2P−1　ソート機能をもった機能メモリ型並列プロセッサＦＭＰＰ

渡辺章弘　安浦寛人　田丸啓吉
京都大学工学部

1．まえがき

逐次処理による処理速度の限界を打破するために，並列処理に関する研究は古くから行われてきている．従来集積度の点で実用化されなかったアーキテクチャが，急速な進歩を遂げてきた集積回路技術を用いて実現されるようになってきた．また，メモリ回路はその時期の集積技術の指標として用いられているように最も集積度の高いアーキテクチャであるが，これはメモリのもつ規則性によるところが大きい．

我々は，メモリのもつ規則的な構造が並列処理アーキテクチャの処理効率向上に有効であると考えている．　本論文ではこの考えに基づき，ソーティング機能をもつ機能メモリ型並列処理アーキテクチャを提案しその性能について考察する．このアーキテクチャはホスト計算機からはメモリがソート機能をもっているように見える．

2．機能メモリ型並列プロセッサＦＭＰＰ

論理機能をメモリブロック内に分散したメモリは一般に機能メモリと呼ばれている．メモリ内に記憶と論理を分散した機能メモリは本質的に並列性をもっている．我々はこれらをプロセッサの集合とみなし，機能メモリ型並列プロセッサＦＭＰＰ(Functional Memory type Parallel Processors architecture と呼ぶ．　ＦＭＰＰはメモリであることを基本概念としているが，高い並列度をもつ SIMD 型の並列アーキテクチャとみることもできる．各プロセッサのもつ機能は必要最小限に抑え，メモリのもつ規則的な構造を維持することで高並列度を得ることができる．ＦＭＰＰの応用として，組合せ最適化[1]や画像処理[2]などが考えられている．

3．並列計数ソート法

ソーティングは実用的に非常に有用な問題であり，そのアルゴリズムや専用ハードウェアの研究は古くから多くの研究者によって行われてきた．また現在でもデータベース等において利用が高まっており，その高速化は今なお重要な課題である[3]．

ソーティングには順序づけと並べ換えがあり，実際多くのソート法はこの両方を行っているものが多い[3]．しかし，ソートが行われる状況によってはどちらか一方でよい場合もある．計数ソート法(Enumeration Sort)[4]は順序付けだけを行い，並べ換え即ちデータの移動を伴わない．そのアルゴリズムを図１に示す．図１に於て N はデータ要素数，D[1..N] はソート対象，C[1..N] が各要素の順序である．

```
procedure SORT(var D: array[1..N] of integer;
                  C: array[1..N] of inteher;)
  var
    i of integer;
    j of integer;
      begin
        for i:=1 to N do C[i]:=0; /* 初期化 */
        for i:=1 to N do          /* 比較及びカウント */
          for j:=1 to N do
            if D[j] > D[i] then C[j]:=C[j]+1;
      end
```

図１　逐次型計数法

このソート法は逐次型処理では O［N^2］の計算量であり効率は悪い．しかし，内側のループ内の比較及びカウントはそれぞれ独立であり，並列に行うことができる．このことを利用して O［N］ の計算量でソートを行える並列計数法[5]が考案されている．

本アーキテクチャでも並列計数法を用いる．　１ワードのメモリセルをデータ部とカウンタ部に分け，データ部にソート対象を入力する．（図２）ソート終了後にはカウンタ部が順位を示している．

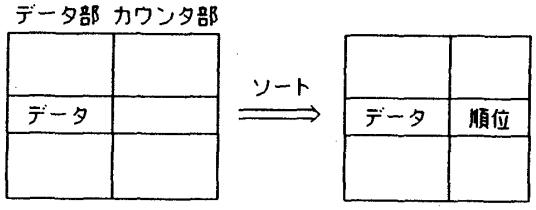


図２　ＦＭＰＰによる並列計数法

4．ＦＭＰＰ上での並列計数ソートの実現

4．1．ハードウェア構成

ＦＭＰＰの全体図を図３に示す．これはＣＡＭ(Content Addressable Memory)を基にしており１ワードを１プロセッサとみなす．またワード並列を基本とし SIMD 的動作をする．各ワードはビットセル列とフラグ部からなり，このフラグが各プロセッサを局所的に制御する．

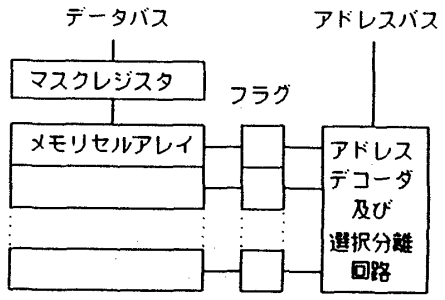


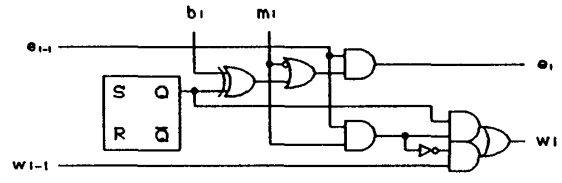
図３　ＦＭＰＰの構成

計数法の主な演算は大小比較とカウントである．ＣＡＭのマスク付一致検索，マスク付並列書込み機能だけでも並列計数法は行うことができるが，大小比較及びカウントはビット直列な処理となるためO［NlogN］となる．

マスク付大小比較及びカウントをビット並列に行う回路を図４に示す．大小比較信号線は一致信号線とともにフラグ部へ送られる．また，カウンタ部は桁上げ伝播による遅延を抑えるため，１ビットあたり２つのメモリセルを使用した桁上げ保存型になっている．桁上げ伝播はカウンタ部の最下位ビットへの桁

Functional Memory type Parallel Processors FMPP with Sorting function
Akihiro WATANABE, Hiroto YASUURA, Keikichi TAMARU
Kyoto University

上げ入力が0になった後，ビット幅分のサイクルで終了する．
このカウンタは通常の加算への拡張が可能である．
これらの機能は既存のＣＡＭの構造を大きく崩すことはない．
各ワードにワード方向の配線が一本増える程度である．



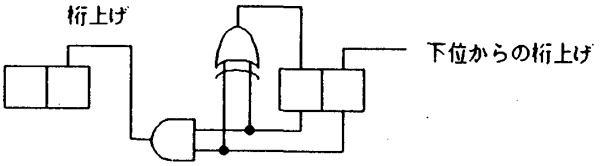


図4　マスク付大小比較回路とカウンタ回路

4．2．命令セット

本アーキテクチャの命令セットを定義する．ここではソーティングに必要な命令だけに限定する．

1)マスクレジスタへの書込み：MASKSET <data>
　マスクレジスタに data を書込む．
　マスクレジスタが0のビットは，2)REF，3)WRITESの対象とならない．

2)検索：REF function <data>
　各プロセッサのデータと data との比較を行い，function で示される結果に該当するプロセッサのフラグがセットされる．function として equal（一致），greater（大），less（小）がある．

3)選択プロセッサ並列書込み：WRITES <data>
　フラグがセットされているプロセッサにデータを書込む．

4)選択プロセッサのカウントアップ：COUNTS
　フラグがセットされているプロセッサをカウントアップする．

5)ノーオペレーション：NOP
　何もしない．（カウンタの桁上げ伝播だけが行われる）

6)アドレス指定による読出し：READA <address>
　通常のアドレス指定によるデータ読出しを行う．

4．3プログラム

前節で定義した命令セットを用いて，昇順にソーティングを行うプログラムを図5に示す．ここでソート対象のデータは既にメモリに入力されているとする．図5では，N はデータ数，M はカウンタ部のビット幅，any は任意のデータをあらわす．また，'<' と '>' の間の ',' はデータ部とカウンタ部の区切りである．

```
procedure SORT
  var
    data of integer;
    i of integer;
  begin
    MASKSET <0..0,0..0>;
    REF equal <any>;              /* 全ﾌﾟﾛｾｯｻ選択 */
    MASKSET <0..0,1..1>;
    WRITES <any,0..0>;            /* ｶｳﾝﾀ部ｸﾘｱ */
    for i:=1 to N do
      begin
        data:=READA <i>;          /* i 番目のﾃﾞｰﾀ読出し */
        MASKSET <1..1,0..0>;      /* ｶｳﾝﾀ部ﾏｽｸ */
        REF greater <data>;       /* data より大きいﾃﾞｰﾀ
                                     をもつﾌﾟﾛｾｯｻを選択 */
        COUNTS;                   /* ｶｳﾝﾄｱｯﾌﾟ */
      end
    for i:=1 to M-1 do NOP;       /* 桁上げ伝播 */
  end
```

図5　ＦＭＰＰによる並列計数法のプログラム

5．応用と性能評価

4．で述べたハードウェア及びアルゴリズムで得られた結果は，データ部・カウンタ部ともに一致検索の対象とした場合，次のように柔軟な利用が可能である．

1)i を入力として i 番目のデータをの出力する．
　データ部をマスクし，カウンタ部に対して一致検索を行い，選択プロセッサの読出しを行う．昇順・降順に順次読み出すことも，これと同様である．

2)データ d を入力として，その順位を出力する．
　カウンタ部をマスクし，データ部に対して一致検索を行い，選択プロセッサの読出しを行う．

また，データ部を複数のフィールドに分割しマスクつきの大小比較を行えば，データの一部分だけに注目したソーティングが可能である．

次に本アーキテクチャの速度について評価を行う．ＣＡＭチップとして実際に試作されたもの[6]の値を参考にして，4．2で定義した各命令が全て 200(ns)（5(MHz)）で動作すると仮定し，この場合データ数 Nに対して N プロセッサのＦＭＰＰを用いるとするとカウンタ部は logN ビットとなるから，図5のプログラムは 4+4*N+(logN-1)=4*N+logN+3 命令で終了する．よって N=1024 のとき 4109*200(ns)=8228(us)でソーティングが行える．

6．まとめ

本論文ではソーティング機能をもった機能メモリ型並列処理プロセッサＦＭＰＰを提案した．通常の計算機の主記憶の一部をこのようなＦＭＰＰで置き換えることによって，処理効率の大幅な改善がはかれることがわかった．今後，更に下位レベルの設計を行い，速度やハードウェア量のより正確な評価を行う必要がある．また，ソーティング以外の機能について考察してみる必要がある．

文献

[1]安浦 寛人，辻本 泰造，田丸 啓吉："組合せ問題に対する機能メモリ形並列プロセッサアーキテクチャ"，信学論，Vol.J72-A No.2（平1）
[2]A.Nakano,H.Yasuura and K.Tamaru:"Functional Memory Type Parallel Architecture for Image Processing",VLSI89
[3]Clark D.Thompson:"Hardware Algorithms for Sortng -The VLSI Complexity of Sorting",1983 IEEE
[4]Donald E.Knuth:"THE ART OF COMPUTER PROGRAMMING Volume 3/Sorting and Searching",1973
[5]安浦 寛人，高木 直史："並列計数法による高速ソーティング"，信学論，Vol.J65-D No.2
[6]小倉 武，山田 慎一郎，山田 順三，長沼 次郎：20Kb CAM(Content Addressable Memory)LSI，信学技報CPSY87-33(1988)