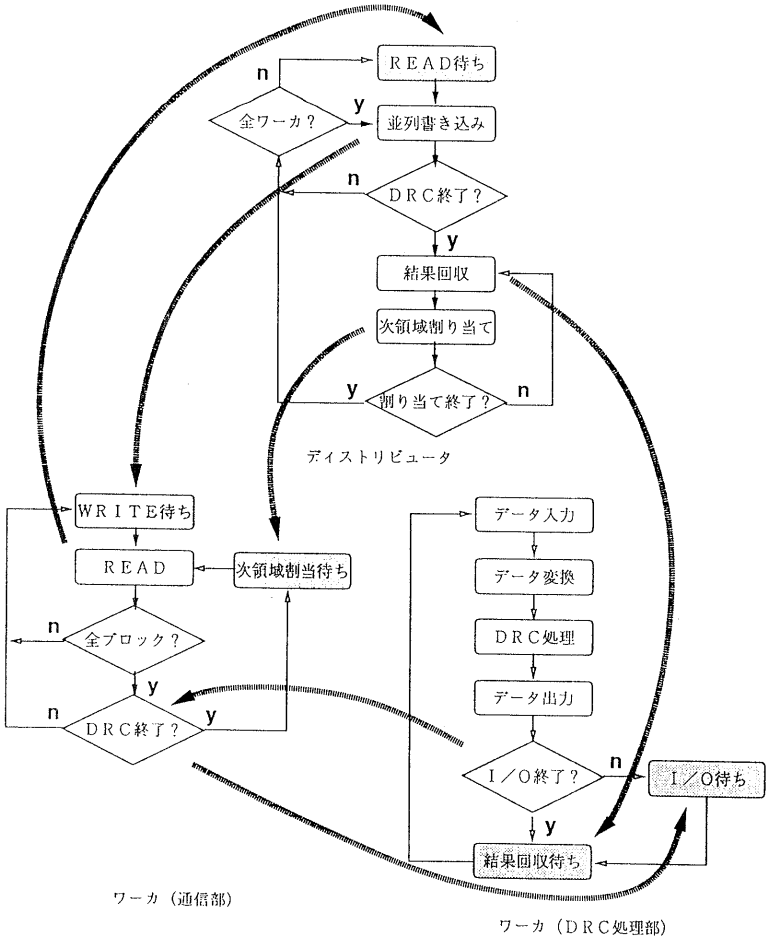


図2 第一モデルのブロックダイアグラム
Fig. 2 The block diagram of model No.1.

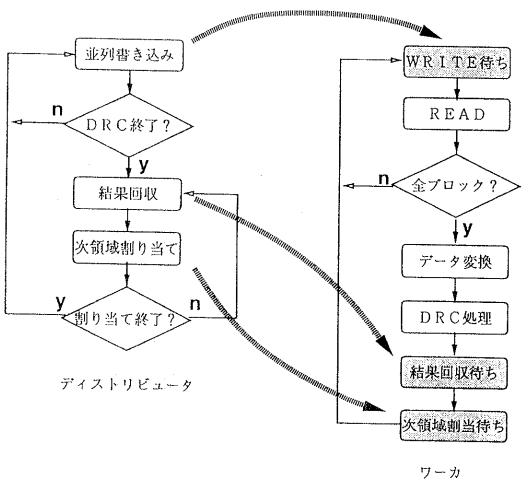


図3 第二モデルのブロックダイアグラム
Fig. 3 The block diagram of model No.2.

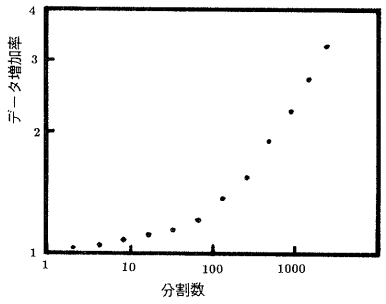


図4 分割によるパターンデータの増加率
Fig. 4 Increasing rate of data.

めるためと，このシミュレータの検証をするため第一モデルに対応する実験機の作成を行った．**図5**に実験機の構成を示す．ディストリビュータとしてはオムロン社のEWS（LUNA-2）に外部VMEとの接続ユニットを付けて使用した．EWSの外部に相互通信用

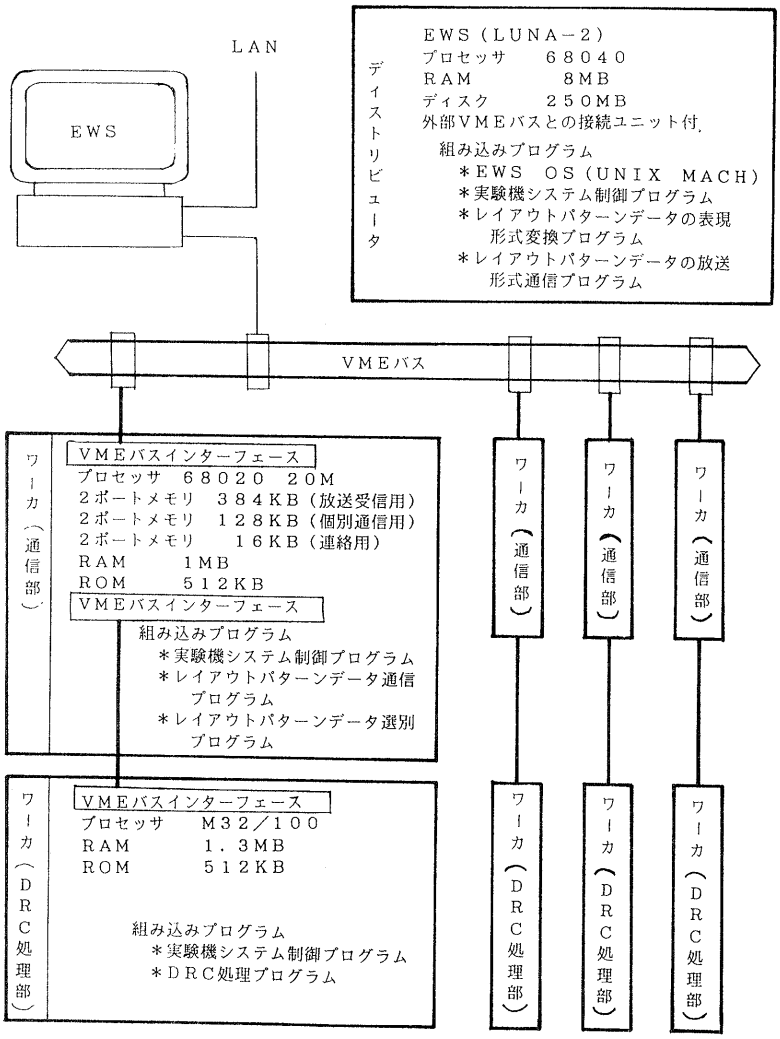


図5 実験機の構成
Fig. 5 The schematic diagram of DRC engine.

のVMEバスを設けEWSからレイアウトパターンデータをこのバスに出力し4台のワーカが同時にこのバスから受信できるようにした．各ワーカは通信部とDRC処理部（これに使用したプロセッサM32/100については文献12)参照）の2枚のボードから構成され通信部はVMEバス側から入力したレイアウトパターンデータの中から必要なデータを選別してDRC処理部に渡す必要があるため入力側，出力側の両方にVMEバスインタフェースの回路を持っている．通信部とDRC処理部のボード間はケーブルで接続し一対一のデータ転送が行えるようにした．EWSのOS（オペレーティングシステム）はそのまま利用し，このEWSのLAN機能を利用してレイアウトパターンデータは外部から入力し，並列に各ボードに転送される．実験機の内部の通信はいずれも双方向に可能なのでこれを利用してシステム全体の制御を行い最終的にはディストリビュータに検証結果が集まるようにした．これに必要なプログラムを図中に示す．この実験機でDRC実験を行った結果，ワーカ4台の範囲で実験機とシミュレータとはよく一致した[10]．

表1に示したパラメータ群について説明する．区分領域DRC処理速度は使用するDRC処理プログラムとそれを実行するコンピュータの種類に関係するが特に後者については数倍から数十倍も違う場合がある．パラメータの値としてSPARC Station2程度のプロセッサをワーカに使用した場合（11.8 Kbyte/sec）と実験機（M32/100）程度のプロセッサをワーカに使用した場合（1.0 Kbyte/sec）の2種類を想定し，参考の

表1 シミュレーションに用いたパラメータ群
Table 1 Parameters of the simulation.

区分領域DRC処理速度	: D	可変	Kbyte/sec
放送データ転送速度	: T	可変	Kbyte/sec
データ選別速度	: S	可変	Kbyte/sec
転送1ブロックデータ量		128	Kbyte
パターンデータ誤り率		0.1	percent
Dis. からワーカへの書き込み通知時間		20	msec
ワークから Dis. へのデータ選択終了通知時間		20	msec
次領域割当時間		20	msec
次領域データバッファへの転送速度		4096	Kbyte/sec
処理結果回収速度		4096	Kbyte/sec
次領域データバッファからの転送速度		4096	Kbyte/sec
出力バッファへの転送速度		4096	Kbyte/sec
データ形式変換速度		2048	Kbyte/sec
ワーカのメモリ		64	Mbyte
DRC処理を行うレイアウトの大きさ		180	Mbyte

ためさらに0.2 Kbyte/sec, 0.09 Kbyte/secを加えて4種類のパラメータの値で実験した．また並列書き込み速度とデータ選別速度については実験機の実測値はそれぞれ173 Kbyte/secと310 Kbyte/secであった．これら二つのパラメータについては図8に示す4種類の値の組合せを想定し，DRC処理速度の組合せで16種類の組合せでシミュレーションを行った．これら以外のパラメータについては実験機の値を用いた．

5. 実験結果と検討

対象となるレイアウトパターンデータの量は200メガバイトを想定し，前節に述べたパラメータのすべての組合せについてシミュレーション実験を行った．並列処理の評価尺度として実効プロセッサ数を以下に定義する．実効プロセッサ数はそのときのDRC処理部に使用したプロセッサの何倍の処理能力が実効的に達成されているかを示す数字で，例えば64台の並列動作で実効プロセッサ数が10であれば64台で10台分の能力を達成するアーキテクチュアということになる．次に，モデル1は通信選別部分とDRC処理部分とが並列に動作できるが，モデル2の場合は必ず直列に動作する．実行プロセッサ数においてモデル1がモデル2を上まわる範囲を通信選別並列化有効範囲（以下並列有効範囲）と定義する．

図6に並列有効範囲の概略を示す．**図7**は図6のワーカが32台の場合を取り出した詳細図で，通信速度（T），DRC処理速度（D）の各パラメータの条件について太い実線で示した部分が並列有効範囲である．点線はモデル1とモデル2とが同等であった領域，細い実線は境界を示す線および実験限界線で，これから図6が得られた．また点線で示したA，B，Cの領域では実効プロセッサ数でモデル2がモデル1を上回っているがそれぞれ原因が異なっているので以下に説明する．ワーカが繰り返し割当を受けて多くの領域を処理する場合，モデル1は通信選別とDRC処理が並列になっているので時間的に短い方の処理が長い方の処理

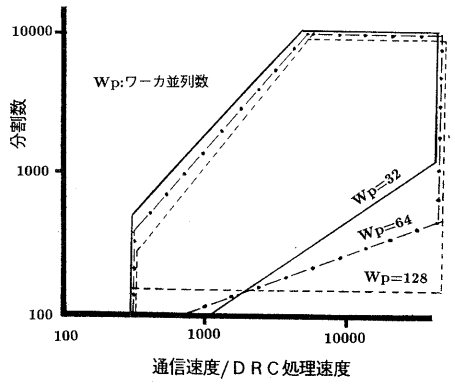


図6 並列有効範囲の概略
Fig. 6 The effective ranges in which model No.1 is superior to model No.2 for each case.

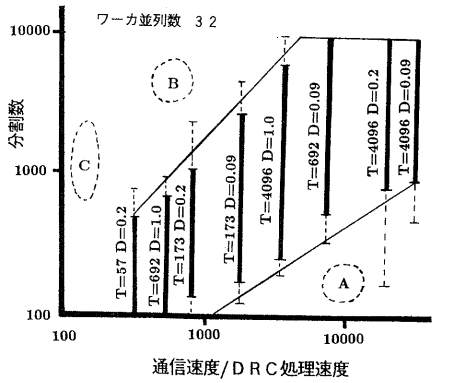


図7 並列有効範囲の領域の確定
Fig. 7 Details of an effective range.

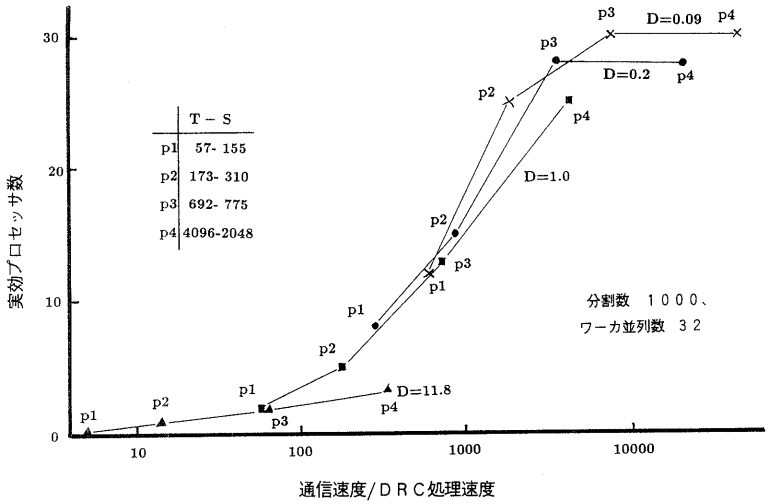


図8 実効プロセッサ数の変化
Fig. 8 A variation of effective processor numbers.

の陰に入ってしまいモデル1の方が有利であるが，繰り返し回数が少なく例えば1回の場合はどちらのモデルも直列に動作して差がなく，モデル1では通信部分からDRC処理部へのデータ転送分余分の時間がかかる．したがって明らかにモデル2の方の性能が優れる．A領域は分割数が少なくしたがって繰り返し回数が少ないのでモデル2がモデル1を上回っていると考えられる．また通信速度/DRC処理速度（以下 T/D）が大きな領域ではモデル1は通信時間がDRC時間の陰にはいって有利になるのであるが通信時間が小さくなるとモデル1とモデル2の差は小さくなる．したがって T/D の大きな（通信時間の小さい）ところほど繰り返し回数が多いところまでモデル2の優位が続く，これが領域Aが右上がりになっている理由である．

次にB領域については図2，図3を見比べるとモデル1の方がシステム全体の制御が複雑である．両方のモデルに共通のワーカ側のWRITE待ちの部分は全部のワーカがこの状態にならなくても1台でもこの状態に入ればディストリビュータは書き込みができる．他の所の転送またはDRC処理を行っているワーカは後でWRITE待ちになってから全体の処理に参加すればよいのである．これに対してディストリビュータのREAD待ちの方は全部のワーカからREADずみの返事がくるまでディストリビュータが待つことになる．モデル1ではモデル2に比べて通信部からDRC処理部への転送する時間とDRC処理部，ディストリビュータ間の転送時間およびこれらの転送制御をする時間が必要で，この時間そのものは短くてもディストリビュータの待ち時間になるので他のワーカ全部が休んでしまう．分割数が大きくなった場合これらの転送の回数が多く，この時間が大きくなってモデル1の実効プロセッサ数を引き下げる方向に働きモデル2より下まわることになる．モデル1に生じるこれらの待ち時間はワーカ並列数と分割数に対してある一定時間生じるものと推定され，モデル1の通信時間がそれだけ増加したと考えられる．

一方 T/D が約300以上の領域でモデル1は通信時間がDRC処理時間の陰に入りだしモデル2より優れているのであるが，上記の待時間を吸収する余裕は T/D に比例して大きくなると考えられる．これがこの領域の下限が右上がりになっている理由である．

C領域については T/D の値が約300以下の領域で通信時間が大きくDRC処理時間は小さい．モデル1はこの領域ではDRC処理時間が通信時間の陰にはいり，モデル2に対する有利さはB領域において述べた制御の複雑さによって打ち消されてしまっていると考えられる．シミュレーション結果ではこの領域はすべてモデル2がモデル1より優位であった．

図8は図7と同じワーカ並列数32の場合の結果を実効プロセッサ数で示したものである．図8の各点は正確にはモデル1とモデル2の少しずれた二つの点なのであるが中間をとって一点で表した．この図を見ると分かるように T/D が大きい方では実効プロセッサ数が上限に近いところで飽和しており，また図6，図7と見比べると並列有効範囲であることが分かる．一方 T/D が小さい方ではその反対である．すなわち T/D が約300から約3000までの範囲では実効プロセッサ数が急に立ち上がっており，この範囲では並列DRC

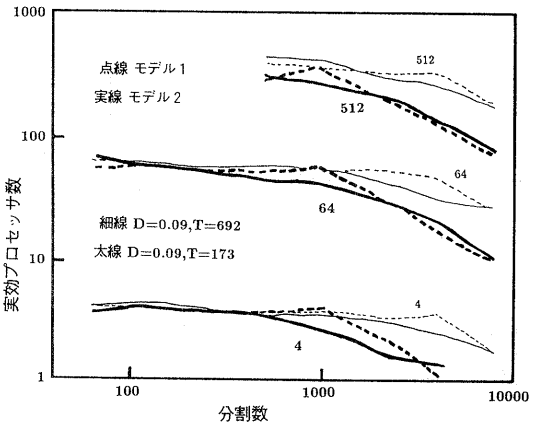


図9 シミュレーション結果の例（T/D が1000以上の場合）
Fig. 9 An example of simulation ($T/D > 1000$).

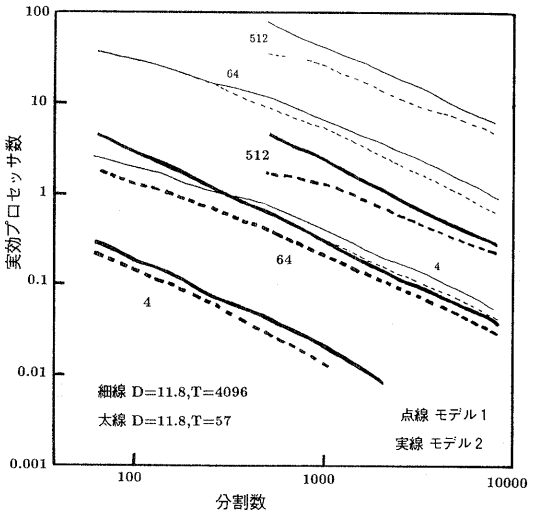


図10 シミュレーション結果の例（T/D が400以下の場合）
Fig. 10 An example of simulation ($T/D < 400$).

全体の所要時間もこれに応じて小さくなっている．実際のシミュレーション結果でも並列有効範囲の内か外かではっきりした特徴を示す曲線が得られる．**図9**，**図10**にその例を示す．この両図は T/D の大きな場合（T/D が1000以上）と小さな場合（T/D が400以下）についてワーカ並列数をパラメータにして実効プロセッサ数を表示したものである．この両図でいずれの場合も右の方にいくに伴い実効プロセッサ数が下がっているのは分割数の増加に伴い各区分領域の重複部分のデータ量が増加し，したがって対象総データ量が増加するためと考えられる．また図9の場合は実効プロセッサ数が高い値に保たれたままでデータ量の増加に突き当たり減少が始まる．一方図10では有効プロセッサ数の値が全般に低く，分割数の増加と共に各部の待時間が増加するため全体に右下がりの傾向を示している．データ転送速度（T）データ選別速度（S）の組合せは図8に示されているが図9，図10では T を表示した．

6. まとめ

LSIのDRCを高速化する目的で2種類の並列アーキテクチュアモデルを考え，これらを使用したプロセッサの実効効率について比較した．その結果

（1）通信速度とDRC処理速度の比（T/D）が並列アーキテクチュアの実効効率に決定的な影響をあたえる．

（2）T/D と領域分割数の空間で通信選別並列化有効範囲が確定した．すなわち T/D が40000から約2000までは分割数の広い範囲でモデル1が優越している，T/D が300以下ではモデル2が優越している．T/D が2000から300までは分割数の値によって異なる．

（3）領域分割で並列にDRCを行う場合，分割数の増加については限界がある．すなわち約数万分割でこの方法による高速化は飽和する．

ということがわかった．また並列有効範囲から考えると，例えばSPARC Station2程度のプロセッサ（$D = 11.8$）をDRC処理部に装着したとすると通信部が束縛条件にならないためには約20Mbyte/secの転送速度が必要である．この値はハードウェアソフトウェア両方の改善によって達成できるものと考える．

今後の問題としては，効率的なシステムを組むためには，プロセッサや通信部のハードウェアの性能だけでなく，通信部と，データを供給するディスクをもつディストリビュータとを含めて全システムを管理する制御ソフトウェアの最適設計を検討する必要がある．

参考文献

1) Macomber, S. and Mott, G.: Hardware Acceleration for Layout Verification, *VLSI DESIGN*, Vol.6, No.7, pp.18–27 (1985).
2) 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).
3) Nielson, R.N.: Algorithmically Accelerated Cad, *VLSI SYSTEM DESIGN*, Vol.7, No.2, pp.65–66 (1986).
4) Tailor, G.S. and Ousterhout, J.K.: Magic's Incremental Design Rule Checker, *Proc. of*

21st Design Automation Conference, pp.160–165 (1984).
5) Suzuki, G. and Okamura, Y.: A Practical On-line Design Rule Checking System, *Proc. of 27th Design Automation Conference*, pp.246–252 (1990).
6) Marantz, J.D.: Exploiting Parallelism in VLSI CAD, *Proc. IEEE ICCD'86*, pp.442–445 (1986).
7) Bier, G.E. and Pleszkun, A.R.: An Algorithm for Design Rule Checking on a Multiprocessor, *Proc. 22nd DAC*, pp.299–304 (1985).
8) 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, FTS93-32, VLD 93-56, pp.39–46 (Oct. 1993).
9) 上坂達生, 田丸啓吉：LSI 設計検証に適したレイアウトパターンデータの表現方法, 情報処理学会九州研究会資料, 3-36, pp.43–47 (1991).
10) 上坂達生, 池田 浩, 河野一郎, 路 圭明, 田丸啓吉：平面分割によるデザインルールチェックの並列処理方式, 信学技報, CAS93-10, VLD93-10, DSP93-20, pp.65–72 (May 1993).
11) M.H. マクドウガル, 小林 誠 訳：シミュレーションによるコンピュータシステムの性能評価—テクニックとツール, p.168, 工学社, 東京 (1987).
12) 森 昭助：TRON プロジェクトの現状と展望—CHIP サブプロジェクトの現状と展望, 情報処理, Vol.35, No.10, pp.926–933 (1994).

(平成 7 年 2 月 15 日受付)
(平成 7 年 9 月 6 日採録)

上坂 達生（正会員）

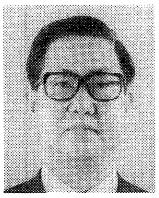
1937 年生. 1960 年大阪大学理学部物理学科卒業. 同年三菱電機株式会社に入社. 1984 年三菱電機セミコンダクタソフトウェア株式会社に出向. 1989 年から熊本電波高専情報工学科に勤務. この間コアメモリ, ASIC, LSI 用 CAD などの研究に従事. 電気学会, 電子情報通信学会各会員.

南 利秋

1966 年生. 1990 年京都大学工学部電子工学科卒業. 1992 年同大大学院修士課程修了. 同年キャノン株式会社に入社. LSI 設計技術の研究に従事. 電子情報通信学会会員.

松木 俊寿

1971 年生. 1993 年京都大学電子工学科卒業. 現在京都大学大学院工学研究科修士課程電子工学専攻に在学中. LSI 用 CAD の研究に従事.



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 年同大大学院修士課程（電子工学専攻）修了. 工学博士. 1960 年（株）東芝に入社. 1979 年から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ュア, LSI の設計手法, LSI 用 CAD などの研究に従事. 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）などを著作. 電気学会, 電子情報通信学会, ACM, IEEE 各会員.