文　献　紹　介

A: 数 値 解 析　　B: プログラミング　　C: 計 算 機 方 式
D: 回路および機器　　E: オートマトン　　F: 応 用 そ の 他

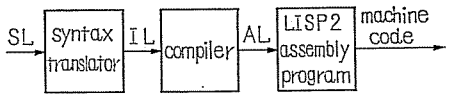
B-34. LISP 2 プログラム言語とシステム

P.W. Abrahams et. al.: The LISP 2 Programming Languages and System [Proc. FJCC, 1966, pp.661～676]

LISP 2 は複雑な記号処理とぼう大な数値計算を含む問題に対処するために設計された新しいプログラム言語である.

LISP 1.5 はその syntax が数学的にきれいなことと強力な記号処理能力で知られているが, 数値計算が不得意なことと言語形式が直感的にわかりにくいという欠点を持っている. LISP 1.5 の特長をそのまま生かし, 欠点を取除き, さらに COMIT のようなパターンによってデータ処理を決定する機能を付加したプログラム言語が LISP 2 である. この論文では LISP 2 言語と SDC で作成されている LISP 2 システムについて説明している.

LISP 2 のソース言語 (SL) は ALGOL に似ていてブロック構造や変数の declaration を含む. データは数値, 記号列およびリストである. ソースプログラムは syntax translator によって中間言語 (IL) に変換され, IL はさらにコンパイラによってアセンブラ言語 (AL) に変換され, 最後に AL はアセンブラによって機械語に翻訳される (第1図参照).



第1図　システムの構成

IL (LISP 1.5 とほぼ同じ言語である) と AL は, ユーザが利用できるようになっている. またユーザはシステムプログラムを自由に用いて自分自身のために新しいシステムを構成することができる. 特に, ライブラリ・プログラムである META compiler を用いて新しい syntax translator を作り, SL の syntax を変えることもできる.　　(二村良彦)

B-35. COGENT プログラミングシステム

J.C. Reynold: An Introduction to the COGENT Programming System [Proc. ACM, 20 th National Conf., 8. 1965, pp. 422～436]

syntax directed compilation における問題は, 適当な meta-language で与えられた入力言語と出力言語の記述に基づいて, 入力記号列の解析および翻訳を行なうことである.

COGENT (COmpiler and GENeralized Translator) は, BNF (Backus Normal Form) で記述された入力言語のシンタクスに基づき, 入力記号列を generation tree (リスト構造) に変換する syntax analyzer およびその generation tree を出力言語の generation tree に変換するためのリスト処理ルーチン (generator) から構成される syntax directed compiler である. 出力言語のシンタクスも BNF で記述でき, かつまた generation tree を出力記号列に変換するための generator も持っているので, COGENT は記号列およびリストを処理するための言語として用いることもできる.

〔例〕 代数式のシンタクスを, たとえば, 次のような式を含むように, 与えておいたとする.

(TERM)=(FACTOR),
　　(TERM)*(FACTOR),
　　(TERM)/(FACTOR).

このとき下に示す PRODUCT は, 代数式を意味する二つのリスト構造 X, Y の積を表現するリスト構造を作り出す generator である. (ユーザはこのように新しい generator を定義することができる)

```
$GENERATOR PRODUCT((X,Y)
  +1 IF X=/(EXP/(TERM)),X.
        X/=(TERM/(()(EXP)())),X.
1/+2 IF Y=/(EXP/(FACTOR)),Y.
        Y/=(FACTOR/(()(EXP)())),Y.
2/      X/=(EXP/(TERM)*(FACTOR)),X,Y.
     $RETURN(X). ).
```

ただし，=/ は左辺と右辺が同じ構造であれば後続する変数に適当な値を代入して＋の直後の数字をラベルとして持つステートメントに行き，そうでなければ次のステートメントに行くことを意味する．/= は後続する変数の値を右辺のリスト構造の対応する場所に代入し，かつそのリスト構造を左辺の変数に代入することを示す．　（二村良彦）

C-36．マルチプロセサ・システムにおいて割りこみを処理するプロセサを選ぶ方法

R.J. Gountanis and N.L. Viss: A Method of Processor Selection for Interrupt Handling in a Multiprocessor System 〔Proc. IEEE, Vol. 54 No. 12, Dec., 1966, pp. 1812～1819〕

入出力装置からの割りこみをどのプロセサで処理するかは，マルチプロセサ・システム固有の問題である．通常とられる手段は，

（1） 最初その入出力動作を起こさしめたプロセサに割りこむ．

（2） 割りこみ処理は，一つのプロセサに専任せしめる．

（3） ハードウェア・レジスタの内容により，どのプロセサに割りこみを送るかを決める．

などがあるが，これらのルート決めはすべて，割りこみが発生する以前に設定されているものである．

これに対し本論文では，入出力割りこみに対してすべてのプロセサが同格である一方式を提案し，割りこみ発生時に処理すべきプロセサを，ダイナミックに選定している．

プライオリティ・レスポンスを最適化するために，各プロセサはその状況にあって最も重要なタスクを実行するよう選ばれなければならない．そのために二つのパラメータを導入する．Interruptibility (II) は，プロセサで処理される各タスクのプライオリティであり，Interrupt Priority (IP) は，入出力割りこみの，他の割りこみ並びに II に対する相対的レスポンスの度合いを示す．II, IP ともにプログラムの支配下にあり，スーパバイザにより設定される．

入出力割りこみ発生時，II と IP の比較を行なわしめる割りこみディレクトリなるハードウェアを用意する．このディレクトリは，各プロセサ・タスクからは II を，入出力装置からは IP を入力として受取り，

（1） 最低の II をもつプロセサを探し，

（2） 最高の IP をもつ割りこみ要求を選び，

（3） IP>II かどうかを調べ，

選ばれたプロセサに選ばれた入出力割りこみをルート付けする．

このディレクトリ回路は，せいぜい簡単な並列加算器程度のものであるが，信頼度を上げるため三重の多数論理が用いられる．

ハードウェアによってプロセサ選択がなされるためソフトウェア処理に比べてレスポンス時間が速く，またプログラム支配下で処理されるため融通性に富むのが本方式の特徴である．　（平栗俊男）

C-37．アソシアティブ・メモリを用いた時分割りシステム

A.B. Lindquist, R.R. Seeber and L.W. Comeau: A Time-Sharing System Using an Associative Memory 〔Proc. IEEE, Vol. 54, No. 12, Dec., 1966 pp. 1774～1779〕

時分割システムにおけるソフトウェアの構造と，ハードウェアの問題を調べるために，IBM ではシステム 360 モデル 40 を用いて実験を行なっているが，この論文は主としてそのハードウェアの実験装置であるアソシアティブ・メモリを用いたマッピング装置について述べてある．

ここでは Ferranti Atlas で提案された虚の記憶装置（virtual memory）の概念をさらに拡大して，メモリだけでなく，CPU 周辺装置を含めた虚システム（virtual system）の概念を導入している．虚システムから実際のシステムの翻訳のために，ソフトウェアであるコントロール・プログラムではメイン・メモリだけでなくテープ，ディスクなどの周辺装置も含めた物理的な番地を虚システムに割付けてその対応をきめ，一方ハードウェアであるマッピング装置では動的な番地割付（dynamic rellocation）を行なう．

マッピング装置は 1 語 16 ビット，64 語のアソシアティブ・メモリがメイン・メモリのアドレス・レジスタとアドレス・デコーダの中間にあり，アソシアティブ・メモリの 1 語はメイン・メモリの 1 ページ（4,096 byte）と 1 対 1 に対応している．アソシアティブ・メモリは使用者の番号，虚ページ番地，コントロール・ビットの三つのフィールドに分かれていて，新たに追加された命令により，コントロール・プログラムが書きこむ．コントロール・ビットにはコントロール・プログラムのために use, activity, change, lock, transit, 未使用の 6 ビットが用意されている．

アドレス・レジスタに新しい番地がおかれるとその上位6ビットをあらかじめ質問レジスタにセットされている使用者番号とでアソシアティブ・メモリに質問され，一つだけ一致した語があれば，その語のエンコーダ出力がアドレス・レジスタの下位 12 ビットと共にメイン・メモリのデコーダに送られる．

一致した語でアソシアティブ・メモリにない場合はコントロール・プログラムが番地割付を行ない，二つ以上一致した語があればエラー・ルーチンが実行される．システム360ではページを越えて命令，データ共に使用できるため，命令・データの両方が実際のメイン・メモリにある時にのみ命令は実行され，そうでない場合は先に番地の割付を行なったのちに実行される．

マッピング装置は SLT で作られており9枚のボードに収っている．マッピングに要する時間は 220 ns で，アソシアティブ・メモリで50 ns，エンコードのために 30 ns，残りは布線と論理の遅れである．

（箱崎勝也）

C-38. 関数発生器合成のためのアリスメティック・マイクロシステム

A. Avizienis: Arithmetic Microsystems for the Synthesis of Function Generators [Proc. IEEE, Vol. 54, No. 12, Dec., 1966, pp. 1910～1919]

初期の計算機では2演算数加算器を中心に構成した一組の演算装置をもつだけであったが，現在の大形計算機は非常に複雑な演算装置をもつようになってきている．IC 論理素子を用いることにより，大きさ，価格の点でより複雑な演算装置が実現可能となってきた．現在の逐次論理回路を結合論理回路と置き換え，プログラムによるサブルーチンを演算関数発生器に置きかえ得る．現在カリフォルニア大学の可変構造計算機にはそのような装置が用いられている．IC 1 個に多数の理論素子を含み外部へのコネクタを少なくし，種類を少なくして大量に生産することが IC の価格を下げる方法であるが，現在の演算装置は内部構造上，細分化して統一したパッケージを作ることは困難である．

この論文はアリスメティック・マイクロシステムと呼ぶパッケージの設計に符号付ディジット数系（signed digit number system）を応用したことについて述べている．符号付ディジット数の演算に関する説明をし，3種類のマイクロシステム（Augmented 2-digit Sum Unit-ASU, 2-digit Product Unit-PU, Multidigit Sum Uuit-MSU）の内部の設計を基数 16 のシステムを例にとり説明している．マイクロシステムの応用として基数 16 の乗，除算装置を述べ，関数発生器として説明している．通常の演算と比較して符号付ディジット数の利点は演算回路をかなり複雑なマイクロシステムに容易に細分できること，および加算遅れが演算数の長さに対して独立であることである．

現在の研究は遅延時間を最少とするためにユニットの内部を最適化すること，指定された関数発生器をスピードと価格を制限してプログラムで合成する方法などについて進めている．

（小野田勝洋）

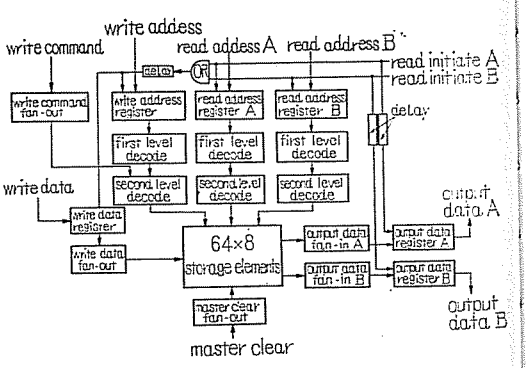
D-39. 集積回路による高速スクラッチパッドメモリ

I. Catt, E.C. Gorth and D.E. Murray: A High-speed Integrated Circuit Scratchpad Memory [Proc. FJCC, 1966, pp. 315～331]

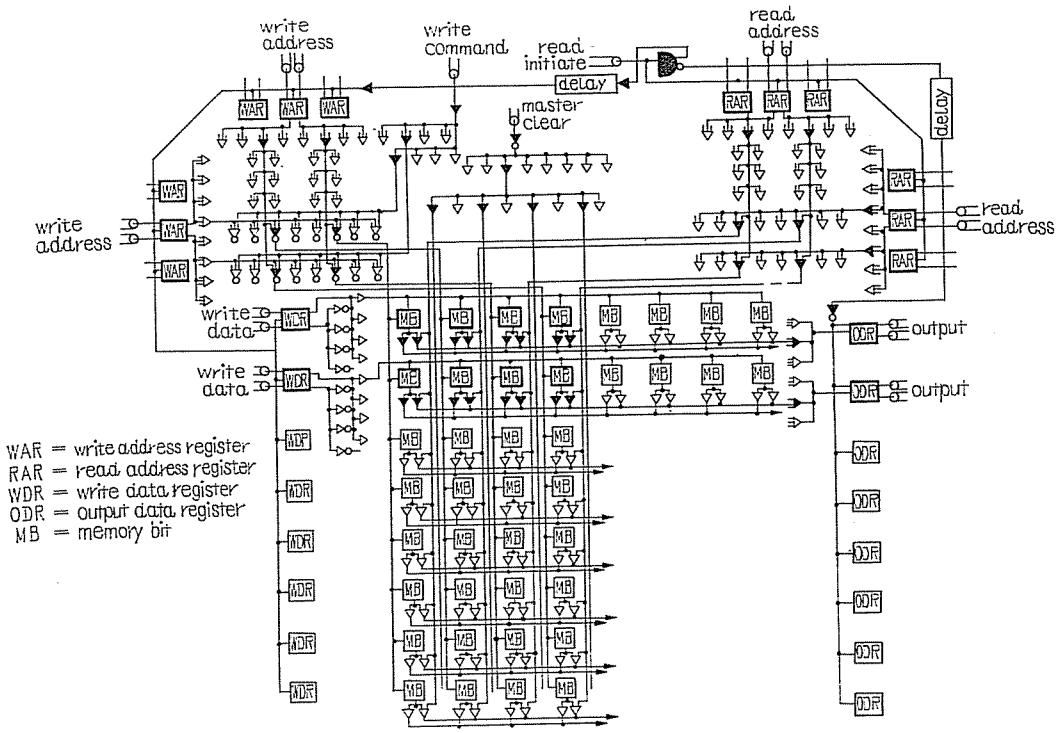
集積回路による記憶装置の試作検討結果について述べられている．第1図が試作装置のブロックダイヤグラムで 64 語8ビット構成とし，各種特性試験は第2図で示すように部分実装によった．記憶細胞の基本回路は第3図で示すもので2ビットが単位となり，50×55 ミルの寸法の矩形におさめられている．リード端子は 14 ある．単位細胞あたりの放熱量は約 275 mW である．

この回路の特徴はメモリ内の2個所の番地を独立に同時に駆動でき，システムの機能に融通性をもたせることができることである．

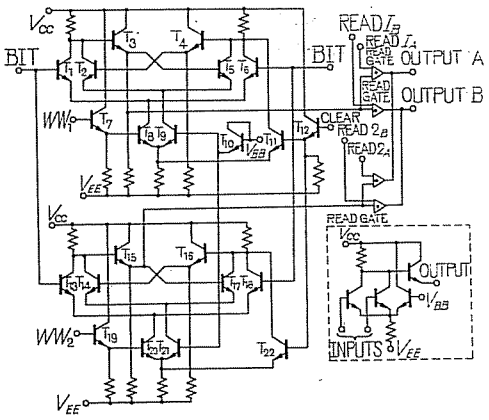
非破壊読み出しで，読み出しサイクル 17 ns，書きこみサイクル 10 ns である．また同時に書きこみと読み出しを2個所の番地に対して独立して行なえる．レジスタも記憶細胞に用いたフリップフロップ回路と同



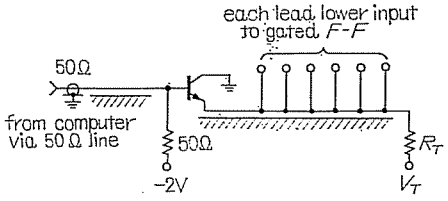
第1図 Block diagram of memory system



第 2 図　Partially populated model



第 3 図　Schematic of memory bit circuit



第 4 図　Method of dislribution of clock pulses to register circuits

じものを用いている．レジスタに対する負荷インピーダンスは 50 Ω にとり，エミッタフォロアにより各フリップフロップを駆動している．第 4 図にその回路を示す．回路内の信号の遅れは 14 ns あり，線路の遅れが 3 ns である．これらの遅れ時間はサイクルタイムに関係し，実装方法についても工夫されている．その他プリント板パッケージに対する実装方法が工夫され，温度試験の結果にもとづき部品材料が選らばれている．　（富永英義）

D-40．高密度化集積回路が機器構成とソフトウェア/ハードウェアの妥協点におよぼす影響

L.C. Hobbs : Effects of Large Arroys on Machine Organization and Hardware/Software Tradeoffs, [Proc. FJCC, 1966, pp. 89～96]

電子計算機は，初期の頃はハードウェアが大変高価なものであったが，最近部品のコストが下がり，特に高密度集積回路が用いられるようになると，CPU そ

のもののコストが下がる．それに反して，システム（ソフトウェア）の方は次第に複雑化して，cost, reliability の両面からみて，バランスがくずれ，トータルコストの大部分はソフトウェア，メモリ，入出力となり，CPU は次第に無視できるようになってくる．このような技術動向の結果，これからの計算機を設計する者にとって，次のような問題が現われる．（1）LSI を生かすような機器構成とする．（2）mass storage, input/output device を減らし，システムの cost, reliability を上げる．（3）CPU のハードウェアを増してでも operating システムや user のソフトウェアコストを下げる．

Machine Organization Implication

高密度化集積回路（LSI）の作り方として，次の三つを考える．

（1） Cellular logic: 異なった回路を標準パターンで構成する．

（2） discretionary array: 個々の単位回路をテストし，良い物を用いて所要の LSI を作る．

（3） fixed array: 現在の I/C を拡張したもの．batch-fabrication の発達によって，unit の大きさを決めるものは，パッケージの lead となり logcial element を増してでも module 間の配線を減らすように機器を構成する必要があり，したがってコストを下げるために同一の module を用いる努力が必要である．一つのやり方として，very small standard modular computer システムにすることである．

Hardware/Software Tradeoffs

「システムについての最初に要するコスト（ハードウェア， etc）とシステムを使用し，維持に要するコスト」を「cost-of-ownership」とすれば初期の計算機ではプログラミングコストがしめる total cost-of-ownership の割合は小さかったが，最近は大きな複雑なプログラムをかけるようになり，そのしめる割合は非常に大きくなった．machine language はこれまで，プログラミングをしやすくするよりも，CPU を効率よくするように設計してきた．そして user はプログラミングしやすいようにソフトウェアを開発してきた．ハードウェアが安いならば higher order language を使うことも一案である．たとえば B-5000 は Polish notation, pushdown-list store を持っている．

Additional Hardware Functions in Conventional Operations

binary-to-decimal, decimal-to-binary 変換，code 変換，…などの function をハードウェアで行なうと，プログラミングの労力と storage space を減らすのみならず speed も改善する．多くのプログラムの大部分は register-to-memory, memory-to-register transfer, index register の内容の operation であるが，このようなものが大きく減少するだろう．たとえば general-register, small high speed control メモリなど．

Future Progress

新しいハードウェア・ソフトウェアの関係，ハード・ソフト両面を含めた total system コストという評価がシステム設計において必要である．（山本昌弘）

D-41. 高密度集積回路の将来のコストに関する展望

R.N. Noyce: A Look at Future Costs of Large Integrated Arrays [Proc. FJCC, 1966, pp. 111～114]

技術的な面は別として，集積回路を高密度化（Large Scale Integration LSI）するときに，コストがどのようになるか，また高密度化するにはコストがどうならねばならないかについて述べる．現在の半導体工業では同じものを多く製造するために，設計コストは，全体のコストの小部分しかしめず，製造コストがその大部分をしめている．しかし単体素子から I/C 化されたとき研究，開発費がやや増加したが，その結果製造コストが低下し，全体のコストが下った．これと同様なことが高密度な I/C を作るときにいえるが，その反面，以下の二つのコストが必要である．

（1） basic production コスト

（2） custom コスト

（1） は全体の装置のコスト，プロセス開発コスト，技術開発コスト，（2）に属すコストを下げるために必要なコスト（例えば design automation）(2) は配置（array）の設計，客の要求に合致する回路の設計，客の要求を確認するテストプログラムを作る，マスクを作る，などのコストである．（1）に関するコストは通常新技術導入時必要なコストであるが，（2）は LSI にしたことによる特殊なコストで，各客に応じて必要なコストで，ここでは（2）に関して述べるが，custom コストは array の種類の数に比例することになるから，このコストを下げるには，array の標準化が必要である．

もっともコスト以外の performance, reliability, size, weight からの要求はあるが，少なくとも，I/C を用いたときと compatible なコストで LSI ができなければならない．

そのためには I/C において基礎的なゲートにおいて，標準化を行なったと同じように，LSI においても，基礎的な element において，標準化を行なって，custom コストを下げるようにしなければならない．

またそれと同時に，安く容易にマスクを設計し製造する方法，design antomation 用のソフトウェア，テストプログラムの作成方法などの研究，開発も必要である．　（山本昌弘）

D-42. 高密度集積回路の技術的基礎と将来の方向

R.L. Petritz: Technological Fondations and Future Directions of Large-Scale Integsated Electronics [Proc. FJCC, 1966, pp. 65～88]

エレクトロニクス産業は過去，真空管，トランジスタ，I/C と発達し，いまや完全な equipment component を1枚のチップ上に作るようになりつつある．これは Large Scale Integration (LSI) など，いろいろな名前で呼ばれているが，厳密にいえば「Large Scale Integrated Electronics」と呼ぶべきもので，この技術より得られるものを integrated equipment component (IEC) と呼ぶ．LSI とは，“1枚のチップ上に powerful logic や memory function のような logical power を作ること”である．多くのゲートよりなる，たとえばメモリ，central processor を作るとき，単体素子で作るより，I/C で作れば，mechanical は結合の可成の部分が，internal な結合となって減少したと同様に，LSI にすることによって，この interal な結合が，material processing の技術（たとえば蒸着）によってなされて，さらに減少し，従来生じていた mechanical は結合による障害を可成除くことができる．また LSI による利点として，5 ns 以上のスイッチング回路を取り扱うとき問題となる（1）transmission time,（2）低インピーダンスによる終端，に関する解が得られる．

LSI のやり方として，（1）単体から全体の IEC を作る Device Base IEC,（2）NAND または NOR などの回路をもとにして全体の IEC を作る Circuit Base IEC に分類することができる．

前者の利点は，高密，小面積，欠点は100% 歩留りのチップが必要，後者の利点は高い柔軟性をプロセスに入れることができる．欠点は一つの IEC につき，一組の完全なマスクが必要，discretionary wiring の技術が必要，などがいえる．

次に LSI に適する素子として，MOS か，bipolar か．packing density の点では MOS トランジスタでは 100,000 個/in², bipolar トランジスタでは 50,000 個/in²，抵抗はインピーダンスが小さいものなら，bipolar でも可能であるが，大きくなると，大きな面積が必要となる．しかし MOS では 100 kΩ 程度の大きなものでも小面積で作ることができる．そのためインバータを作ると，MOS の方が bipolar の 1/3 の面積で作ることができる．

speed は当然 bipolar の方がよい．speed を示すと思われる g_m は，1 μA 程度の電流では，bipolar の方が 10 倍ほど良く，高電流になるにつれてますます良くなる．結論として，speed が間に合い，control する電流も間に合うものでは，高密度（歩留りが良くなる）が可能な MOS がてきとうである．

IEC の集積度は（1）device の density として，discretionary wiring では，良い所のみを用いるので歩留りが悪くても用いられ，その結果，redundacy を持っていることになり，余分な area が必要となり，density が小になる．（2）チップの大きさの点として，光学的な制限結晶の大きさ，製造技術上からくる歩留りなどによってその大きさが制限される．

またコストの面から考えたとき，IEC の標準化が必要となり，同時にこの点についても考えねばならない．

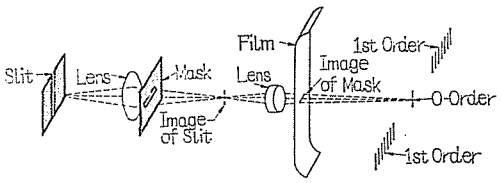
過去第1世代の真空管はコストが高いため，第2世代のトランジスタは信頼性があまり良くないため，第3世代の I/C 複雑なことと，高い performance の要求には不十分である，などの条件により順次発達し，いまや IEC が生まれつつあり，ここでは I/C 時代のハードウェアのある部分がソフトウェアに組みこまれつつある．その結果エレクトロニクス産業は，material technology, ソフトウェア technology が増加し，mechanical technology が次第に減少して行くであろう．と結論づけている．　（山本昌弘）

D-43. 格子模様の重ね合わせで写真フィルムにディジタル・データを記録するシステム

R.L. Lamberts and G.C. Higgins: A System of

Recording Digital Data on Photographic Film Using Superimposed Grating Patterns [Proc. FJCC, 1966, pp. 729～734]

記録媒体として写真フィルムを用いる場合，従来のビット情報を単に濃淡で記録する方法では機械的な位置ぎめの困難さとよごれによるビットの欠損によってフィルムの持っている高い解像度を有効に利用することができなかった．本文に述べられている方法は1ビットの情報を小さな格子模様としてフィルム上に記録し，単色光源を用いた簡単な光学系の中でその回折像をつくり，その1次線スペクトルを光学的に検出するもので，回折像の性質として格子模様が動いても線スペクトルの位置は動かないので機械的位置ぎめが楽になる．さらに格子間隔の異なる7種の格子模様を7ビットの各々に対応させ，フィルム面にビット情報に応じてそれらを重畳するという方法で7ビットを／単位として記録している．このためビット密度を等しくとれば，1ビットの記録される面積が通常の方法より広くなり，よごれに対してそれだけ強い．



第1図

重畳された格子模様の回折像をつくると図のように各ビットに対応する1次線スペクトルがそれぞれ異なる位置に生じるので各々に検出器をもうけることで読み出せる．1次線スペクトルの中央（0次線スペクトル）からの距離は格子模様の空間周波数に比例するので，各ビットに対して等差的に空間周波数を選べば1次線スペクトルは等間隔にならぶ．また2次線スペクトルの影響をさけるため，すべての空間周波数は1オクターブ内に納まっている必要がある．一区境に重畳して記憶できる実用上の限界は7～8ビットで，それ以上では一つの線スペクトルの光の強さが急激に低下する．

格子模様の最小寸法は，光学系のスリットの寸法によって決まる各線スペクトルの分散の大きさと，隣り合う1次線スペクトルの間隔が等しくなる点が限界となる．実際には間隔を5倍（冗長度5）以上にしないとほこりで使用できなくなる．

実際に試みた系の一つはブラウン管を7個のオシレータの組み合わせで輝度変調して組み合わせ格子模様を発生させ，15,000 patterns/sec でフィルム面上に記録している．1パターンの大きさは 500μ×7μ で，格子の空間周波数としては 10 cycle/mm おきに 70～130 cycle/mm の7種を用いている．ビット密度は 6×10^5 bits/in^2 である．その他にプリズムを組み合わせた光学系による記録も試み 60 kc の速度が可能としている．

読取りは高圧水銀アークを光源とし光電子増倍管を検出器として図示の系で行なった．500μ 長の格子で横方向に 25μ の動きが許容できる．

ヘリウムネオン・レーサを光源として光度を増せば光ダイオードで検出でき，読み取り速度は 100 kc まで上げられる．（田中範夫）

D-44. 大容量ランダム・アクセスメモリ用光電技術

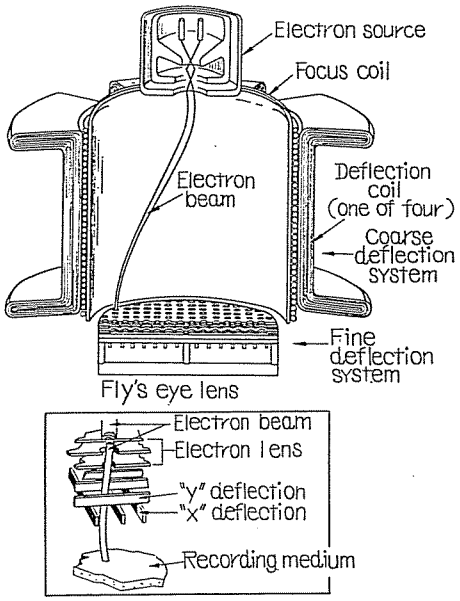
S.P. Newberry: An Electron Optical Technique for Large-Capacity Random-Access Memories [Proc. FJCC, 1966, pp. 717–728]

電子ビーム記録のメモリで機械的動きを完全に除去することは，アクセスを早くするために望ましい．しかし一つの電子レンズでは視野の制限があるため記憶面が増やせない．そこで fly's eye と呼ばれる昆虫の複眼の原理に似た電子レンズマトリックスが考えられたが，本文ではその構造，試作装置，試験結果について述べている．

図は装置の断面図で，右下に一つの小レンズが示してある．小レンズは共通軸上にある三つの孔よりなっている．小レンズの中央の孔は全部共通の金属板に含くまれ，カソード電位近くに保たれ，外側の孔は各々やはり全部孔とも共通の金属板に含くまれアノード電位に保たれている．そこで小レンズのマトリックスは孔の各板ごとに一つずつの3本のリードでよく，電子ビームの向う一つの小レンズまたはいくつかの小レンズだけが動作するので，全部小レンズごとにリードはいらない．各レンズのすぐ次にXとYの回折板があり，こまかい回折を行なう．やはり回折板すべてに電圧を加えるのでリードは少なくてよい．回折板の次に記録する板がある．この報告の試験には Lippman-type の写真用感光乳剤が使われた．

試作したのは，1.5 mm 間隔に 10×10 の 100 個のレンズマトリックスをもつもの，およびこれを改良

した，0.75 mm 間隔で 32×32 の約 1,000 個のレンズマトリックスをもつものである．レンズの孔は直径 0.25 mm で各板の間隔も同じにした．この fly's eye のセットは真空系とつながった光電ベンチで試験が行なわれた．電子源は 4 keV ビームを使用した．



第 1 図 Schematic diagram of microspace concept employing fly's eye lens

その結果，一時に全記憶面で焦点があうことができる新しい光学素子であること，記録面での電流密度一定で，記録面の大きさはスケールファクタの平方でへらせることがわかった．また 1 インチ平方当たり 10^8 ビットの密度が直径 1 ミクロンのビームで示された．

（佐藤 武）

D-45. 高密度集積回路におけるシステム設計者と半導体設計者との関係

W.B. Sander: The System/Semiconductor Interface with Complex Integrated Circuits [Proc. FJCC, 1966, pp. 105～110]

要約 高密度化集積回路 (Complex Integrated Circuit) の技術の方法とこれを導入したときのシステム設計者と半導体設計者との関係について述べる．

高密度化集積回路を作る方法として次の三つが考えられる．

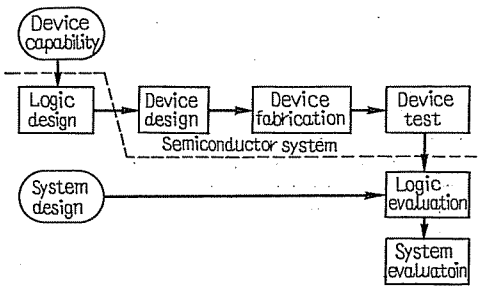
（1） Specialized Design: 今日の I/C を拡張したもの．

（2） Test and Connect: 各 unit をテストして，良い所だけを用いて所要の function をウェファ上で作る．

（3） Cellular: 同じようなセルを作り，テストせずにロジックブロックを作る．

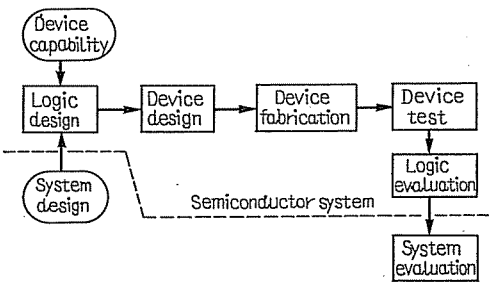
次に高密度化集積回路の発達に伴う system/semiconductor の interface について，いろいろなやり方について述べる．

（1） Standard Products (第 1 図)



Advantages	Disadvantages
* Fast delivery * Low cost per device * Minimum interaction	* Designs rarely optimum in performance and not always lowest cost

第 1 図 Interface using standard products.



Advanteges	Disadvantages
* Close coordination between logic and device design * Testing at major black box level	* Subcontractor type of interface * Poor design control by system manufacturer

第 2 図 Interace at subsystem specification.

semiconductor designer は system designer からの何ら情報を得ずに自ら設計し作る．それを買って system designer が logic を評価して，システムを作る．

長所: 速くでき，device 当たりのコストは安い，双方の接触は最小である．

短所：必ずしも，total コストは安くない．不要な労力がいる．

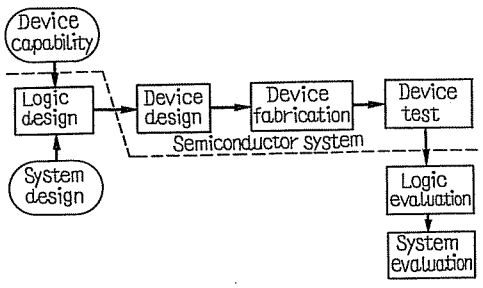
（2） Black Box Specification Interface（第2図）

system designer は semiconductor designer にブラックボックスとして入出力のデータを与える．

長所：device と logic の間のかなりの接触がある．

短所：semiconductor designer は system designer に半ば従属する．semiconductor designer は logic design についての知識が必要である．

（3） Logic Block Diagram（第3図）



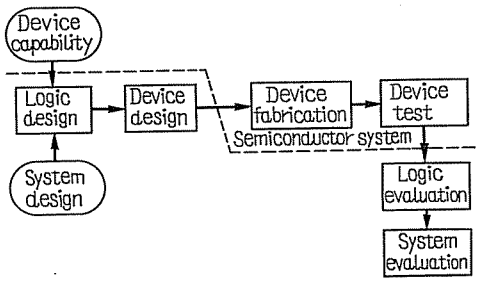
Advantages
* Good control of logic design by system mamufacturer
* Good control of device design by device manufacturer

Disadvantages
* Definition of test requirement to satisfy everyone is difficult

第3図 Interface at logic description.

両方の意見により logic を設計する．

長所：device と logic の良いコントロールが可能である．



Advantages
* Best control of design by system manufacturer

Disadvantages
* Current design rules must be maintained
* Suitable control of artwork quality can be difficult
* Who tests what?

第4図 Interface at mask design.

短所：双方に満足なテストを行なうことは非常に困難である．

（4） Mask Design（第4図）

system designer が強力な場合この方法がとられる．

長所：system designer によって設計の最良のコントロールができる．

短所：設計の規則の維持，テスト基準が難かしい．

結論として，small system manufacture にとっては（1）の方法が良い．最も良いのは，job, user の能力によって異なる，といえる． （山本昌弘）

F-46. コンテクスト・フリーな言語におけるあいまいさ

S. Ginsburg and J. Ullian: Ambiguity in Context Free Language [JACM. Vol. 13, No. 1, Jan., 1966, pp. 62～89]

言語における ambiguity の問題は，たとえば構文解析の場合などに重要な問題となってくるが，この論文では context free language における ambiguity に関する問題をいくつか取り上げている．まず ambiguity について簡単に定義を述べると

定義 1. grammar G が ambiguous であるとは，G によって生成される language $L(G)$ の中に二とおりの derivation を持つ word が存在することである．

定義 2. language L が本質的に ambiguous であるとは，$L=L(G)$ となるどんな grammar G をとっても G が ambiguous であることである．

Cantor によって，任意の CF grammar が ambiguous か否かを決定するアルゴリズムは存在しないことが証明されているが，ここでは次の定理が証明される．

定理. 2文字からなる任意の CF language が本質的に ambiguous か否かを決定するアルゴリズムは存在しない．

この系として，この論文の一つの主要な結果を得る．

系. 任意の CF language が本質的に ambiguous か否かを決定するアルゴリズムは存在しない．

次に CF language のうち「限定された」という条件を持つものについての ambiguity に関する問題が検討される．

定義 3. $\sum$ をアルファベットとし，$\theta(\sum)$ を $\sum$ か

ら生成される word の集合とする．$w_1, w_2, \cdots\cdots, w_n$ を $\theta(\sum)$ の任意の元としたとき，$\theta(\sum)$ の部分集合 X が $X \subseteq w_1^* \cdots\cdots w_n^*$　$(w_1^* \cdots\cdots w_n^* = \{w \mid w = \prod_{i=1}^{N_1} w_1 \prod_{i=1}^{N_2} w_2 \cdots\cdots \prod_{i=n}^{N_n}, 0 \leqq N_j \leqq \infty, j=1, \cdots\cdots, n\}$ ならば X は $\langle w_1, \cdots\cdots, w_n \rangle$ に限定されているという．さらに，$X=L(G)$ のとき，grammar G が限定されているという．

ここで次の定理が証明される．

定理．任意の限定された CF grammar が ambiguous か否かを決定するアルゴリズムは存在する．

さらに，限定された CF language が与えられたとき，それが本質的に ambiguous であるための必要十分条件について述べられている．しかし著者も述べているように，与えられた language がこの条件を満たしているか否かを決定するアルゴリズムが存在するかどうかはまだわかっていない．このように，限定された CF language についていろいろ述べられているが，この「限定された」という条件がどういう内容を持っているのかあまりはっきりしていないように思われる．　（小野寛晰）

F-47. 計算機による誤りの自動修正

G. Carlson: Techniques for Replacing Characters that are Garbled on Input [Proc. SJCC, 1966, pp. 189～192]

大容量ファイルを使用することが多くなるにつれて大量のデータをインプットすることが必要になってくる．データの量が多くなるにつれて，誤りも多くなるが，従来の人手による検孔や，2度打ちは高価であるので，データが誤ってインプットされた場合，それを計算機によって自動的に修正させる方法について検討した．

たとえば光学的文字読取り装置について考えると，現在のところ，2ないし 5%の誤り率があるようである．幸いなことにこの装置は定まった位置の文字について誤りを起こすことが多い．すなわち，第1，第3，第4の文字については正しく読むか，第2の文字に誤りが出るといったたぐいである．このような誤りについては計算機による直接修正の可能性がある．そのデータの性質によって前後の文字との関連性を用いて修正を行なうのである．

ここでは英語の人名を例にとって，誤って入力された文字をどの程度修正できるか調査した．すなわち人名にあらわれる文字の連続の統計をとり，それによって入力された文字の修正の可能性を調べた．この方法によって，多くの場合誤って入力された文字を 90% 以上の精度で訂正できることがわかった．作成されたプログラムは，必要な統計をとるものと，それにより誤りを修正するものとからなっている．これらは，IBM-7040 の COBOL によって組まれた．

チェックに用いる前後の文字の関連性には，2文字の関連性，3文字の関連性，4文字の関連性などがあるが，2文字の関連性によるものは駄目であるが，3文字の関連性によるとかなりよい結果がえられた．表

例——MA？Y

MA？	頻度	A？Y	頻度
MAB	5	ABY	1
MAG	5	ALY	4
MAL	5	AMY	46
MAR NE*	8,316	ARY End	6,466
MAR End	5	ARY NE*	20
MAS	22		
MAT	12		
MAU	2		
MAW	1		
MAY	4		

（注）NE は Not End で，語の途中にあることを示す．

はその例で，MA？Y という3番目の文字がわからない場合，MA？ および A？Y の二つの3文字関連 (trigram) の二つを調べ確率の高い R を入れるわけである．　（関　栄四郎）

F-48. 混成計算機法を使った無作為探索による助変数最適化

G.A Bekey, M.H. Gran, A.E. Sabroff and A. Wong: Parameter Optimization by Random Search Using Hybrid Computer Techniques [Proc. FJCC, 1966, pp. 191～200]

複合力学系の助変数最適化問題を考えるとき，適化規準函数の設定と助変数修正の algorithm および計算機技術の面が問題になる．この algorithm の典型としては relaxation や steep descent の方法によるものがあるが，これは主として規準函数の極値が唯一である場合に適し，非線形系ではこの条件はみたされない．また規準函数が助変数につき部分的に微分可能の場合や背筋状をなすときなど，収束しないかまたは非常におそい．

そこで，この論文では，混成計算機により，一種の無作為探索で総体的最適値を見出す方式をとる．混成

計算機を使った無作為探索については，助変数2個で step の大きさ一定，かつ単一の代数的規準函数を用いた研究があるが，ここではそれを拡張して，9個の助変数を持つ非線形力学系に適用した．実例として，人工衛星姿勢制御について説明がある．

系を記述する方程式の解を

$$x=x(x_0;\ t;\ \alpha)$$

x_0 は初期条件，α は助変数 vectior とする．各々の x_0 と α について，規準函数を

$$J(x_0;\ \alpha)=\int_0^t g(x;\ t;\ \alpha)dt$$

で定める．例としては，衛星の制御に要する燃料または時間など．

調べたいすべての初期条件空間 X_0 にわたる助変数設定の効果を見るため，X_0 を q 個の $X_1\cdots, X_q$ に分割して次の規準函数を導入する．

$$F(\alpha)=\sum_{j=1}^{q} J^*(x_{0j},\ \alpha)\qquad (x_{0j}\in X_j)$$

$$J^*(x_{0j},\ \alpha)=\max_{x_{0j}\in X_j} J(x_{0j},\ \alpha)$$

探索の方法としては，まず極小値を求め，次に総体的最適値を見出す2段階に分ける．第1段は absolute positive and negative directional biasing を用いて次のような手続きを定める．

（1） analog 高周波雑音発生機より無作為に $N^{j\prime}$ をとる．

（2） 分散行列 $C^{j\prime}$ を適当に選んで

$$\Delta\alpha^j=C^jN^{j\prime}$$

を計算する．

（3） $\alpha^{j+1}=\alpha^j+\Delta\alpha^j$

として，analog により

$$F'=F(\alpha^{j+1})$$

を計算し F^j と比較する

$F'<F^j$ ならば　$F^{j+1}:=F'$

$\Delta\alpha^{j+1}:=\Delta\alpha^j$

$F'>F^j$ ならば　$\Delta\alpha^{j+1}:=-\Delta\alpha^j$

として再び F 値をくらべる．

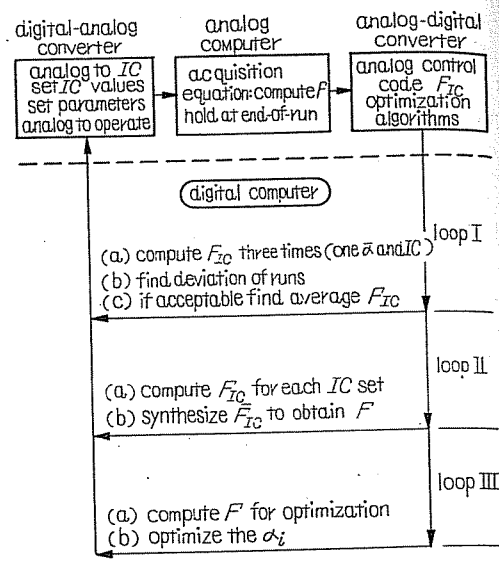
（4） F 値が改善されない α^j の回数が規定の値になったところで，上記の loop をうちきる．

第2段は，第1段で得た極値から出発して

$$\Delta\alpha^j=C^jN^j$$

で，普通の無作為探索を行なう．分散 C^j は初め十分小さくとり，F 値の改善されない α^j の回数の増加につれて，次第に大きくしていく．

後半は実例で，衛星の一つの軸をある方向に向け，この軸のまわりの回転を零とする方法を扱っている．計算機構は下図のようである．



第1図　Optimization master logic flow diagram

規準函数を選んで loop I では，ある X_j と α につき，J を3回計算し，analog によるバラツキを検定する．その偏差が規定以内の場合は，三つの値の平均をとって次の計算に進む．しからざるときは再び J を3回計算し，前の値はすてる．この方法により，analog 計算の精度をあげたことになる．

loop II は，すべての $X_j(j=1\sim q)$ に関するくりかえし．

loop III で $F(\alpha)$ の値を計算し，$\Delta\alpha$ の step により，改善されたかどうかを調べる．

このほか，計算機の誤動作や助変数設定の不都合により計算時間が異常に長びくとき，適当に中止して規準函数値を大きな値にかえ計算を続ける機構を入れている．

以上の計算の結果を解析すると

（1） 収束の点で absolute biasing の方法は単純な無作為探索を効果的に改良したといえる．

（2） 極小値を求める段階では，分散 C^j を有効に変えるうまい手続が見あたらない．他の研究によると，第1段では C^j を一様にとっても収束性において変わらないという結論が出た．

（3） 無作為総体探索は非常に有用である．そこで，ここでとりあげた方法による助変数の最適化は，非線形力学系について，有効かつ program しやすい，そして混成計算機に適した方法であると結論してある．

（岡崎精一）