計算機アーキテクチャ　68－1
(1987・11・12)

# 配線処理用ＣＡＤエンジンについて

CAD Engines for routing

鈴木　敬　　井出　進博　　大附　辰夫
Kei Suzuki　Nobuhiro Ide　Tatsuo Ohtsuki

早稲田大学理工学部電子通信学科
Waseda Univ. School of Science and Engineering
Department of Electronics and Communication Engineering
Ohkubo 3, Shinjuku-ku, Tokyo 160, Japan

ＶＬＳＩの設計は集積度の向上とともに扱うデータ量も増え非常に時間のかかる部分になっており専用ハードウェアにより高速な処理を実現する試みが成されている。我々はＶＬＳＩのレイアウト設計の中の配線処理を高速化する２種類のハードウェアを提案し試作を行っている。その一方は迷路法を実行する並列プロセッサで、もう一方は図形処理により径路を求めるグリッドレス・ルータを実行する連想メモリを用いたハードウェアである。本稿ではこれらのハードウェアの構成と実験結果を示す。

We present two CAD Engines for routing. One of them is based on Lee algorithm. This processor requires only O(N) prosessor elements for N x N grid plane, and find a path with length L in O(L) time. Another is based on gridless routing algorithm (improved Line Search Algorithm). This processor uses Content Addressable Memory (CAM), which can solve geometric search problems in constant time. We describe these architectures and performance measured on prototype machines.

## 1．迷路法のハードウェア化

### 1－1．迷路法を処理するハードウェア

LEEの迷路法[1]はアルゴリズムが単純なため、以前からハードウェア化が検討されている[2]-[15]。ハードウェア化については２種類の考え方がある。

一つはラベル付けの処理をパイプラインにより並列化する考え方である。これはラベル付けの作業をいくつかの処理に分け、これらの処理をパイプライン化し高速化を図るものである[2]-[4]。パイプライン型のプロセッサは少ないハードウェアで処理の高速化が可能であるが、見付かる径路の長さLに対してソフトウェアで最悪の場合O(L²)かかっていた径路発見の手間を改善することは出来ない。

もう一つの考え方は多数のプロセッサ（以下ＰＥ：プロセッサ・エレメント）を格子状に接続し、一つのＰＥが一つのセルに対するラベル付けの処理を行うようにした並列プロセッサである（図１）[5]-[7]。各ＰＥは隣接するＰＥ（セル）からのラベル付けの信号を受け取り、そのＰＥの担当するセルへのラベル付け処理を行う。このアレイ型の並列プロセッサは扱う配線領域の格子数（一辺をＮとするとＮ²）のＰＥを用意することにより処理の手間をO(L)にすることができる。しかし、この方式では１つのセルに対するラベル付け作業は単純なので１つのＰＥの機能は低くてもよい反面、実用規模の配線領域を扱うには、膨大な数のＰＥを必要とする欠点を持つ。また、ラベル付け時に稼動しているＰＥはウェーブフロント上のＰＥ（Ｎ²個のＰＥに対しO(Ｎ)個）だけが稼動しているだけなのでハードウェアに無駄が多い。

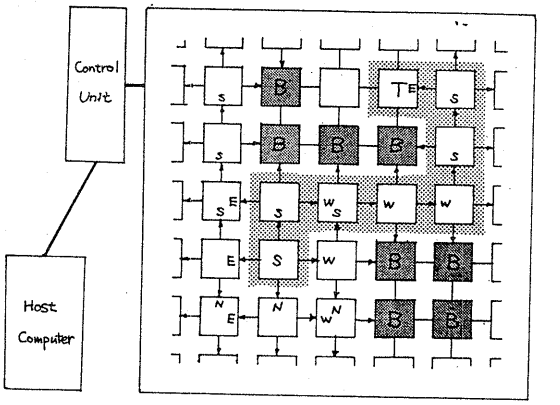



図１　並列プロセッサによる迷路法

そこで膨大なハードウェアを必要とするアレイ型プロセッサをうまく畳み込んで少ないＰＥで同等の効果を得ようという試みが成されている[8]-[11]。

我々も迷路法のラベル付けの性質を考慮し、適切なフォールディングにより少ないハードウェア量で迷路法を実現する方法を提案した[14]-[15]。この方法では、セル－ＰＥの割り当て規則が悪いと１つのＰＥに処理が集中して並列度が下がり高速化が図れない。そこでウェーブフロントの性質を考えて割り当て規則を決める必要がある。

１－２．セル割り当て規則

フォールディングの方法はいくつか考えられる。各ＰＥに連続した領域を担当させるもの、各ＰＥをトーラス状に接続して平面上にプロセッサの並びを繰り返させるもの、ＰＥアレイの境界でウェーブフロントが反射するようにしたものなどがある。

ウェーブフロントの性質を考えてみると、１つには『格子上の１点から拡がる』ことがあげられる。領域ごとにＰＥを割り当てるとウェーブフロントの存在する領域を担当するＰＥのみが処理を行なうので、ラベル付けの処理が分散されず、並列性が上がらない。そこで１つのＰＥが担当するセルは、とびとびの座標値（かつハードウェア、ソフトウェアを単純にするため規則的な値）を持つことが望ましい。２つめに『ウェーブフロントは２種の斜線（45°,135°）により構成される』ことがあげられる。この性質からこの斜線上では、セル割り当てが均等になるようにすることにより処理を分散させることができる。３つめは『格子上のセルをチェス盤のように白黒２色に塗り分けたとき、ある世代のウェーブフロントは全てが同じ色のセル上に乗る（格子グラフは２部グラフである）』ことである。ここからＰＥを２群に分けて接続しＰＥの制御を容易にすることができる。以上３つの条件を満たすように、ＰＥを“ひねったトーラス”（Twisted Torus）型[16]に接続する（図２）ことによりラベル付け処理の均等化が図れる。

この接続による格子上でのＰＥ番号（それぞれのＰＥが持つ識別番号）の配列は、一般化すると2a*2bの長方形を2cづつずらした形となる（図３）。ただし、この時以下の関係が成り立たなければならない。

G.C.M.（｜a－c｜，b）＝１

G.C.M.（｜a－b＋c｜，b）＝１

この条件が満たされていればウェーブフロントを構成する２種の斜線上には、Np/2（NpはＰＥの数）個毎にしか同じＰＥの担当するセルは現れない。我々が試作したルーティング・プロセッサ（以下ＲＰ）は、a=2,b=8,c=1（ＰＥ数64）である。

またウェーブフロントの長さ（セル数）は配線領域の大きさをN²とすると平均的にはO(N)程度である。処理が各プロセッサに均等に割り当てられればO(N)台のプロセッサでもラベル付け処理の手間を最悪O(L²)からO(L)にすることが出来る。我々はシミュレーションによりN²の配線領域に対してN／２台のプロセッサを用意すればO(L)の手間で径路を発見できるという結果を得た。ＲＰは、ＰＥの数は64台で128＊128セルで２層の配線領域を扱うことができる。

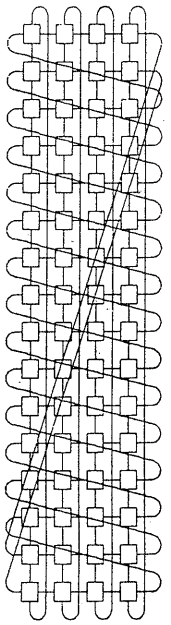


図２　ＰＥ６４台のプロセッサアレイ
（Ｔｗｉｓｔｅｄ　Ｔｏｒｕｓ）

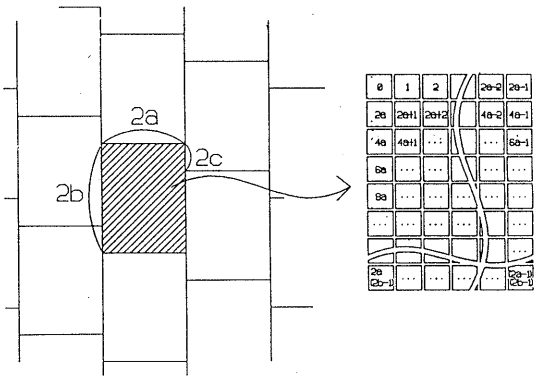



図３　一般化したセルの割当規則

１－３．ハードウェア構成

ＲＰはホスト・コンピュータのバックエンド・プロセッサとして利用することを想定している。ＲＰ自体は、図４のように複数のＰＥとＰＥ群を管理しホスト・コンピュータとの間でデータの転送を行なうコントロール・ユニット（ＣＵ）から成る。ＰＥとＣＵ間は、バスにより接続されており、３線のハ

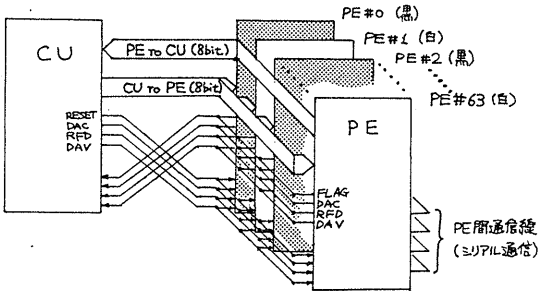



図４　ＲＰの構成

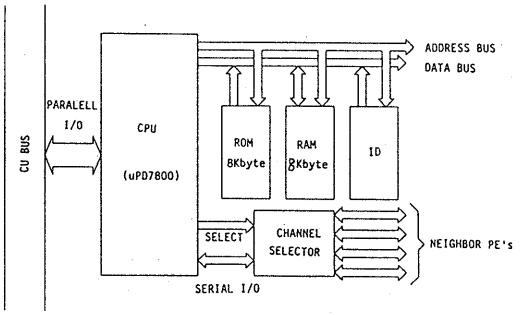



図５　ＰＥの構成

ンドシェィクによりデータを転送している。この構成はＣＵからのコマンドにより動作するSIMD型のマルチ・プロセッサである。しかし、各ＰＥは内部状態により異なる処理を行うこともある。各ＰＥは４本のシリアル通信線を持っており、各ＰＥ間はセル割り当て規則に基づいた関係で接続されている。ＰＥ間の通信はこのシリアル通信線を利用している。

各ＰＥには、汎用の8bitマイクロプロセッサμPD7800を用いており、１ＰＥ当り１１チップ（RAM 8KByte,ROM 8KByte）で構成している（図５）。１枚のプリント基板（40cm*25cm）に４台のＰＥを乗せ、このプリント基板を16枚並列に接続して64台のＰＥアレイを構成している。

１－４．実験結果[17]

ひねったトーラス型の接続を採用したＲＰの評価を行うため実行時間の測定の他に２種類の評価値を設定し、トーラス型の接続との比較を行った。評価値は以下の２つである。

１）プロセッサ稼働率
　PU （Processor Utilization）

２）平均重畳度
　MAF （average Multiple Assignment Factor）

どの指標も各ＰＥに均等に処理が割り当てられているかを示すものである。ＲＰではラベル付けにかかる時間Ｔは次の式で与えられる。

$$T = T0 + 4\,Qmax\ (Tg + Tl) \quad （式１）$$

T0：ＣＵからのコマンド・パラメータの受信などＰＥの負荷に関わらない一定時間

４：ラベル付けのための通信は隣接する４つのＰＥへ行われる

Qmax：各ＰＥ毎のラベル付けされたセルの数（Queueに入っている座標値の数）$Q_i$の最大値

Tg：ラベル付け毎にＰＥ間の同期を取るための時間

Tl：ＰＥ間通信で一回のラベル付け信号を送る時間

T0の項は次の項に比べて十分小さいので式１は次のように書き表せられる。

$$T = k\,Qmax \quad （式２）$$

この時、各指標の実際の算出方法は以下の通りである。

$$PU = \frac{\sum_{i=1}^{L}\sum_{j=0}^{Ntp-1} Qij}{Ntp \sum_{i=1}^{L} Qmax\ i} \quad （式３）$$

L　：径路長

Np：ＰＥ数

Ntp：送信側のＰＥ数　＝Ｎｐ／２

Qij：ｉ番目のウェーブフロントにおけるｊ番目のＰＥのQueueの深さ

Qmax i：ｉ番目のウェーブフロントにおける全ＰＥの中のQueueの最大値

$$MAF = \frac{\sum_{i=1}^{L}\sum_{j=0}^{Ntp-1} Qij}{L} \quad （式４）$$

PU は実際の処理時間の内、各ＰＥの実働時間の割合を示し値が大きい程良い、またMAF はラベル付けの１ステップ当りのＰＥ中の最大負荷の平均値を示し値が小さいほど良い。

実験は64台のＰＥで（１）8*8のトーラス型、（２）4*16のトーラス型、（３）8*8のひねったトーラス型、（４）4*16のひねったトーラス型の４つの接続についていくつかの配線の例題（w1～w5:表１）を実行し処理時間と２つの評価値を調べた。

・時間測定

４種類の接続に対して同じ例題を実行し配線径路長と処理時間の関係を測定した。この結果を図６に示す。ここからこの４種の接続の中では4*16のひねったトーラス型の接続が最も適した接続であることが判る。このグラフの傾きはおよそ1.2でほぼ配線長に比例した処理時間で径路を探索できることが判る。またこの例題における配線長とラベル付けされるセ

表１　配線の例題

| 配線例 | ネット数 | 平均径路長 | 平均端子数 |
|---|---|---|---|
| w１ | ５６ | ３０．５ | ２．５ |
| w２ | ５ | １１６．６ | ４．４ |
| w３ | １４ | ９５．９ | ３．８ |
| w４ | ６９ | １７．３ | ２．４ |
| w５ | ７６ | ２２．８ | ２．４ |

ル数の関係を図７に示す。ソフトウェアでの処理ではラベル付けされるセル数に比例した処理時間を必要とするがこの傾きはおよそ1.9でありソフトウェアでのラベル付けの処理がほぼO(L²)であることが判る。

・PU の比較

表２に各例題に対するＰＵの値を示す。例題によって度合は異なるがひねったトーラス型の接続の方がPU の値が高くなっていることがわかる。

・MAF の比較

表３に各例題に対するMAF の値を示す。これも例題によって度合が異なるがひねったトーラス型の接続の方が低い値となる。

以上の結果からひねったトーラス型の接続の有効性を確認できた。

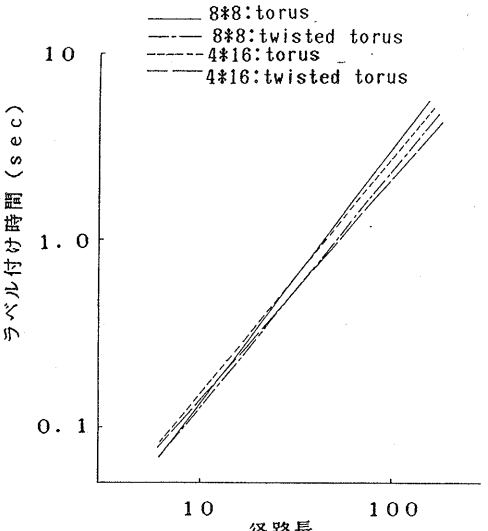



図６　配線の径路長とラベル付け時間の関係

表２　プロセッサ稼働率（％）

| 接続<br>配線 | ８＊８<br>Torus | ４＊１６<br>Torus | ８＊８<br>Twisted | ４＊１６<br>Twisted |
|---|---|---|---|---|
| w１ | ２２．３ | ２４．４ | ２５．９ | ２６．９ |
| w２ | ３５．４ | ３１．８ | ５０．４ | ５１．６ |
| w３ | ３５．１ | ３７．５ | ４５．２ | ４７．１ |
| w４ | ２１．３ | ２２．０ | ２２．４ | ２２．３ |
| w５ | ２１．８ | ２２．５ | ２３．６ | ２３．５ |

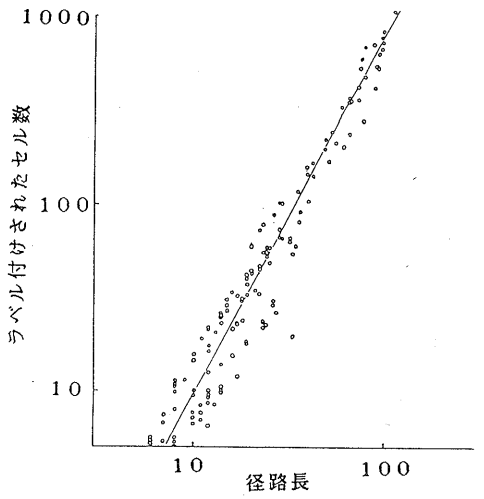



図７　配線の径路長とラベル付けされたセル数の関係

表３　平均重畳度

| 接続<br>配線 | ８＊８<br>Torus | ４＊１６<br>Torus | ８＊８<br>Twisted | ４＊１６<br>Twisted |
|---|---|---|---|---|
| w１ | ２．６８ | ２．４５ | ２．２３ | ２．１８ |
| w２ | ７．５０ | ８．１８ | ５．２０ | ５．０６ |
| w３ | ６．１３ | ５．８１ | ４．６８ | ４．５１ |
| w４ | １．９０ | １．８２ | １．８０ | １．８０ |
| w５ | ２．１５ | ２．０３ | １．９５ | １．９３ |

１－５．ハードウェア・ルータを用いた引き剥し・再配線手法

迷路法のように配線を一本毎に逐次的に結線して行くと既に引いた配線が障害物となって結線不可能な配線が生じて来る。この場合、障害物となっている配線を引き剥して順番を入れ換えて結線を行う手法（Rip-up and Reroute）が用いられる。しかし一般的には順番を入れ換えるだけでは結線できない場合もある。この様な場合は、

（１）障害となっている配線を見つける[18]
（２）指定した配線を引き剥す
（３）指定したセルを通る径路を発見する
（４）指定したセルをなるべく通らない径路を発見する。

といった機能を用いて配線を進めて行く。

我々は試作したＲＰ上に上の機能のうち（３）を除く３つの機能をインプリメントした。またある配線領域において径路が必ず通るセルをボトルネック・セルと呼ぶが、このセルを発見するアルゴリズム[19]もＲＰ上にインプリメントした。

現在、ＲＰをワークステーションに接続し、対話型の配線処理システムを構築中である。このシステムではＲＰ上に実現した幾つかの機能の有効な利用法、配線処理を進めて行く上で必要な新しい機能の検討などを行う予定である。

## 2. グリッドレス・ルータのハードウェア化

### 2－1. グリッドレス・ルータと図形処理

線分探索法[20][21]は迷路法と共に古くから知られている配線手法である。迷路法と比べて次のような特徴を持つ。

| 迷路法 | 線分探索法 |
|---|---|
| ・最短径路 | ・曲がりが少ない径路 |
| ・配線領域の大きさに比例したメモリ | ・少ないメモリ |
| ・見つかる径路の長さの自乗の手間 | ・速い |
| ・径路があれば必ずこれを発見する | ・径路探索能力が低い |

線分探索法の最大の欠点は径路探索能力が低い点である。この点を改善するために計算幾何学の手法を用いた径路を探索する手法がいくつか提案されている[22]-[25]。この様な配線手法を総称して”グリッドレス・ルータ”(gridless router)と呼んでいる。グリッドレス・ルータは径路が存在すれば必ずこれを発見することが出来る。

我々はこのグリッドレス・ルータを高速化する専用ハードウェアを提案した[25]。グリッドレス・ルータは迷路法のような並列化が難しいので処理の中で行われる基本的な図形処理操作を高速化する方針をとった。基本的な図形処理操作は幾つか存在するが、我々がハードウェア化を想定しているImproved Line Search Algorithmでは以下の二つの図形探索問題が基本となっている。

Ａ）線分探索問題

図８のように、平面上にｘ軸と平行な横線分の集合Ｓと１点Ｐ(X0,Y0)が与えられた時、点Ｐからｙ軸に平行に延ばした線分が最初に出会う横線分を報告する。

Ｂ）線分交差問題

図９のように、平面上にｘ軸と平行な横線分の集合Ｓとｙ軸に平行な線分Ｖ(X0,YU,YL)が与えられた時、Ｖと交差する横線分を列挙する。

こういった問題はソフトウェアでは、木構造のデータ構造を用いてO(log n)（ｎは集合の要素数）の手間とメモリ（あるいはO(log² n)の手間とO(ｎ)のメモリ）で解くことが出来る。これに対し連想メモリを用いるとO(１)の手間で処理できる。

### 2－2. 連想メモリを用いた図形処理プロセッサ

我々は３－１で示した二つの図形探索問題を連想メモリ（Content Addressable Memory:以下ＣＡＭ）を用いて解く方法を提案した[26]。ＣＡＭを用いることにより処理の手間をデータ数によらず一定時間に抑えることが出来る。

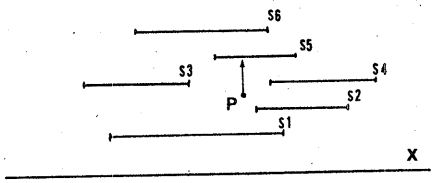



図８　線分探索問題

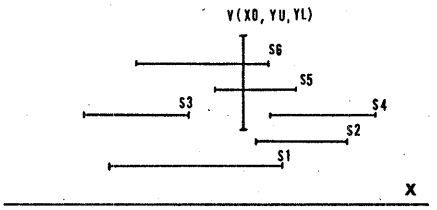



図９　線分交差問題

最近のLSI技術の発達により、大容量のＣＡＭチップを作ることが出来るようになった[27][28]。図10がＮＴＴのＣＡＭチップの構成を示すブロック図である。１ワード３２ビットで１２８ワードの記憶アレイにデータを記憶し、複数のチップを接続してワード数を増やすことが出来る。記憶アレイの部分はＲＡＭと似た構成であり、ＲＡＭがアドレスによりデータをアクセスするのに対し、連想メモリではデータの一部分を示し、その部分が一致するデータをアクセスする（具体的には各ワード毎に存在する応答レジスタに”一致”したことが記録される）ことができる。この動作を一致検索と呼ぶ（図11）。一回の一致検索に要する時間（データの読み出し時間を除く）はワード数によらず一定である。

ＣＡＭは複雑な検索を行えるように一致検索の際、各ワード毎に一致検索の結果と応答レジスタ値の論理演算（AND,OR）の結果を新たに応答レジスタの値とする論理演算機能を持つ。この機能を用いて、データの最大値（maximum）、最小値（minimum）、あるいはある値より大きいデータ（greater than）、小さいデータ（less than）を検索することができる。これらの操作を関係検索と呼ぶ。関係検索は１ビット毎の一致検索を組み合せて行う。この操作はデータ数には関係なく、データのビット幅に比例した手間がかかる。通常、データのビット幅は固定値なので関係検索も一致検索と同様にO(１)の手間となる[29]。

問題Ａに対しては、ＣＡＭの１ワードを図12のように４つのフィールドに分け、次の４段階の処理により解となる線分を求める。

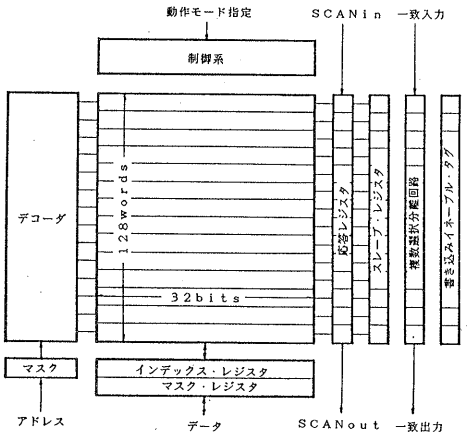
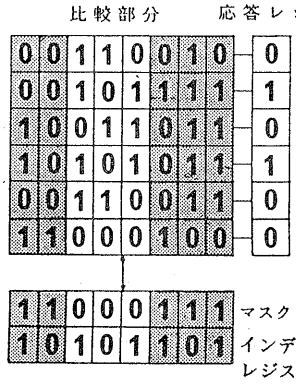



図１０　４Ｋｂｉｔ連想メモリチップの構成




① インデックスレジスタに値を書き込む。

②マスクレジスタに値を書き込む。

③比較命令を行う。

④応答レジスタに結果が入る。

図１１　連想メモリの基本動作

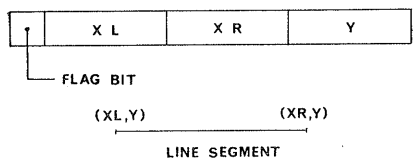



図１２　線分データの表現

①　XR≧Xoを満たすデータを選ぶ。
②　①を満たし、XL≦Xoを満たすデータを選ぶ。
③　②を満たし、Y≧Yoを満たすデータを選ぶ。
④　③を満たすデータの中で、最小のＹ値を持つデータを選ぶ。

このように４回の関係検索で解を求めることができる。これにより、ＣＡＭを用いることにより今まで複雑なデータ構造を用いてO(log n)の手間を必要とした処理を一定時間で行える。問題Ｂについても同様に解くことが出来る。またソフトウェアの場合は各問題を効率よく解くためには、問題毎にデータ構造を用意する必要があったが、ＣＡＭを用いると同じフォーマットのデータで各々の問題を解くことが出来る。

グリッドレス・ルータでは図形データの集合に対する挿入・削除・検索の３つの操作がランダムに発生する動的な問題を扱う。

図形処理を行うハードウェアとしてはシストリック・アレイを用いたハードウェアも提案されている[30]。ｎ個のプロセッサから成るシストリック・アレイでは挿入・削除・検索とも一回の操作に対してはO(ｎ)の時間を必要とするが、互いに独立したｍ回の挿入・削除・検索のシーケンスはO(ｎ＋ｍ)の時間で処理できる。しかし互いに独立していない場合、即ち検索の結果をもとに挿入・削除を行うような場合、シストリック・アレイではO(ｎｍ)の手間が必要となり効率が悪くなる。

ＣＡＭを利用したハードウェアでは挿入・削除・検索ともデータ数に依存せずO(１)の手間で行え、かつ独立でないｍ回の操作に対してもO(ｍ)の手間で実行できる利点がある。この比較を表４に示す。

表４　各ハードウェアの比較　　データ数：ｎ　問題数：ｍ

| | 独立な問題 | | 非独立な問題 | |
|---|---|---|---|---|
| | 時間 | (ﾌﾟﾛｾｯｻﾒﾓﾘ) | 時間 | (ﾌﾟﾛｾｯｻﾒﾓﾘ) |
| ソフトウェア<br>Linked List | $O(mn^2)$ | $O(n)$ | $O(mn^2)$ | $O(n)$ |
| Segment Tree | $O(m\log n)$ | $O(\log n)$ | $O(m\log n)$ | $O(\log n)$ |
| Priority Search Tree | $O(m\log n)$ | $O(n)$ | $O(\frac{m}{\log^2 n})$ | $O(n)$ |
| Systolic Array | $O(m+n)$ | $O(n)$ | $O(mn)$ | $O(n)$ |
| ＣＡＭ | $O(m)$ | $O(n)$ | $O(m)$ | $O(n)$ |

２－３．ハードウェアの構成

図13に我々が試作したハードウェアの構成を示す。ＣＡＭにはＮＴＴで試作された４Kbitのチップを用いている。ＮＴＴのＣＡＭチップはクロック・パルスに同期して与えられるコマンドによりその動作が決まる。４Kbitのチップの場合、３０種のコマンドを持つ。各コマンドは全て１サイクルのクロック内で処理され、クロック・パルスの周期は最高140nsである。このタイミングで動作させるためには、ＣＡＭを使うＣＰＵが直接コマンドを与えるのでは遅い。そこで、シーケンサを用いてマイクロコードを駆動し、これにより処理を行うのが効果的である。

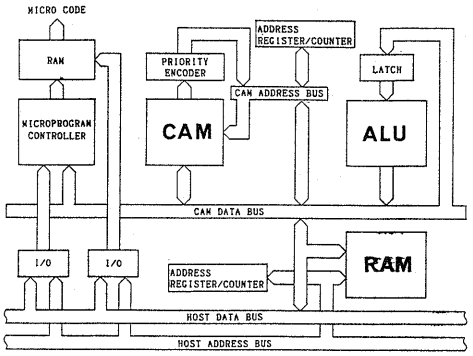



図１３　連想メモリを用いた図形処理ハードウェア

このハードウェアは、ホスト・コンピュータのバスに接続しており、スレーブ・プロセッサとして使われる。内部は大きく４つの部分からなる。シーケンサはバイポーラのビットスライスプロセッサのシリーズの中のAm2910を用いている。このシーケンサでＲＡＭに記憶したマイクロ・コード（１ステップ１００ビット、４Ｋステップ）をクロックに合わせて送り出しハードウェアの各部分の制御を行う。ＲＡＭ（１ワード３２ビット、４Ｋワード）は、ホスト・コンピュータとのデータの交換や作業用として使われる。ＡＬＵは検索データやマスクデータの加工などに用いる。これもバイポーラのビットスライスプロセッサAm2900シリーズのLSIを利用している。ＣＡＭ部はＣＡＭチップ、複数分離回路、アドレス・レジスタ等からなり、最大６４個のＣＡＭチップを登載できる。現在は８個のチップが実装してある。

ハードウェアは200nSのクロックに合わせて送り出されるマイクロ・コードに同期して動作する。マイクロ・コードはハードウェアの各部分の動作を並列に記述したものであるがこのマイクロ・コードを効率よく作成するために専用のアセンブラを作成した。またマイクロ・コードのデバッグ用にハードウェア・エミュレータなどのツールも開発した。

２－４．実験結果[31]

今回試作したハードウェア上で問題Ａを行い処理速度を測定した。線分及び質問点の座標は一様乱数を用いて求めており、1000回の探索の平均時間を測定した。また比較のために単純な線形リストを用いる方法とPriority Search Treeを用いる方法[32]をプログラム化し（Ｃ言語）、VAX11-785(1.5MIPS)で実行時間を測定した。図14に結果を示す。ここから線分数が多いほどＣＡＭが有効であることが判る。

現在、このハードウェア上に配線処理のアルゴリズムをインプリメントしている。またグリッドレス・ルータで用いる配線データ上での有効なRip-up and Rerouteの手法についても研究を進めている。

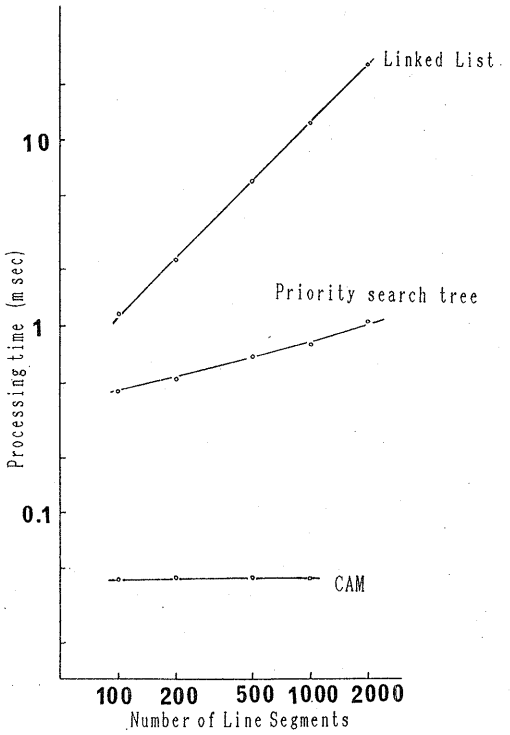



図１４　線分探索問題の実行時間

３．まとめ

配線処理には処理時間の短縮とともに配線の100%自動結線という目的がある。そのためには有効なRip-up and Rerouteの手法の確立が必要でありこれが今後の配線処理用のハードウェアの研究課題であるといえる。

＜参考文献＞

[1] C.Y.Lee,"An Algorithm for Path Connection and its Applications",IRE.Trans.,EC-10,pp.346-365(1961).

[2]R.A.Rutenbar et al.,"A Class of Cellular Architectures to Support Physical Design Automation",IEEE Trans. on CAD, Vol.CAD-3,No.4,pp264-278(1984).

[3]緑川 他、”データ駆動型画像処理プロセッサの配線処理への応用”、情報処理第３０回（Ｓ60年前期）全国大会、2H-2,pp.1873-1874(1985).

[4]Y.Won et al.,"A Hardware Accelerator for Maze Routing",Proc. of 24th DAC,pp.800-806(1987).

[5]A.Iosupoicz, "Design of an Iterative Array Maze Router",Proc.of ICCC,pp.908-911(1980).

[6]M.A.Breuer and K.Shamsa,"A Hardware Router",Degital Systems,vol.4,pp.393-408.
[7]T.Watanabe etal,"A Parallel Adaptable Routing Algorithm and its Implementation on a Two-Dimensional Array Processor,"IEEE Trans.on CAD,Vol CAD-6,No.2,pp.241-250,Mar.1987.
[8]T.Blank et al,"A parallel bit map architecture for DA algorithms," Proc. 18th DAC,pp.836-845(1981).
[9]S.J.Hong at al.,"A physical design machine",VLSI81,J.P.Gray,London Academic Press,1981,pp.257-266.
[10]H.G.Adshead,"Towards VLSI complexity: The DA algorithm scaling problem: Can special DA Hardware help?", Proc. of 19th DAC,pp.339-344(1982).
[11]永安 他、"ＰＡＸによる配線問題の並列処理"、情報処理全国大会第30回（Ｓ60年前期）、6B-5,pp.133-134(1985).
[12]E.Damm and H.Gethoffer,"Hardware support for automatic routing",Proc. of 19th DAC,pp.219-223(1982).
[13]進藤 他、"プロセッサの協調動作に基づく並列配線法"、信学技法、CAS85-152,pp.33-40(1986).
[14]K.Suzuki et al,"A Hardware Maze Router with Application to Interactive Rip-up and Reroute,"IEEE Trans.on CAD,VolCAD-5,No.4,pp.466-476,Oct.1986.
[15]橘 他、"並列ルーティング・プロセッサの試作研究"、情報処理論文誌、Vol.27,No.6(1986).
[16]C.A.Seqin,"Doubly Twisted Network For VLSI Processor Arrays",Proc.of 8th Ann. Symp. on Computer Architecture,pp.471-480(1981).
[17]二上 他、"Processors-to-cells Mapping in a Hardware Maze Router",信学技法,CAS86-203,pp.1-8(1987).
[18]松永 他、"ハードウェア・ルータを利用した再配線手法",信学技報,CAS85-153,PP.41-48,(1986).
[19]松永 他、"並列化を考慮したボトルネック発見のアルゴリズム"、信学技報、CAS86-54,PP.33-38(1986).
[20]K.Mikami and T.Tabuchi,"A Computer program for Optimal Routing of Printed Circuit Connections",Proc. IFIPS,Vol.H47,pp.1475-1478(1968).
[21]D.W.Hightower,"A Solution to Line-Routing Problem on the Continuous Plane",Proc. of 6th DA Workshop,pp.1-24(1969).
[22]佐藤 他,"グリッドレス・ルータ －格子を用いない二層配線径路探索手法－",信学論,Vol.J-69-D,No.5,pp.802-809,May(1986).
[23]M.Sato et al.,"A Fast Line-Search Method based on A Tile Plane",proc. of ISCAS 1987,pp.588-591(1987).
[24]H.Imai and T.Asano,"Dynamic Segment Intersection Search with Applications",Proc. of the 25th Annual IEEE Symp. on Foundations of Computer Science,pp.393-402(1984).
[25]K.Suzuki et al,"A Gridless Router:Software and Hardware Implementation," VLSI'87 pp.121-131(1987).
[26]鈴木 他、"連想メモリによる図形処理問題の解法",信学技報,CAS84-117,pp.13-20(1984).
[27]小倉 他、"4Kb CMOS連想メモリLSI",信学技報、SSD83-78,pp.45-53(1983).
[28]小倉 他、"20Kb CMOS連想メモリLSI",昭和61年信学総全大447,pp.2-235(1986).
[29]奥川、"連想メモリとその応用",bit,Vol.15,No4,pp.318-329(1983).
[30]梅尾 他,"シストリック計算幾何アルゴリズムに関する最近の研究",情報処理,Vol.27,No.11,pp.1270-1281,Nov.(1986).
[31]井出 他,"連想メモリを用いた図形処理装置の試作",信学技法,VLD87-106(1987).
[32]E.M.McCreight,"Priority Search Trees,"SIAM J. Comput.,Vol.14,No.2,pp.257-276(1985).