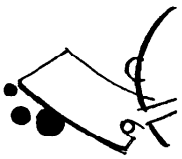
報　告



電子装置のCAD (1)†

—方式設計，論理設計自動化技術の最近の動向—

電子装置設計技術研究委員会

1. まえがき

最近における電算機，通信機などの電子装置開発に関するDA (Design Automation) システムもしくはCAD (Computer Aided Design) システムの役割りの重要性については論をまたないところであろう．一口にCADといっても，部品技術から，装置，方式までを含んでおり，最近の電算機のすべての応用技術をカバーするといっても過言でない．電子装置設計技術研究委員会では，最近の話題を中心に，三つのテーマに分けて調査を進めて来た．今回を含めて4回にわたり，その主な傾向について紹介したい．

この種の調査報告は，過去には何度か試みられており，昭和48年には，計算機設計自動化研究委員会報告[1]があり，最近では，倉地氏の論文[2]に詳しい．

今回は，第1回として，方式設計，論理設計を中心に報告する．この分野に限らず，最近のCADシステムの特徴は

1) 設計言語，設計プロセスの面で融通性を高めて，多様化に応えられるものとする．

2) 汎用性をもたせるため，入力言語の高級化，ファイルのデータベース化，使用部品・構造等のライブラリ化をはかる．

3) 応用プログラム設計，維持の容易化をはかる．

4) 方式設計，論理設計，実装設計など，すべての設計プロセスにCADが適用され，かつ一貫した思想でシステムを構築する．

傾向にあることである．

2章では，まず始めに構造化設計を指向するシステムおよび，PLAを含む設計手法を中心に紹介する．3章では，最近のハードウェア記述言語のトピックスと今後の動向について示す．さらに4章以降では，最近ますます適用範囲が広がりつつあるマイクロプログラムに関するサポートシステムについて示し，さらに最近ほぼ一般化した設計データのデータベース化，人間機械インタフェースの例としてELISシステムなどについて報告する．　　(中林　撰：武蔵野通研)

† Computer Aided Design of Electronic Equipments (1) —The Recent Trends of the CAD Technology in Hardware Systems Design and Logic Design— by Research Comittee on The Design Technology of Electronic Equipments.

2. デザイン・メソドロジー

2.1 構造化設計とCAD

ディジタルシステムの方式設計や論理設計の分野では，設計対象の多様化，開発サイクルの短期化および設計保守費用の増大等の環境変化があり，併せて半導体技術の急激な進歩により，設計方法を見直す動きがある．

構造化設計（ストラクチャードデザイン）やトップダウン設計といわれるものは，ソフトウェアからの影響を受けた，実用性を目指した設計手法と考えられ，この手法を前提としたサポートを行うCADシステムが現われ始めている[3)~5)]．

SCALD[5]は，構造化設計を考慮したCADシステムで，設計対象装置を表現するため階層を持った図形マクロを入力要素とし，グラフィックCRTによるインタラクティブな入力を可能としている．設計作業は，最上位のマクロ，すなわち設計対象の全体を示すマクロを構成するための一連のマクロ登録と考えられる．

実際の設計では，上位のマクロから下位のマクロにブレークダウンするトップダウン過程と，下位のマクロもしくは実現可能な回路（素子を含む）によって上位のマクロを構成するボトムアップ過程が混在するが，最終的には，トリー構造を持つ設計データ(**図-1**)が得られる．このようなすっきりした構造が，各レベルでの詳細に応じた理解を可能とし，設計生産性を向上させ，保守を行いやすくするといわれている．SCALDでは，信号の集合的扱い（ベクター，バス）や，くり返し表現等を許すなどして構造化設計を進めやすくし，5500 ICのプロセサを2人年で開発した実績を持つ．**図-2**は，SCALDによって出力されたブ

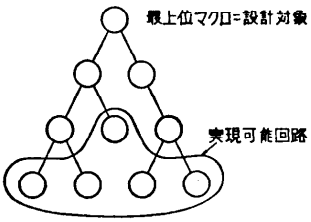


図-1 SCALD のマイクロ構造

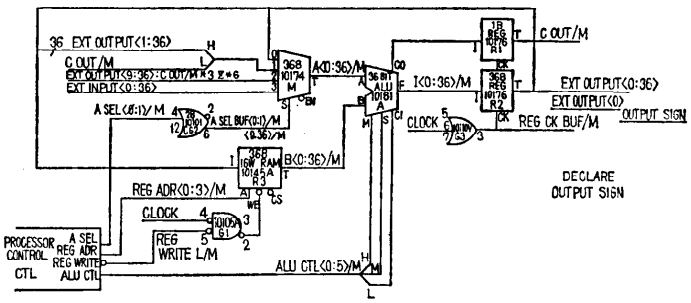


図-2 SCALD 出力 (simple processor macro)

ロック図を示す．一見簡単に見えるが，一本の線が36ビット分を表現したりしていて，これだけでもかなりのハードウェア量になるであろう．

構造化設計は，さらに LSI 化等の技術の変化に対しても，再設計を容易にする効果を持つ[5),6)]．

以上のように，設計対象のハードウェアそのものの構造を明確化し効果をあげる試みのほかに，設計過程を明確にし品質を向上させる設計手法として，トップダウン設計がある．この例として，IBM 社の LCD[4)]ではハードウェアのシステム動作による仕様化から，詳細な論理設計までのサポートを意図し，以下の手順で設計を進めるものである．(言語については3.2参照)

1) 設計するマシンの仕様の記述
2) 基本設計の記述
3) 1) と 2) をシンボリックシミュレーションで比較して等価であることを検証
4) 2) より完全な論理設計の記述
5) 1) と 4) をシンボリックシミュレーションで比較して等価であることを検証
6) ハードウェアの記述
7) 検証

すなわち，仕様から基本設計，基本設計から論理設計，論理設計からハードウェアへと上から下へ設計を進め，常に仕様と各設計段階の同等性をシンボリックなシミュレーションで比較しながら進める方法である．設計内容は'人間によって'与えられ，検証のみを自動化している点で実用的な方向といえよう．

このほか，システムの動作記述をもとに，ハードを generate するシステムも試みられている[7),8)]．

以上のような構造化設計やトップダウン設計は，設計作業を効率化し，内容の理解を容易にする等数々の利点を有するため，今後も発展していくとおもわれる．

2.2 LSI の設計手法

LSI の設計手法は，量産製品では依然かなり人手によるものが主流のようであるが，多種少量製品については，マスタースライス法等の CAD サポートが使用でき，ここでは，素子以上のブロックを要素とするマクロライブラリにより論理設計が進められる．しかし，その製造上の制約から，高度の設計精度が要求されるため，シミュレータ等の検証ツールが現在は主役を占めている．今後，VLSI の進展により素子数の増大や回路の複雑さが増加することが予想され，構造化設計等新手法の導入が望まれている．

2.3 PLA を含む論理設計

PLA (Programmable Logic Array) は，ランダムロジックをレイアウト設計なしでカスタマイズできる LSI として注目されている[9)]．しかし，PLA を用いる設計は以下のような困難さを含んでいる．

i) 論理機能を持つ素子を接続して回路を構成するのではなく，アレーの交点により回路の機能を表現するため，従来の手法に慣れている設計者には直感的に理解しずらい．

ii) ROM や PROM と同じく，一度プログラム化してしまうと修正が不可能のため，事前のベリフィケーションが不可欠である．

iii) 内部に FF を持つシーケンシャル PLA では，状態割当等のオートマトン的手法によって設計せざるを得ないため，設計者の負担を増す．

iv) 与えられた機能がその PLA に収まるか否かの判定がつきにくい場合がある．

これらの状態を克服するため，i) に対しては，論理式による入力[10)]，ii) に対してはシミュレーション[11)]，iii) に対しては状態遷移とタイムチャートを表現する言語[12),13)]，および iv) に対してはアレーの交点を減らすことにより間接的に収容量を引き上げる方法[14)~16)]等の試みがなされつつある．

将来，PLA は自動設計の一部に組み込まれるのに好適な素子であり，3) ではその可能性が示されている．このためには，PLA-CAD の研究の進歩とともに，PLA 素子自体の改善も必要となろう．

（寺本雅則：日電(株)）

3. ハードウェア記述言語 (HDL)

3.1 HDL の応用分野と記述レベル

ソフトウェア設計者が問題を記述するときに Fortran や Algol のようなプログラミング言語を使うのと同様に，ハードウェア設計者がディジタル・システムを表現するために HDL で記述することが考えられ，これまで数多くの HDL が開発されてきた[17)~21)]，HDL はハードウェアの構造と動作を簡潔かつ正確に表現することを目的としているが，その応用分野としては次に示すものがある．

（1） ハードおよびソフト設計者間の標準化された情報交換の道具

（2） 各種のドキュメンテーション

（3） 設計検証，テストパターン生成，性能評価などのためのシミュレーション・モデルの記述

（4） ハードウェアの自動合成の入力

（5） マイクロコード生成のための入力

（6） 構造化設計における入力

HDL はその用途に応じて異なった記述レベルを有する．この記述レベルは次の五つに分けられる[22),23)]．

（1） システム・レベルあるいは PMS (Processor Memory Switch) レベル：システムの構造をプロセッサ，メモリ，周辺機器ユニット，スイッチング・ネットワークなどにより記述する．

（2） プログラミング・レベルあるいはインストラクション・レベル：プロセッサの動作を機械命令の演算とその解釈規則により記述する．

（3） レジスタ・トランスファ・レベルあるいはマイクロ命令レベル：レジスタなどのリソース間のデータの転送と変換の規則を記述する．

（4） スイッチング・サーキット・レベルあるいはロジック・レベル：ゲートやフリップ・フロップにより構造を記述し，ブール式や状態遷移表などにより動作を記述する．

（5） サーキット・レベル：トランジスタや抵抗などによりゲートを記述する．

通常，HDL にはそのもっとも適した記述のレベルが存在し，それぞれの適用分野がある．たとえば，Gordon Bell の PMS は(1)，ISP (Instruction Set Processor) は(2)，Yaohan Chu の CDL は (3) のレベルの記述に適している[23)]．これに対し，多レベルで記述可能な構造化設計向きの言語も開発されてきている[4),6),24)]．

3.2 HDL 開発の現状

これまで，CDL, DDL, APL, ISP など多くの HDL が開発されているが，ここでは主として 1977 年～1978 年における，HDL の側面からみた CAD システムの研究開発を例にとり，今後の動向を探ることにする（表-1）．

HDL がもっとも多く用いられてきた応用分野の一つに論理シミュレーションがある．システム・レベルからサーキット・レベルにいたる各レベルのシミュレータの入力として使われ，ゲート，フリップ・フロップ，IC, LSI, メモリ，バスといった素子間の接続と，新しく定義する素子の機能記述を行うものが基本である．すでに定義された素子をライブラリに登録して用いる場合には接続情報の記述だけで済む．SALOGS-IV[25)]，LOGOS 2[26)] の入力単位はゲートおよびゲート

表-1 HDL からみた CAD システム開発の現状

言語名，システム名	文献番号	所 属	記述レベル	応 用 分 野
SALOGS-IV	25)	サンディア・ラボラトリ	(4)	論理シミュレーション
LOGOS 2	26)	日本電気	(4)	論理シミュレーション
MULTI-SIM	27)	ペンシルバニア大学	(1),(3)	マルチマイクロプロセッサ・シミュレーション
FDA	28),29)	日本電気	(1),(3)	機能シミュレーション
MACSIM	3)	慶応義塾大学	(1),(3),(4)	LSI 装置の論理シミュレーション
LCD	4)	IBM	(2),(3)	設計検証のシミュレーション
SDL	6)	スタンフォード大学	(1),(3),(4),(5)	構造記述を使うすべての DA システムへの入力
構造化設計システム	24)	スタンフォード大学	(1),(3),(4),(5)	
LOGAL/LADS	30)	UNIVAC	(3)	設計検証のシミュレーション，ハードウェアの自動合成システム，機能シミュレーション
CDL 改良型	31)	アーバン大学，GM	(3)	
SCIRTSS	32)	アリゾナ大学，オクラホマ大学	(3)	テスト・シークエンス作成
FLOW WARE	33)	ミズーリ・ローラ大学	(3)	設計検証シミュレーション
代入文とデータ・タイプに関する研究	34)	コロラド大学		

を組合わせたモジュールやメモリであり，MULTI-SIM[27]の入力単位はパス，クロックおよびコンポーネント（SN 7400 など）である．FDA[28),29)]では接続と機能による記述の他に，状態遷移とタイムチャートによる記述が可能である．MACSIM[3)]の入力単位はLSIからゲートまで任意に選べる．MULTI-SIM, FDA, MACSIM ではレベルの混在したシステムを記述することができ，LSI を用いた多様化するシステムの取扱いを意図した HDL を考えている．

計算機システムの設計と検証を階層的に行うことを意図した HDL に LCD[4)] がある．トップダウン設計を考えた場合，まず高レベル（たとえば ISP レベル）でシステムの構造と動作を記述し，この設計が仕様と一致していることをシミュレータにより検証する必要がある．LCD ではこのシミュレーションの結果，アブストラクト・モデルと呼ばれるシステムの機能表現のテキストを出力する．次のステップにおいて実装を意識した低レベル（たとえば RT レベル）で同じシステムを記述し，これに対するアブストラクト・モデルを再びシミュレーションの結果出力する．LCD は，これらの異なるレベルのアブストラクト・モデルが機能的に同じものであることを確かめることにより設計の検証を行おうとするものである．

SDL[6)] は，ハードウェアの構造を記述する言語であり，すべての記述レベルに対して設計の目的に応じて使えることを目指している．高レベルの記述から低レベルの記述へのマッピングはユーザの定義したマクロ展開により行う．PCB レイアウト・システム，論理シミュレータ，回路解析プログラムなどへの入力として使われている．グラフィック入力を SDL 言語におとす構造化設計システムが新たに開発されている[24)]．

この他に，トップダウン設計用の LOGAL/LADS[30)]，ソフトウェアの開発も同時に行おうとする CDL の改良型[31)]，テスト・データ作成のための SCIRTSS[32)]，グラフィック入力言語の FLOW WARE[33)]，HDL における代入文とデータ・タイプの研究[34)]などがある．

3.3 HDL の研究動向

構造化設計の手法は，設計のモジュラリティ，テクノロジー独立な設計，機能分割およびテスト，ハードウェアの自動合成などの分野で多くの可能性を持っている．

MSI, LSI の普及にともない，これらの素子やこれらの素子を用いたシステムの機能をより一層容易に表現できる HDL が望まれる．シミュレーションの分野からはいくつかの研究が報告されている[28),29),35)]．

各種のマルチプロセッサ・システムや分散処理システムの開発される傾向に対応して，これらの多種多様なプロセッサ・アーキテクチャを能率的に記述できる HDL が期待される[27)]．

この他，共通 HDL の開発，自動設計検証技術の確立，自動合成技術の実用化など，HDL に関連して技術的に解決すべき多くの項目がある．なお，HDL に関し，米国では精力的な検討がなされているが，最近の専門家の検討グループとしては CONLAN（議長，テキサス大 Lipovski）があり，その結果が注目される．（田村英二：慶大）

4. マイクロプログラム

マイクロプログラム（以下，μP と記す）によるファームウェア設計においては，その設計結果がシステムの性能を左右するため，μ コードのコンパクト化やスピード向上が要求されてきた．しかし最近では，制御メモリの低価格化や，計算機のもつ機能の多様化の要請により，μP を用いた計算機が増え，ソフトウェアの場合と同様に，その設計およびデバッグの能率向上が要求されてきている．この章では，設計自動化の立場から後者の最近の動向について述べる．

μP 用の CAD システムとして，μP アセンブラ（コンパイラ）と，μP シミュレータが用いられている．

4.1 マイクロプログラム・アセンブラ

μP アセンブラには，特定の計算機専用のものと，汎用のものがある．最近では μP を使用する機器が増え，μ プロセッサの出現などによる多様化に伴い，汎用のものが多く作られ，実用に供されている[37)]．さらに，大学・研究所を中心として，μP 記述言語を高級化し，構造を表わす文を可能にして設計能率を高める一方，生成される μ コードの増大やスピードの低下を防ぐため，最適化を施すシステムが作られている．最近の例では MPG[38)]，MPL 200[39)]，があり，MPG では μP 実行文として，

(1) assign statement
(2) flipflop set statement
(3) counter control statement
(4) procedure coll
(5) return statement
(6) goto statement
(7) if statement
(8) case statement

を，使用している．また，BARGEN/BASS[40]，PMS[41]では，汎用化によるアセンブル時間の増大を防ぐため，アセンブラを二つのモジュールに分離し，一方で言語定義（超言語）から変換テーブルを発生させ，もう一方でそのテーブルを高速アクセスする事により，効果を上げている．

4.2 マイクロプログラム・シミュレータ

μPシミュレータは，μPアセンブラによって作られるμコードのデバッグや性能評価のため用いられる．シミュレータにもアセンブラと同様，専用のものと汎用のものがあり，汎用方式では HDL（ハードウェア記述言語）により，ハードウェアを記述し，これをμP に従ってシミュレートする．この方式には，コンパイル方式とインタプリタ方式がある．最近の例では，汎用 μP コンパイラの出力結果を入力としたシミュレータ[42]が発表されており，高級言語による記述に応じたシミュレート情報が得られる事や，コンパイラと共通の計算機記述の使用が可能となっている．μPシミュレーションは，論理シミュレーションに比べ，対象が大きくなるため，レジスタレベルのシミュレーションが行われることが多いが，論理シミュレーションにおいても，μプロセッサの出現により，大規模な論理回路のシミュレーションの必要が生じ，機能ブロック化されたモデルのシミュレーションが要求されており，両方の目的で使えるシミュレータが試みられている[3]．

4.3 今後の動向

今後，μP を使用した計算機が一般的となり，ダイナミックμプログラミングや，ユーザが μP をコーディングできる計算機が使われるにつれ，必ずしも高速性，コンパクト性が必要とされない．このような部分から μP 言語の高級化が進むと思われる．また，同時に μP シミュレータも，クロスソフトウェアとしてデバッグ能率向上のため，広く使われるであろう[43]．

（三好昭夫：東芝(株)）

5. 設 計 入 力

設計入力は，設計者のアイデアを何らかの形で，CAD システムに入力する工程である．多くの CAD システムでは，論理回路を言語で表現して入力する方法を採用している．言語としては CDL[44] をはじめ数多く提案されているが，これらの言語の設計入力という観点から見た特色は，レジスタトランスファレベル以上のマクロな記述がしやすいように工夫されていることであろう．しかし，ゲートレベルで記述する場合には記述量が多くなり，記述段階で誤りの混入する恐れがある．さらに，パンチミスなどの誤りも発生しやすい．一方現在の技術では，機能レベルの記述から論理合成までを完全に自動化することは困難であるためゲートレベルでの入力はある程度避けられない．論理シミュレーション等において，かなりの計算機時間が入力ミスを発見するために使われており，設計入力において誤りを少なく，しかも迅速に入力できる方式が望まれている．

これに対しては現在のところ大別して二つのアプローチがある．一つは，Curve Tracer などの図形入力装置で回路図を読込み，論理情報を自動的に抽出する方法である．専用のテンプレートを用いて描かれた回路図を認識できるシステムが実験的に作られているが，パタン認識としてのむずかしさ（特に手書きの場合）や入力装置のコスト高などの理由によって，殆んど実用化されていない．

もう一つのアプローチは，人手にはよるが，回路を図面のまま入力することで誤りを減らし効率を上げようとする試みである．これまでに提案されたものでは入力したい回路図をタブレットやディジタルタイザの上に固定し，特別のカーソルで必要な点を指定してゆく方式のものが多い．また，表示修正用としてグラフィックディスプレイが組合されることもある[45]．処理は会話的に進められ，人間機械インタフェースについてさまざまな工夫が考えられている．端子位置の制限のない完全に手書きの回路図でも入力可能なシステムも提案されており[46]，かなりの省力化が期待できよう．

このアプローチの例として，ここでは，東京大学の ELIS (Editor of Logic Circuit System) を紹介する[47]．

このシステムの入力方式は，基本的に文献[46]に紹介されている方式に沿っており，手書きの回路図をタブレット上に固定し，それを専用のペンでペンタッチしてゆくことによって入力編集できるシステムである．タブレットなどの入出力装置は，汎用のマイクロプロセッサによって制御され，ペンタッチされた座標はこのマイクロプロセッサによって信号基準記述の言語に変換される．そして，言語形式となった論理回路情報は回線を介して TSS 入力としてホスト計算機に送られる．ペンタッチの座標データをそのままホスト計算機に送る方式と比べて，ターンアラウンドタイムや回線の負担の点で有利と考えられている．

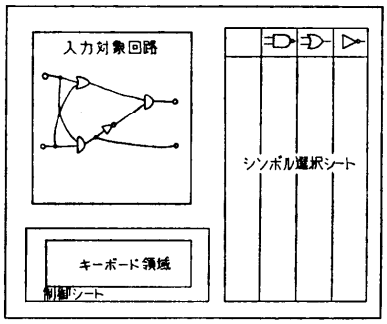


図-3 ELIS による論理回路の入力

入力方式は次の通りである．ELIS では，論理回路の入力の単位をモジュールと呼んでいる．モジュールの登録は，①入出力端子，②そのモジュール内に現われるサブモジュール，および③それらの接続関係（ネット）を入力することにより行われる．入力されたモジュールは，サブモジュールとして登録でき，階層的に入力することができる．

タブレット上には図-3 に示すように，3枚のシートが置かれる．制御シートは文字を入力するためのキーボード領域および各種の機能指定部から成っている．シンボル選択シートには，入力の対象となる回路で使用されるサブモジュールが描かれている．シンボル選択シートは必要に応じて何枚でも用意でき，適宜取替えることができるので，多くのサブモジュールを登録しておくことが可能である．これらのシートを用い，前出の①②③の手順で入力し，Abstraction と呼ばれる操作によってシンボル選択シートに登録してゆく．

このように構造的な入力が可能であり，大規模な回路でも少しずつモジュール化して入力することができる．

ELIS のような設計入力システムを今後，CAD システムに組入れ，総合的に省力化を図ることが一つの実用的傾向であろう． （三坂敏夫：東大）

6. CAD 用データベース

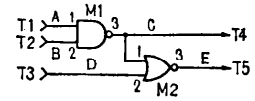
6.1 データベースの必要性[48]

データベースは2次記憶上に CAD に関する設計データファイルと規則ファイルを一元的に蓄積し，データの編集，回路図作成，シミュレーション，実装情報の生成のための情報を提供するものである．

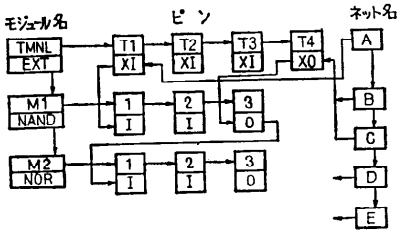
設計データのデータベース化は特定のアルゴリズムとは独立にデータを蓄積する必要がある．データベースを利用した場合の処理効率は低下する恐れはあるが，反面，設計対象，アルゴリズムや使用する計算機とは無関係にデータを維持できる可能性が強いし，一元的なデータ管理のための修正に伴うシミュレーション，実装設計等で扱うデータの同時性が確保される．処理効率についてはデータベース管理システムの強力なバッファリング機能や，アクセスパス（データのとり出し方）の改善により，分散してファイルをもつ方式と同程度に処理できる可能性もある．

6.2 データベースの構造[49]

CAD に適したデータベースの構造としては，a）ネットワーク構造と，b）関係モデルとがあるが，前者が多く用いられており，後者は，最近注目されている．ネットワークモデルは図-4(b)に示すように，ネットや，モジュール，IC 等のレベルで設計データを階層的に結合する．結合に従ったデータ処理，例えば，特定のネット名に属するすべてのピンの取り出しは容



(a) 回路例



(b) (a)のネットワーク表現

<モジュール表>

モジュール名	機 能 名
M 1	NAND
M 2	NOR
THNL	EXT

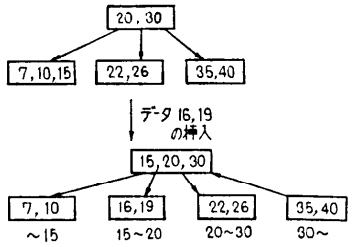
<ピン表>

主 キ ー			
ネット名	モジュール番号	ピン名	機能
A	1	1	I
B	1	2	I
C	1	3	O
C	2	1	I
D	2	2	I
A	3	T1	XI

(c) (a)の関係モデル表現

モジュール名は一意とする

図-4 回路とその表現例



ページ内のデータ数は2以上，4以下

図-5 ページ分割

易である．しかし，ネットとモジュール間に結合がない場合は，そのモジュールに属するネット名をとりだすには手間がかかる．このように，ネットワークモデルでは，ある特定の設計対象を前提として組上がっていることが多く，性能がアルゴリズムに依存することになるため，いわゆるデータ独立性が弱い．

関係モデルは図-4(c)に示すようにデータを重複のない表の形に展開して蓄積する．構造が簡単で，一様な表現形式であるからデータ独立性は強いが，主キー以外の特定のキーを指定したデータの高速とり出し（例えばモジュール名 NAND を持つピン名のとりだし）は，大きな二次インデクスを必要とする．

CAD の場合，データへのアクセスは順序処理が多いが，時間的な変動の激しい（あるいはデータの分布の不明）なデータを順序ずけて記憶する手法として，B氏木の手法が注目されている．これは，データをページに区切り，各ページ内でデータを順序ずけて記憶すると共に，各ページはページ内データの値域により木状に構成する(図-5)．データの更新に伴いページ内データがオーバフローしたときは，ページを分割し，分割したページを木状に再結合する．

データベースシステムとしては，2次記憶の管理以外に，

a）主記憶のバッファ管理

b）複数の要求（例えばモジュール名 NAND で，ネット名Aを持つピンのとりだし）を満たすデータの高速アクセス手法

c）一部データの変更に伴う波及効果の自動処理等が重要な課題である．

6.3 CADデータベース[20]

CADデータベースは，

1. 設計者の直接入力データ
2. CAD システムの生成する設計データ
3. IC 構造，基板構造，配線規則，IC 機能，図面記法等の各種の規則データ（ライブラリデータ）

を蓄積する．CAD の特殊性の一つとして図形データベースがある，IC マスク，論理回路図，基板配線パタン等はすべて多層の2次元図形であり，複雑な処理に対する設計者の介入は，これらの図形に対する編集の形で行うと便利なことが多い．CAD の図形処理はプログラム言語に対する流れ図以上の重要性を持っており，設計者にとって，文書より図形の方が理解し易いことが多い．CAD の図形データは基本図形（素子，端子），線，および文字のみであるがコンパクトで扱い易い蓄積方法が必要である．

その他，データベースとして蓄積するものとして，

4. よく利用される回路部分を登録する
5. 機能モジュールの動作記述

が考えられる．4. はアセンブラのシステム・マクロと同じ考え方である．5. は機能シミュレータあるいは，新しい部品の説明として利用できる．

6.4 今後の動向

CAD の分野におけるデータベース化の発展過程については，Fosterらの報告に詳しいが[2),51)]，それらは設計技術の高度化と表裏一体をなすものであった．すなわち CAD の初期段階においては，論理シミュレーション，布線設計などの設計過程の一部分に電算機が導入され，必要な設計ファイルもローカルなもので済んだ．しかしながら，部品技術等の進歩発展に伴って，より複雑，高度の設計を要求されるに及んで，一貫した設計技術が求められ，設計データも一元化されたデータベースにより維持管理されるようになって来た．

今までの汎用データベースは，CAD 側にとって，扱いにくい，性能的にも未知のものが多かった．このため，CAD 専用のデータベースを開発する傾向もみられたが[50)]，今後は，より能率の良い汎用データベースを経済的に使用することが考えられる．また，設計データの構成についても，従来のネットワークモデルの他，関係モデルを指向するもの[47)]が増えていく可能性がある．

（伊藤 誠：山梨大）

7. あ と が き

最近における方式設計，論理設計の主な動向について述べた．HDL がさらに高級言語化し，使い易くなり，トップダウン設計が試行され，設計データはデータベースによって一元管理される方向にあるといえよう．

次回以降は次のようなテーマで，最近の話題を紹介する予定である．

（2） テスト・パターン設計と故障診断の最近の話題

（3） 実装設計自動化技術の最近の動向

（4） CAD システムの現状調査 （中林）

参 考 文 献

1) 元岡他：計算機設計自動化研究委員会報告，情報処理学会研究会報告 73-1 (1973)．

2) 倉地 正：コンピュータの設計自動化，情報処理，Vol. 18, No. 6-9 (1977)．

3) Tokoro Mario, et al.: A Module Level Simulation Technique for Systems Composed of LSI's and MSI's, Proc. of 15th DAC, pp. 418-427 (1978).

4) Evangelisti, C. J., Goertzel, G., and Ofek, H.: Designing with LCD, Language for Computer Design, Proc. of 14th DAC, pp. 369-376 (1977).

5) McWilliams, T. M. and Widdoes L, C. Jr.: SCALD Structured Computer-Aided Logic Design, Proc. 15th DAC, pp. 271-277 (1978).

6) van Cleemput, W. M.: An Hierachical Language for the Structual Description of Digital Systems, Proc. of 14th DAC, pp. 377-385 (1977).

7) Snow, E. A. et al.: A Technology-Relative Computer-Aided Design System. Abstruct Representation, Transformations and Design Tvadeoffs. Proc. of 15th DAC, pp, 220-226 (1978).

8) Hafer, L. J. and Parker, A. C.: Register-Transfer Level Digital Design Automation, The Allocation Process. ibid. pp. 213-219.

9) 田中：ランダム論理の設計法 PLA，日経エレクトロニクス，3-6 (No. 181) (1978)．

10) 早水，南谷：PLA 設計の自動化，情報処理学会第 19 回全国大会，p. 609 (昭 53)．

11) 寺本他：モジュール化設計向RTLシミュレータ情報処理学会第 19 回全国大会，p. 607 (昭 53)．

12) 壺屋他：PLA 設計サポートシステム，情報処理学会第 19 回全国大会，p. 611 (昭 53)．

13) PLA による論理回路設計をサポートするシステム，日経エレクトロニクス，12-25, pp. 43-48 (1978)．

14) 杉山他：アレイ論理 LSI のレイアウト構成，昭 52，信学半導体全大，p. 82 他．

15) Hong, S. J. 他：MINI, A Heuristic Approach for Logic Minimization, IBM J. Res. Develop., Vol. 18, No. 5 (1974).

16) Arevalo, Z. and Bredeson, J. G.: A Method to Simplitfy a Boolean Function into a Near Minimal Sum-of-Products for Programmable Logic Arrays, IEEE Trans. Computer, C-27, No. 11 (Nov. 1978).

17) Hardware Description Language 特集号，IEEE Computer (Dec. 1974).

18) Hardware Description Language Applications 特集号，IEEE Computer, (June 1977).

19) Proceedings of International Symposium on CHDL'S and Their Applications (1979).

20) Breuer Melvin A.: Digital System Design Automation. Languages, Simulation & Data Base, Computer Science Press Digital System Design Sevies (1975).

21) Rein, W. Hartenstein: Fundamentals of Structured Hardware Design—A. Design Language Approach at Register Transfer Level, North Holland Publishing Co. (1975).

22) Mario R. Barbacci: A Comparison of Register Transfer Languages for Describing Computers and Digital Systems, IEEE T. C. Vol. C-24, No. 2, pp. 137-150 (Feb. 1975).

23) Su, Stephen Y. H.: A Survey of Computer Hardware Description Languages on the U. S. A., IEEE Computer, pp. 45-51 (Dec. 1974).

24) Bechtolsheim, Andreas: Interactive Specification of Structured Designs Proc. of 15th DAC, pp. 261-263 (1978).

25) Case, Glenn R. and Stauffer, Jerry D.: SALOGS-IV A Program to Perform Logic Simulation and Fault Diagnosis Proc. of 15th DAC, pp. 392-397 (1978).

26) 黒部，根本，志方，可児：LSI 論理シミュレーションシステム LOGOS 2，情報処理学会設計自動化研究会資料 31-3 (1977 年 1 月 20 日)．

27) Chen, C. Robert and Coffman, E. James, MULTI-SIM, A Dynamic Multi-Level Simulator Proc. of 15th DAC, pp. 386-391 (1978).

28) 11) に同じ．

29) 12) に同じ．

30) Stewart, J. H.: LOGAL. A CHDL for Logic Design and Synthesis of Computers, IEEE Computer, pp. 18-26 (June 1977).

31) Heath, J. R. and Cwik, T. T.: CDL—A Tool for Concurrent Hardware and Software Development?, Proc. of 14th DAC, pp. 445-449 (1977).

32) Hill, Frederick J. and Huey, Ben: A Design Language Based Approach to Test Sequence Generation, IEEE Computer, pp. 28-33 (June 1977).

33) Ching, S. Shingfat and Tracey, H. James: An Interactive Computer Graphics Language for the Design and Simulation of Digital Systems, IEEE Computer, pp. 35-42 (June 1977).

34) Jordan, Harry F. and Smith, Barten J.: The

Assignment Statement in Hardware Description Languages, IEEE Computer, pp. 43-49 (June 1977).
35) Chappell, S. G., Menon, P. R., Pellegrin, J. F., and Schowe, A. M.: Functional Simulation in the LAMP System, Proc. of 13th DAC, pp. 42-47 (1976).
36) 倉地 正：コンピュータの設計自動化(4)，情報処理 Vol. 18, No. 9, pp. 950-957 (1977, 9).
37) 藤井他：機械適合性を有するマイクロアセンブラ，情報処理 Vol. 18, No. 9 (1977).
38) 馬場他：MPG マイクロプログラムコンパイラ，情報処理 Vol. 19, No. 1 (1978).
39) 重松他：マイクロプログラミング言語 MPL 200 とその最適化技法，情報処理 Vol. 19, No. 8 (1978).
40) 田村他：高級汎用マイクロアセンブラ BARGEN/BASS の開発，通信学会誌 Vol. 60-D, No. 7 (1977).
41) 国立他：超言語記述によるマイクロプロセッサ用汎用クロスアセンブラ，情報処理 Vol. 19, No. 7 (1978).
42) 馬場他：MPG マイクロプログラムシミュレータ，情報処理 Vol. 19, No. 5 (1978).
43) Johnson, G. R. and Mueller, R. A.: Automated Generation of Cross-System Software for Microcomputers, IEEE Computer (Jan. 1977).
44) Chu, Y.: An Algol-like Computer Design Language, CACM, Vol. 8, No. 10 (Oct. 1965).
45) Wieczner, S.: Interactive Graphics in Design Automation, IEEE COMPUTER (Jan. 1974).
46) Rvtman, R.: Non-Gridded Graphic Input, Proc. of the 13th DAC. (June 1976).
47) 伊藤他：関係データベースによる論理設計データ編集システム，電子通信学会電子計算機研究会資料 (昭和53年3月).
48) Nash, Dan: Topics in Design Automation Data Bases, Proc. of 15th DAC (1978).
49) Martin, J.: Computer Data-base Organization, Prentise-Null (1975).
50) 中林他：DA のための設計データ管理システム，情報処理学会設計自動化研究会資料，DA 30-3 (1976).
51) Foster, J. C.: The Evolution of an Integrated Data Base, Proc. of 12th DAC (1975).

(昭和54年6月11日受付)