計算機アーキテクチャ　91－10
（１９９１．１１．２２）

資源共有型並列計算機“砂丘”の制御方式の検討

瀬﨑　直裕・井上　倫夫・小林　康浩
鳥取大学工学部

筆者らは，資源共有型並列計算機“砂丘”を開発している．“砂丘”は，複数のマイクロプロセッサを密結合したマルチマイクロプロセッサシステムであり，最大６４台の汎用のマイクロプロセッサが接続可能である．一つの仕事を複数のプロセッサを用いて処理する場合，プロセッサ間での通信，同期が必要である．

本報告では，本システムでの通信，同期，排他制御，プロセス制御について述べ，並列処理の実行制御について検討する．本システムで取り扱う問題は，特に限定せず，科学技術計算全般を予定している．そこで，具体例として，数値計算において最も処理量の多いループ文を，本システムで並列処理する場合について検討を行った．

A Study on the Control Scheme for The Multimicroprocessor System "SAKYU"

Naohiro SEZAKI, Michio INOUE, Yasuhiro KOBAYASHI
Faculty of Engineering, Tottori University

"SAKYU" is a parallel processor system of MIMD type. It employs 64 tightly coupled microprocessors and is designed to solve any problems efficiently in the field of numerical analysis. Performance of such a parallel processor system depends on control of mutual communication between processors, syncronization of individual operations, mutual exclusive and process control.

In such aspects, performance of "SAKYU" is evaluated. An actual criterion is settled by time necessary for solving some problems containing multiple loop processing.

1．はじめに

近年，高性能なコンピューターの要求に応えるため多数のマイクロプロセッサ（以下μPと記す）を並列に接続してシステム全体の処理能力の向上を図るマルチμPシステムが各種開発されている．筆者らは，実験室レベルで特定ユーザーが使用する数値シミュレーションマシンとして，複数のμPを密に結合した並列計算機“砂丘”を開発中である．密結合型を用いた場合，データの送受が高速かつ容易に行えるので様々な処理アルゴリズムに対応できるような利便性の良いシステムの構築が可能である．

密結合のマルチプロセッサシステムを構築する場合の検討課題は，複数のプロセッサが同一のメモリユニットにアクセスしようとする際に起こる競合問題である．アクセス競合によるオーバーヘッドはシステムが大規模になるほど増え，競合が緩和されなければ，プロセッサの接続台数に比例した処理速度の向上が期待できなくなる．しかし，アクセスパスの分散，リードライト比の適正化，メモリの多重ポート化，メモリの階層化など適当な緩和策を構じると，アクセス競合無しに数十～百台程度までのプロセッサを並列処理させることが可能になる［10］．

また，複数のプロセッサが協調して1つの仕事を行う場合，プロセッサ間での通信，同期などが必要である．これらの処理が効率よく実行できなければ処理効率が低下する．システムを構築する際に，このような処理をどのように実行するかを検討する必要がある．

本報告では，現在開発中の資源共有型並列計算機“砂丘”のシステム構成及びその制御方式について述べる．

2．システム構成

当研究室で開発中の並列計算機“砂丘”のシステム構成図を図1に示す．

本システムは，最大64台のプロセッシングユニット（ＰＵ）が接続可能な密結合型マルチμＰシステムである．その構成は，バスコントロールユニット（ＢＣＵ）を介して1本のローカルバスを共有する4台のＰＵを1グループとし，その4グループを1ブロックとする．この4ブロックをマトリックススイッチ（ＭＴＸ）を介してシステムエリア，メインメモリユニット（ＭＭＵ），オメガネットワークに接続した．ＭＴＸは4×4のバススイッチである．

共有メモリは，頻繁にアクセスされるメインメモリと比較的アクセスの少ないシステムメモリ（システムエリア）から成る．

メインメモリは，合計64Ｍバイトの容量をもち，演算用ワークエリアとして使用する．一般に，演算データに対してはライトアクセスよりリードアクセスの方がより多く行われる．そこで，メインメモリを4つのユニットに分割して各ＰＵブロック毎に配置し，各メモリユニットのリードパスは2インターリーブ方式とし，ＭＴＸの2出力に接続した．従って，同じブロック内でのみリードアクセスが可能となる（図2）．またライトアクセスパスはオメガネットワークを介して全てのメモリユニットに接続される（図3）．ライトアクセスパスにオメガネットワークを用いたことにより，メインメモリは次の様な利用形態が可能である．

1）マルチリード・ワンライト方式

オメガネットワークの一斉放送モードを用いて4つのメインメモリユニットの内容を同一に保持することができる．従って，見かけ上のメモリ容量は4分の1になるが，全ＰＵで一斉に同一データに対し処理する場合，リードアクセスのアクセス競合を緩和することが出来る．しかし，全てのＰＵが1つのメモリユニットにライトアクセスするのと同等の動作になるためライトアクセスの競合に注意を要する．

2）パイプライン接続

あるＰＵブロックで処理した結果を，任意の隣

のメモリユニットにライトアクセスする．そしてライトされたデータに対し新たな処理を行い，結果を更に隣のメモリユニットにライトアクセスする．この動作を繰り返すことによりタスクレベルでのパイプライン処理が可能である．

3）個別接続

それぞれのＰＵブロックでリードアクセスするのと同じメモリユニットに，ライトアクセスする．この形態は，各ブロックが４つのメモリユニットをそれぞれ独立して使用することになり，ＰＵブロック別に分散処理が可能となる．

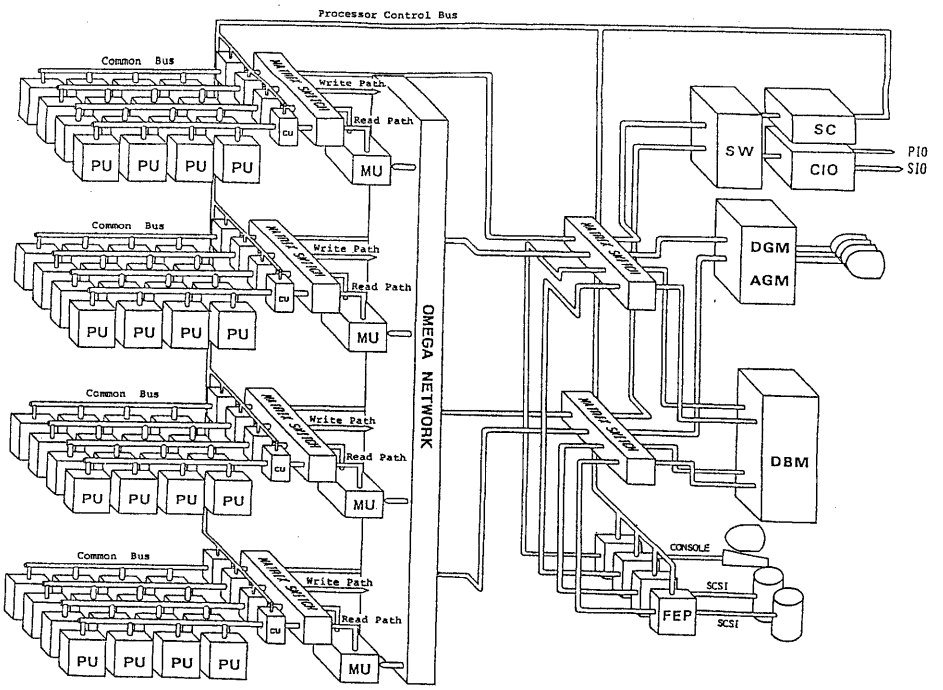
システムメモリは全プロセッサで共有するデータエリアとしてそれぞれ機能化されており，２段目のＭＴＸの出力に接続されいる．このエリアはグラフィックメモリ，データバッファメモリ及びシステムワークエリアより成る．

グラフィックメモリは計算結果を分かりやすく図示するために用意した．

データバッファメモリは入力データ，計算結果などを格納し，メインメモリと外部記憶装置等とのバッファ領域として用いる．

システムワークには，並列処理を効率よく実行するためのハードウェアとして，システムコントローラ，メイルボックス，排他制御回路，プロセス番号割当回路等を配置している．また，他の機器との入出力を行うためのコミュニケーションＩ／Ｏが設けてある．

また，ＰＵを演算専用とするために，マンマシンインターフェース及びシステムの制御を受け持つフロントエンドプロセッサ（ＦＥＰ）を用意し，２段目のＭＴＸの入力に接続している．ＦＥＰは必要な機能を分散して処理するため複数用意している．



PU : Processing Unit
CU : bus Control Unit
MU : Memory Unit
SW : System Work area
SC : System Controller
CIO : Communication Input/Output
DGM : Digital Graphic Memory
AGM : Analog Graphic Memory
DBM : Data Buffer Memory
FEP : Front End Processor

図１　システム構成図

3．システムの制御

マルチプロセッサシステムで並列処理を行う場合，プロセスの起動，プロセス間での共有データの授受といった通信，共有変数の更新の際の排他制御，処理の順序関係を保証するための終了時の同期など並列処理のためのオーバーヘッドが存在する．このような制御を効率よく実行できなければプロセッサを増加しても処理効率の向上が望めない．

3．1　通信制御

密結合のマルチプロセッサシステムではプロセッサ間の通信を行う場合共有メモリと割り込みを使用する．あるプロセッサが他のプロセッサにメッセージを送るには共有メモリの所定のエリアに設けたメイルボックスにメッセージを書き込み，相手のプロセッサに割り込み信号を送り，割り込まれたプロセッサはメイルボックスよりメッセージを読み取る．

本システムでは，システムコントローラとして全てのＰＵに接続される割り込み制御回路を設けた．各ＰＵは，この制御回路及びシステムワークエリア内に設けたメイルボックスを用いて通信を行う．

3．2　同　期

１つのまとまった処理を並列処理する場合，全ての部分が並列に実行できるわけでないので，並列に実行される処理の終了時に同期を取る必要がある．

システムコントローラ内には，通信時の割り込み要求時にセットされる同期用フラグが設けられている．従って，プロセスの起動の通信時にこのフラグがセットされ，処理を実行する各ＰＵがプロセスの終了時にこのフラグをリセットする．プロセスの制御を行うプロセッサはこの同期フラグを管理することにより各プロセッサの同期をとることができる．

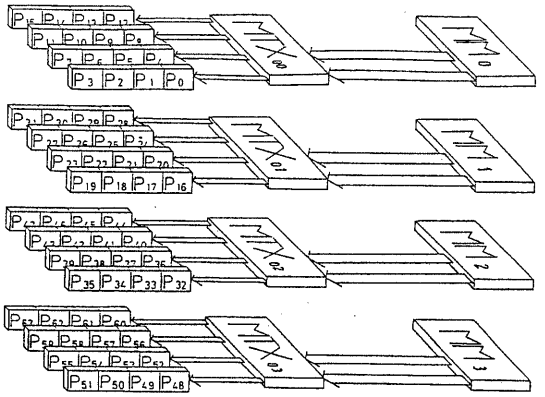


図２　リードアクセスパス

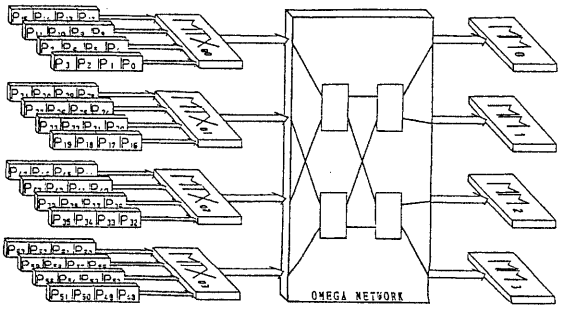


図３　ライトアクセスパス

3．3　排他制御

多数のプロセッサが共有データを更新するといった排他制御を必要とする処理を実現する場合，共有データへの並列アクセスを直列化して専有アクセスを実現する機構が必要である．

本システムでは，この排他制御の際必要とされる，共有変数に対する不可分なリード・モディファイ・ライト動作をハードウェアで実現している．共有メモリの一部を排他制御用メモリとして確保し，このメモリに対してはリード動作とライト動作を不可分に行う．

排他制御を必要とする処理において，各ＰＵはこの領域に先ずアクセスし，専有権獲得の確認を

行った後，共有データの更新を行う．

但し，この種の排他制御回路は専有権争奪に於いて競合が起き，処理効率の低下を招き易いため，専有権を得られなかったプロセッサの振る舞いに注意をすることが必要である．

3．4　プロセス制御

複数のプロセスの生成を行う場合，各プロセスが生成時に違ったプロセス番号を得ることにより，プロセスがそれ自身を認識し，同じ処理の違った部分（または，完全に異なる処理）を行うことが可能となる．従って，多数のプロセスを限られた数のＰＵで効率よく並列処理するには，各ＰＵの処理するプロセスの番号管理が必要である．この管理を通常の共有変数を用いた場合，この変数に対する排他制御が必要である．この際，専有権争奪に於いて競合が起こり，プロセス起動のオーバーヘッドが増大する．

本システムでは，この機能をハードウェアで実現している．この番号をカウンタで管理し，ＰＵからのリードアクセスが終了すると自動的に指示値を増すようにした．従って，プロセスの番号操作のためのアクセスは１回となりプロセス起動時のオーバーヘッドを抑えることができる．

4．検　討

本システムで並列処理を行う場合の手順を以下に示す．並列処理されるプログラムはあらかじめＰＵのローカルメモリにロードされているものとし，プロセスの起動時にプロセス制御ブロックのポインタを受取り，処理を実行する．

1）プロセスの管理を指名されたプロセッサがプロセス制御情報のポインタをシステムワークエリア内のメイルボックスに書き込む．
2）プロセス番号割当回路の初期化を行う．
3）複数のＰＵに対しプロセス起動の通信を行うためシステムコントローラにアクセスする．このとき，各ＰＵに応じた同期フラグがセットされ，ＰＵに対し割り込み要求が行われる．
4）割り込み信号を受けたＰＵはメイルボックスより必要なプロセス制御情報のポインタを読み出す．
5）その後プロセス番号割当回路にアクセスを行い自身のプロセス番号を得る．
6）得られた番号が起動されたプロセス数を越えていなければ，得られた番号に対応するプロセスを実行する．
7）プロセスの実行が終了したＰＵは，５）から６）までの操作をプロセス番号がプロセス数を越えるまで繰り返す．得られた番号がプロセス数を越えている場合には，ＰＵはシステムコントローラ内の同期フラグをリセットして処理を終了する．
8）プロセスを管理するプロセッサは，割り込み要求を行った全てのＰＵに対応する同期フラグのリセットを確認し同期をとる．

4．1　ループ処理の実行

本システムで扱う問題は特に限定せず科学技術計算全般を予定している．この種の計算のプログラム中で最も処理量が大きいのは，大規模な配列データの各要素に対して一定の処理を行うループ文である．このようなループ文はdo_all文等の形に記述して実行される．この時，すべてのプロセッサは配列の要素を読み込み，同一の処理を行い要素の更新を行う操作を全ての要素に対して繰り返す．この場合，各プロセッサの処理する要素をうまく配分しなければならない．

並列処理実行の具体例として，図４のようなdo_allで記述された処理を行う場合を考える．

ループ内の処理プログラムは各ＰＵにロードされているものとする．

プロセスの管理を指名されたプロセッサは，プロセス番号割当回路を初期化し，システムコントローラを用いて，ループ内の処理のポインタを全

てのＰＵに送り，プロセス起動の通信を行う．プロセス起動の通信を受けたＰＵは，このポインタをメイルボックスより受取る．

次にプロセス割当回路にアクセスする．得られた値がこの場合は制御変数iの値に対応する．各ＰＵはiに応じたループ内の処理を実行する．

処理が終了すれば，再びプロセス番号割当回路にアクセスし，得られる値がmを越えるまで各値に応じたループ内の処理を繰り返す．

得られた値がmを越えたＰＵはシステムコントローラにアクセスし同期フラグのリセットを行い，処理を終了する．

プロセスの管理を指名されたプロセッサは，この同期フラグが割り込み要求を行った全てのＰＵについてリセットされたことを確認し同期をとり，ループ処理の終了の確認を行う．

プロセス番号割当回路をこのように用いることにより，制御変数iの更新が１回のリードアクセスのみで実行でき，このiの専有のための排他制御及び，専有権争奪に於ける競合に因るオーバーヘッドを無くすることができる．また，このような処理プログラムを内蔵しておけば，プログラム作成時に，実際に並列処理をプログラムに行うＰＵの台数を意識する必要がない．

この処理を行う際のプロセッサ間の通信は，プロセス起動の際に行われるだけで，ループ処理実行中には通信する必要がない．このため，ループ処理全体から考えると通信のオーバーヘッドは問題にならないと考えられる．

一方，この処理のように負荷を均等に配分した場合，演算データを格納するメインメモリにアクセスする際に競合が起き易く，各ＰＵの待ち時間の累積の増大が処理効率低下の大きな要因となる．

そこで，ＰＵの持つ能力をどれくらい利用できたかを見積もるために，実際に計算に要した時間の内，各プロセッサが純粋に計算に専念できた時間の割合として稼働率を定義し，検討する．

筆者らは，各ＰＵの待ち時間の累積が最大となる（均等負荷で動作している）ときのシステムの

```
do_all  i=1,m
  •
  •
 a(i)=b(i)*c(i)
  •
  •
continue
```

図４　ループ処理のプログラム

性能を陽的に解析できることを示した[6]．

そこで，システムのハードウェア特性（共有メモリのアクセス時間，サイクル時間等）及びソフトウェア特性（プロセスの処理時間，共有メモリのアクセス回数）とを用いて，並列動作時の各ＰＵの平均稼働率を次のように表すことができる．

$$P(n,Th) = \frac{100}{1 + \dfrac{t_{ac} + m_n \cdot t_S}{Th}} \qquad (1)$$

ここで，

$$m_n = \begin{cases} 0 & (n \leq i_B \cdot n_0) \\ \dfrac{n}{i_B} - n_0 & (n > i_B \cdot n_0) \end{cases} \qquad (2)$$

ただし，

n ：同時に動作しているＰＵの台数

Th ：プロセスの平均シンクタイム［Ｓ］
（ＰＵが共有メモリをアクセスする平均時間間隔）

t_S ：共有資源のサイクル時間［Ｓ］

t_{ac} ：共有資源をアクセスするとき各ＰＵが要する平均アクセス時間

$$t_{ac} = t_B + t_A + \frac{1}{2}t_S \qquad (3)$$

t_A ：共有資源のアクセス時間［Ｓ］

t_B ：ＢＣＵ，ＭＴＸ等での遅延時間［Ｓ］

i_B ：共有バスのインターリーブ数

n_0 ：同一の共有バスでアクセス競合による待ち時間の累積無しに動作できるＰＵの台数

$$n_0 = 1 + \frac{Th}{t_S} \qquad (4)$$

本システムでは，全ＰＵで並列処理を行う場合，メインメモリの制御にマルチリード・ワンライト方式を採用している．

リードアクセス用パスは，１ユニット当たり２インターリーブ構成としたので，メインメモリ全体で８インターリーブ方式と同等とみなすことができる．

この方式ではアクセス競合を起こさずに接続できる台数がアクセスのリード・ライト比（ソフトウェア）に依存する．メインメモリへのアクセス全体を１としたときのリードアクセスの割合をｒとする．このとき（４）式を拡張すると，リードアクセスにおいてアクセス競合を起こさずに動作できるＰＵの最大台数ｎ$_R$は，

$$n_R = 8\left(1 + \frac{Th/r}{t_s}\right) / r \qquad (5)$$

で表させる．同様にライトアクセスの場合の最大台数ｎ$_w$は，

$$n_w = 1\left(1 + \frac{Th/(1-r)}{t_s}\right) / (1-r) \qquad (6)$$

で表すことができる．

（５）（６）式の関係を図５に示す．この図から，シンクタイムが6[μs]でリード・ライトアクセス比が0.65以上であれば，アクセス競合を起こさずに64台のＰＵで並列処理を実行可能であることがわかる．

また，稼働率を表す式は，（１）（２）式に（５）（６）式を用いて，

$$P(n) = \frac{100}{1 + \frac{t_{ac} + m \cdot t_s}{Th/r} + \frac{t_{ac}' + m' \cdot t_s}{Th/(1-r)}} \qquad (7)$$

と表せる．

（７）式において，ｒが0.8の場合の，ＰＵの台数と稼働率の関係を図６に示す．この図から，シンクタイムが6[μs]であれば，90%程度の稼働率で実行できることがわかる．

5．おわりに

以上，資源共有型並列計算機“砂丘”の共有メモリの制御方式及び通信，同期等の制御のための専用ハードウェアと制御方式につい述べた．さらに，本システムで目的としている数値計算において最も処理量の多いループ処理を並列処理に実行する場合について検討し，プロセス番号割当回路等を用いることにより効率良く実行できることを示した．また，このように負荷が各ＰＵに均等に割り当てられるような場合，処理効率低下の最大の要因である共有資源へのアクセス競合を起こすこと無く処理が実行できることを示した．ここで示した処理は通信のオーバーヘッドが最も小さい処理であるから，複雑な制御を必要とする問題に対しては，オーバーヘッドによる処理効率の低下について検討を行う必要がある．

今後の課題としては，本システムを使用して各種の具体的な問題に対する評価を行うことである．

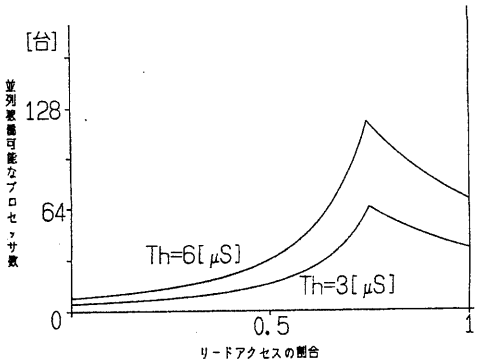


図５　メインメモリ利用時のリード・ライト比と競合を起こさずに稼働できるＰＵ台数

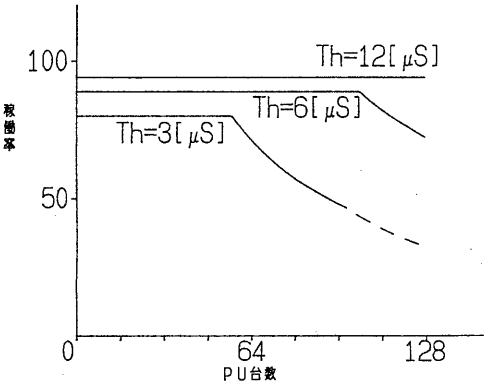


図６　メインメモリ使用時のＰＵ台数と稼働率

参考文献

[1] 白川 他："並列計算機ＰＡＸ－１２８"
通信学論, Vol.J67-D, No.8, pp.853-860,
Aug. (1984)

[2] 出口 他："コンピュータグラフィックシステムＬＩＮＫＳ－１における画像生成の高速化手法" 情報処理論文誌, Vol.25,
No.6, pp.944-952, Nov. (1984)

[3] Rodrigue. G.: "Parallel Computation"
Academic Press (1982)

[4] Paker. Y: "Multi-microprocessor"
Academic Press (1983)

[5] 井上・小林："マルチマイクロプロセッサシステムα－16のアーキテクチャ"
情報処理論文誌, Vol.25, No.4,
pp.632-639, July (1984)

[6] 井上・小林： "α－16マルチマイクロプロセッサシステムの性能評価"
情報処理論文誌, Vol.25, No.4,
pp.640-646, July (1984)

[7] 井上・小林："資源共有型マルチマイクロプロセッサシステムにおけるアクセス競合の調停について" 電子情報通信,
回路とシステム研究会資料, CAS84-206,
pp.9-16 (1984)

[8] 山根 他："並列計算機"砂丘"のハードウェアアーキテクチャ" 電子情報通信,
コンピュータシステム研究会資料,
CPSY87-3, pp.15-22, June (1987)

[9] 加納 他："マルチマイクロプロセッサシステムの大容量共有メモリの一構成法"
情報処理学会
計算機アーキテクチャ研究会資料
CA-66-2, pp.1-8, July (1987)

[10] 井上 他："マルチマイクロプロセッサシステム"砂丘"の共有メモリアーキテクチャについて" 情報処理学会
計算機アーキテクチャ研究会資料
CA-66-2, pp.9-16, Nov. (1989)

[11] 荒川・橋本他： "資源共有型並列計算機"砂丘" 情報処理学会
計算機アーキテクチャ研究会資料,
CA-66-2, pp.1-6, Nov. (1990)