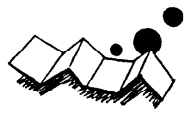
解　説



VLSI設計へのコンピュータの応用†

吉田憲司††　檀　良†††
山田尚志†††

1. まえがき

1万ゲートを超える論理 LSI, 256 Kb メモリなどいわゆる VLSI の開発における大きな問題の一つは設計であり，その困難さは今後の回路規模の増大とともに急激に増加すると考えられる．膨大な設計工数を減らし，設計期間を短縮するためには CAD (Computer Aided Design) に期待するほかないが，残念ながら期待と現実との間にはまだまだ大きなギャップがあるといわざるを得ない．

VLSI の設計手法は回路内容によって異なるが，主な設計工程と利用される CAD としては図-1 のようなものがある．メモリのような量産品の場合は，新しいプロセスで高度に最適化された設計が要求されるのでデバイスシミュレーション，回路解析が重要で，またレイアウトも人手設計で行われる．一方カスタム LSI の場合は，標準的なプロセスや基本回路が用いられ，各品種の設計では，システム設計，論理設計あるいはテスト発生が重要となる．またレイアウト設計も基本回路ブロック単位の配置，配線が中心で，自動化も進んでいる．

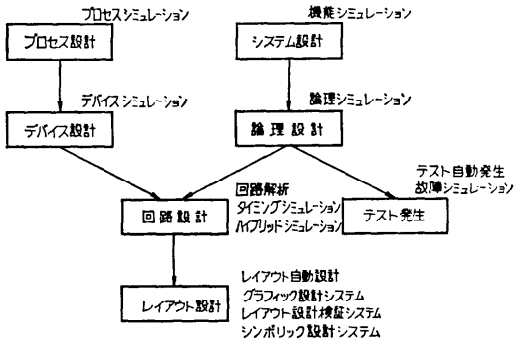


図-1　VLSI 設計工程と CAD

上記の各設計工程へのコンピュータ，特にスーパコンピュータの応用を考える場合に考慮を要する点としては，処理速度のほかにデータ量及び処理コストがある．処理の高速化はほとんどのプログラムで要求されており，所要時間ゆえに実用化が制限されている場合も多い．この点はスーパコンピュータに最も期待できる所であろう．しかし，VLSI のパターンのデータ量が膨大であることから，外部記憶とのデータ転送が頻繁に起こり，CPU のみが高速化されても全体の処理速度があまり上がらないという事態も考えられる．また営利を目的とした VLSI の開発においては，設計にかけられる費用には自ら制限がある．いかに高性能なコンピュータであってもコスト・パフォーマンスの点でも優れたものでなければ使うことができない．

以下では，VLSI 設計におけるさまざまな局面におけるコンピュータの利用と問題点及びスーパコンピュータへの期待について述べる．なお，各 CAD の詳細な内容についてはほかの文献[1),2)] を参照されたい．

2. デバイスシミュレーション

デバイスシミュレーションとは素子の寸法や不純物濃度などデバイスの物理的パラメータを用いてデバイスの振舞を支配しているいくつかの基本物理方程式を解いて，デバイス特性を求めることである．半導体デバイスは種類が非常に多いがここでは現在の VLSI に最も広く用いられている MOSFET だけ例に取ってデバイスシミュレーションの現状を紹介する．

一般に MOSFET の特性をシミュレーションする場合次の三つの基本方程式[3)] を連立にしかもセルフ・

† Applications of Super Computer in VLSI Design by Kenji YOSHIDA (Semiconductor Division, Toshiba Corporation), Ryo L.M. DANG (Research and Development Center, Toshiba Corporation), Hisashi YAMADA (Research and Development Center, Toshiba Corporation).

†† 東京芝浦電気(株)半導体事業部
††† 東京芝浦電気(株)総合研究所

コンシステントに解くものである．

$$\left.\begin{aligned}\frac{\partial n}{\partial t}&=\frac{1}{q}\mathrm{div}\,\boldsymbol{J}_n+G-R\\ \frac{\partial p}{\partial t}&=-\frac{1}{q}\mathrm{div}\,\boldsymbol{J}_p+G-R\end{aligned}\right\}\quad\cdots\cdots(1)$$

$$\left.\begin{aligned}\boldsymbol{J}_n&=qD_n\,\mathrm{grad}\,n-q\mu_n n\,\mathrm{grad}\,\phi\\ \boldsymbol{J}_p&=-qD_p\,\mathrm{grad}\,p-q\mu_p p\,\mathrm{grad}\,\phi\end{aligned}\right\}\quad\cdots\cdots(2)$$

$$\mathrm{div}\ \mathrm{grad}\,\phi=-\frac{q}{\varepsilon_s}(N_D-N_A+p-n)\cdots\cdots(3)$$

ここで，q：素電荷量，ε_s：半導体の誘電定数，ϕ：電位，n, p：電子，正孔密度，D_n, D_p：電子，正孔の拡散定数，μ_n, μ_p：電子，正孔の移動度，$\boldsymbol{J}_n, \boldsymbol{J}_p$：電子，正孔の電流密度，$N_D, N_A$：ドナー，アクセプタ不純物濃度，$G, R$：キャリアの発生，再結合率，とする．(1)式は通常電流の連続式といい，(2)式は電子と正孔に関する電流の定義でそれらの右辺の第1項と第2項はそれぞれ拡散成分とドリフト成分で，(3)式は Poisson 方程式である．G, R と D_n, D_p 及び μ_n, μ_p はそれぞれ $N_D-N_A(\equiv N_{sub})$ と n, p 及び電界強度 $|E|=|\mathrm{grad}\,\phi|$ の非線形関数で与えられている．また(2)式中の $n\,\mathrm{grad}\,\phi$ や $p\,\mathrm{grad}\,\phi$ などは二つの未知量の積で表わされているものも非線形量である．これらの非線形性が互いに関連しているので(1)式～(3)式のまともな解析解を厳密に求めることは不可能である．数値解法を用いても決して容易ではない．そこで問題を簡単にするため通常の MOSFET シミュレーション・プログラムでは次のような仮定を置くことが多い．まず定常解のみ考えて（つまり時間の変化を考えない），またキャリアの発生や再結合をすべて無視してしかも単一キャリア（つまり n チャネル MOSFET では電子，p チャネルものは正孔）だけ考える．すると(1)式は次のように簡単になる．

$$\mathrm{div}\,\boldsymbol{J}_n=0\,;\ \mathrm{div}\,\boldsymbol{J}_p=0\quad\cdots\cdots(1)'$$

しかし，このように基本方程式がかなり簡略化されたとはいえ，すべての未知量（$\boldsymbol{J}_n, \boldsymbol{J}_p, n, p, \phi$）は空間の x, y, z の関数であり，さらに N_{sub} のようなパラメータも通常は三次元の関数になっているので MOSFET のシミュレーションには非常な困難が伴うものである．

上記の基本方程式の数値解法として最もよく用いられているものは次の通りである．すなわち**図-2** の MOSFET の三次元構造を三次元的な格子に細分し，基本方程式の差分近似式をまず求めてから**図-3** のような手順によって電流の連続式と Poisson の方程式を連立に解くものである[4),5)]．なお，三次元構造の MOSFET において，ゲート幅が十分広い場合ではその方向における諸量の変化が無視できるので問題は二次元問題として近似できる．しかし，今後の VLSI の趨勢を考えると MOSFET の寸法は段々小さくなり，そのゲート幅方向の寸法もチャネル長方向やチャネル深さ方向に較べて決して十分大きいとはいえないので，デバイスシミュレーションとしては三次元シミュレーションが大勢をしめるものと考えられる．

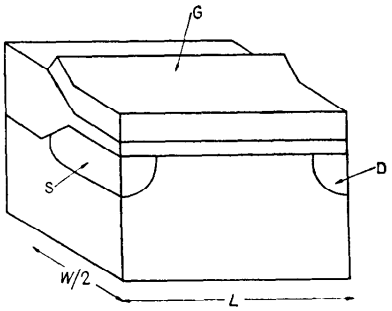


図-2　MOS トランジスタ

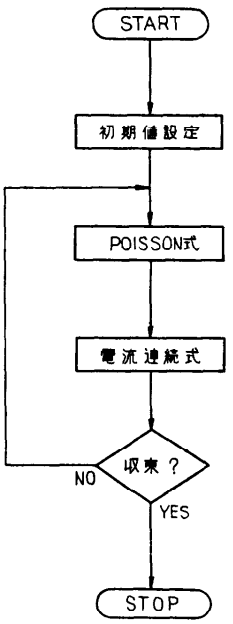


図-3　二次元や三次元デバイスシミュレーションのフローチャート

三次元デバイスシミュレーションにおいて，基本方程式を精度よく解くには素子領域を前記三次元格子に細分する際，格子間隔を十分細かく取る必要がある．

表-1 三次元デバイスシミュレーションに必要な計算機メモリと実行時間

格子数	メモリ	計算時間*
30×30×15	～200 kW	2,400 sec
50×50×25	～1.5 MW	4,800 sec

* 1 MIPS マシンに換算した結果

通常，この格子間隔（いわゆるメッシュサイズ）は半導体基板の Debye 長$\left(=\sqrt{\frac{\varepsilon_s}{qN_{sub}}\frac{kT}{q}}\,;\,\frac{kT}{q}:\text{Boltzman 電圧}\right)$程度小さく採る必要がある[6]が VLSI の MOSFET を想定する場合（つまり素子寸法は数 μm 以下），三次元領域の格子数は 30×30×15 ないし 50×50×25（ゲート幅方向の対称性を考えると素子の半分だけ取ればよい）程度取る必要がある．このため三次元 MOSFET のシミュレーション・プログラムを実行させる場合，必要なメモリと計算時間は**表-1** にまとめてあるがこれを見てわかるように今後の素子シミュレーションにおいてその実用性を確保するには超大型計算機（スピードとしては 15 MIPS 以上，ユーザ・メモリは 2 MW 以上）の導入が是非必要である．

3. 回路解析

回路解析は，LSI 設計現場では，すでに十分実用の域に到達しており，モデルの精度向上，アルゴリズムの進歩によって，特に MOS 大規模集積回路設計においては，必要不可欠のものになりつつある．現在の設計者の要望は，もっと大規模な回路をもっと短時間で解析したいという方向に移りつつあり，スーパコンピュータの利用を検討しはじめているというのが，一般的情勢である．またアルゴリズムの面からも，必要十分な精度を確保しつつ計算時間を短縮するという試みもなされている．

通常の回路解析の機能を**表-2** に示す[7]．実際に LSI 設計を考える場合には，ロジック回路の過渡応答が計算の主体であり，過渡解析が問題となっている．この

表-2 回路解析の機能

DC 解析	（DC の動作点を求めるもの）
AC 解析	（回路の周波数特性を求めるもの）
過渡解析	（回路の実時間応答を求めるもの）
感度解析	（回路内の素子の変化が全体の特性を変化させる度合を求めるもの）
統計解析	（回路素子の統計的バラツキによる特性のバラツキを求めるもの）
歪解析	（正弦波入力に対する歪を求めるもの）
雑音解析	（回路の発生する雑音を求めるもの）

表-3 回路解析のハイアラーキ

- 回路シミュレーション
- タイミングシミュレーション
- ハイブリッドシミュレーション

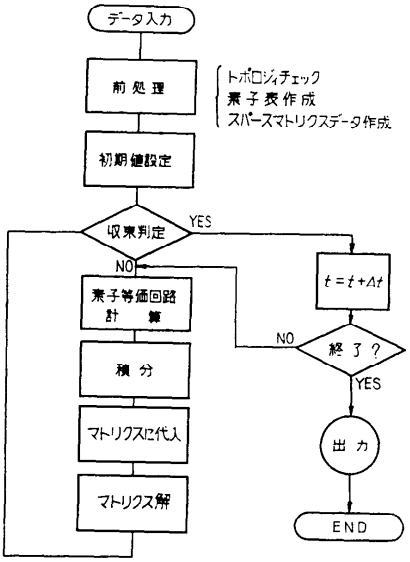


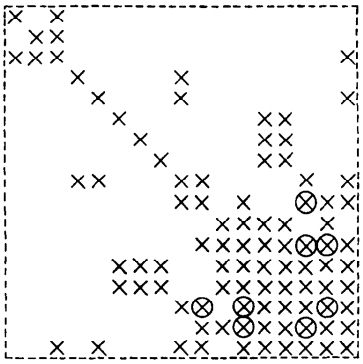
図-4 過渡解析の計算手順

過渡解析については，種々の提案があり，**表-3** に示すような提案が行われている．

過渡解析においては，何といっても計算の数が膨大になることが問題であり，かつ，回路解析の特殊性から並列演算が難しいことにある．**図-4** に過渡解析の流れを示す．この中で繰返し計算のループの中にはいっている部分のスピードが問題となるが，これには，回路解析独特のスパースマトリクスの問題がある．

3.1 スパースマトリクス[8]

回路は，一般的に，一つのノードに対して，平均 5～10 程度の素子が接続されており，これは回路規模にはあまり関係しない．これを回路マトリクスに表現した場合，その節点（ノード）の数を n とすれば，$n\times n$ のマトリクスとなるが，互いに直接素子が接続されていないノード間のマトリクス要素は 0 となるため実際は，マトリクスの中は零要素ばかりで，各行，各列に 5～10 程度の非零要素を持つスパースマトリクスになっている．**図-5** にスパースマトリクスの一例を示すが，これは MOSIC の回路に対応したスパースマトリクスの例である．このようなスパースマトリクスの場合，$n\times n=n^2$ のメモリを用意することは効率



⊗; fill-in

図-5 MOS 回路マトリクスの例

が悪いためリンクド・リスト形式のデータとして蓄積されている．このマトリクスの形は，回路の接続が変らない限り不変である．またこのマトリクスをガウスの消去法（実際はこれを変形したLU分解法）により解く場合に，最初のマトリクスで零だった点が零でなくなる場合が生じ（fill-in と呼ぶ），これを最初のマトリクスのリンクド・リストに付け加えねばならないが，全体として見れば，スパース性はあまり変らない．繰返し計算の中では，マトリクス構造，計算の順序は不変である．このマトリクスを解く時間は，ほぼ節点数の自乗に比例するといわれており，回路規模が大きくなるに従いマトリクス解の時間が増大する．

このスパースマトリクスの解については，リンクド・リストとなっているために必ずしも計算しようとするエレメントが連続しているわけではなく，常にインダイレクトアドレッシングを行う必要があり，またフルマトリクスで提案されている種々の近似解法が使えないため，現在の所並列演算による方法はあまり試みられていない．前述したごとく，リンクド・リストによる計算時間の増大を防ぐため，計算の手順が常に一定なのに着目し，手順をすべて機械語で記述して途中のアドレッシングを省略する方法では，計算時間の90％が節約されるという報告があるが，100素子程度の計算で 50〜60 k 語のメモリを必要とし，大規模回路の計算には使えない弱点がある．

3.2 素子モデルの計算[9]

図-4 のフローで，マトリクス解と同等に問題化しているのが，素子の等価回路の計算である．MOS 等価回路は，VLSI の微細加工化に伴い，二次元効果，三次元効果等が考慮されねばならなくなり，かえって複雑化している．一方では，プロセスのマージンも減少する傾向にあることからより精密なモデルが求められるようになってきている．実際のモデルの式は，単純な四則演算ばかりでなく，√‾，EXP，LOG 等の関数演算を含むためきわめて長い計算時間を必要とし，四則演算で，100〜1,000 演算にも相等するぐらい複雑化している．このため，**図-6** に示すごとく，実際にはマトリクス演算そのほかの演算より時間を浪費しているという報告もある．特に回路規模の小さい場合にはこの計算の占める割合が大きい．しかし，この等価回路の計算は，前の繰返し計算で求まった動作条件から，素子の非線形等価回路を求めるもので，マトリクス解と異なり，全素子について並列に演算することが可能であり，計算機を並列運転することで，確実に計算時間の短縮が図れる．またこういった計算の煩雑さを避けるため，素子の特性を表にしてその表を参照する Table-Look-Up 方式[10]も提案されており，最近の報告では，三次元のテーブルを用いることにより，計算時間の短縮と精度の向上の両方が達成されている[11]．このため，いずれは，マトリクス解が問題となると思われる．

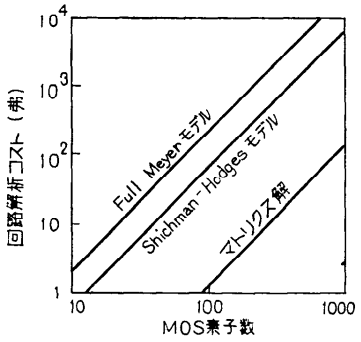


図-6 SPICE の解析コストとモデルの複雑さの関係

3.3 タイミングシミュレーション[10),12)]

タイミングシミュレーションは，マトリクス解と，積分ルーチンの両方の時間の短縮化を図るもので，実際には現在の回路解析が必要以上の精度で計算を行っているため，時間がかかることに着目している．この方法によれば，回路を素子のモデルではなく，マクロモデルといわれる，単位回路のモデルに表現し，入出力の関係を単純化すると共に，積分の際にマトリクス解を求めない前進オイラー法を用いている．この方法により，必要十分な精度で，通常の回路解析の約 10倍〜100倍の計算速度で計算できた[12]という報告がある．実用上の問題点としては，シミュレーション精

度，適用限界が今一つ明確にならないため，ユーザ側に不安があることであろう．

3.4 ハイブリッドシミュレーション[13)]

ハイブリッドシミュレーションは，ダイナミックRAM のセンスアンプのごとく全体がディジタル系のクロック動作の中で，アナログ回路が動作し，その応答波形を精密に知りたいという場合に，クロック系の膨大な回路部分をそのまま回路解析することはあまりにも無駄であるため，このクロック系は，後述するロジックシミュレーションで，解析し，問題となる部分を回路解析するというもので，実際に回路実験を行う時にクロックジェネレータからクロックパルスを供給して，アナログ回路の応答を見るのと原理的には同じである．どのくらい時間が早くなるかは，対象とする回路に依存しており，現在では，結局回路解析による時間が最終的には，全体の解析時間を決定してしまう．

しかしながら，通常の計算では，かなりのスピードアップが期待できる．

3.5 今後の方向

実際に設計現場から，回路解析に要求されることは際限がない．最終的には，IC マスクから回路を浮遊容量，配線抵抗なども含めて読み取り，それを全面的に解析して，その IC が動くかどうか検証するということを要求されている．しかし，現実に計算の量を考えてみると，たとえば，節点数 1,000 を仮定する．クロックにもよるが，少なくとも一つの波形応答を求めるには，1,000 点程度の計算は必要である．このため1点当り平均3回の繰返し計算を行うとすると 3,000 回の計算となる．素子数は節点数の約倍で 繰り返し計算，1回当り 2,000×100 の演算がある．またマトリクス解は，厳密にはむずかしいが，ほぼ，n^2 程度の計算があるとすると，マトリクス解と，等価回路ではマトリクス解の比重が大きくなり，計算の数は，

$$1,000\times3\times(2,000\times100+1,000^2)=3.6\times10^9$$

となる．つまり 1 MIPS の計算機で，$3.6\times10^3=1$ 時間の計算時間となり，メモリは，約 200 k 語必要となる．もし，VLSI で，素子数 10 万，クロック数 10 万等となれば，まさに天文学的数字となり，いかにスーパコンピュータを用いても解決し得ない問題であることは明らかである．しかし現実には，一人の設計者が，波形をリアルタイムで考え得る範囲は，約 1,000 素子程度までで，それ以上の規模は，各ブロックの接続条件を規定し，ブロックごとの設計を行うのが常であるから，実際は 1,000 素子程度の回路が，完璧に，実用的時間内で解けることが要求されている．このためには CRAY-1 等の計算機はきわめて有効であり，カリフォルニア大学，Berkeley では，CRAY-1 を5台並列に運転し 1,000 素子程度の回路解析を行うことが試みられている．しかしながら，一般的にいえば，この回路解析のコストは，LSI を実際に作ってみるコストを上回らないことが条件である．現在は，全費用の 10% 程度は，回路シミュレーションに使用しても良いのではないかという認識になっているが，スーパコンピュータを用いても計算コストが安くなるわけではないので問題は残っていると思われる．またパイプライン処理等で，等価回路部分の計算は短縮されるが，スパースマトリクス部分は，データ形式，アルゴリズムを変更して，n^2 でなく n に比例する計算時間が達成されそうであろうが，10万素子といった最近の VLSI の計算は，いずれにしてもかなり遠い所にある．当分の間，VLSI と，コンピュータのイタチゴッコは続きそうである．

4. 論理設計とテスト発生

4.1 論理設計における計算機応用

数万ゲートの VLSI の論理設計を考える場合，従来の LSI の設計手法をそのまま延長していったのでは済まされぬ問題がいくつも発生する．これを解決するために，現在必要と考えられている手法には，(1)階層化設計を有効なものとする設計記述言語とデータベース，(2)ブロック・ライブラリ等の活用による自動設計システム，(3)高能率のシミュレータや自動論理比較を用いた設計の検証システム，(4)LSSD に代表されるテスト容易化設計とこれを前提としたテスト自動発生システム及びテストベクトル評価のための故障シミュレータ利用等があげられる．

VLSI の構造的な複雑さを人間あるいは計算機の処理できるレベルへまとめる各種の方法が確立しても VLSI を対象とする CAD システムが取り扱わねばならぬデータ量は飛躍的に増大し，プログラムの計算処理時間も驚くほど増加する．そこで，従来より論理設計の CAD プログラムとして最もよく用いられている論理シミュレータと，現在の集積度においても計算機処理が困難とされているテスト発生 CAD プログラムについて，スーパコンピュータの与えるインパクトについて簡単に述べてみることにする．

4.2 論理シミュレーション

イベント・ドリブンでゲートレベルのシミュレータを考える．計算処理時間はどの程度まで詳しく回路をシミュレーションするかにもより，従来，4値（論理0，論理1，不明状態，ハイインピーダンス状態）のものが多く用いられていたが，MOS 回路のダイナミック状態までシミュレーションできる7値[14]のものを例にとる．ゲートあたりイベントを一つ処理するに要する時間が 1 m sec（1 MIPS の計算機）であったとすると，50 k ゲートの VLSI に対し 50 k ステップの入力パタンを与えてシミュレーションするには約 70 時間必要となる．ただし，この時各パターンに対して，全回路のおよそ1割の部分が動作しているものと仮定している．通常，このようなシミュレーションを何回も繰り返して回路のデバッグを行うわけであるから，一つの VLSI の全シミュレーションに要する計算機時間は数 100 時間のオーダになることが十分予想され，このままでは到底実用的なものとはいえなくなってしまう．また，現状では大きな回路の論理シミュレーションはバッチモードで行われることになってしまうが，少なくとも回路の初期デバッグ時には，数 100 から 1,000 パターン程度の入力信号に対してインタラクティブにシミュレーションが行えることが望ましい．このためには，どんなに遅くても数分で短い入力系列に対するシミュレーションを終了させる必要がある．階層化設計のメリットを生かしたミクストレベルシミュレータの活用やモジュールごとのシミュレーションで等価的な回路規模を減らすことを考え，その上に現在の計算機の 10 倍以上の能力を持ったスーパコンピュータの利用が望まれる．

また，スーパコンピュータでは，並列処理・パイプライン方式等による処理の高速化が行われているので，ロジックシミュレータのアルゴリズムで並列化できる所はできるだけこれを行う工夫も必要である．データ・フロー計算機上で処理の並列化を試みに報告[15]，同時刻に処理されるイベントをベクトル処理し，パイプライン方式の計算機を用いて大幅に効率を改善した報告等が発表されており[16]，今後の研究が期待される．

4.3 テスト発生 CAD

回路の集積度があがり，複雑化するに従って，人手のみによるテスト発生が難しいものとなり，VLSI の場合には完全なテストベクトルを人手のみで準備することはほとんど不可能に近いとまでいわれている．一方計算機による自動的なテスト発生は高度な順序回路では，スーパコンピュータをもってしても手に負えぬ計算処理量を必要とし，テスト容易化設計手法の確立がまず第一の問題となろう．テスト容易化設計により回路が完全に組み合わせ回路としてテストできる場合にもゲートレベルでのテスト発生に必要な計算機処理時間は最低でもゲート数の2乗に比例すると考えられている[17]ので，VLSI のテスト発生には，テスト容易化設計を採用した回路を入力とし，スーパコンピュータをホストとして高速のテスト自動発生を行い，使いやすいマン・マシン・インタフェースを持つ会話形のテスト作成システムが必要となろう．

テスト・ベクトルから故障辞書を作成し，故障検出率を算出して評価を行う故障シミュレータでは，パラレル法・ディダクティブ法・コンカレント法の三種のアルゴリズムがよく用いられている．この内，コンカレント法が回路への適応性，実行速度等の点からすぐれているとされているが，同一規模のデータに対し論理シミュレーションの 10 倍～100 倍以上の実行時間が通常必要とされている．VLSI の故障シミュレーションではスーパコンピュータにより2～3桁以上の処理の高速化をはかると共に，並列処理をできる限り導入して，スーパコンピュータの能力を最大限に生かすことが必要となる．

5. レイアウト設計への応用[18]

VLSI のパターン設計には，回路によって種々の手法が用いられる．論理 LSI 特にカスタム LSI では設計期間を短くし，費用を安くするため，自動設計が用いられることが多い．メモリやマイクロプロセッサのような標準品では，チップサイズや電気的特性の点でより最適に近い設計を追求するため，現在は手設計が用いられることが多い．手設計の場合に用いられる CAD ツールとしては，グラフィック設計システムとともに，設計ミスを検査するための各種設計検証プログラムが重要である．また手設計を簡略化するためのシンボリック設計手法も用いられるが，そのために自動的にパターンを圧縮するプログラムも提案されている．

5.1 自動レイアウト設計

一般に自動設計の場合は基本的な論理機能をブロックとしてあらかじめ設計し，ライブラリに登録しておき，ブロック間相互接続情報を与えて，ブロックの配置位置と相互配線経路の決定を自動的に行う．品種ごとにすべてのマスクを設計するビルディング・ブロッ

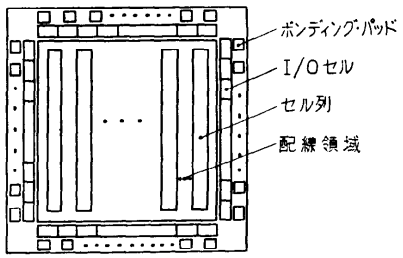


図-7 マスタ・スライス LSI のレイアウトの例[19]

ク（フル・カスタム）方式と，あらかじめ拡散工程まで終えた共通ウェーハ上に品種に応じて配線のみをほどこすマスタ・スライス（セミ・カスタム）方式とがある．**図-7** はマスタ・スライス方式のチップの一例である．前者ではチップ・サイズ最小が目標となり，後者では配線率100%が目標となるという違いはあるが，ほぼ共通の手法が用いられる．

一般にこの種のプログラムを考える場合，

（i） 問題の定式化

（ii） アルゴリズム

の二点が問題となる．定式化にはレイアウトのモデルを定め，レイアウトの良さの評価関数を定めることが含まれる．本来配置と配線は同時に最適化すべきものであるが，問題を単純化するために配置問題と配線問題の二つに分けて，それぞれ独立に最適化されるのが普通である．このため配置の評価関数の決め方が難しい．いずれにせよ現状では相当単純化された定式化が行われており，自動設計が設計の質の点で人手設計より劣るのはこのことによる場合が多い．

現在定式化されているレイアウト問題の多くは，いわゆる NP-完全問題のクラスに属する難しい組合わせ問題である[20]．すなわち，問題の規模を n として計算手数が n の多項式のオーダですむようなアルゴリズムの存在が知られていない．したがって，最適解を求めるためには，基本的には“しらみつぶし”以外になく，$n!$ のオーダとなるが，たとえば $n=20$ でも 2.4×10^{18} の手数を要することになり，非現実的な計算量となる．そこで実際には，最適解を与える保証はないが，多項式オーダの手数ですむ，いわゆるヒューリスティック・アルゴリズムを用いて，実用的に容認できる程度の解を求めることで満足することが多い．また最近は取り扱う問題の規模を制限するため階層的設計法が種々提案されている[21]．

スーパコンピュータを“しらみつぶし法”に用いても根本的解決にはならないが，ヒューリスティック・アルゴリズムに用いることによって，処理時間の短縮，問題規模の拡大あるいはより最適に近い解を求めることが可能となる．一般にこの種の最適化問題を並列演算によって高速化することは容易でないが，問題を多数の局部的な最適化問題に分割し，並列化する試みが発表されている[22]．

5.2 レイアウト設計の検証

主として手設計の場合の設計ミスを検出するプログラムで，内容としては次の三項目に分かれる．

（i） 幾何学的設計規則に対する違反の検査

（ii） 回路接続ミスの検査

（iii） 電気的特性の確認

（i）は図形線分間の相対位置関係のチェックであり，また（ii），（iii）の処理の主要部は図形間の関係をもとに，素子及びその相互接続配線を認識することである．いずれにしても，すべての二図形の位置関係を調べる処理に集約され，図形数 n に対し，n^2 オーダの手数ですむ．しかし，問題は図形数が非常に多い（たとえば 10^6）ことで，補助記憶とのデータ転送が多いことも相まって n^2 オーダでは非現実的な計算時間となる．そこで，隣接図形間の関係のみを調べれば良いことに着目して，図形を座標値でソートしておき，順次とり出せるように工夫することにより，n^1 オーダに近いプログラムが開発され実用化されている．

最近，手設計用の CAD ツールとして，パターンの自動圧縮プログラムが提案されている[23]．これは手設計されたラフ・レイアウト図を，設計規則が許す限りコンパクトに，たて及びよこ方向に圧縮するプログラムで，主としてシンボリック設計手法を対象としている．現在の所 $n^{1.5}$ オーダといわれており，適用回路規模に制限があるが，今後のコンピュータの性能向上により LSI レベルへの適用も可能と考えられる．

上記の各処理は，今後さらに高速化が要求されるので，スーパコンピュータへの期待は大きい．しかし注意すべきはデータ量で，メインメモリには到底入り切らないことから，補助記憶装置とのデータ転送が頻繁に発生するので，CPU のみが高速化されても総合的な処理時間はあまり変らないということも十分考えられる．

6. あとがき

VLSI 設計へのスーパコンピュータの応用について述べた．VLSI の CAD はきわめて難しい問題をかか

えており，その解決のため，現状に比べて格段に強力な計算機の出現を待ち望んでいる．特に，シミュレーション関係はいわゆるスーパコンピュータの最も有効な適用分野と考えられる．しかしながら，VLSI 設計工程全体について考えると，扱うデータ量がきわめて多いこと，人間の創造的，直観的能力の利用はさけられないことなどから，CPU の演算能力だけでなく，記憶容量，データベース，マン・マシン・インタフェースなどの面でも同様に強化されることが望まれる．さらに，VLSI 設計への応用の場合，計算コストもきわめて重要な要素であろう．

参 考 文 献

1) 樹下行三：論理装置の CAD，情報処理学会，東京 (1981).
2) 小特集：VLSI の CAD，情報処理，Vol. 22, No. 8 (1981).
3) Shockley, W.: BSTJ, Vol. 28, p. 435 (1949).
4) 笠井他：電子通信学会，研究会資料，Vol. SSD 80-16, p. 63 (1980).
5) 檀他：(近日発表).
6) Wada, T. et al.: IEEE J. Solid-St. Ccts, Vol. SC-14, p. 398 (1979).
7) Nagel, L. W.: SPICE-2 A Computer Program to Simulate Semiconductor Circuits, ERL Memo No. ERL-M 520, Electronics Research Lab., University of Calfornia, Berkeley (May 1975).
8) Berrey, R. D.: An Optimal Ordering of Electronic Circuit Equations for a Sparse Matrix Solution, IEEE Trans. Circuit Theory, Vol. CT-18, No. 1, pp. 40-50 (1971).
9) DeMan, H.: Computer Aided Design: Trying Bridge the Gap, Proc. ESSCIRC (1978).
10) Chawla, B. R. et al.: MOTIS-An MOS Timing Simulator, IEEE Trans. CAS, Vol. CAS-22, No. 13, pp. 901-910 (1975).
11) Shima, T. et al.: Three-Dimensional Table Look-Up MOSFET Model for Precise Circuit Simulation, Proc. ESSCIRC (1981).
12) Hirabayashi, K. and Watanabe, J.: MATIS-Macromodel Timing Simulator for Large Scale Integrated MOS Circuits, Proc. 3rd USA-Japan Computer Conf., pp. 457-462 (1978).
13) DeMan, H.: Hybrid Simulator, Proc. IEEE ISCAS, pp. 249-253 (1979).
14) Watanabe, J., Miura, J., Kurachi, T. and Suetsugu, I.: Seven Value Logic Simulation for MOS LSI Circuit, Proc. of ICCC 80, pp. 941-944 (1980).
15) 深沢友雄，栗原 謙，鈴木達郎，田中英彦，元岡 達：データフローマシン "TOPSTAR" による論理回路シミュレーション，情報処理学会第21回全国大会，pp. 89-90 (1980).
16) Krohn, H. E.: Vector Coding Techniques for High Speed Digital Simulation, Proc. of 18th DA Conference, pp. 525-529 (1981).
17) Goel, P.: Test Generation Costs Analysis and Projections, Proc. of 17th DA Conference, pp. 77-84 (1980).
18) 吉田憲司：LSI のレイアウト設計，信学誌，Vol. 61, No. 5, p. 737 (1978).
19) Shiraishi, H. and Hirose, F.: Efficient Placement and Routing Techniques for Master Slice LSI, Proc. 17th DA Conf., pp. 458-464 (1980).
20) 白川 功：実装設計における配置配線技法の動向，信学誌，Vol. 61, No. 3, pp. 245-255 (1977).
21) Preas, B. T. and Gwyn, C. W.: Methods for Hierarchical Automatic Layout of Custom LSI Circuit Masks, Proc. 15th DA Conf., pp. 206-212 (1978).
22) 上田他：並列処理ハードウェアによる論理モジュール自動配置の検討，信学論誌 D, Vol. J 63-D, No. 10, p. 907 (1980).
23) Dunlop, A. E.: SLIM-The Translation of Symbolic Layouts into Mask Data, Proc. 17th DA Conf., pp. 595-601 (1980).

(昭和56年8月3日受付)