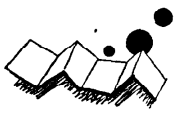
解　説



カスタム VLSI 設計システム†

須藤常太††　杉山　吉††

1. はじめに

近年，LSI プロセス技術の進歩に伴って，数万〜数十万素子／チップの LSI の実現が日程に上がっており，改めてこの設計技術の重要性がクローズアップされている．

まず最初に，VLSI 設計時の問題点をまとめてみよう．

（1） 高集積化の進展に伴って，システム的要因からデバイス的要因までが1つの LSI チップの中で複雑に絡みあってくることである．このため分野の異なるさまざまな境界条件を考慮にいれながら設計を行う必要が出てくる．

（2） 大規模化に伴って，当然のことながら設計データの絶対量が増加してくる．このため，量の増大に従いデータの作成，検証の困難度が加速度的に増大する．

（3） 論理的要因と物理的要因を総合的に捉えて最適化する必要が生じる．これは従来 IC をプリント板に並べて論理を設計していたのとは大きな違いである．

（4） 設計データの完成度がきわめて厳しく要求されることである．プリント板の設計においては，多少の設計ミスも製造段階のボード配線で救済することができたが，LSI 上でのマスク・パターンのミスは致命的である．

（5） LSI 製造技術が日進月歩である現状を考慮に入れる必要がある．使用プロセスや素子の基本性能が改良されるたびに設計全体をやりなおさなければならないのでは賽の河原の石積みになる．

このような問題点を解決するために下記の特徴を持つ VLSI 設計システムが必要である．

（1） 大規模化するデータを効率良く設計処理するために無制限の階層が扱える構造化設計手法をサポートすること．

（2） システム全体で共通に使用できる共通設計記述言語を設定し，設計段階間でデータの共用を計ること．

（3） 設計データ・ベースを開発し一元的にデータの管理を行う一方，設計資産として蓄積利用が計れるよう工夫すること．

（4） 人手がデータに直接関与して，ミスを誘発することを避けるため，ルーチン化した各種処理のプログラム化を進めること．

（5） 設計システムによる設計結果が満足のゆくものであるかどうか，設計者が容易に判断できるよう使いやすいマン・マシン・インタフェースを設定し，人間の経験が十分生かせるインタラクティブな切り口を設定すること．

2. カスタム VLSI 設計法

2.1 カスタム VLSI の構成

従来多品種少量生産の LSIの コストを下げるため，各種のカスタム化技術が用いられている．マスタスライス方式，PLA に代表されるセミカスタム LSI，ビルディングブロック方式に代表されるフルカスタム LSI がそれであり，これらの LSI で共通した事柄は，(1)最も複雑な素子設計，回路設計・マスク・パターン設計をセルの設計に集約し，論理機能に対する柔軟性を比較的単純な配線技術に帰着せしめたこと，(2)自動設計に向いたレイアウト構造としたこと，(3)各設計工程に自動設計プログラムやチェックプログラムを用意し，かつ設計工程間で，設計情報の授受を円滑化したことである．実用され，かつ自動設計の最も成功したカスタム LSI は，マスタスライス方式 LSI である．計算機メインフレーム用に 200 種類にものぼる 550〜1,500 ゲート規模のバイポーラ LSI[1] がきわめて短い期間で設計され，作成された．PLA については，ユーザ・サイトで論理を設定できる高速で 200 ゲート規模

† Custom VLSI Design System by Tsuneta SUDO and Yoshi SUGIYAMA (Musashino Electrical Communication Laboratory, N. T. T.).

†† 日本電信電話公社武蔵野電気通信研究所

の FPLA が現われつつあり，従来使用量が少なくLSI 化の遅れていた部分が多数あったが，これらへの適用が今後期待される．ビルディング・ブロック方式 LSI についても種々自動化が試みられたが，対象とした LSI の規模が人手でも設計でき，人手の方がより小さなチップを実現できる領域であったので，自動化システムはあまり用いられなかった．LSI の集積度が増加した場合，セルを配列した形式のマスタスライス方式は，5,000 ゲート規模の LSI でその実用性が実証されたが[2]，ROM, RAM, PLA などの混在を要求する場合については未解決である．トランジスタをチップ一面に配列し，ポリシリコン抵抗と配線とを個別に変えてやる形式のマスタスライス方式は，比較的効率良くチップを使って，RAM などを作ることができるが[3]，全くの白紙からスタートし，拡散パターンまでも個別に変えてやるフルカスタム方式には及ばない[4]．一方チップ上の全配線長は，素子数をNとした場合，人手設計，自動設計に関わりなく，およそ $0.6\,N^{1.2}$ であり多層配線になってもこの傾向はあまり変わらないという報告がある[5]．このことは，集積度が増すほど，配線領域面積／チップ面積が増えることを示す．

セミカスタム VLSI のフルカスタム VLSI に対する短所は，上述のように，①構成面での柔軟性に乏しい，②チップの実効ゲート密度が現在でも低いが，これがますます助長される，③長い配線長となり，性能に重大な影響を及ぼすなどである．

このため，フルカスタム VLSI は大規模化の進展した場合に1つの解となろう．

2.2 階層構造的 VLSI 設計

カスタム VLSI の設計は，VLSI 仕様が明確化された後，機能設計，論理設計，レイアウト設計，テスト・パターン発生の順に行われる．これは，大まかな流れを示したものであって，実際の設計に当たっては，テストやレイアウトを十分考慮した機能設計や論理設計がなされる．機能・論理設計では，数万ゲートにもおよぶ論理を一度に取り扱うことは困難なので，論理設計者の取り扱いやすい単位に分割し，最も便利なように階層設定し，設計を進める．設計に当たっては，テストのために端子や付加回路を設けたり，レイアウト後の配線遅延を考慮したクリティカルパスの設計を行う必要がある．

レイアウト設計に当たっては，配線がチップ面積の大半を占め，そのパターンの良否がチップ寸法や性能に重大な影響を与えるため，冗長配線のないレイアウトをしなければならない．同時に，要求性能を満足するよう，まずブロックのレベルで設計・確認し，次にブロック間配線に基づく遅延をチェックし，ゲートのドライブ能力を上げるなどの対策を施す必要がある．このため，レイアウト設計は，レイアウトに最も適した規模のブロックおよび階層で表現された論理記述を用いて行われる．

このように VLSI 設計は，多くの階層を設定し，その時々の状況により階層をダイナミックに変化させて進められる．設計は，機能／論理設計およびレイアウト設計の初期では全体的視野に立ってトップダウンに進められ，後期では，それぞれの実情を反映してボトムアップに進められる．また，設計過程の途中に数多くのチェックポイントを設定し，きめ細かく確認，修正を行いながら進められる．

3. システム構成

以後6章まで，武蔵野電気通信研究所で昨年稼動を始めた LSI 設計システムについて説明しよう[6]．**図-1** にそのシステム構成を示す．システムで共通に使用する記述言語は，階層仕様記述言語 HSL (Hierarchical Specification Language) である．この HSL で記述された設計データは，システムによってさまざまな処理を経たのち，しかるべきマスク・パターンや試験のためのテスト・パターンなどの形で出力される．システム中央に設計データベースを配置してあり，HSL で記述された設計データは，ここに蓄積管理される．データベースの周辺には，HSL 記述のコンパイラ，

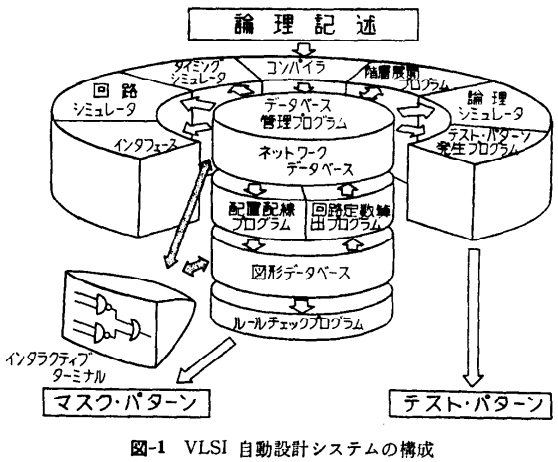


図-1 VLSI 自動設計システムの構成

逆コンパイラ，階層展開プログラムやワイアード論理生成プログラムなどの HSL 処理ソフトウェア群，論理シミュレータ，タイミング・シミュレータ，回路シミュレータなどの各種シミュレータ群，そのほかテスト・パターン発生プログラムやインタフェース・プログラムなどが配置されている。各種のシミュレータなどの支援ソフトウェアで検証された設計データは，自動配置配線プログラムによって自動レイアウトされた後マスク・パターン・ジェネレータなどのデータとして出力される．配置配線の結果が妥当かどうかの判断は，インタラクティブ・ターミナルやプロット図での判定のほか，配線結果を電気特性にフィード・バックさせて詳しいシミュレーションを行う路が用意されている．この VLSI 設計システムにおいては，一度 HSL によってデータを入力すれば以後の処理はすべてシステムによって自動的に行われる．人間は要所要所における判定を行って，GO/NOGO の指示や自動設計手順の指示を出しさえすれば良い．このシステムにおいては，設計途中のデータを手作業で修正したり，引き残した配線を人手で加えたりすることは一切ない．

4. 階層仕様記述言語

VLSI 設計過程で発生する各種データの統一的表現が可能であれば，各設計プロセス間でのデータ授受が明確でかつ容易になる．HSL[6] は，このような目的に作られた言語であって，階層構造が自由に表現できるとともに，仕様の追加・拡張が容易であるなどの特徴を持つ．HSL の記述は，(1)共通管理情報の記述，および(2)モジュールの記述からなっている．共通管理情報部では LSI のチップ名，版数，設計年月日やデータの共有および機密保護などの管理項目を記述する．モジュールは LSI 設計を行う際の記述の単位となるもので，以下のようなデータを記述する．

(i) モジュール管理情報：モジュールの名称，使用目的，プロセス，階層名．

(ii) 外部ピン属性：モジュールの外部ピン（外枠に並んだ端子）の名称とその属性，およびデフォルト値，等価ピン．

(iii) 回路図データ（参照図形，結線ピン位置，サブモジュール挿入位置）．

(iv) 接続情報：モジュール内の接続情報，引用している下位のモジュール名など．

(v) 遅延情報：ネットワークの持つ配線遅延や素子の遅延情報

共通管理情報部	IDENT : CPU; VERSION : VR 01.00; DATE : 80/08/03; AUTHOR : DENDEN TARO; PROJECT : VLSI 03; COLLECTION: VLSI; COMMENT: MICRO PROCESSOR;
モジュール管理情報部	NAME : BUSSW; PURPOSE : LOGSIM, ROUTER, CKTANAL; PROCESS : EDMOS 2 U; LEVEL : BLOCK;
外部ピン属性	EXT : A, B, E, O, DATABUS⟨0:31⟩, VDD, VSS; DEFAULT : 1, 1, 1, A; INPUTS : .A, .B, .E; OUTPUTS : .O; POWERS : VDD, .VSS; BUS : .DATABUS⟨0: 31⟩;
座標情報	TYPES : INV, TRINAND; INV : G 1, I 2, I 3; TRINAND : G 2, G 3, SW 1⟨0: 31⟩, SW 2⟨0: 31⟩; MPOSITION: . A(1, 1),. B(1, 30),. E(1, 50), .O (100, 50), G 1 (30, 10), G 2 (30, 20);
接続情報	NETA =FROM (.A) TO (G 2. 2); NETO =FROM(G 2. 3, G 3. 3) TO (.O);
遅延情報	MDELAYS : DEL 035: NETA, NETO; DELAYS : DEL 035, 35, 32, 37, 35, 32, 37;
機能情報	FUNCTION: TRINAND, 2, 1, DTRINAND; DELAYS : DTRINAND, 6, 5, 7, 4, 3, 6; ELEMENTS : PMS3, 15, 2; PARAMETER: BULK=15, VT=.7, BETA=2. 7 E−5, MOB=99, CLM=1, GZI=5.1 E−5, GAMMA=.5, ALS=.436, GMD=.01, PHI =.32;

図-2 HSL の記述例

(vi) 機能情報：ネットワークを構成する基本素子の論理機能や回路素子の種類．ワイアードゲート．

以上のような記述を必要なだけ繰り返し行うことにより階層的に LSI 全体の設計を進める．図-2, 3 に，HSL のコーディング例を示す．記述はキーワード方式の 80 カラム・カード・イメージでフリー・フォーマット形式を採用している．このほかに，論理図によるグラフィック入力の方法も設定されている．

5. 設計データベース

HSL データを格納したデータベースを中心に据えデータベースとほかの DA プログラムとの間でデータの引き渡しをするための処理システムが使われる．このシステムは，HSL コンパイラ，逆コンパイラ，マクロエクスパンダ，ワイアゲート・ジェネレータ，切り出しプログラムなどから成る．

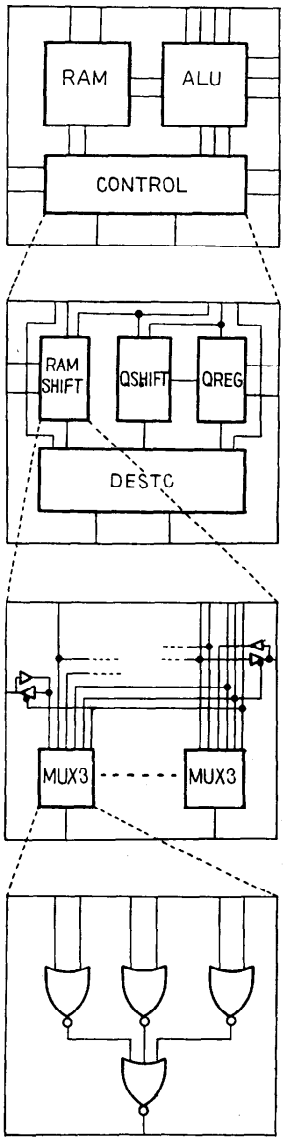


図-3 階層設計例

コンパイラ，逆コンパイラは HSL のソース入力とオブジェクトとの双方向変換を行う．オブジェクトはデータベース上では，個人ファイル，プロジェクトで共用できるファイル，多人数の人の共用できるライブラリファイルがあり，データの保護と設計資産の共用を計っている．マクロエクスパンダは階層設計されたデータを必要に応じて組み合わせ展開処理を行うプログラムで，HSL 処理の要となる．マクロエクスパンダを使えば，同じ論理を論理シミュレータや回路シミュレータに使うことができる．また同じ論理仕様の LSI を新しいデバイス技術で設計することは，ブロックレベル（時にはゲートレベル）のデータをそのまま残し，トランジスタレベルの回路の記述を書き換えるだけで済む．ワイアゲート・ジェネレータは仮想ゲートの自動発生のほか BUS ピンなどの処理を行う．切り出しプログラムは論理の再分割や再構成を論理構造に注目して自動的に行うユーティリティである．これらの HSL 処理プログラムによって適当な形に準備された設計データは目的とするシミュレータや自動配置配線プログラムに引き渡される．

設計データは，時には，数Mバイトからのデータ量となり，モジュールの数も数百から成る場合があり，データベースへのアクセスの容易さと，そのパフォーマンスが問題となる．階層構造の親子関係をたどってそれに含まれるモジュール・データを集めるためのコマンドや，モジュールの集合操作を司るコマンドなどが用意されている．表-1 にデータベースの管理項目を，表-2 にデータベース管理システムコマンドを示す．

表-1 データベース管理項目

分類	管理項目	内容
データの所属	USER	各設計者
	PROJECT	プロジェクト名称
	LIBRARY	ライブラリ名称
データの属性	NAME	モジュール名称
	IDENT	チップ名称
	PURPOSE	使用目的名
	PROCESS	使用プロセス名称
	VERSION	版数名称
	LEVEL	階層レベル名称

表-2 データベース管理システムコマンド

システム制御	起動，終了
データ入出力制御	コンパイラ，逆コンパイラ起動 各種 AP とのインタフェースプログラム起動
データアクセス	データ書込，読出し，削除，コピー，統計情報出力
データ操作	マクロエクスパンダ，ドット処理，切り出し処理 展開後のオブジェクトの圧縮

6. 自動配置配線

LSI において配置配線の設計品質ができ上がりレ

イアウトのパッキング密度にあることは論を待たない．配線がチップ面積の大半を占めるので，配線領域を極力減少させる．具体的には，ⓐ配線長を極力短く，迂回配線や余分な折れ曲りが発生しないように配線する．また，ⓑ配線ピッチの広い配線ほど直線とする．ⓒ‘第2層目の配線領域を有効に使うことによってチップ寸法に大きな影響をおよぼす第一層配線を減らす’ように配置する．ⓐはブロック間配線で特に大きな効果を持つ．チップレベルで最短となるように，ブロックを迂回させずに，ブロックの中を貫通するように配線する．ⓑは，セルを構成する拡散やポリシリコンを性能劣化が重大とならない程度の短い距離に限って配線代りに使うことで実現される．ⓒは，一次元配線プログラムによってネットワークの特徴を抽出し，それに基づいて配線層の有効利用をはかるべく二次元配置を行うことで，目的を達成される．これに使われる配置配線ソフトウェア群は次の4つの特徴を持つ．

（1） 設計階層に合わせて各段階でレイアウト品質を判定できるようにしたこと．

（2） プログラムの構成を機能モジュール単位で分割し，いろいろな配置配線アルゴリズムを自由に組み合わせたり，実行順序を入れ換えるなどの自由度を持たせたこと．

（3） 設計者は知恵を用いるのみで直接には，手を下さないシステムとすること．

（4） グラフィック・インタフェースを駆使してシステムの出力の良否を設計者が容易に判定できるようにすること．

次に具体的な手順を説明する．HSL で記述されたデータは，論理設計上の階層構造に基づいて準備されている．レイアウト時には，論理設計上の階層構造が必ずしもレイアウト上好都合でないので，マクロエクスパンダや切り出しプログラムなどのユーティリティを使って，ブロックの再構成を行う．次に，詳細なレイアウトに先だって大まかなブロック配置を行う．ここでは，ブロックの大きさ，形状，ブロック間配線の通しかた，ブロック端子のおよその位置などを決める．この作業によって LSI チップの外観が決まる．次に，セルの配置配線を自動で行う．このセルを用いてブロック内の配置，配線，ブロック間の配線処理を進め，チップ周辺部を自動配線して完成する．前に述べたように，各レイアウトのステップが進むたびに設計者による可否の判定を行う．この時設計品質が十分高くないと判定されれば，場合によっては，ブロックの再分割にまで，さかのぼってやりなおす．しかしすべての処理が自動で行われるので一回の試行に要するターン・アラウンド時間はきわめて短い．レイアウトの品質がOKとなれば設計データはマスク・パターンの形に変換されて出力される．

7. テスト・パターン生成

LSI の高集積化の進展に伴い，故障検出率の高いテスト・パターンを論理設計後に自動生成することは，著しく困難になってきた．論理設計時からテストを考慮した設計ができるようなサポートプログラムがいくつか作られつつある．

LSSD 方式

対象論理回路内のフリップフロップをすべて直列に接続し，シフトレジスタとして動作させることにより等価的に組み合わせ回路に分割して，試験を容易にする方法（一般にスキャンパス方式と呼ばれている）との適合性，および信号の立上がり，立下がり，回路や配線の遅延などを考慮した設計ルールとの適合性を，論理設計の段階で自動チェックする．

階層的テスト・パターン発生

対象システムをいくつかのブロックに分割し，ブロックごとにテスト・パターンを発生する．外部端子から所定のブロックへの入力信号の伝播，およびブロック出力信号の外部端子への伝播が可能なように機能／論理設計を進める（ブロック活性化法）．各ブロックの機能設計が終り，論理設計を進めてゆく過程でテスト・パターンを発生しながら，不都合なものは論理を変更しながら進めることができる[7]．

8. VLSI 設計システムのためのハードウェア構成

VLSI 自動設計システムを用いた場合，設計者の行う作業は，

（i） 設計データや各種設計パラメータの投入，変更．

（ii） データベース管理プログラムや各種アプリケーションプログラムの起動．

（iii） 結果の評価・確認

であり，これを繰り返して設計作業が進む．オンラインシステムを使えばこれを効率よく行うことができる．現在は論理図や配置配線結果などのグラフィックデータをインタラクティブ図形処理装置に表示したり

プリンタやプロッタに描かせているがオンライングラフィック端末が普及すれば一層効率良く設計が進むと思われる．VLSI 設計用計算機システムは，多くの資源を使うジョブを処理するホスト計算機，データの投入編集作業や資源をあまり使わないジョブを処理し，ホストの負荷を軽減するフロントエンド計算機，ホストの機能をより多く分担し，ホストの負荷を一層軽減するノード計算機などの複合体が望ましい．

9. 終りに——後の課題

現在の VLSI 設計システムを用いれば，4,000 ゲートの論理図は，220 人時で入力でき，12,000 ゲート規模の 32 ビット VLSI プロセッサのレイアウト設計が 1 カ月で完了する[9]．また，自動設計によれば，1 回の試行時間の短縮により，多重試行が可能となる．これにより，人手設計よりも高密度なレイアウトが得られ，2,000 ゲート規模ですでにこの傾向が表われる[9]．VLSI 化とその高集積度化は，今後もさらに進んでいくものと思われるが，新しいカスタム VLSI 用設計・製造技術および DA 技術の開発が重要である．

（1） カスタム VLSI 構成技術

カスタム VLSI 構成法として，現時点で最も見通しの良い技術はフルカスタム方式である．FPLA の持つユーザサイトでの論理設定機能は，VLSI になっても受け継がれるべき性質のものであろう．また，PLA, ROM, RAM, レジスタなどを標準的に作り付けておき，ROM コードと配線の 1 部を個別設計する MSP (多目的スーパーコンポネント)[6] をはじめとする各種セミカスタム技術の開発が今後望まれる．

（2） 関連技術の統合化

DA 指向のプロセス・回路技術により，MOS を凌ぐ高速高密度のバイポーラ LSI プロセッサを作ることができる[9]．VLSI 関連技術（含プロセス）を統合化することでより進歩したカスタム VLSI 技術が生まれてこよう．

（3） アプリケーションプログラムの機能拡大

機能／論理設計では機能設計のシミュレーション，ベリフィケーション，論理回路へのトランスレーションの自動化[10]，高性能化が望まれ，レイアウト設計では，より一層の自動化，高密度化とともに，多層配線構造への進展が望まれる．テスト・パターン生成では，機能／論理設計時でのテスタビリティチェックのシステム化，LSI 開発段階でのフォールトロケーションの効率化が重要である．

（4） 将来の設計システム

将来，急増するであろう VLSI 設計資産を有効に維持管理するためには，ディスクや MSS に高速にデータを読み書きできる機械，シミュレーションや各種アプリケーションプログラムを高速に実行する機械などが必要であり，これらで機能分散ネットワークを作る必要がある．VLSI 設計人口の増大に伴って，分散データベースへの進展が必要となろう．

参 考 文 献

1) 日立が 3081 対抗の M-280 H 発表，日経エレクトロニクス，pp. 62-64, 1981 年 3 月 30 日号.
2) Feuer, M. et al.: Computer-aided design wires 5000-circuit chips, Electronics Vol. 53, No. 22, pp. 144-145.
3) Akazawa, Y. et al.: A High-Speed 1,600 gate Bipolar LSI Processor, ISSCC 78 Digest of technical papers, pp. 208-209 (1978).
4) Watanabe, M.: CAD Tools for Designing VLSI in Japan, ISSCC 79, p. 242 (1979).
5) Hightower, D.: Can CAD meet the VLSI design problems of, the 80' S 16th DAC, pp. 552-553 (1979).
6) 唐津，須藤：LSI 階層仕様記述言語 (HSL)，電気学会電子デバイス，システム制御合同研究会，EDD 81-12, pp. 57-66 (1981).
7) 和田他：機能記述によるテスト発生システム (FOREST)，昭和 56 年度信学会総合全大 456 (1981).
8) 杉山他：VLSI 設計システム，情報処理学会電子装置設計技術研究会 7-2 (1980).
9) Sugiyama, Y. et al.: A subnanosecond 12 K gate bipolar 32 bit VLSI Processor, Custom Integrated circuit conf. (1981).
10) 上原他：DDL による機能設計のシミュレーションとベリフィケーション，電子デバイス，システム制御合同研究会，EDD-81-7 (1981).
11) 須藤他：CAD による論理 VLSI の設計，日経エレクトロニクス，1981 年 4 月 13 日号.

(昭和 56 年 4 月 20 日受付)