並列ルーティングプロセッサの試作研究†

橘 昌良††* 鈴木 敬†† 大賀 忠†††
中島 聡†††† 大附辰夫††

本論文では，Lee の迷路法を並列化，ハードウェア化した並列ルーティングプロセッサの構成法，試作結果，及び対話型設計システムへの応用について論じる．Lee の迷路法は，グリッドに対応する $N\times N$ のプロセッサアレイを用意することにより，径路探索時間を $O(N^2)$ から $O(N)$ に短縮することが可能である．しかし，必要とするハードウェア量は $O(N^2)$ で増加し，実用規模のシステムでは膨大なものとなる．これに対して，本論文で提案するアーキテクチャは，ウェーブフロント上のセルのみにプロセッサが割り当てられる方式であるため，$O(N)$ 台のプロセッサにより，処理時間を $O(N)$ に短縮することが可能となっている．また，実際に 128×128 のグリッドを扱える，64 台のプロセッサによる構成の試作を行った．迷路法には，数々の変形があり，これらのアルゴリズムは，配線領域が混雑していない段階では，ソフトウェアによる実行でも平均的には，$O(N)$ の時間で径路を発見できることが知られているが，処理が進み配線領域が混雑している段階では，$O(N^2)$ の時間を必要とする．これに対して，並列ルーティングプロセッサは，配線領域の混雑度に関係なく $O(N)$ の時間で径路を発見することができる．この特徴は，配線工程の最終段階において，対話型システムを使用し，障害となっている既配線の引きはがし，及び再配線を行う際には，さらに有効なものとなる．

1. はじめに

本論文では，配線径路探索において最も一般的な Lee の迷路法[7]を並列化，ハードウェア化した並列ルーティングプロセッサの構成法，試作結果，及び対話型設計システムへの応用について論じる[14]．

Lee の迷路法は，2 点間を結ぶ径路が存在すれば，必ず最短な径路を発見することができるため一般の CAD システムにおいて広く用いられている．しかしながら，ソフトウェアにより実装された逐次実行では，径路長の 2 乗に比例した時間を必要とする欠点がある．このため，実際の配線処理工程では，初期の段階においては線分探索法[9]や，チャネル配線法[10]を使用して大部分の配線を決定し，これらのアルゴリズムでは結線できなかった結線要求に対して Lee の迷路法のアルゴリズムを使用するというのが一般的である．

しかし，このような使い分けを行っても，最も処理時間を必要とするのは，この迷路法による処理の部分である．このため，ソフトウェアにより実装されているアルゴリズムを並列化，ハードウェア化することにより，設計期間の短縮をねらった研究が活発になってきた[1)~5)]．

従来より発表されている迷路法の並列化，ハードウェア化の方式は，グリッドに対応する $N\times N$ のプロセッサアレイを用意することで処理時間を $O(N)$ に短縮するものである．しかし，これらの方式では，必要とするハードウェア量が $O(N^2)$ となり，実用規模のシステムでは，膨大なものとなる可能性がある．

本論文で提案するアーキテクチャは，$O(N)$ のプロセッサアレイによって処理時間を $O(N)$ に短縮することが可能となっている．実際に必要とするプロセッサ数は，シミュレーション結果より，$N\times N$ のグリッドに対して $N/2$ で十分であり，この場合の処理時間は従来の方式の 2～3 倍程度となることが予測された．そこで，$N=128$ のグリッドに対して $N/2=64$ 台のプロセッサにより構成される並列ルーティングプロセッサ（以下では，RP とする）を試作し，上記のシミュレーション結果を実証した．

迷路法には，高速迷路法[8]をはじめ数々の変形がある．これらのアルゴリズムは，配線領域が混雑していない段階では，ソフトウェアによる実行でも平均的には，$O(N)$ の時間で径路を発見できることが知られている．しかし，処理が進み迷路法を必要とするほど配線領域が混雑している段階では，$O(N^2)$ の時間を必要とする．

これに対して，RP は，配線領域の混雑度に関係な

† A Wavefront Machine—New Parallel Processing Architecture for Interactive Routing Design by MASAYOSHI TACHIBANA, KEI SUZUKI (School of Science and Engineering, Waseda University), TADASHI OHGA (Matsushita Communication Industrial Corporated), SATOSHI NAKAJIMA (NTT Telecommunication Networks Laboratories) and TATSUO OHTSUKI (School of Science and Engineering, Waseda University).

†† 早稲田大学理工学部電子通信学科
††† 松下通信工業(株)電波事業部移動通信技術部
†††† NTT 武蔵野電気通信研究所基幹交換技術部
* 現在 (株)東芝

く $O(N)$ の時間で径路を発見することが可能である．この特徴は，配線工程の最終段階において，対話型システムを使用し，障害となっている既配線の引きはがし，及び再配線を行う際には，さらに有効なものとなる．

2. 迷路法の並列化，ハードウェア化について

Lee の迷路法の並列処理アルゴリズムのハードウェア化の方式として，各セルに対して1台のプロセッサを割り当て，これをグリッド状に相互接続するアーキテクチャが考えられる．以下では，この方式により実装された装置を1セル1プロセッサ型と呼ぶことにする．この構成では，各セルに対応する処理の相互関係と，プロセッサの相互接続が同じ形であるため，その動作が理解しやすい．この方式では，ラベル付けと逆追跡を径路長に比例する時間，ラベルのクリアは一定時間内で実行可能である．このため既に発表された論文において提案されているアーキテクチャは，この方式に基づいている．

1セル1プロセッサ型の欠点は必要とするハードウェア量が膨大になることである．LSI, PWB などのレイアウト設計においては，2,000×2,000 程度もしくはそれ以上の規模のグリッドが必要とされている．この大きさのグリッドを処理できる装置では少なくとも 4×10^6 台のプロセッサが必要となる．したがって，実用規模の装置を作成するためには，複数それもかなり大量のプロセッサをまとめて LSI 化し，これを組み合わせる必要が生じる．ところが，LSI チップ内に搭載できるプロセッサ数は，各プロセッサが隣接する四つのプロセッサと相互接続されなければならないため，チップから引き出せるピン数によって大きく制限を受け実際に集積可能な台数を大きく下回ってしまう．

このため，1セル1プロセッサ型で実用規模の装置を作成する場合，その構成は①すべてのプロセッサを1チップに搭載するか，②非常に多数のチップにより構成するかのいずれか一方の方法をとらざるを得ない．4×10^6 台ものプロセッサを1チップに搭載するのは現在の技術ではまだ不可能であり，多チップ構成では信号の伝播遅延が問題となる．また，いずれの方法でも集積度を上げるためプロセッサ自体の機能は，低くせざるを得ず複雑なアルゴリズムを実行させるのは無理であると思われる．

さらにこの方式においては，すべてのプロセッサが同時に動作するのは，ラベルのクリアを行う時のみであり，ラベル付けの際は，ウェーブフロント上（及び，それに隣接する）のプロセッサのみが動作し，それ以外のプロセッサは，休止状態にある．ウェーブフロントの長さは $N\times N$ のグリッドに対して，たかだか $2N$，平均すれば N 程度であると考えられるのでラベル付けの際に動作しているプロセッサは全体の $1/N$ 程度である．

以上のことより，ラベル付けの際にウェーブフロントへプロセッサを割り当てることができれば必要なプロセッサ数はウェーブフロントの長さ，つまり，$O(N)$ で十分であることがわかる．プロセッサの割当てが効率よく行えれば処理に必要な時間は $O(L)$（L は径路長）を維持できることになる．ただし，ラベルのクリアを行うのに必要な時間は，$O(1)$ から $O(N)$ へと増加する．

本論文では，このようにウェーブフロント上へプロセッサを割り当てる方式をウェーブフロント割当型と呼ぶことにする．

ウェーブフロント割当型の構成では，ウェーブフロントの拡張の際に各プロセッサは複数のセルを担当する可能性があるため，プロセスの管理が複雑となり処理速度は1セル1プロセッサ型と比べて低下する．しかし，必要なプロセッサ数は，$O(N)$ となるためプロセッサの機能は，1セル1プロセッサ型よりも高度なものにでき，多様なアルゴリズムへの対応も容易である．また，プロセッサ数が少ないため，プロセッサを LSI 化しなくとも汎用のマイクロプロセッサチップを使用して装置を試作することが可能である．

以上のことを考慮し，試作したハードウェアは，ウェーブフロント割当型の構成をとっている．

ウェーブフロント割当型の構成では，ウェーブフロント上のセルを動的にプロセッサに割り当てる方式と，セル-プロセッサ割当てを固定しておく方式が考えられる．前者では，ウェーブフロントを集中/分散して管理を行い一回の拡張ごとにプロセッサへ分配する必要があり，このためのアルゴリズムは，Lee の迷路法よりも複雑になってしまう可能性がある．後者では，個々のプロセッサが担当するセルがあらかじめ固定されているため管理アルゴリズムは非常に簡単になる．しかし，個々のウェーブフロントに対して最適な割当てを行うことができないため，プロセッサの負荷，つまり，ウェーブフロント上のセルのうち特定の

プロセッサの担当しているセルの数にばらつきが生じ，ウェーブフロントの拡張の速度は，一番負荷の重いプロセッサによっておさえられてしまうため，良い割当規則を見付ける必要がある．

試作したハードウェアでは，セル-プロセッサ割当てを固定している．セル-プロセッサ割当てにはウェーブフロントの拡張の性質を考慮した規則を採用することにより負荷のばらつきをおさえている．

3. 並列ルーティング・プロセッサの構成

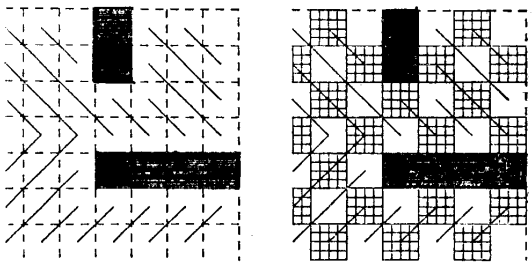
一つの始点から拡がったウェーブフロントには，以下のような性質がある（**図1**）．

①ウェーブフロントは，グリッドの45°または，135°の線分から構成されている．したがって，45°または，135°の線上のセルはできるかぎり異なったプロセッサに分散されることが望ましい．

②グリッドをチェス版状に塗り分けた時，あるセルから偶数距離にあるセルは同色，奇数距離にあるセルは異なった色に塗られている．したがってある時点のウェーブフロントは，すべてが白色，または，すべてが黒色のセル上にある．つまり，データ転送の方向は白色セル⇒黒色セル，または，黒色セル⇒白色セルのどちらか一方である．したがって，プロセッサをこれに対応して白色，黒色の2グループに分けることにより，データ転送の制御を単純にすることができる．

セル-プロセッサ割当ては固定されているのでセル-プロセッサ割当規則は，プロセッサの相互接続の形状としても見ることもできる．上記の要請を満たすセル-プロセッサ割当規則として，試作したRPにおいては，Twisted Torus 型[6]を採用した．

図2に試作した並列ルーティングプロセッサの相互接続を示す．これは，Twisted Torus に白色，黒色のグループ分けを組み合わせたものである．これは，普通の Torus の折り返し部分を**図3**のようにずらしたものに相当する．普通の Torus では斜線に沿って同じプロセッサの担当するセルが，$M\times M$ の Torus の場合，M 個ごとに現れるため，特定のプロセッサへ負荷が集中する傾向がある．これに対して，図2の Twisted Torus では，割当は斜線に対して最適化され同じプロセッサの担当するセルは $M^2/2$ 個ごとに



（a） 45°, 135°. （b） Black and white.

図1 ウェーブフロントの拡がり方
Fig. 1 Wave propagation.

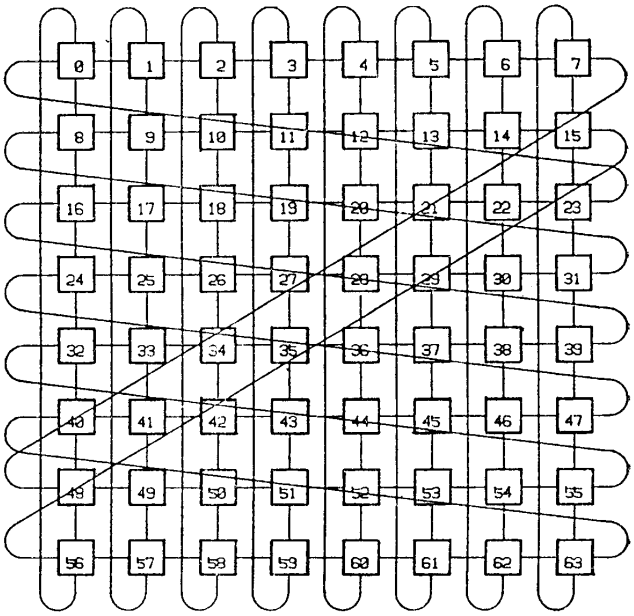


図2 PE の相互接続 (Twisted torus)
Fig. 2 Interconnection of processor elements (Twisted torus).

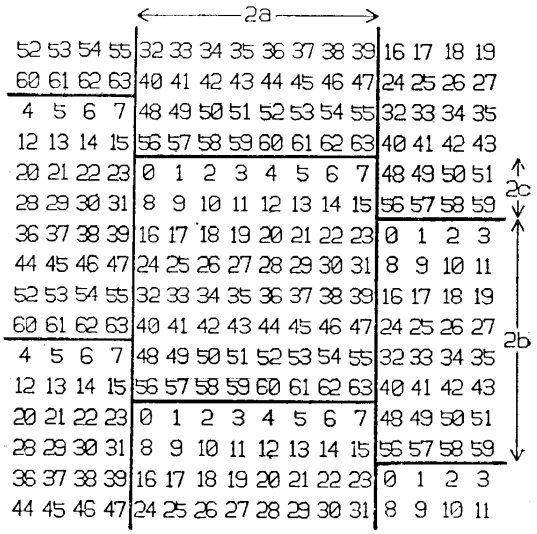


図3 プロセッサのセルへの割当
Fig. 3 A processor-to-cells mapping

現れる.

この接続では，グリッド上で各セルを担当するプロセッサの番号は，$(2a\times 2b)$ の長方形を $(2c)$ ずつずらしたようになる．ただし，このとき以下の関係が成り立つ必要がある．

G. C. M$(|a-c|, b)=1$.

G. C. M$(|a-b+c|, b)=1$.

この条件が満たされれば，ウェーブフロントを構成する2種の斜線上には同じプロセッサの担当するセルが，$2ab$ 個おきにしか現れない．実際にハードウェアを試作する場合は，プロセッサが正方形に配置されると製作が容易になる．試作した RP では，プロセッサの台数を 64 台としたため，$a=4$, $b=4$, $c=1$ となっている．また，扱うグリッドは，発生するウェーブフロントの長さがプロセッサ数と同程度である範囲内の大きさでなければならない．以前に行ったシミュレーション[13]より，一辺でのグリッドの数Nが，プロセッサ数 N_p に対して，$N\fallingdotseq 2N_p$ 程度であればグリッド全体にウェーブフロントを拡げるのに必要とする時間は，Nに比例することがわかっているので，$N=2N_p$ としグリッドの大きさは，128×128 とした．

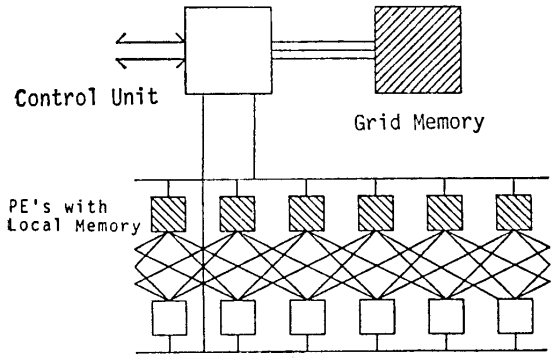


図 4 並列ルーティングプロセッサの構成
Fig. 4 Basic structure of the wavefront machine.

4. 試作した並列ルーティング・プロセッサ

RP は，**図 4** に示すように，コントロールユニット（以下では CU とする）と，実際に径路探索を行うプロセッサユニット（以下では PU とする）によって構成される．CU は，ホストコンピュータとの通信，グリッド情報のグローバルな管理，PU の制御を行う．また，グリッドメモリにグリッド情報のコピーを持つことによりホストコンピュータの負荷を大幅に低減している．PU は，格子状に配置され，相互にシリアルチャネルにより接続されたプロセッサ（以下では PE とする）により構成されている．各 PE は，それぞれ担当する複数のセル情報をローカルメモリに保持している．PE は，二群に分けられ，CU からのコマンドによって同期をとり，相互にデータ転送を行う．

試作した RP では，CU にパーソナルコンピュータ (PC 8801) を使用し，ホスト・コンピュータとしての機能を付け加え単独で使用できるものとした．CU と PU 間は，CU バスで接続している．また，CU バスには，アドレスバスに相当するものはないが，これは，CU が PE に与えるコマンド列中に PE の指定を含むことにより解決している．

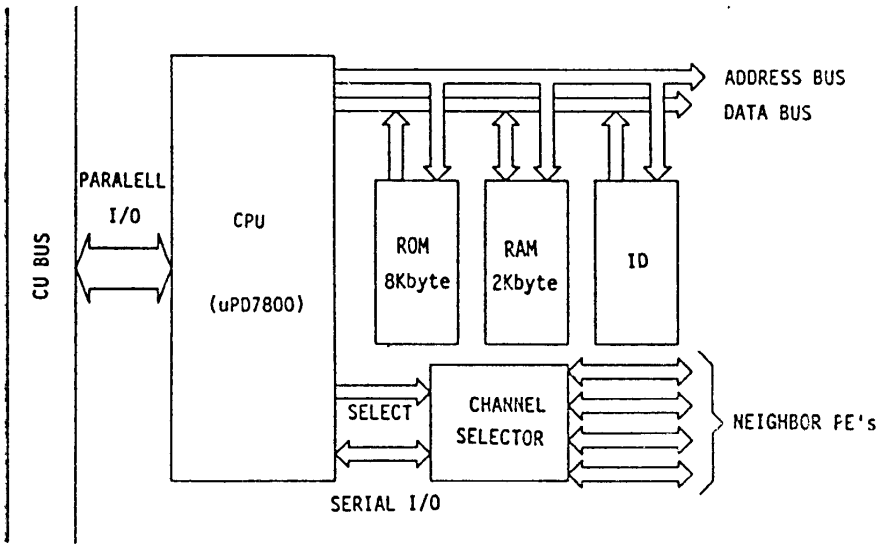


図 5 PE の構成
Fig. 5 Basic structure of a PE.

PE の構成を**図 5** に示す．各 PE は，CPU に1チップマイクロプロセッサ μPD 7800 を採用することにより PE 1台当たり11チップで構成されている．メモリは RAM 2kバイト，ROM 8kバイトを持つ．ROM には PE の動作が記述されているが，その内容はすべての PE で全く同じである．このため，各 PE は ID ナンバをスイッチにより持ち，これにより自分の担当するセルを知ることができるようになって

いる．また，隣接する PE は，シリアルバスにより接続されているが，μPD 7800 はシリアルチャネルを1系統しか持たないため，CMOS スイッチにより4チャネルに切り換えて使用している．

RP は，それ自体が独立して動作するのではなく，**図6**のように，ホストコンピュータのバックエンドプロセッサとして使用することを前提として構成した．この構成により，専用のオペレーティングシステムなどのソフトウェアの開発のための時間を短縮でき，CAD システムへの組み入れが容易となる．また，RP で行うのが効率的でない周辺装置の制御や，配線のための前処理などはホストコンピュータで実行することができるので構成を簡素化することが可能となり開発が容易になった．

ホストコンピュータと RP の処理の分担は，①グリッド情報，ネット情報を収容したファイルの管理，オペレータとの対応などの煩雑な処理は，ホストコンピュータが行い，② RP は，前処理の終わったグリッド情報，ネット情報をホストコンピュータより受け取り，径路情報を送り返すことになる．

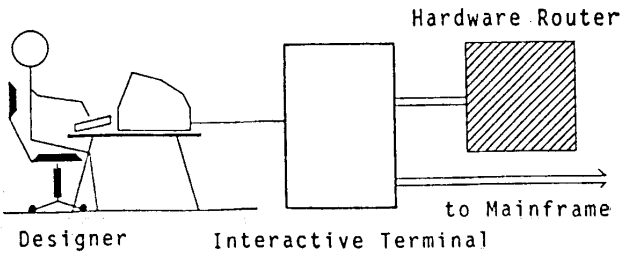


図 6 並列ルーティングプロセッサの運用形態
Fig. 6 A suggested mode of using hardware engine.

表 1 RP の諸元
Table 1 Configuration of the wavefront machine.

Processors	8*8 (twisted torus)
Grid	128*128 (cells)
Data transfer (grobal)	8 bit parallel 650 μs/2 byte
Data transfer (local)	serial (4 direction) 85 μs/1 byte
PE structure	CPU μPD 7800 (2 MHz) ROM 8 k byte RAM 2 k byte

5. 実験結果

実験は，単一径路について，①禁止領域のない場合，②禁止領域のある場合について，ラベル付けにおいて処理時間を，ウェーブフロント1世代（グリッド上での単位距離）ごとに求めた．つぎに四つの配線例について，処理時間と PE の稼動率，ラベル付けされたセルの重畳度* を記録した．

PE の稼動率は，i 番目のプロセッサの動作状態，休止状態の時間をそれぞれ T_{ia}, T_{ib} としたとき，

$$P=\sum T_{ia}/\sum(T_{ia}+T_{ib})$$

とした．

(1) 単一径路

4例についてラベル付け過程の経過時間をウェーブフロント1世代ごとにプロットしたものを**図7**に示す．(1), (2)は禁止領域のない場合であり，(1)がグリッドの中心，(2)がグリッドの隅からウェーブフロントを展開させた場合である．(3), (4)は禁止領域として，それぞれ，10ブロック，20ブロックがランダムに存在している例である．

図7において，(a)はセルの重畳度が1の場合，(b)はセルの重畳度が8の場合の処理時間である．

(1)～(4)の配線例において，すべてのプロットは(a), (b)にはさまれた領域にあり，セルの重畳度は，たかだか7に抑えられている．また，(1)では，距離の短い部分では，距離の2乗に比例して処理時間が増加し，その後，増加が抑えられている．これは，ウェーブフロントがグリッドの端の壁に達し長さが短く

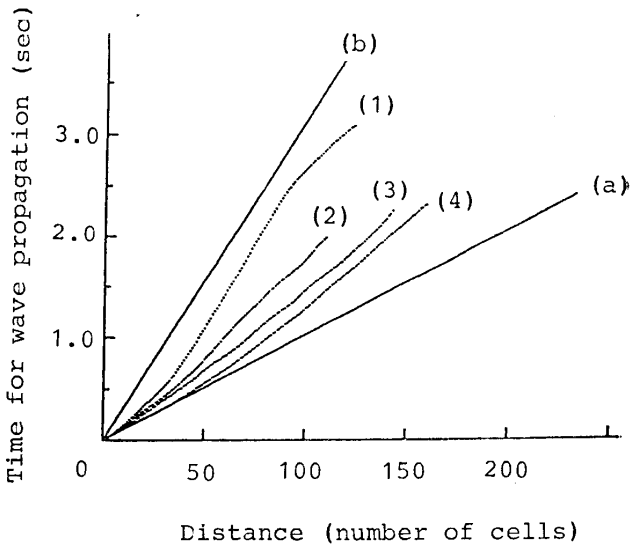


図 7 単一径路のラベル付け時間
Fig. 7 Running time for wave propagation (1).

* ラベル付けの際に，ある一世代のウェーブフロントについて PE の担当したセル数の最大値

なるためである．その他の例では，ほぼ処理時間が直線的に増加している．

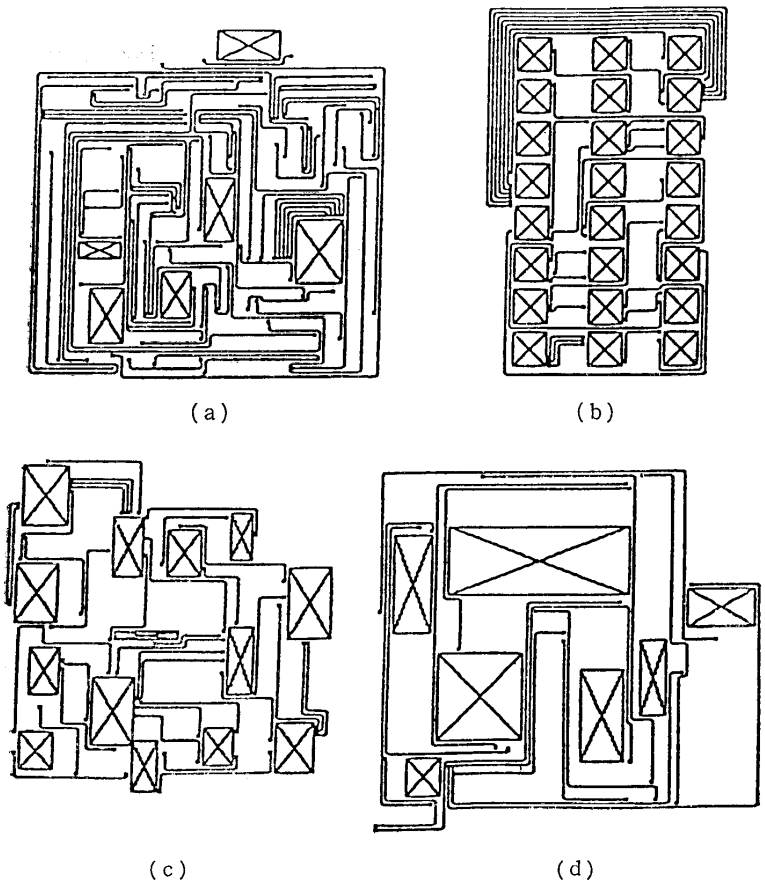


図 8 複数配線例
Fig. 8 Sample problems for one-layer routing.

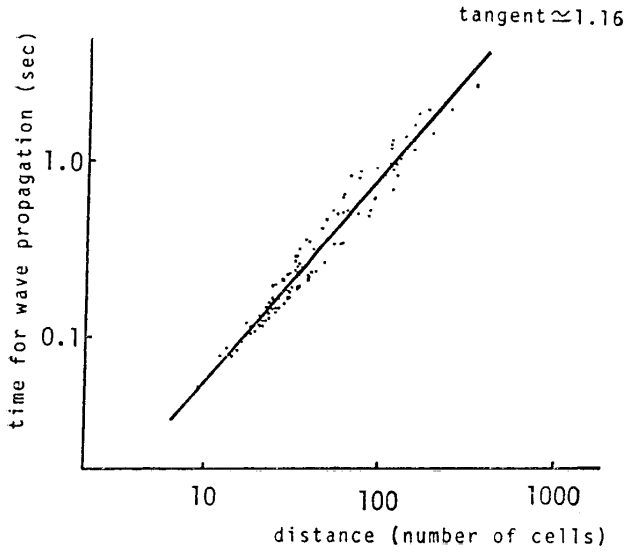


図 9 複数径路のラベル付け時間
Fig. 9 Running time for wave propagation (2).

(2) 複数配線

複数の始点，終点ペアで構成したネットリストについて径路を求めた結果を示す．

配線例は，図8(a)～(d)の4通りで，各径路についてラベル付けに要した時間を縦軸，径路長を横軸に log-log でプロットしたグラフを図9に示す．このグラフを見ると，多少のばらつきはあるものの，ほぼ直線上にプロットが乗っていることがわかる．この直線の傾きは，1.16であるので，処理時間はほぼ距離に比例している，と言える．

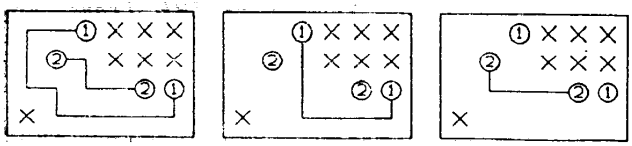
表2に図8(a)～(d)の配線例についての各種測定値を示す．4例において配線時間は，平均配線長，ブロック数などが異なるにもかかわらず，距離に比例して，ウェーブフロント1世代当たり約26 ms である，という結果を得た．稼動率は，ほぼ50% であり，セル-プロセッサ割当てが効果的であることを示している．また，セルの重畳度の平均は，2～3であり，1セル1プロセッサ型と比べて2～3倍の処理時間となることがわかる．

6. 対話型配線システムについて

配線処理の最終段階においては，迷路法のアルゴリズムを用いても径路を発見できないことがしばしばある．これは，既配線により形成された障壁によって既に径路が存在しないことが原因であることが多い．したがって100% 結線を達成するためには，このような障壁を形成している既配線の径路変更

表 2 複数配線例の結果
Table 2 Performance data for the sample problems.

	(a)	(b)	(c)	(d)
ネット数	59	30	35	15
ブロック数	6	24	14	7
総配線長	3,079	1,510	989	1,451
平均配線長	52	50	28	97
処理時間(s)	78	41	23	37
時間/Net (s)	1.3	1.4	0.7	2.5
時間/距離(ms)	26	27	24	26
ラベル付け回数/距離	2.5	3.1	2.0	2.7
稼動率(%)	45	53	50	44



(a) 同時配線　(b) 端子対①を優先　(c) 端子対②を優先

図 10 順序を入れ換えても効果のない例
Fig. 10 A case where changing wiring order doesn't work.

を行う必要がある．このような処理を行う配線システムは，ダイナミックルータ[11]と総称されすでにいくつかの手法が提案されている．

既配線の径路変更は，一時的に配線を引きはがした後再配線することにより行われるが，**図10**のようにただ単に順序を入れ換えただけでは効果がない場合も存在するので，未結線と障壁となっている既配線の概略のルートを指定する必要が生じる．このため，この処理は対話型のシステムで実行されることが望ましい．また，未結線ネットに対して，障壁が複数の既配線によって構成されている場合も多いため，その選択を行う必要もあり，高速の対話型システムが要求されている．

処理の高速化のために，変更する既配線の選択や概略径路の指定に知識工学的手法を応用する研究[12]も行われているが，この工程において最も処理時間を必要とするのは，障壁となっている既配線を発見するための処理である．

なぜならば，障壁を発見するためには，実質的に全配線領域を探索する必要があり，このため，高速迷路法などの探索領域の限定による高速化の手法が意味を持たないからである．

このように逐次処理では $N\times N$ のグリッドに対して $O(N^2)$ の時間を必要とする処理を，並列ルーティングプロセッサは，ラベル付け処理の並列性を利用することにより，$O(N)$ の時間で処理することが可能であり，対話型配線システムのバックエンドプロセッサとして使用した場合，大きな威力を発揮するものと思われる．

以下に迷路法で径路が発見できない場合の障壁となっている既配線の探索，および，引きはがしの効果がある既配線の選択の手順を示す．ただし，ここでは1本の既配線を引きはがす場合についてのみ述べる．

①ある始点，終点ペア (A, B) が結線不能となったとすると，A点からラベル付けされたセルを含む既配線は，AB 間の障壁の候補である．B点より新たにラベル付けを行う．この時ラベル付けされたセルを含む既配線のうち，A点からラベル付けされたセルを含むものは，1本で AB 間の障壁となっている既配線である．特に，両点よりラベル付けされたセルの集合は，極小 AB セパレータ* を形成している（**図11**）．

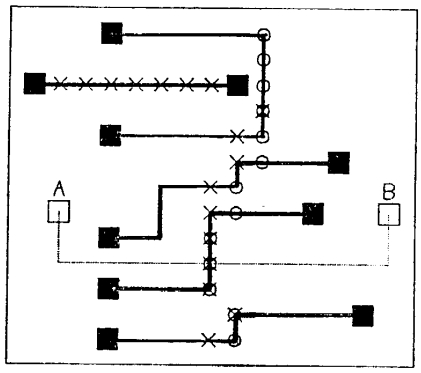


図 11 極小セパレータ
Fig. 11 Minimal separators AS and BS.

② AB 間の障壁を形成する個々の既配線Wについて極小 AB セパレータに含まれるセルの数を数え，$N(W)$ とする．もし，Wの端子が極小セパレータによって分離される二つの領域に分かれて存在するならば $L(W)=1$，そうでないならば $L(W)=0$ とする．つまり，引きはがした既配線Wを再配線する場合に極小セパレータを通過しなければならない場合に $L(W)=1$，そうでない場合に $L(W)=0$ とする．

③ Wについての評価指標 $P(W)$，ただし

$$P(W)=N(W)-L(W).$$

を求める．この時，$P(W)\geqq 1$ でない限り，Wを引きはがした場合に AB 間の結線，及び，Wの再配線が成功する可能性はない．

④オペレータは，各既配線の評価指標を考慮して，引きはがす既配線の決定と，結線のための径路の概略をディスプレイ上で指定する．なお，評価指標は，引きはがす既配線が2本以上の場合も同様にして求めることができる．したがって，1本の引きはがしにより結線が成功しない場合には，2本以上を引きはがして同様な手順で処理を行うことも可能である．

* A, B をグリッド上の隣接していないセルとする．この時，あるセル集合 S があり，グリッド上から S をとりのぞくと A, B 間に径路が存在しなくなる時，このセル集合を AB セパレータと呼ぶ．さらに，AB セパレータのいかなる真部分集合も AB セパレータとならない時，このような集合を極小 AB セパレータと呼ぶ．

7. おわりに

ウェーブフロント割当型のハードウェアルータの試作を行った．この装置では，$N \times N$ のグリッドに対して $O(N)$ 台のプロセッサを使用し径路長に比例する時間で径路を発見できる．この方式では，従来より発表されている N^2 台のプロセッサを必要とする1セル1プロセッサ型とくらべて大幅にハードウェア量を減らすことができプロセッサの平均稼動率も高い．それにもかかわらず速度は 1/2 から 1/3 に低下するだけである．

さらにこのウェーブフロント割当型の並列ルーティングプロセッサが配線処理工程，特に障壁を形成している既配線の発見，除去及び再配線を必要とする段階において有効であることを示した．

この方式で速度を低下させずに扱えるグリッドの大きさを大きくするためには，プロセッサの台数を増やす必要があるが，汎用のマイクロプロセッサを使用し個別部品で組み立てるのでは実装密度が低く実用規模の装置を製作するのは無理である．したがって，実用規模の装置を製作するためには，カスタム LSI によりプロセッサを構成し，これに汎用のメモリを組み合せる，といった方法をとり実装密度を上げる必要があると思われる．

参考文献

1) Iosupovicz, A.: Design of an Iterative Array Maze Router, *Proc. of ICCC*, pp. 908-911 (1980).
2) Breuer, M. A. and Shamsa, K.: A Hardware Router, *Digital Systems*, Vol. 4, pp. 393-408 (1981).
3) Blank, T., Stefik, M. and vanCleemput, W.: A Parallel Bit Map Architecture for DA Algorithms, *Proc. of 18th DA Conf.*, pp. 837-845 (1981).
4) Kondo, T., Nakashima, T., Aoki, M. and Sudo, T.: An LSI Adaptive Array Processor, *IEEE JSSC*, Vol. SC-18, No. 2, pp. 147-156 (1983).
5) Hong, S. J., Nair, R. and Shapiro, E.: A Physical Design Machine, *VLSI 81*, pp. 257-266 (1981).
6) Sequin, C.: Doubly Twisted Network for VLSI Processor Arrays, *Proc. of 18th Ann. Symp. on Computer Architecture*, pp. 471-480 (1981).
7) Lee, C. Y.: An Algorithm for Path Connection and Its Applications, *IRE Trans.*, Vol. EC-10, pp. 356-365 (1961).
8) Soukup, J.: Fast Maze Router, *Proc. of 15th DA Conf.*, pp. 100-102 (1978).
9) Hightower, D. A.: A Solution to Line Routing Problems on the Continuous Plane, *Proc. 6th DA Workshop*, pp. 1-24 (1969).
10) Hashimoto, A. and Stevens, J.: Wiring Routing by Optimizing Channel Assignment within Large Apertures, *Proc. 6th DA Workshop*, pp. 159-169 (1969).
11) 松田庸雄，高見澤一彦，藤田友之，後藤 敏：ダイナミック・ルータ，情報処理学会第25回（昭和57年後期）全国大会資料，pp. 1301-1302 (1982).
12) Mitsumoto, K., Mori, H., Fujita, T. and Goto, S.: AI Approach to VLSI Routing Problem, *ISCAS '84*, pp. 449-452 (1984).
13) 橘 昌良，中島 聡，大附辰夫：並列ルーティング・プロセッサの一構成法，信学技報，CAS 83-75, pp. 25-30 (1983).
14) Tachibana, M., Suzuki, K. and Ohtsuki, T.: A Hardware Engine Architecture for Interactive Routing Design, *ISCAS '85*, pp. 209-212 (1985).

（昭和60年8月19日受付）
（昭和61年3月20日採録）

橘 昌良（正会員）

昭和34年生．昭和56年早稲田大学理工学部電子通信学科卒業．昭和58年同大学大学院理工学研究科博士前期課程修了．昭和59年同大学理工学部助手，昭和61年同大学大学院理工学研究科博士後期課程修了．工学博士．昭和61年(株)東芝入社．マルチプロセッサシステム，マンマシンインタフェース，LSI のための CAD システム等に興味を持つ．IEEE，電子通信学会各会員．

鈴木 敬

昭和36年生．昭和61年早稲田大学大学院理工学研究科修士課程修了．同大学院博士課程在学．CAD，計算機アーキテクチャに興味を持つ．電子通信学会会員．

大賀　忠

昭和36年生．昭和59年早稲田大学理工学部電子通信学科卒業．同年，松下通信工業(株)入社．現在電波事業部において主として移動無線のアナログ部の設計・開発，アナログ LSI の設計に従事．

中島　聡

昭和35年生．昭和60年早稲田大学大学院理工学研究科電気工学修了．同年 NTT 入社．現在，武蔵野通研，NTT 通信網第一研究所基幹交換部処理方式研究室所属．グラフィックス，CAD，マンマシンインタフェースに興味を持つ．作成したソフトウェアで最も満足しているのは CANDY（アスキー，1984）．

大附　辰夫（正会員）

昭和15年生．昭和38年早稲田大学理工学部電気通信学科卒業．昭和40年同大学院修士課程修了．同年日本電気(株)入社．昭和55年同社退社．現在，早稲田大学理工学部電子通信学科教授．工学博士．電子回路の CAD 及びこれに関連した基礎研究に従事．昭和44年度電子通信学会論文賞，1974年 IEEE Guillmin-Cauer 賞各受賞．電子通信学会，IEEE 各会員．