単一バス同期データ交換型マルチコンピュータシステムの高能率化†

出口光一郎†† 森下巖†† 小笠原司††
小野輝†† 平沢裕†† 渡辺淳††

単一バス同期データ交換型マルチコンピュータシステムは，基本的にハードウェアが簡単，プログラミングが容易，また，データ交換に際してデッドロックが生じ得ないという特徴があるが，一方，これまでのシステムでは，一般的な問題に対してはコンピュータの停止時間が長くなるという欠点があった．本論文では，この欠点を改良する高能率化方式を報告する．ここでは，各メンバコンピュータに各ステップで割当てられたタスクを外部用タスクと内部用タスクに分離する．外部用タスクは他のメンバとの間で伝送するデータを処理するタスク，内部用タスクはそのメンバ内部でのみ用いるデータを処理するタスクである．各ステップでまず外部用タスクを実行し，すべてのメンバコンピュータでこれが完了するとデータ交換を開始する．これと同時に内部用タスクの処理も進める．これによりデータ交換とタスク処理の並列進行が可能になり，メンバ間の負荷不均等による停止時間も短縮する．また，データ交換用に高速メモリと高速コントローラを用い，データ交換時間が各ステップの処理時間の支配的要因とならないようにする．

以上の高能率化方式に基づいて，16ビットマイクロコンピュータを用いてシステムを試作したので，制御方式，回路構成およびシステムのパフォーマンスについても合わせて報告する．

1. まえがき

マルチコンピュータシステムでは，本来一体である計算ジョブを並列処理可能な多数のタスクに分割し，それぞれを複数のコンピュータで分担処理させる．しかし，本来一体であったジョブを分割したのだから，各コンピュータは他から完全に独立してタスク処理を進めることができず，相互にデータ交換を行いながら計算を進めなければならない．したがって，マルチコンピュータシステムでは，システム構成要素であるコンピュータ間にどのような通信用バスを用意し，それを用いてのデータ交換をどのように制御するかが重要な問題となる．

通信用バス構成方式およびデータ交換制御方式については，多くの研究が報告されている[1)~3)]．そのなかで，バス構造，制御方式ともももっとも簡単なものが，シーメンス社の開発した SMS 201 である[4),5)]．各コンピュータは専用ローカルメモリのほかに通信用メモリを持っており，これはすべて一本の通信用バスに接続されている．各コンピュータに割当てられたタスクは一連のステップに分割され，あるステップの処理が終ると通信用メモリに書込まれた内容を通信用バスを用いて1語ずつ順番にすべて交換する．これは「単一バス同期データ交換方式」と呼ぶのが適当な方式である*．

SMS 201 はハードウェアが簡単でプログラミングも容易になるという特徴を持っているが，一般的なジョブに対する処理能率の点では問題がある．第一に，データ交換実行中すべてのコンピュータが動作を停止するため能率が低下する．第二に，各ステップにおけるコンピュータの負荷が不均等になる場合，データ交換待ちのためコンピュータの動作停止が発生して能率が低下する．短時間でタスク処理が完了したコンピュータももっとも長時間を必要とするタスクが完了するまで待たなければならないからである．

本論文では上記の欠点を改良する高能率化方式を報告する．この方式においては，各ステップのタスクを外部用タスクと内部用タスクに分離し，まず外部用タスクを実行して，すべてのコンピュータでこれが完了するとデータ交換を開始する．これと同時に内部用タスクの処理も進める．この方法により，

† High Performance Design for Single-Bus Synchronized Data-Exchange Multi-Computer Systems by KOICHIRO DEGUCHI, IWAO MORISHITA, TSUKASA OGASAWARA, AKIRA ONO, YUTAKA HIRASAWA, and JUN WATANABE (Department of Mathematical Engineering and Instrumentation Physics, Faculty of Engineering, University of Tokyo).

†† 東京大学工学部計数工学科

* 開発者は SIMD 方式 (Switched Multiple Instruction-Multiple Data Stream 方式) に沿ったものと考えているようである．

(i) データ交換と内部用タスク処理の並列進行が可能となってコンピュータの停止時間が短縮し，

(ii) 各ステップ内で早期にデータ交換が行われるため内部用タスクを完了したコンピュータは直ちにつぎのステップのタスク処理に入ることができ，負荷不均等によるコンピュータ停止時間が短縮する．

また，データ交換時間が各ステップの処理時間の支配要因とならないよう，

(iii) バイポーラ通信用メモリと高速コントローラによるデータ交換の高速化

をはかる．

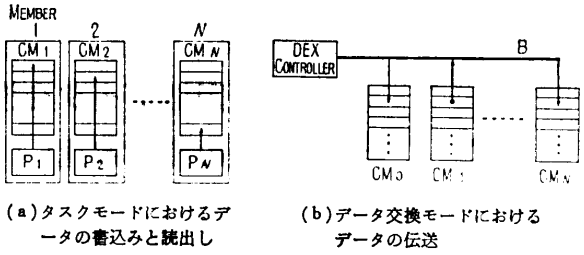
以上の高能率化方式に基づいて，システムを試作したので，制御方式，回路構成およびシステムのパフォーマンスについても合わせて報告する．なお，試作システムは，TI 社の 16 ビットプロセサ TMS 9900 を用いて構成した．

2. 単一バス同期データ交換方式

単一バス同期データ交換方式によるマルチコンピュータシステムの構成は，図 1(a)に示したものである．N台の各コンピュータは，プロセッサ P_i，メモリ M_i，通信用メモリ CM_i よりなり，CM_i は通信用バス B に接続されている．G_{i1}, G_{i2} はそれぞれバッファゲートである．タスクモードでは，ゲート G_{i1} が開き，G_{i2} が閉じて図 1(b)の構成となる．このモードでは，各プロセッサはメモリに収められたプログラムとデータを用いて，それぞれ独立したコンピュータとして処理を進める．処理中に，他のコンピュータに伝送しなければならないデータは通信用メモリに書込む．データ交換モードでは G_{i1} が閉じ，G_{i2} が開いて図 1(c)の構成となる．このモードでは各コンピュータは作業を休止し，データ交換コントローラ (DEX Controller) の制御の下にバス B を用いてデータ交換を行う．

通信用メモリ CM_i を用いたデータ交換は図 2 に示す方法で行われる．すなわち，各コンピュータの通信用メモリはすべて同一構造になっているが，第 i 番目のコンピュータにはその中の第 i ブロックが割当てられている．この CM_i はタスクモードでは，M_i と連続したアドレス空間を持っていて，ふつうのメモリとして読み書きが行われるが，第 i 番目のコンピュータより他のコンピュータへ伝送するデータは，この CM_i 内の第 i ブロックに書込む (図 2(a))．データ交換モードでは，CM_1 内の第 1 ブロックを他のすべての CM_i の第 1 ブロックへ転写し，続いて，CM_2 の第 2 ブロックの内容を他のすべての CM_i の第 2 ブロックへ転写する (図 2(b))．すべての割当てブロックに対してこの動作が行われると，データ交換モードは完了する．データ交換の完了した時点では，すべての CM_i は同一内容となっており，それぞれの CM_i 内の第 k ブロックには第 k 番目のコンピュータから伝送されたデータが入っている．データ交換が終了すると各コンピュータはタスクモードへ戻る．このとき，各コンピュータは自分の CM_i 内の第 k ブロックを読むことにより，第 k 番目のコンピュータからの伝送データを知ることができる．

この方式を用いたときのシステム全体のジョブの進行の流れは図 3 に示したものとなる．各コンピュータでタスクの進行状況が異なっている場合には，各タス



(a)タスクモードにおけるデータの書込みと読出し　(b)データ交換モードにおけるデータの伝送

図 2 データ交換動作のあらまし

Fig. 2 Illustratings of the data exchange procedure. (a) Writing in the task mode. (b) Broadcasting in the data exchange mode.

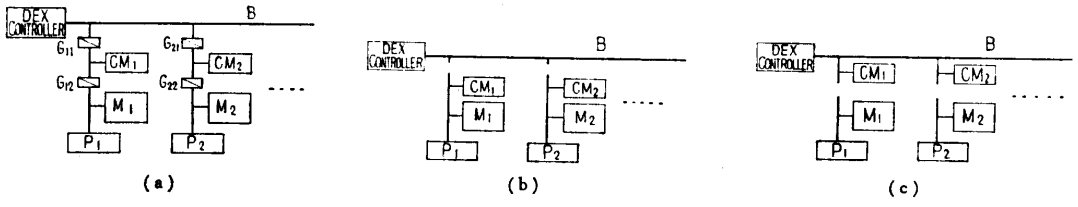


図 1 (a)単一バス同期データ交換型マルチコンピュータシステムの構成 (b)タスクモードにおける結合 (c)データ交換モードにおける結合

Fig. 1 (a) Architecture of a multi-computer system with single-bus and synchronized data-exchange control. (b) Configuration in the task mode. (c) Configuration in the data exchange mode.

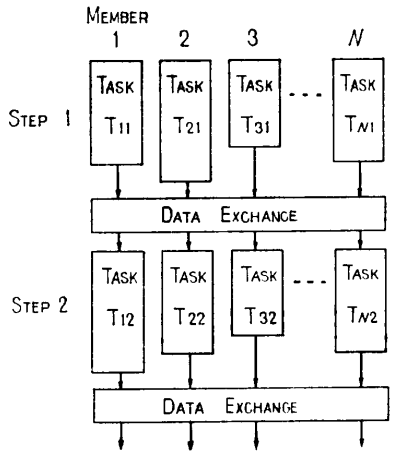


図 3　コンピュータ N 台のシステムにおけるタスクの流れ
Fig. 3　Task flowgraph for N-computer system.

クが終了し全コンピュータのデータ交換の用意が整うまで，早く進んだコンピュータは動作を停止して待つことになり，データ交換によって各コンピュータの作業進行の同期がとられる．この方式は，システムのハードウェア構成が単純であり，さらに，ジョブの進行状態が図3に示す形でプログラマに完全に把握できるため，プログラミングも容易である．

なお，前記の SMS 201 では，プロセッサには 8080 を用い，台数は 128 台，通信用メモリは各 4 k バイトで，1台あたりの通信データ数，すなわち前記の1ブロックは 32 バイトである．データ交換モードでは，1バイトあたりの伝送を 1.4 μs で実行しており，4 k バイトを伝送するのに 5.7 ms 必要となっている．

3. 高能率化方式

前章で述べたように，単一バス同期データ交換方式マルチコンピュータシステムには，ハードウェアが簡単でプログラミングが容易になるという実用上重要な特徴がある．しかし，一般的なジョブに対しては，データ交換と同期のためのコンピュータの停止時間が無視できなくなり，能率が落ちるという欠点を持つ．そこでこの特徴を生かしつつ高能率化をはかることを考えた．

図 1(c)から知れるように，通信用バスと通信用メモリを用いてデータ交換を実行しているときでも，各プロセッサ P_i はメモリ M_i 中のプログラムとデータを用いて計算を進めることができる．これを利用してデータ交換と計算を並列進行させれば，処理速度が向上するはずである．問題は，データ交換を実行する前に，交換用データがすべて計算され，通信用メモリ内に収められていなければならないことである．そこで，外部用処理と内部用処理を分離し，まず外部用処理を実行することを考える．

第 i 番目のコンピュータが第 j 番目のステップで割当てられたタスクを T_{ij} とする．これを外部用処理 R_{ij} と内部用処理 S_{ij} とに分割する．ここで，外部用処理 R_{ij} とは，他のコンピュータとのデータ交換に関係した処理である．他のコンピュータから通信用メモリ CM_i を通じて受取ったデータをレジスタやメモリ M_i 内に退避させたり，次のデータ交換で他のコンピュータに伝送するデータの計算をし，通信用メモリ CM_i 内に格納する処理を行う．内部用処理 S_{ij} は，各コンピュータがこれまでのステップで計算し自己の内部に保持しているデータを用いて実行でき，その結果もさしあたっては他のコンピュータに伝送する必要のない処理である．各タスク T_{ij} において，まず R_{ij} を処理し，つづいて S_{ij} を処理する．

ハードウェアとしては，R_{ij} が完了したこと，したがってデータ交換の準備が整っており，この後はいつデータ交換が行われてもよいことを示す「データ交換準備完了フラグ」DEX と，S_{ij} も終了し，したがって T_{ij} が終了して，それ以上は他のコンピュータとのデータ交換なしには作業を進められない状態に達したことを示す「タスク完了フラグ」STEP を各コンピュータごとに用意する．プロセッサにはこの2つのフラグを立てるための命令 DEX と STEP を用意する．プログラムとしては，R_{ij} 処理プログラムの次に DEX を書き，そのあとに S_{ij} 処理プログラムを入れ，最後に STEP を置くことで T_{ij} を構成する．命令 DEX が実行されると DEX フラグが立つが，コンピュータはそのまま次の命令（S_{ij} 処理プログラムの第1命令）の実行に入る．データ交換コントローラは，すべてのコンピュータの DEX フラグが立つとデータ交換を開始する．DEX を発信した後，データ交換が完了する前に命令 STEP に達したコンピュータは，ホールド状態に入れてデータ交換を待たせる．一方，データ交換が終了した後で STEP 命令が実行された場合には，ホールドさせずにそのまま次のステップのタスク $R_{i,j+1}$ の処理に入らせる．

図 4 に，第 i 番目のコンピュータのためのプログラムの例の一部を示す．これは，後述の試作システムのアセンブラ言語によるものである．それぞれのコンピュータのプログラミングでは，他のコンピュータの仕

```
        .
        .
      BLWP  @FML
      MOV   TEMP, AC
      DATA  NEXT
NEXT  MOV   AC, @R
      MOV   @Y, @BUF
      DEX
      MOV   @Y, @P *4+X
      LI    DS, X
      LI    TEMP, BUF
      STEP
LOOP  MOV   *TEMR+, *DS+
      CI    TEMP, BUF+16
      JLT   LOOP
      MOV   @R, @BUF
        .
        .
```

R_{ij} T_{ij} S_{ij} R_{ij+1} T_{ij+1}

図 4 メンバコンピュータのためのプログラムの例
Fig. 4 Program example for a member computer.

事の進行状況を気にする必要はなく，この例のように R_{ij} の最後に命令 DEX を，そして S_{ij}，したがって第 j ステップのタスク T_{ij} の最後に命令 STEP を挿入するだけでよい．このとき，プログラム上の DEX と STEP の間で必ず1回データ交換が行われ，それ以前に次のステップのタスク $T_{i,j+1}$ に処理が入ることはない．すなわち，自動的に他のコンピュータとの間での仕事の進行の同期がとられる．

この方式は，各ステップにおいて各コンピュータの負荷が均等でないことによる能率低下に対しても有効である．SMS 201 の採用した単純な方式では，タスク T_{ij} の処理時間を t_{ij} とすると第 j ステップの処理時間は，

$$t_j=\max\{t_{1j}, t_{2j}, \cdots, t_{Nj}\}$$

で与えられ，**図 5**(a)に示したようにコンピュータの休止時間が長くなる．外部用処理と内部用処理を分離すると，データ交換中すべてのコンピュータを停止させたとしても図 5(b)のようになり，処理速度が向上する．

もちろん，内部用処理とデータ交換を並列進行させればさらに処理速度が向上し，図 5(c)に示したものとなる．R_{ij}, S_{ij} の処理時間を r_{ij}, s_{ij} とすると，

$$t_{ij}=r_{ij}+s_{ij}.$$

このとき，

$$\min_i\left\{\sum_{j=1}^{k} t_{ij};\ 1\leq i\leq N\right\}$$
$$\geq\max_i\left\{\sum_{j=1}^{k-1}(t_{ij}+r_{ij});\ 1\leq i\leq N\right\}$$

がすべての $k\leq K$ で成立すれば，第 K ステップまでのコンピュータの休止時間は 0 となる．なお，SMS 201 ではタスク完了フラグしか用意されていなかった．

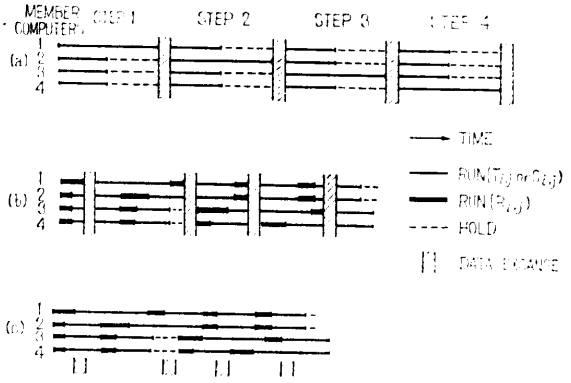


図 5 高能率化システムによるパフォーマンスの改善
(a)単純システムにおけるパフォーマンス
(b)タスク T_{ij} を2つのサブタスク R_{ij}, S_{ij} に分割することによる改善．ここで，R_{ij} は伝送データを処理する外部用タスク，S_{ij} はメンバ内部でのみ用いるデータを処理する内部用タスク (c)データ交換と S_{ij} の処理を並列化させることによる改善
Fig. 5 Illustratings of performance improvement with the proposed techniques. (a) Performance in the prototype system. (b) Improvement by dividing each task T_{ij} into two subtasks R_{ij} and S_{ij}. R_{ij} is a computation of data to be transmitted to other computers, while S_{ij} is a computation of data to be used only in itself. (c) Further improvement by parallel running of the data exchange and the subtask S_{ij} processing.

上記のように，外部用処理と内部用処理を分離し，データ交換と内部用処理を並列進行させる方式を採用した場合でも，データ交換が低速度でデータ交換に必要な時間が内部用処理の時間より長くなれば，やはりコンピュータの停止が発生し能率の低下が生じる．したがって，当然のことながらデータ交換はできるだけ高速化するのが望ましい．そこで，後述の試作システムでは通信用メモリにバイポーラメモリを用い，データ交換を 50 ns/語と高速化している．

4. 試作システム

前章で述べた高能率化方式に基づくマルチコンピュータシステムを試作した．パイロットシステムとしてコンピュータ接続台数 16 台のものを設計し，通信用メモリの記憶容量は 32 語／台とした．32 語のデータ交換に要する時間は 1.6 μs であり，16 台程度ではタスク処理時間に比べてデータ交換時間は無視できる．このため，このパイロットシステムでは図1のゲート

G_{12} を省略し，データ交換中プロセッサの動作を停止させる方式で運転している．コンピュータを100台程度接続する実用機では，もちろん，データ交換とタスク処理の並列進行によるメリットがゲート G_{12} のコストに比べてはるかに大きくなる．

4.1 システムの構成

システム構成の概要を**図6**に示す．1本のバスBに16台のそれぞれ独立したコンピュータとして働くマイクロコンピュータが接続されている．うち1台が，各コンピュータへのプログラムのロード，デバッグなどの機能を持つホストコンピュータであるが，他の15台のメンバコンピュータとの共同の仕事に加わることもできる．データ交換の制御は，バスBに接続されたデータ交換コントローラ (DEX Controller) によるが，すべての制御機能をこれに集中しているのではなく，各コンピュータ内に制御論理回路 ($CL_H, CL_1, CL_2, \cdots$) として分散配置し，若干の制御信号線で結ばれている．データ交換の点では，ホストとメンバは同一の構造を持つ．

各構成要素の概要は以下のとおりである．

1）プロセッサ：TI社 TMS 9900
　　語長16ビット

2）通信用メモリ：バイポーラメモリ（サイクルタイム25 ns），32語／台

3）一般用メモリ：NMOS RAM および EPROM
　　13 kバイト／台（ホストコンピュータは33 kバイト）

4）台数：16台（現在5台実装）

5）データ交換時間：1.6 μs (50 ns×32語)

4.2 同期データ交換制御回路

図7は同期データ交換制御回路のあらましである．ホストおよびメンバコンピュータの通信制御回路

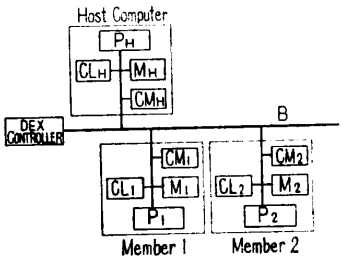


図6 試作システムにおける制御回路の構成
Fig. 6 Control circuit distribution in the pilot system.

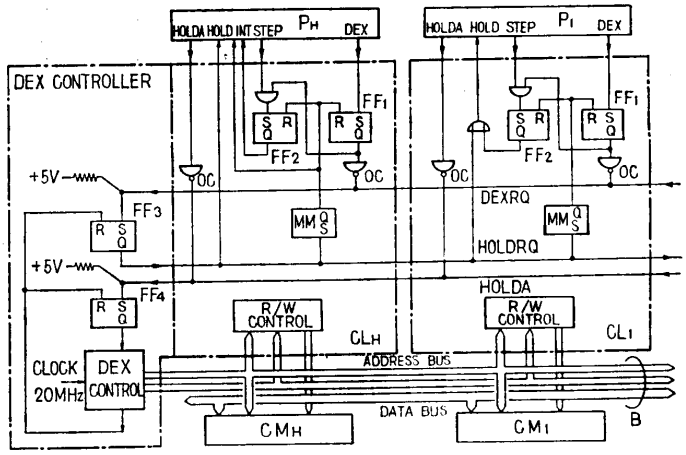


図7 データ交換制御回路
Fig. 7 Schematic diagram of the data exchange control circuit.

CL_H, CL_1 および DEX コントローラの主要部を示す．プロセッサには，データ交換準備完了信号 DEX およびタスク完了信号 STEP を発する一語命令が用意されている．プロセッサが DEX を発信すると，フラグ・フリップフロップ FF_1 がセットされ，その出力がオープンコレクタゲート接続した信号線 DEXRQ によって集計される．すべての FF_1 が "1" を発信すると，DEX コントローラ内の FF_3 がセットされ，信号線 HOLDRQ によって各プロセッサにホールドを要求する．各プロセッサのホールド承認は信号線 HOLDA によって集計され，FF_4 がセットされると，その出力によってデータ交換開始が指令される．DEX コントローラは，アドレス信号とメモリ制御信号をバスBに乗せて発信する．$CL_H, CL_1, \cdots$ 内の R/W コントロールによって，通信用メモリ CM_i へのデータの読出しと書込みが制御され，前述のデータ交換が1語あたり50 nsで実行される．データ交換が完了すると FF_3 がリセットされ，これをモノステイブル MM で検出して FF_1 をリセットし，初期状態に帰る．DEX が発信されて FF_1 がセットされた後，まだデータ交換が完了する以前にプロセッサが STEP 信号を発すると，FF_2 がセットされ，そのプロセッサは HOLD がかけられ休止する（ホストコンピュータのみは，FF_2 によって割込みがかかるよう変更されている）．FF_2 も FF_1 と同時にリセットされる．データ交換完了後に STEP 信号を発信しても，FF_1 がリセット状態にあるので無視され，プロセッサは HOLD 状態に入らず次のステップの命令実行に移る．

なお，プロセッサは，データ交換制御命令として，このほかにコンピュータがデータ交換に参加しないこ

とを宣言する命令 ISOL を持つ．図8には省略したが，プロセッサが ISOL 命令を実行すると，ISOL 信号が発せられ，以後，DEX またはリセット信号を発するまで，そのコンピュータはデータ交換には参加せず，バスBから切離されて独自の仕事を行うことができる．

4.3 計 算 例

試作システムは，現在，5台のコンピュータが実装されている．試作システムを用い種々の問題をプログラムして，パフォーマンスを調べた．ここでは，線形連立方程式の解法と，FFT 計算の例を示す．

線形連立方程式を4台のコンピュータで反復法で解いた（計算アルゴリズムは文献6)による)．このときの計算時間の同じ計算を1台のコンピュータで行った計算時間に対する比 Reff を，反復の回数に対してプロットしたものが図8である．方程式の係数および初期値によって多少計算時間が異なるが，その平均値が示してある．4元連立方程式，反復回数 29 回の例では，1台での計算時間 825.01 ms に対して，4台では 216.95 ms となり，1/3.80 となった．このときのデータ交換時間の合計は 0.11 ms，データ交換待ちのための全コンピュータの停止時間の合計は最大 3.10 ms であった．反復回数が増すにしたがって，各コンピュータへのデータの割付け集計などの前処理，後処理の比率が実質的な計算時間に対して小さくなり，Reff が 1/4 に近づいていくことが確かめられた．

FFT の計算では，256 点の複素数データに対し，同じくコンピュータ4台による並列計算で，1語 32 ビット浮動小数点計算の場合 3.235 sec，1語 16 ビット固定小数点計算の場合 220 ms の計算時間を得た．いずれも1台の場合の 1/4 となっており，16台を実装したときにはそれぞれ 809 ms，55 ms の計算時間になると予想される．

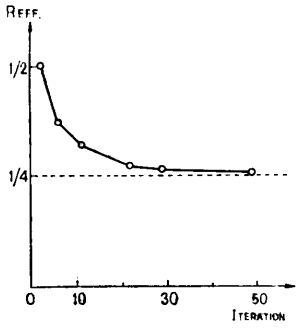


図 8 試作システムを用い，4元連立方程式を反復法で解いたときの反復数に対する計算時間の短縮率

Fig. 8 Computation-time decrease-rate as a function of the number of iteration steps. A solution was calculated for a 4-dimensional linear equation.

5. あ と が き

単一バス同期データ交換型マルチコンピュータシステムは，基本的にハードウェアが簡単であり，また，ジョブの進行状態がプログラマに完全に把握できるため，プログラミングが容易であり，さらに，データ交換に際してデッドロックが生じ得ないという特徴がある．ただし，これまでのシステムでは，一般的な問題に対してはコンピュータの停止時間が長くなるという欠点があった．本論文では，この欠点を，(1)データ交換用に高速メモリと高速コントローラを用い，(2)データ交換とタスク処理の並列進行を可能にすることで改良した高能率化方式を提案した．また，本方式に基づいて設計，試作したマルチマイクロコンピュータシステムについて報告した．これにより，頻繁なデータ交換を必要とする諸問題が能率よく処理できるようになった．

この方式の問題点の一つは，全体としてのメモリの使用量が大きくなることである．共有メモリを持たないため，関数計算サブルーチンなどの各コンピュータが共通して用いるプログラムも，すべてのコンピュータのメモリに格納しておかなければならない．また，本方式はすべてのコンピュータを比較的堅く結合させる方式であり，すべてのコンピュータを同期して動作させない場合には不便である．たとえば，全メンバコンピュータを複数個のグループに分割し，それぞれ独立したタスクを分割処理させたい場合がその一例である．任意のメンバコンピュータ間の割込みを可能とするハードウェアの用意については，なお検討を続けたいと考えている．

参 考 文 献

1) Enslow, P. H., Ed.: Multiprocessors and Parallel Processing, New York: Wiley-Interscience (1974).
2) Baer, J. L.: Multiprocessing Systems, IEEE Trans. Compt., Vol. C-25, pp. 1271-1277 (1976).
3) Fuller, S. H., et al.: Multi-Microprocessors: An Overview and Working Example, Proc. IEEE, Vol. 66, pp. 216-228 (1978).
4) Kopp, H.: Numerical Weather Forecast with the Multimicroprocessor System SMS 201, in Parallel Computers-Parallel Mathematics, ed.

M. Failmeier, pp. 265-268, North Holl (1977).
5) Kober, R., et al.: SMS101-Structured Multi-Microcomputer Systems with Deadlock-Free Operation Scheme, Euromicro Newsletters, pp. 56-64, April (1976).
6) Conard, V. and Wallach, Y.: Iterative Solution of Linear Equations on a Parallel Processor System, IEEE Trans. Compt., Vol. C-26, pp. 838-847 (1977).

(昭和53年10月24日受付)
(昭和54年 1 月18日採録)