

並列メモリスистームの一構成法

Design of a Parallel Memory System

白川 洋 充

H. SHIRAKAWA

立命館大学理工学部

Faculty of Science and Engineering
Ritsumeikan University

〔1〕 緒言

多量のデータを高速に処理する必要がある、一般には多次元の問題である画像処理や偏微分方程式の解法には汎用の計算機は適合せず、各種の並列処理機能を有する計算機あるいは専用のハードウェアが用いられている〔1〕。従来これらの問題は2次元にかぎられていたが、近年3次元の問題を直接解く必要が生じてきた。これらの応用領域の代表的なものとしては、コンピュータグラフィクス、3次元ソリッドモデリング(CAD/CAM)や3次元のNavier-Stokes方程式の解法を挙げることができる。このため、多次元問題に対処できる並列処理機能の研究が行われるようになった。複数のプロセッシング要素とメモリモジュールが存在し、それらがネットワークで結合されるSIMDタイプの並列計算機が画像処理や偏微分方程式の解法に通常使用されている。3次元のアレイをメモリモジュールに埋蔵できたとしても、希望する任意のデータを並列にアクセスするには多くの問題が残されている。2次元の問題に関しては、STARANはビットスライスおよびワードスライスのアクセスを可能にしている。このため2次元の問題の並列処理を可能にしている〔2〕。また、3次元の問題に関しては、任意の軸に関するスライスならびに水平、垂直に並んだスロットにアクセスすることが可能であることが報告されている〔3,4〕。しかしながら、画像処理や偏微分方程式の解法では処理の対象がある近傍のことが多い。このようなデータをアクセスするために従来はスロットあるいはウィンドタイプのアクセスが使用されていたが、これらは必ずしも近傍を表すメトリックと関連するようには取られたものでなく、ハードウェアの構成上の制約から決定されたものが多かったといえる。したがって、不必要なデータ

も同時にアクセスされるという欠点を有する。一方、最近プロセッサアレイを配列する方法として注目されているものにシストリックアレイがあるが、これも1次元か2次元のアレイを基本にしている。3次元の問題をシストリックアレイで解く場合、メモリから必要なデータを一度にアクセスできないためキャッシュメモリを使うなどの工夫が必要である〔5〕。

本論文は、あるメトリックで表される近傍のデータを並列にアクセスできるメモリスистームの構成法について述べるものである。メトリックとしては、各次元における距離の和を用いる街区ノルムと、各次元における距離の平方の和の平方根を使用するユークリッドノルムを考える。メモリスистームを実現するために必要なスキューニングスキームとアドレッシング回路について説明し、最後に実際のメモリスистームの構成方法について述べる。

〔2〕 問題の記述

$M \times M \times M$ の3次元空間を考える。3次元空間の各点を単位立方体で表すことにする。この各立方体をインデクス (i, j, k) で表す。 i, j, k は、

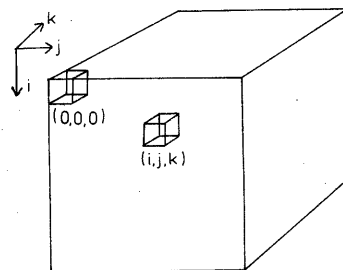


Fig.1 Three Dimensional Coordinate System.

それぞれ、0 と M の間の値をとる。M は必ずしも 2 のべきで表されている必要はない。インデックスのとり方を図 1 に示しておく。この座標系のとり方は行列を表現する従来の約束に従っている。

並列メモリとは、N 個の立方体から構成される形状を少なくとも N 個のメモリモジュールから並列にアクセスできるものをいう。通常、形状としては行、列とファイルに関するスライスおよび水平、垂直に並んだスロットなどが考えられており、これらの形状を構成するデータを並列にアクセスする機構はすでに報告されている。これらの形状の例を図 2 に示しておく。行

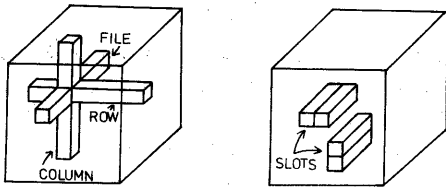


Fig.2 Various Kinds of Modes to Access Memory.

に関するスライスはワードスライス、列に関するスライスはビットスライスと通常呼ばれており、以後この用語を使用する。次に、本論文で考察の対象となっている工学的に見て重要な次の二つの形状について考える。

(1) ある正数 r と任意の点 (i, j, k) の近傍で、街区ノルム (city block norm) [6] が

$$|i - i_0| + |j - j_0| + |k - k_0| \leq r$$

のような関係式を満足する形状。

(2) ある正数 r と任意の点 (i, j, k) の近傍で、ユークリッドノルムの二乗が

$$(i - i_0)^2 + (j - j_0)^2 + (k - k_0)^2 \leq r^2$$

のような関係式を満足する形状。

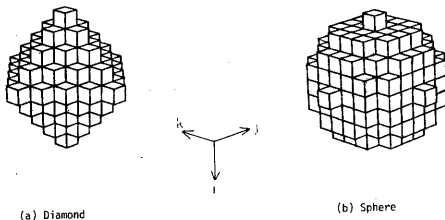


Fig.3 Two Neighbours, (a)Diamond, (b)Sphere.

$r=4$ の場合、(1) と (2) の形状をそれぞれ図 3 に示しておく。(1) と (2) を便宜上、それぞれ、ダイヤモンド、球と呼ぶことにする。問題は (1) と (2) の形状を直接アクセスできるメモリシステムの構成方法を見出すことである。

(3) スキューニングスキーム

並列メモリを構成する際に最も重要な概念である形状とメモリモジュール割当て数との間の対応関係を与えるメモリのスキューニングスキーム (skewing scheme) について考えることにする。いま考えている形状を構成する要素をインデックス (i, j, k) で表し、この全体を Ω で表す。この時、一般にスキューニングスキームは、次に示すような 1 対 1 ではあるが、 N の中への写像 S と定義される。

$$S: \Omega \rightarrow N = \{0, 1, 2, \dots, N\}$$

ここで、 N はメモリのモジュール数である。上の定義より、一般に $S(\Omega) \subseteq N$ が成立する。

実際のスキューニングスキームを考える前にスキューニングスキームが満足しなければならない条件について考えることにする。まず、 Ω の要素の数を $|\Omega|$ とすると、 S は N の上への写像でなくてよいことから $|\Omega| \leq N$ であればよい。次に任意の (i, j, k) に対して S の性質が保存されることが必要である。

このような性質を有するスキューニングスキームとしては次のようなものが考えられる。

$0 < a, b, c < N$ を満足するある整数 a, b, c に対して $S = (ai + bj + ck) \bmod N$ とおく。ここで、 a, b, c は 1 であるか、または N を割切らない数とする。(1) と (2) で与えた形状に対するスキューニングスキームは、上に与えた条件を満足するものの中で最小の N を求めればよい。

(例 1)

$r=1$ の場合 (1) と (2) のいずれも同じ形状になる。この形状は原点の 6-近傍として知られており、3次元のデジタル画像のトポロジカルな性質を議論するのに使用されている。この形状に対しては $S = (5i + 4j + k) \bmod 7$ なるスキューニングスキームが適用できる。図 4 にこのスキューニングスキームによって得られるメモリモジュールの割当てのすべてを示す。容易に知られるように、

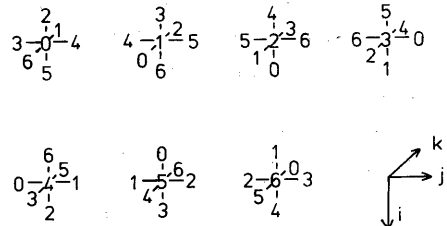


Fig.4 Skewing Scheme for 6-Neighbors.

Table 1 Skewing Scheme for Diamonds and Spheres.

r	Diamonds								Spheres							
	Skewing Scheme				q	Memory Utilization %	Space Occupancy %	N	Skewing Scheme				q	Memory Utilization %	Space Occupancy %	N
	a	b	c	N					a	b	c	N				
1	5	4	1	7	7	100	26		5	4	1	7	7	100	26	
2	10	4	1	27	25	93	22		16	11	1	35	33	94	28	
3	25	16	1	70	63	90	20		49	31	1	152	123	81	44	
4	61	9	1	145	129	89	19		181	21	1	311	257	83	43	

このスキューイングスキームは1対1で、かつ N の上への写像である。

さて、3次元のダイヤモンドおよび球に対するスキューイングスキームについて考察することにしよう。例と同様、ダイヤモンドおよび球に対するスキューイングスキームを一般的に見いだすアルゴリズムは存在しない。したがって、スキューイングスキームの係数 a, b, c とメモリモジュール数は計算機で試行錯誤的に求めた。表1に計算機によって得られた結果をまとめておく。表1において、メモリ利用率 (Memory Utilization) は $100 |q| / N \%$ で定義する。この値は、 $r \leq 4$ において、ダイヤモンド および球とも81%を越えない。また、空間占有率 (Space Occupancy) は $100N / (2r+1)^3 \%$ で定義すると、もしも立方体で構成することと比較すると、44%以内で構成できることを示している。これらのデータを分析すると、メモリ利用率が100%でないということは実現上の障害にはならないと考えられる。

〔4〕 アドレッシングスキーム

提案されたスキューイングスキームを $S = (ai + bj + ck) \bmod N$ とする。このとき、これは次の性質を有する。すなわち、S がスキューイングスキームであるとき、a, b, c は1であるか、あるいは N を割切らないかのどちらかである。すると、任意の位置からとられる長さ N のビットスライス、ワードスライス およびファイルに関するスライスのアクセスが可能である。これは容易に検証できる。

アドレッシングスキームの構成には任意の三方向のスライスのアクセスが可能になった事実を使う。議論を簡単にするために3次元空間には次の仮定をもうける。すなわち、インデックス j に関する M の値が N の倍数になるようにする。アドレスのとりかたの一つとしては、まずワードスライスを $(i, j, k) = (0, 0, 0)$ から j を最優先に、次に i、最後に k と重ならないようにとっていき順序づけられたワードスライスを構成する。各ワードスライスの要素に同じアドレスを与え、アドレスは上にのべた順序で各ワードスライスに割当ていく。図5にアドレスの割

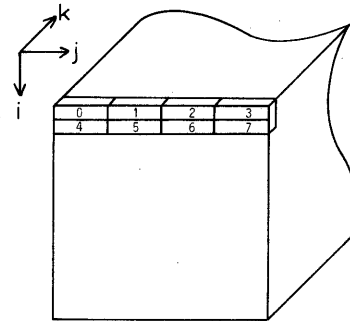


Fig.5 Illustration of the Addressing Scheme.

当の例を示しておく。

さて、 $0 \leq i, j, k \leq M-1$ の任意のインデックス (i, j, k) に対するアドレスを $\text{Address}(i, j, k)$ とすると、

$$\text{Address}(i, j, k) = (Mi) / M + j / N + kM^2$$

ここで、/ は整数の除算における商を表す。

メモリモジュール割当て数はスキューイングスキームによって得られる数に対応している。ところが、形状の置かれた位置によって形状内の特定の位置に対するメモリモジュール割当て数はすべて異なる。これは、メモリモジュール側にしても同様で、このメモリモジュールに割当てられた形状の位置、すなわち、インデックスが求められないと上に定義したアドレスが生成できない。メモリモジュール内でインデックスを逆に求め実際のアドレスを生成するものがアドレッシング回路 (Addressing Circuitries) である。

アドレッシング回路は次のように構成する。各メモリモジュールにそれぞれアドレッシング回路を設ける。この回路には共通に形状の中心のインデックス (i, j, k) がブロードキャストされる。各回路はこのインデックスをもとに、中心に対するメモリモジュール割当て数を計算する。各回路には各々、中心のメモリモジュール割当て数に対する相対的なインデックスの変位をテーブルに記憶しておく。この値を $(\Delta i, \Delta j, \Delta k)$ とすると、このメモリモジュールに割当てられたインデックスは $(i + \Delta i, j + \Delta j, k + \Delta k)$ と得られ、アドレッシング回路はこれをもとにアドレスを生

成できる。

スライスの場合には形状の中心に相当するものが存在しないので、スライスの最下位ビットに対応するインデックスを各アドレッシング回路にブロードキャストすればよい。また、各回路にはスライスの最下位ビットに対するメモリモジュール割当て数に対する相対的なインデックスの変位をテーブルに記憶しておき、それをもとにして、各メモリモジュールは自分自身のインデックスを知ることができる。

並列メモリの特徴は、各メモリモジュールにそれぞれ異なったアドレスを与えるものであるが、次に述べる場合は各メモリモジュールに同一のアドレスを供給できる。これは、アドレス生成の過程からも明らかなように3次元空間からテレビジョンのラスタスキャンに準ずるデータを順番にアクセスする場合である。並列メモリは、特に画像処理に应用することを考慮して設計されたものである。ここではテレビジョンカメラ、グラフィックスディスプレイあるいは周辺入出力機器とのデータ転送がよく使われるが、この場合には上にのべたようなアドレスの生成は不要で、簡単なカウンタでアドレッシングが可能でありさらに高速化が期待できる。

〔5〕データ交換装置

スキューイングスキームを実現するものがデータ交換装置である。これはメモリモジュールに前置し、必要なデータの置換を行う。さて、提案された並列メモリを構成するために必要なメモリモジュール数は必ずしも2のべきにはなっていないことに注意した。したがって、通常使用されている入出力数が2のべきで与えられ、 $\log N$ ステージのシャフルエクスチェンジ (shuffle exchange) のデータ交換装置は次の理由で使用できない。まず、入出力数が一致しておらず、すなわち、 $1 \leq \Omega \leq N$ かつ Ω と N は2のべきでない。次に、シャフルエクスチェンジはすべての置換を生成できないことは知られて

いるが、特に、ここで問題となっているスキューイングスキームを実現するための置換は生成できないことである。このため、専用の $N \times N$ のデータ交換を考えることにする。入出力数は、同じ N にする。これは、可能な限りハードウェアが対称になるとともに同一のサブモジュールに分解されることを考慮したからである。

5.1 基数シャフルネットワーク

N が2のべきで表されていないが基数表現で表されているときデータ交換装置としては、基数シャフルネットワーク (radix shuffle network, 以後、RSNと呼ぶ) が適していることが最近 L.N. BhuyanとD.P.Agrawalによって発表された〔7〕。RSNはネットワークを構成するサブネットワーク自身でルートを決めることができるが、シャフルエクスチェンジネットワークと同様、すべての置換を生成できないことが欠点とされている。これから議論することは、基数シャフルネットワークがスキューイングスキームを実現する置換は生成できることを証明すること、また、サブネットワークにおけるルート決定に必要な制御が単にシフト機能であることを証明することである。

さて、メモリのモジュール数 N が次のように表されているとする。

$$N = n_1 \times n_2 \times \dots \times n_r$$

すると、0と N の間の数 x は $(x_{r-1}, x_{r-2}, \dots, x_0)$, $0 \leq x_i \leq n_{r-i} - 1$ のような r 個の組で表すことができる。ここで、 x_0 は最下位の数 (digit) を表し、 x_{r-1} は最上位の数を表す。また、 n_i は基数である。 x を混合基数で表すとき、基数を示す必要がある場合、 $(x)_{n_1, n_2, \dots, n_r}$ と表す。次に、データ交換の基礎となる基数シャフルの定義を行う。

基数シャフルとは、 $(x_{r-1}, x_{r-2}, \dots, x_0)_{n_1, n_2, \dots, n_r}$ で表される入力を $(x_{r-2}, \dots, x_0, x_{r-1})_{n_2, n_3, \dots, n_r, n_1}$ で表される出力に対応させる写像である。

RSNは r -ステージの結合ネットワークで、 i -

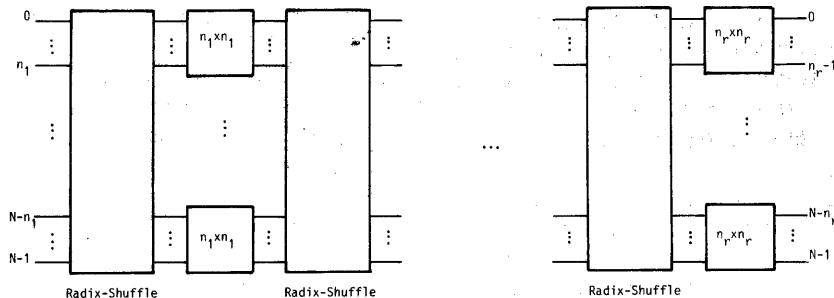


Fig. 6 A $N \times N$ Radix Shuffle Network (L.N.Bhuyan and D.P.Agrawal〔7〕)

ステージはサイズが (n_i, xn_i) のクロスバースイッチが (N/n_i) 個から成り、それらの入力側に基数シャフルが存在するものをいう。図6に $N \times N$ のRSNを示しておく。RSNの入力と出力に番号を付け、それぞれ、 x, y で表す。すなわち、

$$x = (x_{r-1}, x_{r-2}, \dots, x_0) n_1 n_2 \dots n_r$$

$$y = (y_{r-1}, y_{r-2}, \dots, y_0) n_1 n_2 \dots n_r$$

r -ステージにおけるサブネットワークの入力ならびに出力は基数シャフルのため、上のように表すことができないことに注意する。RSNでは、 i -ステージにおける基数シャフルは $(i-1)$ -ステージのクロスバースイッチの出力側で、 $(x_{r-1}, x_{r-2}, \dots, x_0) n_i \dots n_{i-1}$ と表されているものを入力として、これを $(x_{r-2}, \dots, x_0, x_{r-1}) n_{i+1} \dots n_i$ で表される出力に対応させているので、 i -ステージにおけるクロスバースイッチの入力は $(x) n_{i+1} \dots n_i$ と表す必要がある。

スキューイングスキームはRSNを使って次のように実現することができる。まず、ある基準の位置、例えば原点に形状の中心がある場合を考える。これを基本形状とする。この形状の各要素を、スキューイングスキームによつて割当てられる数と同じ番号を有する入力に対応づける。RSNは恒等写像を実現できるので、まず、この基本形状に関してメモリモジュールとの結合がなされる。次に、形状の中心が基準の位置にない場合は、容易に知られるように、中心のインデックスを (i, j, k) とすると、全体の割当てを $\text{mod } N$ の $(ai+bj+ck)$ だけシフトする。したがって、RSNは基準の位置からの変位に相当するシフトを実現すればよいことになる。

スライスの場合も形状の場合と同様の議論で実現問題を考えることができる。ただし、スライスのビットの並びは順序づけられていることが望ましいので、これを修正する必要がある。スライスの最下位ビットが、RSNの最下位の入力番号に対応し、すべてが順序づけられて並ぶためには、ビットスライスの場合、 $ax+(ai+bj+ck) \text{ mod } N$ の写像、ワードスライスの場合には $bx+(ai+bj+ck) \text{ mod } N$ の写像、またファイルに関するスライスの場合には $cx+(ai+bj+ck) \text{ mod } N$ の写像がRSNで実現できればよいことが知られる。ここで、 (i, j, k) は考えているスライスの最下位ビット位置を表すインデックスである。

以上の議論から、スキューイングスキームがRSNによって実現されるには、RSNが $(mx+s) \text{ mod } N$ なる写像を実現することが必要であり、かつ十分である。ここで、 m は、0または a, b あるいは c の値をとる数で、 s ($0 \leq s \leq N-1$) はシフト数である。

5.2 スキューイングスキームの実現

まず、 $\text{mod } N$ のシフトをRSNで実現することを考える。

シフト s を $(s_{r-1}, s_{r-2}, \dots, s_0) n_1 n_2 \dots n_r$ で表すことにする。0-ステージにおいて、

(n_1, xn_1) クロスバースイッチの入力番号が

$$(x_{r-1}, x_{r-2}, \dots, x_1, 0) n_2 n_3 \dots n_r n_1$$

$$\sim (x_{r-1}, x_{r-2}, \dots, x_1, n_1-1) n_2 n_3 \dots n_r n_1$$

で表されているものを考える。これは、下位から $(x_{r-1}, x_{r-2}, \dots, x_1, 0)/n_1$ 番目の (n_1, xn_1) クロスバースイッチのすべての入力を表している。これらはRSNの入力の $(0, x_{r-1}, x_{r-2}, \dots, x_1) n_1 n_2 \dots n_r$ から $(n_1-1, x_{r-1}, x_{r-2}, \dots, x_1) n_1 n_2 \dots n_r$ 番目にそれぞれつながれている。さて、これらをインデックス $p, 0 \leq p \leq n_1-1$ を使って $(p, x_{r-1}, x_{r-2}, \dots, x_1) n_1 n_2 \dots n_r$ と表すことにする。このように表される入力はシフト s によってRSNの出力側では $(y_{r-1}, y_{r-2}, \dots, y_0) n_1 n_2 \dots n_r$ の位置に移される。

一方、0-ステージにおける (n_1, xn_1) クロスバースイッチの制御は y_{r-1} なるタグによってのみ決定される〔7〕。実際に y_{r-1} を求めてみると、 $y_{r-1} = \{p+s'\} \text{ mod } n_1, p=0, \dots, n_1-1$ ここで、

$$s' = [\{ (x) n_2 \dots n_r n_1 \text{ mod } (N/n_1) \} + s] / (N/n_1)$$

したがって、0-ステージにおいては、 (n_1, xn_1) クロスバースイッチは、単に $\text{mod } n_1$ のシフト s' を行えばよいことが得られる。また、 s' は $(x) n_2 \dots n_r n_1 \text{ mod } N/n_1$ に関係することを考えると、 (N/n_1) の (n_1, xn_1) クロスバースイッチですべてシフト数が異なる可能性があることを示している。

同様に、1-ステージにおいては

$$(x_{r-1}, x_{r-2}, \dots, x_{i+1}, 0, \dots, 0) n_{i+2} \dots n_i \dots n_{i+1}$$

$$\sim (x_{r-1}, x_{r-2}, \dots, x_{i+1}, n_i-1, \dots, n_{i+1}-1) n_{i+2} \dots n_i \dots n_{i+1}$$

$n_1, \dots, n_{i+1}$ で表される $n_1 n_2 \dots n_{i+1}$ 個の (n_{i+1}, xn_{i+1}) クロスバースイッチに対し、一様に $\text{mod } n_{i+1}$ のシフト s^{i+1} を施せばよいことが知られる。ここで、

$$s^{i+1} = [\{ (x) n_{i+2} \dots n_{i+1} \text{ mod } (N/n_1 n_2 \dots n_{i+1}) \} + \{ s \text{ mod } (N/n_1 n_2 \dots n_i) \}] \text{ mod } (N/n_1 \dots n_{i+1})$$

最後に、 r -ステージにおいては、すべての (n_r, xn_r) クロスバースイッチに対し、一様に $\text{mod } n_r$ のシフト s_0 を施せばよい。

以上が、RSNはスキューイングスキームに必要な写像を実現し、しかもサブネットワークのルート決定の制御はシフト命令であることの構成的な証明である。スライスのアクセスに必要な $\text{mod } N$ のシフト $(mx+s)$ に関しても (n_i, xn_i) クロスバースイッチの入力のインデックス p を $mp \text{ mod } n$

に変更すれば同様な結果が得られる。

〔6〕 並列メモリシステム

本節では $r=2$ のダイヤモンドを例にとって、実際のメモリシステムを構成する。図7は $r=2$ の

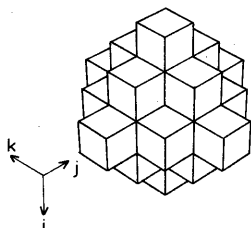


Fig.7 Illustration of a Diamond.

ダイヤモンドの形状を示している。この形状のスキューニングスキームとしては $S=(4i+10j+k) \bmod 27$ が得られている。図8に、このスキューニングスキームによって得られるメモリモジュールの割当て数を示しておく。

スキューニングスキームよりメモリモジュール数は27で、ダイヤモンドの要素の数は25であるが、データ交換装置は 27×27 で設計する。まずシステム全体のブロック図を図9に示しておく。図において、PEは種々の演算を行うプロセッシング要素である。DPUはスキューニングスキームを実現するためのデータ交換装置である。さらに、Mはメモリモジュール、Aはアドレッシング回路である。

メモリのモジュール数は27であり、また $27=3 \times 3 \times 3$ であることから、RSNは全ステージとも対称に構成できる。図10に RSNの回路を示す。図において参考のために $\bmod 27$ のシフト4のパターンを示しておく。前述のごとく、各ステージの 3×3 クロスバースイッチは、単にシフトを実

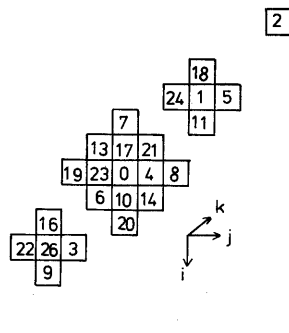


Fig.8 Skewing Scheme for a Diamond.

現すればよいので、クロスバースイッチは必要なく簡単なロジックで構成可能である。なお、0-ステージにおけるシフト数は $\bmod 3$ の $(s_2 + (3x_2 + x_1 + 3s_1 + s_0)/9)$ 、1-ステージにおけるシフト数は $\bmod 3$ の $(3s_1 + (x_2 + s_0)/3)$ 、2-ステージにおけるシフト数は $\bmod 3$ の s_0 である。また、シフト制御が一様になるクロスバースイッチのグループが図11に示されている。したがって、合計13種類の制御信号が必要で、これらは図9に示したごとく、 (a, b, c, i, j, k, N) の組から求めることができる。

ビットスライス、ワードスライスおよびファイルに関するスライスの DPUの制御は次のように行う。ビットスライスに関しては、写像を $4x + (4i + 10j + k) \bmod 27$ で与えればよいが、この場合は都合の良いことに、 $4p \bmod 3 = p \bmod 3$ がすべての $p, 0 \leq p \leq 2$ に対して成立しているのので、前述のごとく、スライスのシフトを実現するには、 3×3 のクロスバースイッチの入力のインデックス p を mp にすればよかったが、この場合 $m=4$ であるので、インデックスは p そのものを使え

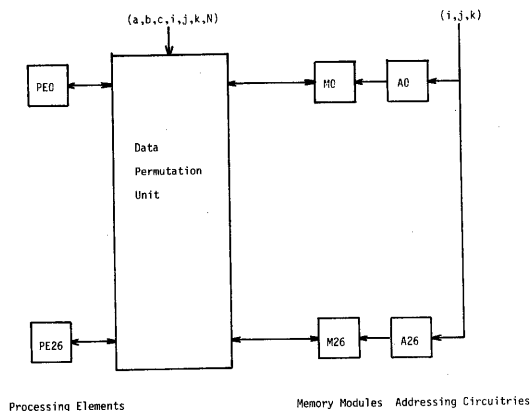


Fig.9 Proposed Parallel Memory System.

ばよい。したがって、各ステージのシフト数は0-ステージでは $s_2 + x_2 + (3x_2 + 4x_1 + 3s_1 + s_0)/9$, 1-ステージでは $3s_1 + x_2 + (x_2 + s_0)/3$, 2-ステージではシフト数は x に関係せず s である。ワードスライスにおいても $10p \bmod 3 = p \bmod 3$ がすべての p , $0 \leq p \leq 2$ に対して成立しており、ファイルに関するスライスも $c=1$ であるのでビットスライスと同様、クロスバースイッチのインデクスを変更することなく制御できる。

〔7〕 結論

データ量が膨大でかつ多次元の問題である画像処理ならびに偏微分方程式の解法を高速に行うための並列処理機構に適する並列メモリの構成について考察した。本メモリシステムの特徴は次のとおりである。(1) 近傍のデータを並列にアクセスできる。(2) ビットスライス、ワードスライス、ファイルに関するスライスのアクセスが可能である。(3) メモリの構成上必要となるスキューイングスキームとアドレッシング回路は現在の技術で充分ハードウェア化できる。(4) 提案されたメモリシステムと周辺機器間のデータ転送ならびに、画像入力、ディスプレイ出力はアドレッシング回路が簡単になり、近傍をアクセスするよりさらに高速に転送できる。

実際のメモリシステムは現在はまだ試作されていない。実際のシステム構成ならびにシステム評価は今後の課題である。

〔8〕 参考文献

- (1) Hochney, R. W. and Jesshope, C. R., "Parallel Computers", Adam Hilger Ltd (1981).
- (2) Batcher, K. E., "The Multidimensional Access Memory in STARAN", IEEE Trans. on Comput., Vol. C-26, No.2, pp. 174-177, (1977).
- (3) Shirakawa, H. and Kumagai, T., "An Organization of a Three-Dimensional Access Memory", 1980 Int'l. Conf. on Parallel Processing, pp. 137-138.
- (4) Shirakawa, H. and Kumagai, T., "Structure of a Three-Dimensional Access Memory System", Memoirs of the Research Institute of Science and Engineering, Ritsumeikan University, No. 41, pp. 27-50, (1983).

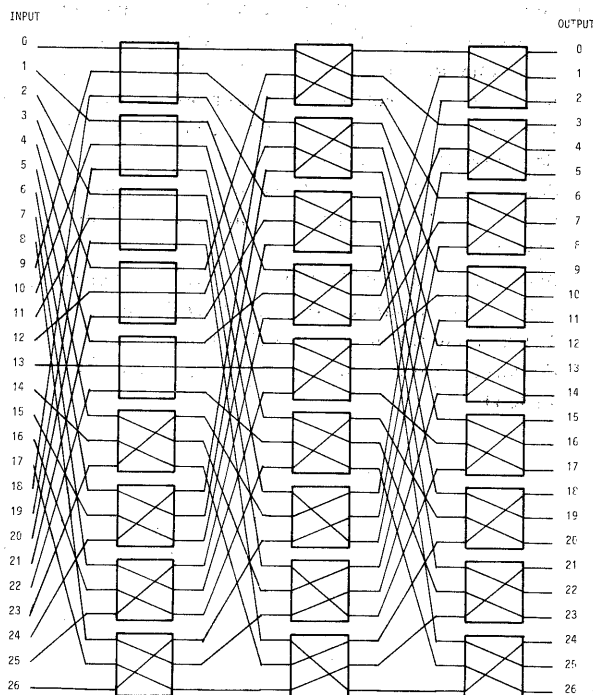


Fig.10 A Three-Stage Radix Shuffle Network.

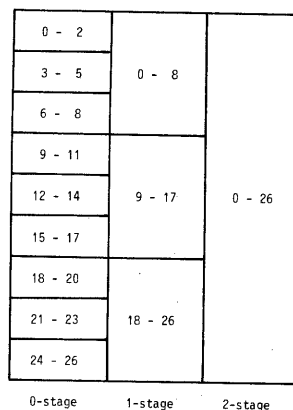


Fig.11 Groups of the Crossbar Switches with the Same Routing Control Signal.

- (5) Kung, H. T. and Picard, R. L., "Hardware Pipelines for Multi-Dimensional Convolution and Resampling". 1981 IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Database Management, pp. 273-278.
- (6) Rosenfeld, A., and Kak, A. C., "Digital Picture Processing", Academic Press. (1976).
- (7) Bhuyan, L. N. and Agrawal, D. P., "Design and Performance of General Class of Interconnection Networks". IEEE Trans. on Comput., Vol. C-32, No. 12, pp. 1081-1090, (1983).