計算機アーキテクチャ 95−17
(1992. 8. 20)

SIMD 型超並列計算機 SM-1(仮称) の概要

松田元彦
住友金属工業 (株)

湯淺太一
豊橋技術科学大学情報工学系

概要

開発中の SIMD 型超並列計算機 SM-1 (仮称) の概要を報告する. SM-1 は SIMD 型計算機による処理が確立されている分野においては実用に供し, その他記号処理等の分野においては研究および開発に適することを目標として開発されたものである.

SM-1 は 1024 個以上の PE (Processing Element) より構成されている. 各 PE は RISC ライクなアーキテクチャであり, 大容量のレジスタと局所メモリを持っている. 全 PE はフロントエンド (FE) であるワークステーションのコプロセッサとして実装されている.

An Overview of the SIMD parallel processor SM-1

Motohiko Matuda
Sumitomo Metal Industries, ltd.

Taiichi Yuasa
Toyohashi University of Technology

abstract

This paper presents a SIMD parallel processor SM-1. Our design goals include that SM-1 should be usable in those fields where SIMD architecture has been proved to be applicable, and that SM-1 should be suitable for research in other fields such as symbolic processing.

SM-1 has at least 1024 processing elements (PE). Each PE is a RISC-like processor with a large number of registers and large local-memory. The array of PEs is connected to the front-end workstation as a co-processor.

1 はじめに

SIMD 型超並列計算機 SM-1 (仮称) は次の 3 点を目標として開発された.

- 数値計算や画像処理等, SIMD 型計算機による処理方式が確立している応用分野においては, 直ちに実用に供すること.
- その他記号処理等の分野においては, SIMD 型計算機による処理方式の研究および開発に適すること.
- SIMD 型計算機そのものの研究に適すること.

このように, SM-1 は実験的性格の強い計算機であり, その目標を達成するために次の特徴を備えている.

大容量のレジスタとメモリ 各 PE は大容量の局所メモリとレジスタ・ファイルを持つ. 大容量のレジスタと RISC アーキテクチャの命令セットを基本にすることで, レジスタ・アロケーション等のコンパイラ技術を SIMD 型超並列計算機へ応用することが可能となる. また, 大容量の局所メモリは, 実用規模のアプリケーションに必要であるとともに, 「仮想プロセッサ」等の実験に不可欠である.

局所メモリの間接参照 個々の PE に自律性を持たせるために, PE ごとにレジスタ・ファイルと局所メモリのアドレスを生成できる機能を持つ. これは, SIMD 型並列計算機の有効利用に役立つとともに, 記号処理等の応用には欠かせない機能である.

ソフトウェアによる通信制御 PE 間通信はハードウェアによる固定した処理ではなく, ソフトウェアによって制御される. これによって, PE 間通信の速度性能には限界が生じるものの, さまざまな PE 間通信ネットワークの実験がソフトウェア・レベルで可能となる.

マイクロ・コードによる柔軟性 各 PE は 8 bit 演算を行う ALU を持ち, FE (フロントエンド) からのマイクロ命令で制御される. 通常のマクロ命令セットは, FE と同様の 32 bit RISC アーキテクチャを基本としておりプログラミングが容易である. 一方, 特殊なアプリケーションに対してはマイクロ・コードの変更により柔軟に対応できる.

FE と PE の密な結合 全 PE は FE のコプロセッサとして接続する. これにより FE から PE への命令発行や局所メモリのアドレス生成が高速に行えると同時に, FE と PE 間のデータ転送がレジスタ間転送として容易に記述できる. さらに対話型の言語処理系の効率の良い実現が可能である.

以下では, システム構成, PE アーキテクチャ, PE 間結合および FE-PE 間結合, 実装の概要, およびプログラミング環境を報告し, 最後に性能ついて触れる.

2 システム構成

SM-1 の現バージョンは 1024 個の PE を PE アレイとして持つ分散メモリの SIMD 型並列計算機である. PE アレイは RISC CPU を使用したワークステーション (現在は SPARCstation) を FE として, そこにバックエンドとして接続される.

PE アレイはマイクロ・コード化されたコントローラを持っており, FE からの命令を解釈・実行する. コントローラは PE に対して, RISC ライクな 32 bit 整数演算, 浮動小数点演算等をマイクロ・コードにより実装する.

FE と PE アレイは, FE のシステム・バスとコプロセッサ・インタフェイスを介して接続されている. コプロセッサ・インタフェイスは, FE の CPU において浮動小数点演算プロセッサ等を接続するためのものである. PE アレイの起動は FE の CPU においてコプロセッサ用として予約されている命令群を使用して行う.

3 PE アーキテクチャ

3.1 PE 演算

PE アレイのマクロ命令セットはマイクロ・コードで実装されており, マイクロ・コードの変更で命令セットを大幅に変更することが可能である. 以下では標準のマクロ命令セットについて述べる. このマクロ命令セットは汎用の 32 bit 演算を実装しており, 並列化した C 言語 (6.1 節参照) による記述を仮定したものである.

PE アレイで実行する演算命令は FE で使用する RISC CPU の命令セットを並列化したものである. 演算は 32 bit 3 オペランド形式の整数演算および浮動小数点演算を基本としている. また, FE に既存のコンパイラを容易に拡張・移植できるように, 演算に関しては FE に使用している SPARC CPU と同様の命令セットを採用している.

たとえば, SPARC の加算命令は

```
add %r1, %r2, %r3
```

と記述される (%r1 等はレジスタ指定) のに対して, PE における加算命令は

```
add %p1, %p2, %p3
```

と記述される (%p1 等は各 PE のレジスタ指定).

レジスタ・セットは PE ごとに 32 個の 32 bit レジスタが用意されている (図 1 参照). それぞれのレジスタは汎用ですべて同じように使用できる. SPARC では整数用と浮動小数点数用のレジスタが別のものであるが, PE のレジスタ・セットにはその区別はない.

3.2 局所メモリ参照

PE による局所メモリの参照はコプロセッサのロード・ストア命令として実装される. メモリ参照のデータ

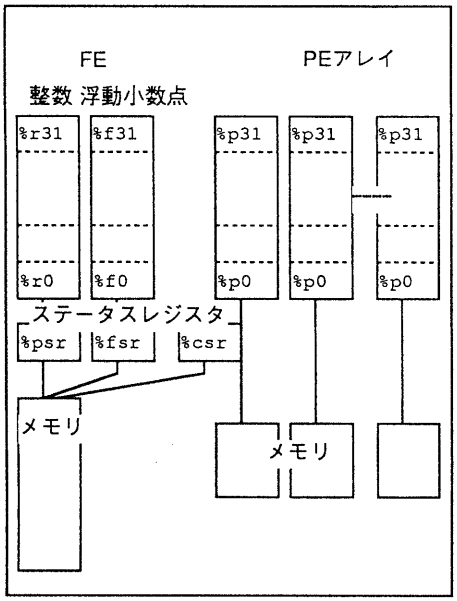


図 1: レジスタ・セット

幅はハードウェアが 32 bit 単位で行うので 32 bit の参照命令のみであり, バイト (8 bit) やハーフ・ワード (16 bit) のメモリ参照命令は標準のマクロ命令セットでは用意していない. その代わりに, バイトとハーフ・ワードのフィールドを扱う命令が用意されている.

メモリ参照にはいくつかのモードがある (図 2 参照).

PEM モード このモードは FE が PE 用のアドレスを生成し, すべての PE が同じアドレスの局所メモリを参照する. これは SIMD では最も頻繁に使われる参照モードである.

FE の仮想アドレス空間上に PE の局所メモリ用の領域 (物理メモリはない) をあらかじめ確保しておく. FE がその領域内のアドレスを指定するコプロセッサ命令を発行すると, PE アレイのコントローラは FE の生成したアドレスから局所メモリのアドレスを生成する. 各 PE は生成されたアドレスにしたがってそれぞれのメモリの内容を参照する.

ループのインデックス等は FE のレジスタに蓄えられているのが普通である. 上の方法により FE のレジスタを使ったレジスタ相対アドレスの計算を高速に行なうことができる.

FEM モード このモードは FE のメモリの内容を各 PE へブロードキャストするのに用いられる. FE の仮想アドレス空間で, 上記の PEM モードに使用される領域外をアクセスすると, そのメモリ (物理メモリがあると仮定されている) の内容がすべての PE にブロードキャストされる. FEM モードは PE へのロード命令のみ意味を持つ.

PEM モードの参照と FEM モードの参照は同一の命令であり, 各モードは参照するアドレスによって区別される. 各 PE に割り当てられる記憶域は PEM モードに指定されるアドレス領域に割り当て, FE がアクセスする逐次用の記憶域はそれ以外のアドレス領域に割り当てることで, 自然に参照のモードが決定づけられる.

間接 PEM モード このモードは各 PE がそれぞれのレジスタの内容によってアドレスを生成するものである. リストたどり等のポインタの並列操作は記号処理を行ううえで必須であり, 各 PE はそれぞれのレジスタの内容に従って独立に生成したアドレスによって局所メモリを参照するハードウェアを持っている.

3.3 アクティビティ

SIMD における実行制御としては, 発行される命令に対して実行する／しないを PE ごとのフラグによって指定する, アクティビティ制御の方法を採っている. アクティブな PE のみが命令を実行しその結果, 状態が変化する. この実行制御の方法は他の SIMD 型超並列計算機 [1, 2] でも採用されている.

SM-1 では, ネストした条件を処理するためにアクティビティを自動的にスタックに保存するような方法は採らず, レジスタ上に明示的に保存場所を割り付ける方法を採っている. これは割り当てをコンパイラに任せた方が柔軟に処理できると考えられるためである.

アクティビティを制御する命令には次のものがある.

jump 命令 jump 命令は, 条件, レジスタ番号, レジスタ中の bit 位置を指定して実行される. 実行時点でアクティブかつ演算結果のフラグによって与えられる条件が成立する PE は, 指定されたレジスタの bit をセットした後アクティブでなくなる.

label 命令 label 命令は, レジスタ番号とレジスタ中の bit 位置を指定して実行される. この命令は, すべての PE により (アクティビティに関わらず) 実行される. 実行時点でアクティブでない PE のうち, 指定されたレジスタの指定された bit がセットされている PE はアクティブになる. 同時に label 命令は指定された bit を再利用可能なようにリセットする.

すなわち, jump 命令でアクティブでなくなった PE は対応する (同一のレジスタ番号と bit 位置を指定した) label 命令の実行によりアクティブな状態に戻る. この結果, jump 命令の条件が成立する PE は jump 命令とその対応する label 命令の間のコードをスキップすることになる (図 3 参照).

jump 命令の条件としては, FE のブランチ条件 (整数演算および浮動小数点演算について) と同様なものが用意されている.

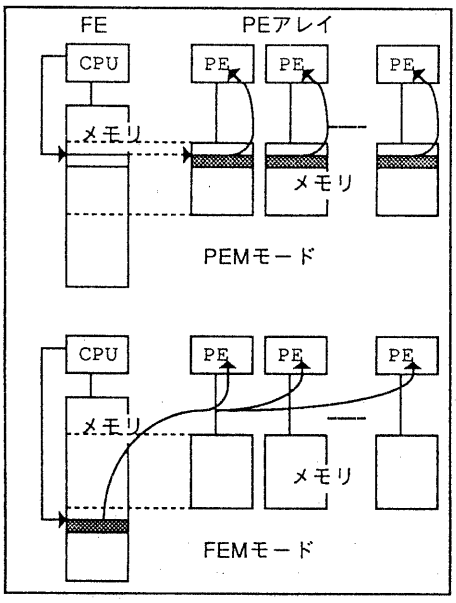


図 2: ロード・ストアの参照モード

アクティビティに関するその他の命令 特別な場合として, 通信などのグローバルな処理では現在のアクティビティの状態によらずすべての PE をアクティブにしたいことがある. この処理のために, catch 命令, throw 命令, doall 命令がある. catch 命令は実行時点でのアクティビティを指定されたレジスタの bit に保存し, すべての PE をアクティブにする. throw 命令は対応する catch 命令で保存したアクティビティに戻す. doall 命令はレジスタを変更しない catch 命令で, デバッグ等に使用する.

4 コネクション

PE 間通信のために 32 bit と 8 bit の通信命令を持っている. 通信には 1 対 1 のものと, バス構成のものがある. バス構成のものは, 同一バスに送られたすべてのデータの (inclusive) OR がとられる.

PE 間通信のために次のハードウェア・ネットワークを持っている.

- 2 次元メッシュ (PE 数 1024 の場合 32 × 32),
- 縦・横の OR バス (縦バス 32 本, 横バス 32 本),
- 全 PE の OR バス (システム全体で 1 本),
- シャッフル・エクスチェンジ・ネットワーク.

ネットワークのルーティング等の複雑で実験の対象になる処理は並列化 C 言語により記述を行うこととしたので, PE アレイのコントローラはプリミティブな処理しか行わず, 命令セットもプリミティブなもののみを提供している.

subcc	%p0,3,%p2	レジスタ p0 から 3 を引いて p2 に入れる
jl	%p31,0	結果が 0 より小さい PE はインアクティブ ···(1)
	⋮	p0 ≥ 3 なら実行
ja	%p31,1	全 PE をインアクティブにする ···(2)
label	%p31,0	(1) でインアクティブになった PE をアクティブにする
	⋮	p0 < 3 なら実行
label	%p31,1	(2) でインアクティブになった PE をアクティブにする

図 3: アクティビティ制御命令の例

4.1 メッシュ

メッシュは, 4 近傍接続でエッジの処理をいくつかのオプションの中から動的に変更することができる. 可能なオプションには次のものがある:

- エッジが閉じたトーラスの構成.
- エッジが開いた状態で全 bit 0 の定数が入力される構成.
- エッジが開いた状態で全 bit 1 の定数が入力される構成.
- 1 次元のらせん状 (あるいは環状) の構成.

4.2 縦バス, 横バス

縦バス, 横バスは 32 × 32 の格子の縦方向と横方向の PE 32 個ずつを (inclusive) OR で接続したものである. それぞれの PE はバスに値を出力すると同時に値を入力することができる. これらのバスを使って, 同一列あるいは同一行に位置する PE の値 (32 bit あるいは 8 bit) の論理和/積や (符号あり/なしの) 最大/最小値を求めるリダクション処理を行うことができる. これらのバスは行列計算において, ある行の行列全体へのコピーなどに使用できる.

4.3 OR バス

全 PE をつなぐ OR バスは, すべての横バスをさらに OR で接続することにより構成されている. 縦・横バスと同じ命令が, 全 PE の OR バスに対しても定義されている.

OR バスの出力は PE アレイのコントローラにも入力されており, 全 PE に対するリダクションの結果をコプロセッサ・インタフェイスを通して FE に送ることができる.

4.4 シャッフル・エクスチェンジ

シャッフル・エクスチェンジ・ネットワークは応用アルゴリズムが多く知られていること [3] と, ルーティングをソフトウェアで実装することを考慮して採用された. グローバルな PE 間通信のルーティング等はシャッフル・エクスチェンジを使用して実現可能である. ハードウェアはシャッフルと逆シャッフルを実装している.

4.5 FE との結合

FE と PE アレイはいくつかの方法で情報を交換することができる.

ブロードキャスト FEM モードを使用して, FE のメモリの内容を PE アレイにブロードキャストすることができる.

FE による PE の局所メモリの参照 FE は PE アレイの局所メモリをシステム・バスを介して読み書きすることが可能である. PE の局所メモリの FE の仮想メモリ空間へのマッピングには以下のモードがある:

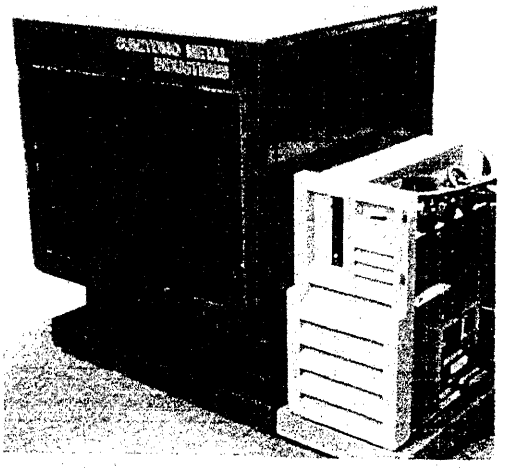
- ワード・モード: 各 PE の指定されたアドレスの 1 ワード (32 bit) が PE の数だけ並んで見える.
- ダブル・ワード・モード: 各 PE の指定されたアドレスの 1 ダブル・ワード (64 bit) が PE の数だけ並んで見える.
- **PE メモリ・モード**: 指定された 1 つの PE の局所メモリがすべて見える.

リダクション PE アレイから FE へは, OR バスによるリダクションを使って 32 bit のデータを送ることができる.

ブランチ条件 PE アレイから FE へは OR バスを使って 1 bit の条件を送ることができる. FE の CPU はその返却値をテストし, 条件ブランチすることができる. 1 bit を返却する命令には次のものが含まれる.

- すべての PE がアクティブか.
- 1 つでもアクティブな PE があるか.

FE から PE アレイへの命令発行 FE は PE アレイに対して, コプロセッサ・インタフェイスを使用して命令の発行を行う. コプロセッサ・インタフェイスにより FE の CPU と PE アレイの間で並列実行が可能となっている. インタフェイスには命令キューがあり, PE アレイが直ちに実行できない命令はキューに保持される. 現在の実装では命令キューは最大 4 命令を保持することができ, それを越える要求に対しては, FE の CPU はホールドされる.



高さ	850 mm
幅	770 mm
奥行き	675 mm

図 4: SM-1 の外観

FE と PE アレイ間の同期 FE と PE アレイは並列に動作しているので, リダクションの結果を参照したり条件ブランチを行おうとする場合, FE はデータが得られるまで待たなければならない. この同期はコプロセッサ・インタフェイスによって FE がホールドされるので, 自動的に行われる.

5 実装

5.1 外観

SM-1 の外観を図 4 に示す. 右にあるのは FE で, 左にあるのが PE アレイ部である. PE アレイ部には, 18 枚のユーロ規格 9U のボード (PE ボード 16 枚, コントローラ・ボード 1 枚, メッシュ端その他のボード 1 枚) が収まっている.

フロント・パネルには, 1024 個の LED が付いている. LED は各 PE のエクスチェンジ出力につながっている. LED を現在のアクティビティにしたがって点滅させる命令が用意されている.

5.2 PE チップ

PE は 8 bit の ALU で構成されており, 4 個の PE を 1 個のゲートアレイに実装している. PE は独自の命令パスは持っておらず, コントローラ上のマイクロコード・シーケンサによって制御される. チップのゲート規模は約 60K ゲートであり, クロックは 20 MHz で動作している.

ALU それぞれの PE は 8 bit の ALU, 20 byte の 3 ポート・レジスタ, 64 bit のシフト・レジスタ, 160 byte の 1 ポート RAM よりなる. この RAM はマクロ命令セットで使用するユーザ用のレジスタとなっている. PE 内の 8 bit のデータパスは 2 段のパイプラインになっている.

また, 8 bit のデータパスとは独立に 1 bit のデータパスを持っている. このデータパスは, 1 bit の演算器 BLU (Bit-wise Logic Unit) と, 32 個の 1 bit レジスタ (ALU からのフラグを含む) より構成されており, フラグ操作, アクティビティ操作, 通信等を担当する.

8 bit の ALU, 1 bit の BLU, そして 64 bit のシフト・レジスタは並列に動作し柔軟な処理が可能になっている.

8 bit ALU は加減算と論理演算を行う. さらに 8 bit ALU には, 8 bit の乗算器が付加されている. これは 32 bit 整数や浮動小数点数の演算に使用されるのはもちろんであるが, 画像処理等をマイクロ・コードで記述する際にも有効である. この乗算器はゲートアレイの実装のため演算結果が得られるのに 2 clock を必要とする.

レジスタ マクロ命令から見えるユーザ用の 32 本の 32 bit レジスタはゲートアレイの 160 byte の 1 ポート RAM によって実装される (128 byte がユーザ用で, 残りの 48 byte はマイクロ・コードで使用する). この RAM は 8 bit ALU と局所メモリのインタフェイスからアクセスされる.

マイクロ・レベルの自律動作 マイクロ・コード・レベルでも PE の実行制御はアクティビティにより行なわれている. マイクロ・コード・レベルでは, 32 個の 1 bit レジスタのうちの任意の 1 個をアクティビティ・フラグとして指定でき, 細かい制御が可能となっている.

さらに, PE ごとの実行制御として, PE ごとに異なる演算を実行する機構を持っている. レジスタの 1 つ (UReg と呼ぶ) は 8 bit ALU の演算命令を保持することができるようになっている. PE ごとに 1 bit レジスタの条件にしたがって, コントローラからの命令か, あるいは UReg 内の命令かのどちらかを選択し実行するようにマイクロ・コードから指定できる.

この機能は, 例えば, 除算を 8 bit ALU で処理する際に PE ごとに加算か減算かをマイクロ・コードで選択し実行するのに使用されている.

5.3 局所メモリ

局所メモリは 1 PE 当たり 1M byte の容量を持っている. PE チップは 4 個の PE を実装しているので, 1 チップ当たり 4M byte の容量になる. チップとメモリとは 32 bit のデータ・バス (ECC 用の bit は除く) で接続されている. チップ上の 4 個の PE はバスを時分割で利用する. メモリは 1 bit エラー訂正, 2 bit エラー検出の ECC によって保護されている.

メモリ参照 メモリ・アドレスは, コントローラで生成される全 PE で共通な場合と, 各 PE ごとに生成される場合がある. 後者の場合のために PE チップはアドレス出力を持っている. チップの I/O pin 数の制約から, アドレス出力は 4 PE 分を一度に出力することは不可能なので, 時分割されている. これに従ってデータバスも 32 bit のバスを時分割で使用するようになっている. さらに, アドレス出力は DRAM へ直接接続するためと, チップの I/O pin 数を減らすためにマルチプレクスされている.

この構成により, 同一チップ上の PE の同一アドレスのメモリは, 実際には 4 PE 分が連続して 1 つの RAM に格納されている. したがって, PEM モードでは DRAM のページ・モードを使用して 4 ワードを連続して参照する.

メモリ・シーケンサ メモリ参照と, PE での演算は並列に行われる. コントローラは主マイクロ・コード・シーケンサとともに, メモリ用マイクロ・コード・シーケンサを持っている. メモリ・シーケンサは PE に対するロード・ストア, FE に対するロード・ストアと DMA, そして DRAM のリフレッシュを担当する.

主シーケンサとメモリ・シーケンサとはレジスタが競合する場合はインターロックする. 主シーケンサは, メモリからのロードが終了していないレジスタを参照する場合, メモリ・シーケンサが終了するまで待つ. メモリへのストアは, ライト・バッファを持っているのでインターロックしない.

5.4 通信ネットワーク

PE 間ネットワーク, および FE への接続である OR バスはすべて 1 bit のシリアル通信で実装されている. マイクロ・コードからは各ネットワークの接続は 1 bit レジスタの 1 つとして見えている. 通信はすべて 1 bit あたり 2 clock を使って行われるので, 通信速度は PE 当たり最高 10M bps である. システム全体では 10 G bps となる.

シャッフル・エクスチェンジの場合, シャッフルとエクスチェンジのどちらから読むかは 1 bit レジスタ中のフラグの値にしたがっているので, PE ごとに決めることができる.

6 プログラミング環境

SM-1 は SIMD 型計算機であり, 並列実行のための特殊な OS は必要でない. 並列プログラムの開発, デバッグ, 実行はすべて FE の OS である UNIX のもとに行われる. また, PE の制御命令は FE のコプロセッサ命令として実装されているので, リンカは UNIX に標準のものを使用している.

6.1 言語等

現在, アセンブラ代わりになるような低レベル記述が可能な C 言語 [4, 5] が利用可能である. この言語は, ANSI 標準の C 言語にデータ並列の拡張を行ったものである. また, SIMD 型超並列計算機能を拡張した Common Lisp[6, 7] の処理系を開発中である. この処理系はすでに MasPar[1] 上でクロス開発を終わっており, SM-1 に移行中である. さらに, Fortran 90 をベースにした Fortran のコンパイラ [8] を開発中である.

上記言語に対応したデバッガも実装中である. SM-1 は SIMD 型であり, PE アレイがコプロセッサとして実装されているため, 機械語レベルのデバッグにおいてもステップ動作等は従来のプロセッサと同様に行える.

6.2 プロファイリング

PE アレイはコプロセッサとして実装されているので, FE の UNIX に備わっているプロファイラが使用可能である.

また, FE が PE アレイとの同期のためにホールドされた時間はコプロセッサ・インタフェイス中にホールドのタイマを持っているので知ることができる.

7 性能

7.1 演算器性能

暫定版マイクロ・コードにおける基本演算に関する性能を表 1 に示す. また, メモリ参照に関する性能を表 2 に示す. 測定は演算を 500 万回連続して繰り返すことにより行った.

演算性能 32 bit 整数演算に関しては, 論理演算その他はほぼ加算と同じ性能である. 浮動小数点数に関しては IEEE のフォーマットに従っている.

局所メモリ参照 PEM モードは FE がアドレスを生成するもので, DRAM のページ・モードを使用しての参照である. FEM モードは FE からのブロードキャストであり, FE のメモリの内容を全 PE にコピーするものである. したがって, 局所メモリは参照しない. 間接モードは局所アドレスによるもので, 各 PE がレジスタの内容からアドレスを生成する. FE による参照は, PE のメモリを FE の仮想アドレス空間にマッピングするもので, FE がロード・ストアを行う.

表 1: 演算性能 (92 年 7 月現在)

32 bit 整数	
加算	860 K ops
乗算	210 K ops
除算	75 K ops
倍精度・浮動小数点数	
加算	63 K ops
乗算	54 K ops
除算	26 K ops

(ops = operation/sec)

表 2: メモリ性能 (92 年 7 月現在)

PEM モード	
ロード	880 K ops
ストア	860 K ops
FEM モード	
ロード	1400 K ops
ストア	なし
間接モード	
ロード	500 K ops
ストア	630 K ops
FE による参照	
ロード	330 K ops
ストア	380 K ops

(ops = operation/sec)

7.2 性能評価

32 bit 整数演算の性能が比較的低いのは, ゲートアレイでユーザ用のレジスタを実装している RAM ブロックが 1 ポート構成であることに起因している. 複数バンク構成等により簡単に改善が考えられるが, 設計の工数の都合上行われなかった.

8 おわりに

SIMD 型超並列計算機 SM-1 の概要を紹介した. 現在 2 台の SM-1 が稼働中であり, 言語処理系等の基本ソフトウェアの最終テストを行っている.

並列化 C 言語を使用して, 大容量のメモリを利用した文献データベースや科学技術計算分野の応用等のアプリケーションを開発中である. また, 並列化 Common Lisp を使用しての応用も検討中である.

9 謝辞

ハードウェアの設計に協力していただいた竹岡尚三氏 (AXE 社) に深謝します.

参考文献

[1] MasPar MP-1 Hardware Manuals, MasPar Computer Corp., 1990.

[2] Hillis, D.: The Connection Machine, MIT press, 1985.

[3] Quinn, M. J.: Designing Efficient Algorithms for Parallel Computers, McGraw-Hill, 1987.

[4] 貴島: 並列 C 言語 "並 C" 言語仕様書, 豊橋技術科学大学湯淺研究室, 1992.

[5] 野田: SIMD 型超並列プログラミング言語「並 C」ユーザーズ・ガイド, 豊橋技術科学大学湯淺研究室, 1992.

[6] Yuasa, T.: TUPLE: An Extended Common Lisp for Massively Parallel SIMD Architecture, In Proc. of the DPRI Symposium, 1992.

[7] Yuasa, T.: Memory Management and Garbage Collection of an Extended Common Lisp System for Massively Parallel SIMD Architecture, In Proc. of the International Workshop on Memory Management, 1992.

[8] 安村, 大谷, 渦原, 小前, 杉森: Bee-Fortran 仕様書, 慶応義塾大学安村研究室, 1992.