試作BCプロセッサアレイとその評価†

小畑正貴†† 宮垣嘉也†† 金田悠紀夫†††

筆者らは以前，隣接プロセッサへの結合端子とバスによる放送端子とを合わせ持つプロセッサ（BC プロセッサ）を提案し，BC プロセッサから構成されるプロセッサアレイが行列の乗算や連立一次方程式の計算を効率よく実行できることを示した．またマイクロプロセッサを用いた1次元 BC プロセッサアレイを設計し，画像処理やデータベース処理へも応用できることを示した．本論文では1次元 BC プロセッサアレイの試作機とその性能評価について論じている．試作機は256台のプロセッサを持ち，CPU には Z8 を使用している．まず試作機の概要とソフトウェアについて述べ，実行例では本システムが行列乗算を $O(n^3)$ で実行できることを示す．また行列乗算における処理時間の内訳を測定し，ブロードキャスト転送の効果やシステムの拡張性について評価を加える．

1. はじめに

近年における LSI 技術の発達に伴い，単純な構成のプロセッサを規則的に配列した高並列計算機システムが注目されるようになっている．代表的な例として Kung らによって提案されたシストリックアレイがあげられる[1]．これは単純で規則的なデータの流れを受け，それにパイプライン制御されたシストリックアルゴリズムを適用することにより目的の計算を高並列に実行することができる．われわれは文献 2) において，隣接プロセッサとの結合端子に加えてバスによるデータの放送端子を持つ演算プロセッサ（BC プロセッサ）を提案し，BC プロセッサから構成される1次元または2次元のプロセッサアレイが行列乗算や連立一次方程式の計算を約 n ステップで実行できることを示した．また文献 3) ではマイクロプロセッサを用いた1次元 BC プロセッサアレイを設計し，データベース処理や画像処理への応用の可能性も示した．

本論文では1次元 BC プロセッサアレイの試作機の概要とソフトウェアについて述べ，実行例として行列乗算の実行時間を示す．また行列乗算の実行における動作解析をもとに BC プロセッサアレイの有効性や拡張性について評価を加える．

† Experimental BC Processor Array System and Its Evaluation by MASAKI KOHATA, YOSHIYA MIYAGAKI (Department of Electronic Engineering, Faculty of Engineering, Okayama University of Science) and YUKIO KANEDA (Department of Systems Engineering, Faculty of Engineering, Kobe University).

†† 岡山理科大学工学部電子工学科
††† 神戸大学工学部システム工学科

2. 試作機のハードウェア

2.1 ハードウェア構成

図1に本システムの全体構成を示す．ホストコンピュータにはパーソナルコンピュータ（PC 9801）を用い，これに256台の BC プロセッサが接続される．各 BC プロセッサは二つの入力ポート（LI と BI），および二つの出力ポート（RO と BO）を持つ．LI (Left Input) ポートは左に隣接するプロセッサからデータを入力し，RO (Right Output) ポートは右に隣接するプロセッサにデータを出力する．

BI ポートはホストコンピュータの出力データバス（Oバス）に接続され，ホストからのデータを入力する．また，BO ポートはホストコンピュータの入力データバス（Iバス）に接続され，ホストに対してデータを出力する．ホストコンピュータは PE アドレスによって特定の BC プロセッサを指定する．また，データの放送（ブロードキャスト）時には全プロセッサを同時に指定する．

各 BC プロセッサの CPU には8ビットマイクロプロセッサ Z8（クロック 8MHz）を用いている．メモリとしては 256 バイトの ROM と，2k バイトの RAM を備えている．BI, LI および RO はそれぞれ8ビット並列ポートで，ハンドシェイクのための信号を備えているが，Z8 の内蔵する入出力ポート数の関係から BO ポートだけは直列ポートになっている．直列出力には，内蔵の直列出力回路を利用した非同期通信（62.5k ボー）による方法と，1ビットごとにホストと同期をとりながらソフトウェアにより行う方法がある．後者はブロードキャストモード（後述）におけるデータ転送に用いる．

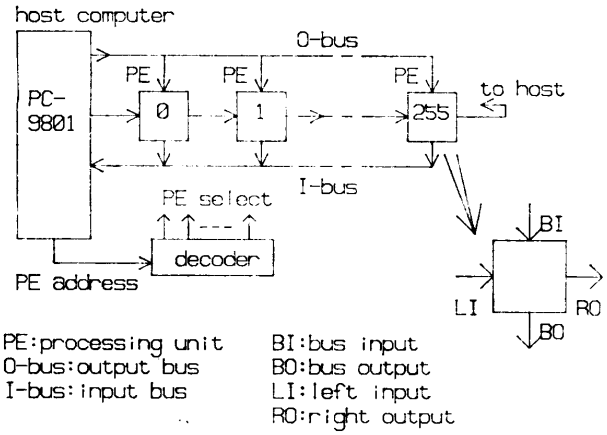


図 1 BC プロセッサアレイシステムの構成
Fig. 1 Structure of the BC processor array system.

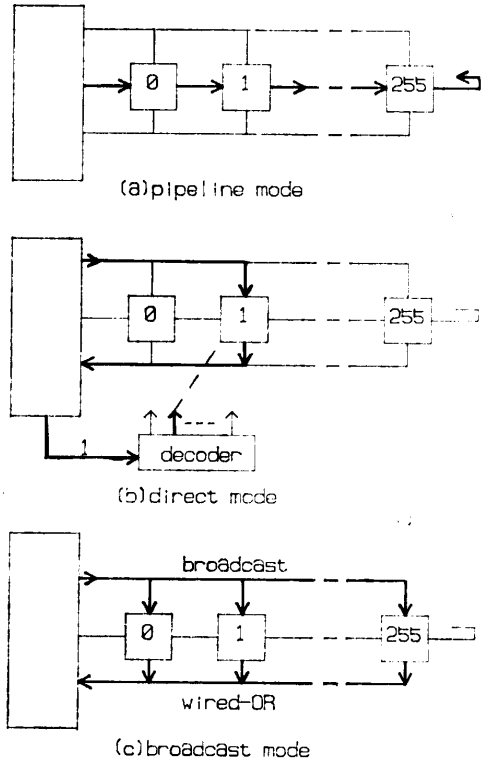


図 2 データ転送モード
Fig. 2 Data transfer modes.

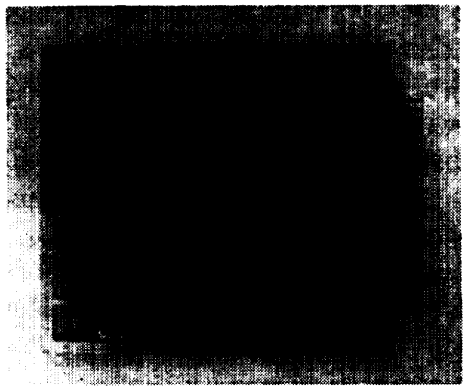


図 3 BC プロセッサ基板
Fig. 3 BC processor board.

2.2 基本動作

本システムには以下に述べる3種のデータ転送モードがあるが，いずれもハンドシェイクによる1バイトずつのデータ転送が基本となる．

(1) パイプラインモード (図 2(a))

各プロセッサは右にデータを出力し，左からデータを入力する．プロセッサアレイ内のデータは全体として右にシフトされる．左端のプロセッサはホストからデータを受け取り，右端のプロセッサはホストにデータを送る．プロセッサ間での同期はハンドシェイク信号によってとられる．

(2) ダイレクトモード (図 2(b))

ホストコンピュータは PE アドレスによって特定のプロセッサを指定し，そのプロセッサに対してバスを通してデータの転送を行う．

(3) ブロードキャストモード (図 2(c))

ホストコンピュータからのデータはOバスを通して全プロセッサに同時に転送される (放送)．ホストからのデータの送出は，全プロセッサの入力レジスタが空の時に可能となる．一方，このモードにおいてプロセッサアレイからホストにデータを転送する場合には，全プロセッサは同時にIバスに出力する．これらのデータはバス上でワイヤードオアがとられ，ホストはこれを読み込む．

2.3 ハードウェアの実装

BC プロセッサ1台を 8 cm×9 cm のプリント基板1枚に実装した (図 3)．ここで RAM と ROM は2階建てに実装される．この BC プロセッサ基板は 40 ピン端子によってマザーボードに接続される．ソケットやコネクタにおける接触不良の問題を避けるため，IC の実装やプロセッサ基板とマザーボードの接続はすべて直接のはんだ付けにより行った．

図 4 に本システムの全景を示す．1枚のマザーボードに 32 台の BC プロセッサを載せ，これを8枚連結することにより，256 台のシステムを実現した．また共通バスに対してはマザーボード1枚ごとにバスバッファを入れ，クロックは全プロセッサ共通でホストコンピュータから供給している．総消費電力は約 400 W

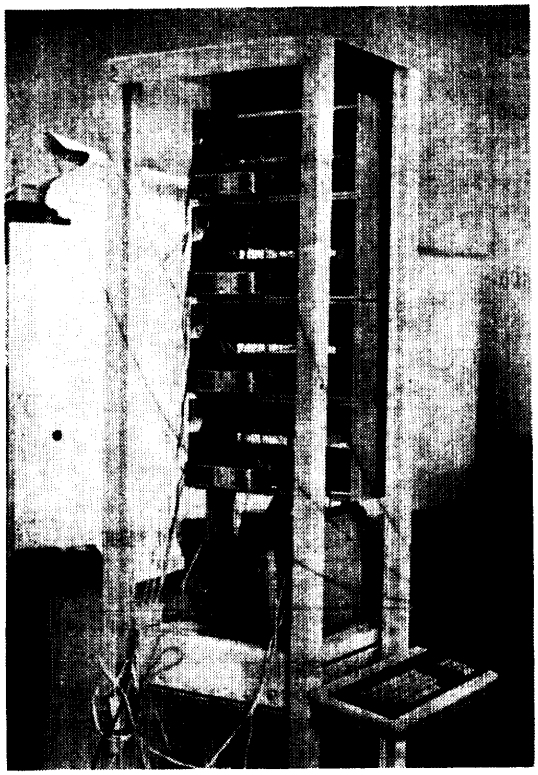


図4 システム全景
Fig. 4 Whole view of the system.

であり，電源は64台ごとに分散して実装した．

3. ソフトウェア

3.1 OSおよびモニタ

OSにはデジタルリサーチ社のCP/M-86を用いている．またプログラム言語に関しては，ホストコンピュータ用にはCコンパイラと8086アセンブラを使用し，BCプロセッサ用にはZ8クロスアセンブラを使用している．

また，ホストコンピュータからBCプロセッサを制御するため，ホストコンピュータ上とBCプロセッサ上には以下のようなモニタプログラムがある．

[ホストコンピュータ側]

BCプロセッサ側のモニタプログラムと通信してBCプロセッサの内部状態をモニタする．また，ホスト側の入出力ポートをアクセスしてプロセッサアレイ側の各ポートを外側からモニタする．ホスト側モニタの以下に機能を示す．

- BCプロセッサ内のメモリとレジスタの表示と変更
- BCプロセッサへの応用プログラムのロード
- BCプロセッサの起動と停止
- BI, BO, LI, RO に対するデータの入出力

[BCプロセッサ側]

BCプロセッサ側モニタはROM上に常駐している．ROM容量の制限から機能は必要最小限のものにとどめている．ホストから送られてくるコマンドとアドレスに従って次のような動作を行う．

- メモリおよびレジスタ内容のホストへの送出
- ホストからの受信データの格納
- 指定番地からの実行

3.2 応用プログラムの実行方法

本システムによってある問題を処理する場合，それに対するプログラムはBCプロセッサとホストコンピュータとに必要となる．ホストコンピュータのプログラムはプロセッサアレイへのデータの転送と結果の回収を行うものであり，BCプロセッサのプログラムはホストから送られてきたデータに対する計算処理を行うものである．BCプロセッサは1台ごとに違ったプログラムを実行することも可能ではあるが，通常は全プロセッサが同じプログラムを実行することを基本とする．この場合のプログラムの転送はブロードキャストモードで行われる．

ホストコンピュータはまずBCプロセッサ用プログラムを全プロセッサに転送し，全プロセッサに起動をかける．つぎに自分のプログラムをロードして実行する．各BCプロセッサは通常は入力待ちの状態にあり，ホストからデータが来るたびにこれを取り込んで計算を進める．

4. 行列乗算の実行

4.1 アルゴリズム

ここではサイズが n で帯幅が w の帯行列どうしの乗算問題を取り上げる（図5(a)）．この場合にはプロセッサアレイ中の w 台のプロセッサのみが計算に参加する．また，各プロセッサにはそれぞれ w 組の作業領域を用意しておく（図5(b)）．

$i=1\sim n$ に対して以下のサブステップを行うことにより結果が求められる．

(1) ブロードキャストモードにより，ホストコンピュータは行列 A の第 i 列の w 個の要素を全プロセッサに転送する．全プロセッサはこれらのデータを各自の作業領域に格納する．

(2) 次にダイレクトモードにより，行列 B の第 i 行の要素を w 台のプロセッサに一つずつ分配する．

(3) 全プロセッサは各作業領域内の中間結果に(1)(2)で転送されてきたデータの積を加え，新しい

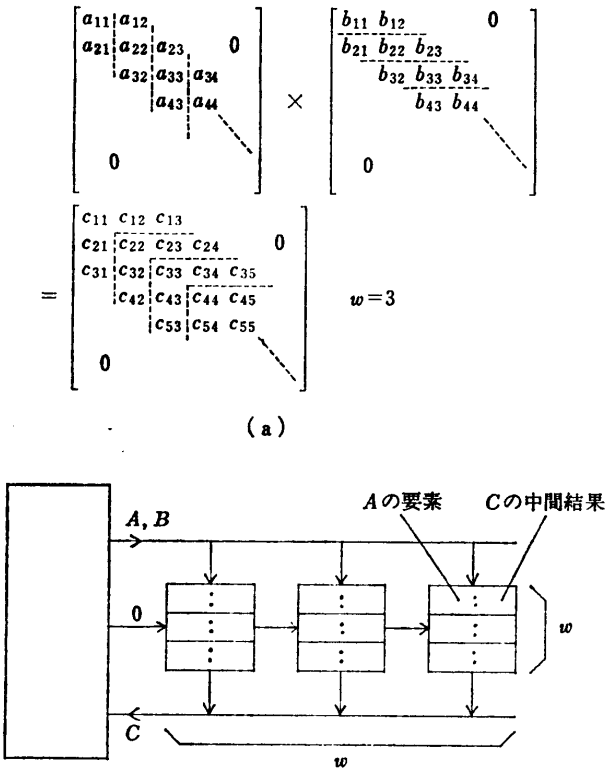


図 5 行列乗算 ($A \times B = C$)
Fig. 5 Matrix multiplication.

中間結果とする．この時，各作業領域の最上段と最後のプロセッサの全作業領域には結果行列 C の一部分（図5のかぎ型の部分）が求められている．

（4） パイプラインモードにより，全作業領域内の中間結果を右上方にシフトする．

（5） 最上段と最右列のデータは BO ポートから出力され，ホストコンピュータに読み取られる．

上記の各サブステップの実行時間はいずれも $O(w)$ 時間であり，全計算時間は $O(nw)$ となる．

4.2 計算時間

試作機による上記計算の実行時間を測定した．行列の各要素は8ビット整数とし，帯幅は $w=n/16$ で行列のサイズに比例するようにした．この場合の全計算量は $O(n^3)$ となる．また，参考として汎用計算機 FACOM-M 380 S による計算時間（FORTRAN による整数計算）も測定した．

図6に行列のサイズに対する実行時間を示す．これより実際に1次元 BC プロセッサアレイが行列乗算を $O(n^2)$ 時間で計算していることがわかる．一方，汎用計算機による計算時間は $O(n^3)$ となっている．試作機

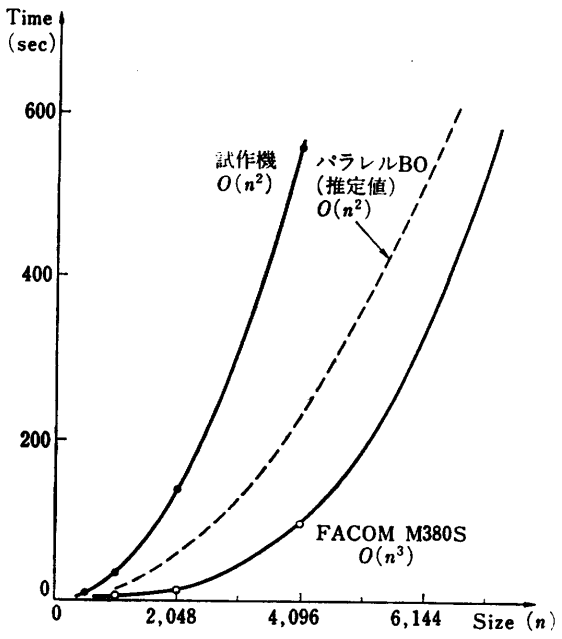


図 6 行列乗算の実行時間
破線は BO をパラレルとした時の推定値
Fig. 6 Execution time of matrix multiplication.

では $n=4,096$ ($w=256$＝プロセッサ総数) までしか計算できずこの範囲では大型汎用機に劣っているが，台数を増すことにより計算速度は逆転する．

本システムでの並列実行においては，常に行列のサイズと同じ数のプロセッサを使用することを基本にしている．したがって一般に行われているように同じ問題について台数を変えて実行し，効率の変化を見るということはできない．行列乗算では，行列のサイズ n に対してデータ転送量と1台当たりの計算量は同じように $O(n^2)$ で増加することになり，台数を増してもボトルネックは生じずプロセッサの効率は変わらない．実際，256 台までの実測値は正確に $O(n^2)$ に従っている．

4.3 動作解析

行列乗算における各処理の実行時間配分を調べるため，4.1 節のアルゴリズムの各ステップを個別にプログラムし実行時間を測定した（**表 1**）．実行時間は $n=4,096$ での値である．各実行時間の合計は図6における全実行時間とほぼ一致している．表1上段の値を総ステップ数 ($4,096 \times 256$) で割って，各処理1ステップの実行時間（表1下段）を得る．この表から試作機の最大性能をまとめると次のようになる．

転送能力（BI ポート） 45 k バイト/秒
転送能力（BO ポート） 5.8 k バイト/秒

演算能力（8ビット整数積和＋パイプライン）1.9 MOPS
(7.4 k×256)

BO ポートがシリアルであるため結果の回収に全処理時間の 66% を要しており，これがシステムの性能を大幅に低下させている．BO ポートをパラレルとした場合に，1データの回収時間が BI ポートへのダイレクト転送時間と同じになると考えると，全処理時間は図6の破線のようになると推定できる．この場合，処理速度は現在の試作機の約2.5倍となる．

表1においてブロードキャスト転送に要する時間はダイレクト転送より 9 μs 長くなっている．この原因として次の3点がある．

（1） BI ポートのハンドシェイク信号の出力はトランジスタを用いてオープンコレクタ出力にしているが，ベース・接地間に抵抗を入れていないため OFF 時にベースの電荷が抜けず約 2.5 μs の遅れが生じている．

（2） ダイレクト転送では同一プロセッサに連続して転送することはなく，他のプロセッサをアクセスする間にデータの格納は終わる．これに対してブロードキャスト転送では同一プロセッサが連続してアクセスされるため，格納時間約 3 μs がきいてくる．

（3） 1台のプロセッサとハンドシェイクするときの平均待ち時間はフラグテストループ時間の 1/2 であるのに対し，複数台になると最も遅いプロセッサに合わせるためループ1回分を要する．これを実証するため，プロセッサ台数を変えてブロードキャスト転送時間を測定した（表 2）．1台の場合のみ 4 μs 早くなっているのがわかる．ただし台数をいくら増しても転送時間は 27 μs 以上増加しない．

4.4 ブロードキャスト転送の効果

ブロードキャスト転送を用いた場合と，用いない場合の計算シーケンスを図7に示す．ブロードキャスト転送を用いた場合の全計算時間は図(a)より，

$$T_b = nw(t_b + 3t_d + t_c + t_p)$$

となる．

一方，ブロードキャストを用いない場合(b)には行列 A の要素を w 回ずつ転送することになる．しかしこの場合には，ブロードキャスト転送時のように一斉同期の必要がないので，計算と他のプロセッサへのデータ転送はすべてオーバラップさせることができる．したがって全計算時間は，

$$T_d = nw((w+3)t_d)$$

となる．

これらの計算時間の比をとると，

$$r = T_d/T_b = (w+3)t_d/(t_b + 3t_d + t_c + t_p)$$

となる．この式は①台数が多く，②1データに対する計算量があまり大きくない場合にブロードキャストの効果が大きいことを示す．プロセッサアレイでの処理はこういった性格が強くブロードキャスト転送は有効である．

上式に試作機の値を代入すると $r = 0.1w$ となる．これは例えば $w = 100$ の時には，ブロードキャストを用いたほうが 10 倍高速になることを示す．データ転

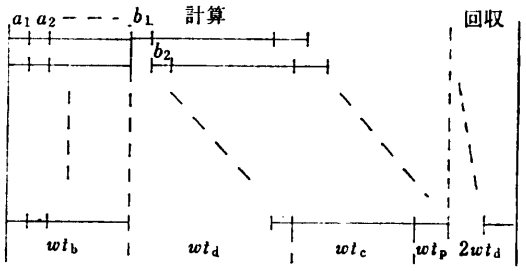
表 1 実行時間の内訳
Table 1 Items of execution time.

	積和 (t_c)	ブロードキャスト (t_b)	ダイレクト (t_d)	パイプライン (t_p)	回収 (t_a)	計
総計算時間 (s)	115	28	19	27	(180×2)*	549
単位時間 (μs)	110	27	18	26	172	—

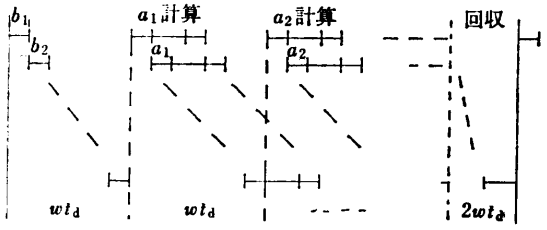
* 結果の回収のみ $2nw$ ステップを要する．

表 2 プロセッサ台数に対するブロードキャスト時間
Table 2 Broadcasting time vs. the number of processors.

台数	1	64	128	192	256
転送時間 (μs)	23	27	27	27	27



（a） ブロードキャスト使用



（b） ダイレクトのみ

図 7 行列乗算の計算シーケンス
Fig. 7 Execution sequences of matrix multiplication.

送に DMA を用い，数値計算に専用プロセッサを用いた場合の転送と計算の処理時間はそれぞれ数百 ns と数 μs になるが，その比は 1：10 程度であり上で論じた関係はそのまま成り立つ．

5. 評価および検討

5.1 試作機について

試作機は行列乗算を理論どおり $O(n^3)$ 時間で実行しており，BC プロセッサアレイの有効性が確認された．また，256 台というプロセッサ台数を持ちながら安定した動作を続けていることから，大規模システムに対する実現性の高さも実証された．本試作機の設計と製作においてはプロセッサ1台当たりのハードウェアを小さくし，台数を増やすことを心がけた．このことで1台当たりの処理能力が小さくなり実用の点では多少無理が生じた（浮動小数点計算など）が，高並列システムの動作や実現性を調べる上では正しい設計方針であったと考える．

1台当たりのメモリ容量が小さいことから，BC プロセッサのプログラミングにアセンブリ言語を使っている点や，モニタが十分なデバッグ機能を持っていない点などで，ソフトウェア開発環境は良いとは言えない．またホストとプロセッサアレイのプログラムを別別に作製するため，データの送受信関係を合わせるのに混乱が生じた．システム全体でのデータの移動をとらえられるソフトウェア開発環境が必要である．

5.2 拡 張 性

まず，プロセッサ台数の拡張について検討する．一般にバス型のシステムではアクセス競合が問題になるが，本システムは同期並列動作が基本であるので問題はない．ハードウェアについてはバスラインの延長による遅延が問題となるが，大規模システムの例として 32 プロセッサ/チップ，64 チップ/ボード，32 ボード/ラック，16 ラック/システムの場合を考えると，2〜3 m のバスラインと4段のバスバッファ（総遅延時間数十 ns）で百万台のプロセッサが接続できることになり，通常の応用に関して問題はない．

結合形態に関しては，1次元システムは実装が容易で広範囲の問題に対応できる点で実用性が高いが，台数を増やして計算時間を2次下げたい場合や問題の2次元性が処理方法に強く関係する場合にはプロセッサアレイの2次元化が必要になる．

2次元 BC プロセッサアレイの例を図8に示す．BI・BO ポートをまとめて一つにし，行方向用と列方向用に用意する（RB と CB）．さらに，上下方向のパイプライン用として OI と UO を設ける．ここで，ななめ方向へのパイプライン転送のため，LI から UO にはバイパス機能を持たせる．また，1次元システムとしても使用できるように，各行の右端プロセッサの RO から1行下の左端プロセッサの LI へのバイパス機能も持たせる．アレイの端（4辺）はバス状にしてホストコンピュータに接続する．

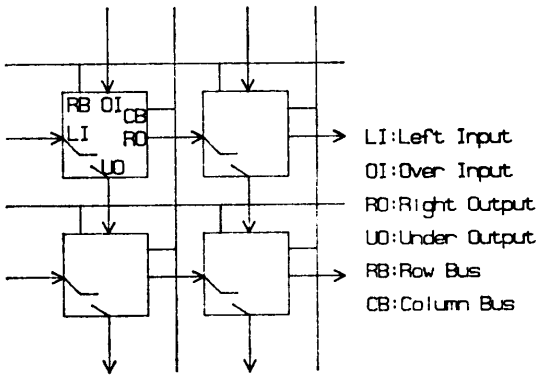


図 8 2次元 BC プロセッサアレイの例
Fig. 8 Example of 2-dimensional BC processor array.

5.3 データ転送によるボトルネック

プロセッサアレイ全般について言えることであるが，計算の多重化に対して，データ転送のそれが対応できなくなる点でボトルネックが生じる．プロセッサアレイ内でのデータ転送がパイプラインやブロードキャストによって多重化できるのに対して，ホストコンピュータや入出力装置との間での転送が直列になる点に特に問題がある．この対策としては転送をできるだけ少なくすることや入出力の並列化が上げられる．

プロセッサアレイの利用は通常次のように行われる．ホストはディスク上の処理データをメモリに読み出し，これをさらにプロセッサアレイに転送する．そして処理結果をプロセッサアレイからメモリ上に移し，次の計算に用いる．もしホストが，メモリをアクセスするのと同じようにプロセッサアレイ内のデータをアクセスすることができれば，ホストはメモリを介さずに入力データを直接プロセッサアレイに転送し，記憶させておくことができる．この場合メモリとプロセッサアレイとの間のデータ転送は必要なくなる．入力データに対する計算はアレイ内で行われ，結果はアレイ内に残る．ホストはプロセッサアレイ内のデータにも変数名を対応付け，必要な時にこれをアクセスする．以上より，プロセッサアレイを演算機能を持った

メモリ（機能メモリ）としてシステムに組み込むことによりデータ転送を少なくすることができる．

また，入出力の並列化はディスクなどに対しては不可能であるが，計測や制御などへの応用において多点の入出力を複数プロセッサが直接行うことは可能であり，実時間処理へのプロセッサアレイの応用が期待される．

6. むすび

256 台のマイクロプロセッサからなる試作 BC プロセッサアレイシステム上で行列乗算問題を実行し，その動作解析結果から有効性や拡張性を評価した．BC プロセッサはデータ線を共通バス接続する点で通常のメモリチップと類似しており，実装が容易で大規模システムの実現性も高い．現在の LSI 技術でも 1 チップ化は容易に可能であり，VLSI 技術の発達により複数プロセッサの 1 チップ化も可能となる．

今後の課題として，連立方程式の計算，データベース処理，画像処理などへの応用が上げられる．

参考文献

1) Kung, H. T.: The Structure of Parallel Algorithms, in *Advances in Computers*, Vol. 19, pp. 65-112, Academic Press, New York (1980).
2) 金田悠紀夫，小畑正貴，前川禎男：BC プロセッサアレイと高並列マトリクス計算，情報処理学会論文誌，Vol. 24, No. 2, pp. 175-181 (1983).
3) Kohata, M., Kaneda, Y. and Miyagaki, Y.: Implementing BC Processor Array, *Proc. of HICSS-19*, pp. 111-116 (1986).

(昭和 61 年 4 月 11 日受付)
(昭和 61 年 6 月 18 日採録)

小畑 正貴（正会員）

昭和 32 年生．昭和 55 年神戸大学工学部電子工学科卒業．昭和 60 年同大学院自然科学研究科システム科学専攻（博士後期）修了．学術博士．現在，岡山理科大学講師．計算機アーキテクチャ，並列処理などの研究に従事．電子通信学会会員．

宮垣 嘉也

昭和 17 年生．昭和 40 年神戸大学工学部電気工学科卒業．昭和 42 年同大学院修士課程修了．現在，岡山理科大学工学部電子工学科教授．通信システムの研究に従事．工学博士．電子通信学会会員．

金田悠紀夫（正会員）

昭和 15 年生．昭和 39 年神戸大学工学部電気工学科卒業，昭和 41 年同大学院修士課程修了．同年電気試験所電子計算機部．TSS システムの研究に従事．昭和 51 年神戸大学工学部システム工学科．現在助教授，工学博士．コンピュータ・アーキテクチャ，ソフトウェア工学，人工知能の研究に従事．電子通信学会，ソフトウェア科学会，ME 学会各会員．