計算機アーキテクチャ 80－8
（1990．1．25）

PSI-IIのメモリ・アーキテクチャ評価

中島 浩　　武田 保孝

（三菱電機）

第5世代コンピュータ・プロジェクトの一環として，我々は逐次型推論マシンPSI-IIを開発した。PSI-IIの設計に当たっては，PSI-Iでの性能評価をもとにキャッシュ・メモリの容量を半減したが，PSI-IとPSI-IIでは実行方式にかなり差異があるため，設計の妥当性を完全に予測することができなかった。そこで，実用的かつ大規模なプログラムであるコンパイラを対象にPSI-IIのキャッシュ・メモリの動特性を評価した。その結果96%以上の高いヒット率が達成されていることや，容量半減の影響は軽微であることが明らかになった。

またこの評価データに基いて，現在開発中のPIM/m要素プロセッサのキャッシュ・メモリやアドレス変換バッファの構成を定めるためのシミュレーションを行った。その結果，命令キャッシュとデータ・キャッシュを分離するハーバード・アーキテクチャの採用や，アドレス変換バッファのエントリ・アドレスの生成方式など，様々な設計パラメータを決定することができた。

Evaluation of the PSI-II Memory Architecture

Hiroshi Nakashima　　Yasutaka Takeda

(Mitsubishi Electric Corp.)

In the Japanese Fifth Generation Computer Project, we developed a sequential inference machine, PSI-II. The cache memory size of PSI-II was reduced to half that of PSI-I, but it was expected that the performance degradation would not be serious. To confirm the prediction, we have evaluated the performance of the cache memory, tracing its behavior in the execution of ESP compiler. The results show that high hit ratio, over 96%, is achieved, and the performance degradation by the size reduction is negligible.

The evaluation data is also used to design the PIM/m element processor which is currently developed. Many architectural decisions, such as the adoption of Harvard architecture and the configuration of address translation buffers, are made by the simulation results.

1 はじめに

第5世代コンピュータ・プロジェクトの一環として，論理型言語ESPを実行する逐次型推論マシンPSI-IIを開発した[1]。PSI-IIは，プロジェクト初期に開発されたPSI-I[2]の後継機であり，PSI-Iの約1/6の実装規模で3~10倍の性能を持っている。

PSI-IIの設計に当っては，実装規模の縮小や性能向上のために大幅なアーキテクチャの変更を行った。その中で重要なものの一つが，キャッシュ・メモリの容量削減であった。PSI-Iは4Kw×2set (1w = 40 bit)のセット・アソシアティブ方式のキャッシュ・メモリを持っており，95%以上の高いヒット率が得られている。また，キャッシュ容量を1/2に削減し，かつダイレクト・マッピング方式とした場合にも，性能低下は1~3%に過ぎないこともシミュレーションによって確認された[3]。そこで，PSI-IIのキャッシュ・メモリは4Kwのダイレクト・マッピング方式により構成することとし，メモリ・チップの数を約60%削減することができた。

さて，PSI-IIではESPの処理方式を大幅に変更したため，PSI-Iでの評価結果がそのまま適用しうるか否かについて，若干の疑問が残っていた。即ち；

(1) 処理方式の変更によって主記憶アクセス特性が変化しないか。

(2) 性能向上に伴う主記憶アクセス頻度の相対的増加により，キャッシュ・ミスヒットの影響が大きくならないか。

(3) ダイレクト・マッピングへの変更によって，スラッシングが頻発しないか。

などの点が問題となっていた。

そこで今回，コンパイラという実用的かつ大規模なプログラムを対象に，キャッシュメモリの動特性を評価し，上記のような疑問に対する回答を得ることとした。更に，現在開発中の並列推論マシンPIM/mの要素プロセッサの設計のために，PSI-IIのメモリ・アーキテクチャを変更した場合の性能をシミュレーションにより評価した。

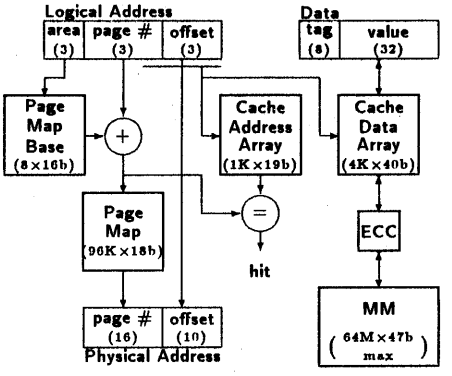

図1: PSI-IIのメモリ・システム

以下本稿では，PSI-IIのメモリ・アーキテクチャ，評価の方法と結果，及びPIM/mの設計のための評価について報告する。

2 PSI-IIのメモリ・アーキテクチャ

PSI-IIのキャッシュ・メモリ及びアドレス変換機構の構成を，図1に示す。論理アドレスは32ビット幅であり，3ビットのエリア番号，19ビットのページ番号，及び10ビットのオフセットから成っている。8つのエリアの内，エリア0~3はプロセス間で共有され，エリア4~7はプロセス固有の領域である。ESPの実行時には各エリアは以下のように使用される。

- エリア0： システム領域
- エリア1： コード及びヒープ†
- エリア2~4：未使用
- エリア5： グローバル・スタック
- エリア6： ローカル・スタック
- エリア7： トレイル・スタック

†書換可能なデータ・オブジェクトを保持する領域であり，WAMのHeap（グローバル・スタック）とは異なる。

一つのエリアに関するアドレス変換テーブルはPage Mapの連続領域に割付けられ，その先頭アドレスがエリアに対応するPage Map Baseに保持されている。

キャッシュ・メモリは4w/blockの構成であり，スワップ方式はストア・バックを用いている。また，スタックの伸長を伴うの書込のために，write stackという操作が用意されている。ブロックの先頭に対するwrite stackでは，スタック・トップを越えた領域のデータは無意味であるので，ミスヒット時にも主記憶からのブロック・ロードを行わない。従ってストア・バックが不要な場合には，ミスヒット時でもヒット時と同じように1サイクルでアクセスが完了する。

3 PSI-IIの評価

3.1 評価方法

キャッシュ・メモリの評価のためには，実行時間，コード量などが充分に大きいプログラムを対象とする必要がある。そこで評価対象プログラムとして，ESPコンパイラを採用した。ESPコンパイラは約50Kwのコード量を持つ大規模プログラムであり，またオブジェクト指向機能などESPの特性を生かした実用的なプログラムであるので，評価対象として適切であると考えた。

評価データの収集は，ESPコンパイラにappend及びquick sortをソース・プログラムとして与え，実行されたメモリ・アクセス・コマンドとそのアドレスをトレースすることにより行った。但し，データの収集と解析に要する時間を節約するために，全実行サイクル（約500Kサイクル）の中から160Kサイクルのスナップ・ショット（40Kサイクルを4ヶ所）を抽出して評価対象とした。

ヒット率などの測定は，収集したデータをキャッシュ・メモリ・シミュレータに与えることにより行った。なお，シミュレータにはキャッシュ・メモリの構成の変更や，コマンド／エリアごとのヒット率の測定などの機能があり，詳細な解析を行うことができた。

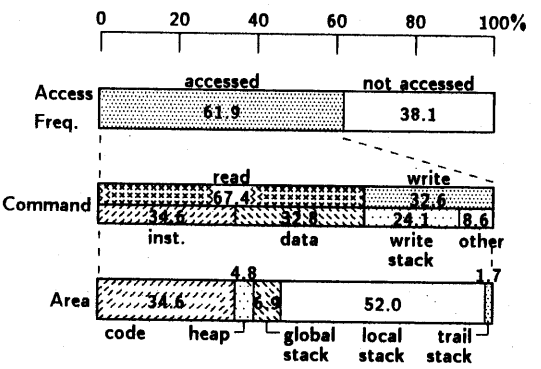

図2: アクセス頻度

3.2 アクセス頻度

図2は，キャッシュ・メモリのアクセス頻度を示したものである。また，コマンドの実行頻度とエリアごとのアクセス頻度も併せて示している。キャッシュ・メモリのアクセス頻度は60%を越えており，PSI-Iと比べると約3倍に増加している。この結果は，PSI-IとPSI-IIではメモリ・アクセス回数に大差はなく，性能が3倍以上に向上した分だけ相対的に頻度が増加したことを示している。

コマンドの実行頻度に関しては，命令の読出，データの読出，及びデータの書込の比率がほぼ1:1:1であり，他の評価結果[4]ともほとんど一致している。データのアクセスが命令フェッチを大きく上回っているのは，PSI-IIがWAMの命令をマイクロプログラムによってエミュレートするCISCであるためである。また，データの書込についてはwrite stackが約3/4を占めており，スタックを伸ばす操作が非常に多いことが明らかである。なお，これらの値に関してはPSI-Iとほぼ同様の結果であった。

エリアごとのアクセス頻度については，ローカル・スタックへのアクセス頻度が全体の約半分，データ・エリアの3/4を占めているのが特徴である。この値も[4]と一致しているが，グローバル・スタックのアクセス頻度がPSI-Iの5~30%に比べて減少傾向にある。これは，構造体の取扱いをstructure sharingからstructure copyingに変更したためと考えられる。

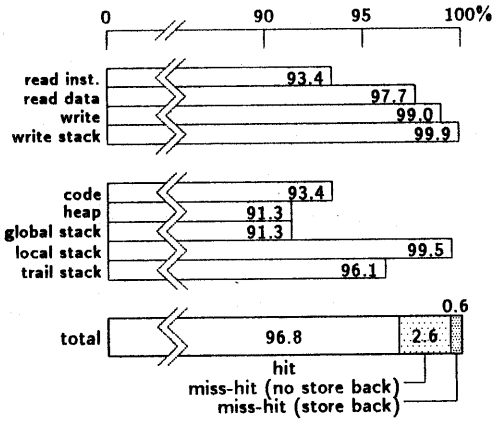

図 3: ヒット率

3.3 ヒット率

図3にコマンド／エリアごとのヒット率を示す。命令に比べてデータ・アクセスのヒット率が高く，特に書込でのヒット率が極めて高い。中でも，write stack のヒット率はほとんど100%であり，スタックの参照の局所性が極めて高いこと，即ちスタックが頻繁に細かく伸縮していることが判る。また，普通の書込のヒット率が高いのは，論理型言語では代入操作（未定義変数のバインド）に先立って，必ず変数セルの値を読出しているためである。

エリア別のヒット率については，ローカル・スタックのヒット率が非常に高く，実質的にローカル・スタック・バッファの役割を果たしていることが判る。全体のヒット率は96.8%と充分に高率である。またミスヒットの内，ストア・バックを伴うものは極めて少なく，ストア・バック方式の有効性が確かめられた。

3.4 容量半減の影響

キャッシュ容量半減の影響を調べるために，同じデータを用いて4Kw×2set のキャッシュ・メモリのヒット率と性能をシミュレートした。図4はPSI-II のキャッシュ・メモリと，4Kw×2set のキャッシュ・メモリを比較したものである。

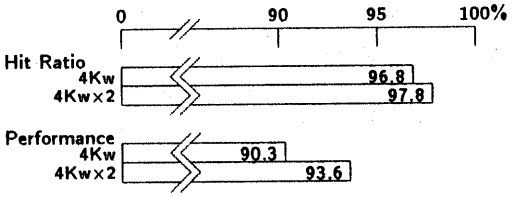

図 4: 容量削減の影響

なお，性能についてはヒット率が100% の時の性能を100 として正規化してある。ヒット率，性能とも差異は極めて小さく，容量削減が誤った選択ではなかったことが確認された。

なお，1% のヒット率の変化が3.6% の性能変化となっているが，この値はPSI-II の約5倍であり，アクセス頻度の増加によってヒット率が性能に与える影響が大きく変化したことが判る。

4 PIM/m 要素プロセッサの評価

PIM/m[5] は256 個の要素プロセッサを格子状に結合した並列推論マシンであり，既に開発した並列推論マシンMulti-PSI/v2[6] の後継機として位置付けられている。従ってPIM/m 要素プロセッサは，Multi-PSI/v2 の要素プロセッサであるPSI-II をベースとして設計され，並列論理型言語KL1[7] とESP の双方を実行する機能を持っている。

PIM/m のプロセッサ数はMulti-PSI/v2 の4 倍であるため，その要素プロセッサの実装規模を1/4 倍に削減する必要があった。また，性能面でも3 倍以上の向上を目標に設計を行った。そのため，メモリ・アーキテクチャについてもいくつかの重大な変更を加える必要があった。まず実装規模の削減のために，メモリ・システムを1μm のCMOS VLSI を中心に構成することとしたが，このVLSI チップに96K×18 ビットのPage Map を搭載するのは到底不可能である。またチップ外部に配置することも，ピン数，実装面積，アクセス時間などの問題から極めて困難であるため，小容量のアドレス変換バッファへの変更が不可避となった。ま

た，キャッシュ・メモリに関しても更に容量を削減できるかを検討する必要があった。

一方，性能面ではパイプラインの導入がメモリ・アーキテクチャに大きな影響を与えた。即ち，命令フェッチとデータ・アクセスが，パイプラインの異なるステージで並行して行われるため，アクセス競合が重大な問題となることが予測された。また，オペランド・フェッチのパイプライン化や，命令実行に要するサイクル数の減少も，アクセス競合の確率を大きく増加させると考えた。

そこで，

- 命令キャッシュとデータ・キャッシュの分離
- アドレス変換バッファの採用

の二つを行うことを決定し，それぞれの容量や構成方式をシミュレーションによって評価することとした。

4.1 キャッシュ容量

キャッシュ容量の決定に際しては，VLSIチップに搭載可能な容量で充分な性能が得られるか否かが最大の問題であった。そこで，キャッシュの容量を変化させてヒット率を測定したところ，図5に示す結果が得られた。即ち，命令キャッシュは1Kw，データ・キャッシュは2Kw程度までは，容量の対数にほぼ比例してヒット率が向上している。

一方，チップに搭載可能な容量は，命令／データ・キャッシュを合わせて1Kw内外と見積もられた。従って；

(a) 命令キャッシュ＝512w
　　データ・キャッシュ＝512w

(b) 命令キャッシュ＝256w
　　データ・キャッシュ＝1Kw

が考えられた。しかし，(a)の場合はデータ・キャッシュのヒット率が，(b)の場合は命令キャッシュのヒット率がかなり低くなる。ヒット率の低下による性能への悪影響は，アクセス頻度の増大や，主記憶とCPUの速度差の拡大によって，PSI-IIに比べてかなり増加すると考えられる。従って，これらの構成では充分な性能を得ることが難しいと判断した。

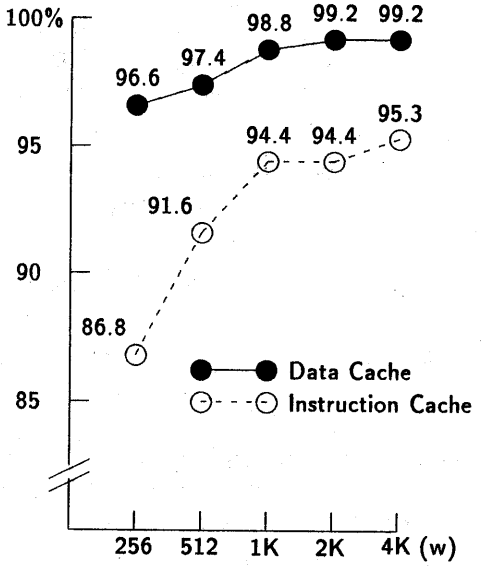

図5: 命令／データ・キャッシュのヒット率

そこで，データ・キャッシュのデータ・アレイをチップ外部に置くことを検討した結果，アクセス時間15nsの高速RAMを用いれば，チップ内部に搭載した場合と同等のマシン・サイクルが得られることが明らかになった。その結果，命令キャッシュは1Kw，データ・キャッシュは4Kwの容量とすることができ，充分に高いヒット率を実現することが可能となった。

4.2 アドレス変換バッファ

アドレス変換バッファのヒット率に関しては，ミスヒット時のペナルティがキャッシュ・メモリに比べてかなり大きいと予想されるため，極めて高い値であることが要求される。ことにPIM/m要素プロセッサではハードウェア設計の簡単化のために，ミスヒット時の処理をマイ

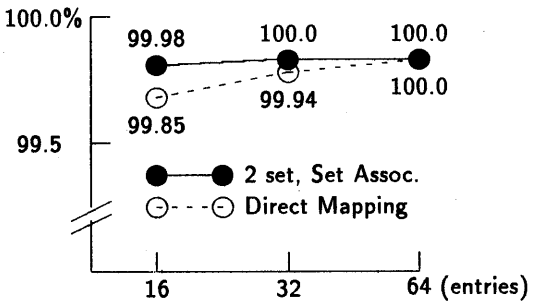

図 6: 命令用アドレス変換バッファのヒット率

クロプログラムで行うこととしたため，ヒット率には細心の注意が必要となった。そこで，アドレス変換バッファに関しては，容量だけではなく構成方式に関しても，いくつかのバリエーションを考えて評価を行った。

まず，命令用のアドレス変換バッファについては，ダイレクト・マッピング方式と2セットのセット・アソシアティブ方式の双方について，16~64エントリの場合のヒット率を測定した。その結果，図6に示すようにダイレクト・マッピングでは64エントリ，セット・アソシアティブでは32エントリで100%のヒット率が得られた。

なお，命令アドレスの分布を詳しく調べると，使用したページの内の約1/3はプログラムの本体から離れた領域にあった。このような分布を示したのは，システムで用意しているライブラリを共有コード方式で実現しているためである。この方式では本体とライブラリが連続領域を形成しないため，命令アドレスの分布が離散的になる傾向がある。従って，アドレス変換バッファの容量が，ライブラリを含めたプログラム・サイズを満たすものであっても，ミスヒットが発生する可能性がある。そこで命令用アドレス変換バッファの構成は，評価結果からかなりの余裕があると見られる，64エントリのセット・アソシアティブ方式とすることとした。

データのアドレス分布については，複数のエリアがアクセスされるため，離散傾向が更に強まることが容易に予測できる。また，スタックの伸びを押さえる処理方式をとっているため，特にローカル・スタックとトレイル・スタックについてはスタック・ボトム付近へのアクセスが集中すると予想される。図7は各エリアの使用ページ数を示したものであり，スタックについては集中し，ヒープについては離散する傾向であることが判る。

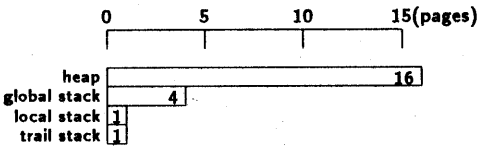

図 7: エリア別使用ページ数

図8は，2セットのセット・アソシアティブ方式のアドレス変換バッファのヒット率を，容量とエントリ・アドレスの生成方法を変化させて測定した結果である。この中でNaiveは，ページ番号の下位ビットをそのままエントリ・アドレスとしたものである。この方法は上記の性質を完全に無視しているため，スタック間でページの奪いあいが頻発し，非常に低いヒット率となっている。特にトレイル・スタックについては容量によらず58.8%という惨憺たる結果になっている。

次にArea Oriented-Iは，エリア番号とページ番号の下位ビットを結合したものをエントリ・アドレスとする方法であり，スタック間でのページの奪いあいがなくなるためにかなり高いヒット率が得られる。しかし，この方法はエリア間で異なるアドレス分布を無視したものであるため，離散傾向が強いヒープに関してのヒット率が悪くなってしまう。実際，ヒープのヒット率は92.9~96.2%であり，充分な値であるとは言えない。

そこで，エントリ・アドレスを下式によって生成する方法Area Oriented-IIを評価した。

$$entry\_addr\langle n:n-2\rangle = area\# \oplus page\#\langle n:n-2\rangle$$
$$entry\_addr\langle n-3:0\rangle = page\#\langle n-3:n\rangle$$
$$(\oplus : \text{Exclusive Or})$$

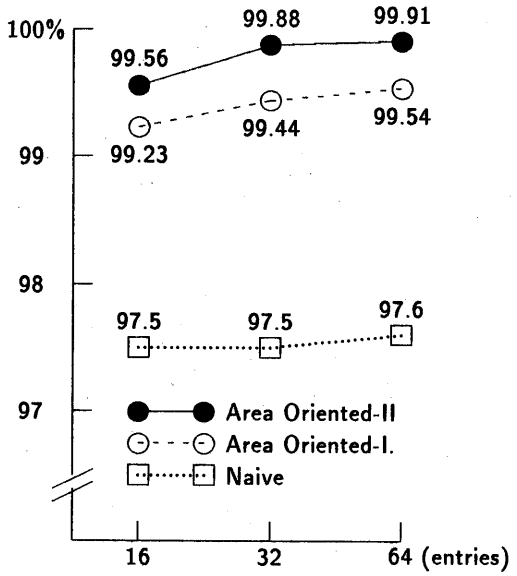

図 8: データ用アドレス変換バッファのヒット率

この方法では，スタック間でのページの奪いあいが抑止されるとともに，ヒープのために多くのエントリが確保される。従って，図8に示すように高いヒット率が得られ，64エントリの場合には100%となった。そこで，アドレス変換バッファの構成を，64エントリ，2セットのセット・アソシアテイブ方式とし，エントリ・アドレスの生成方式はArea Oriented-IIの方法を採用することとした。但し，プロセス切換時のミスヒットを緩和するために，プロセスの識別番号を加味した下式によってエントリ・アドレスを生成することとした。

$$entry\_addr\langle 4:2\rangle = area\# \oplus page\#\langle 4:2\rangle$$
$$entry\_addr\langle 1:0\rangle = page\#\langle 1:0\rangle \oplus process\_id\langle 1:0\rangle$$

5 おわりに

実用的かつ大規模なプログラムであるコンパイラを対象として，PSI-IIのキャッシュ・メモリの評価を行った。その結果，96%以上の高いヒット率が達成されていることや，キャッシュ・メモリの容量半減の影響が軽微であることが明らかになった。また，PIM/m要素プロセッサのメモリ・システムの設計のためのシミュレーションを行って，キャッシュ容量やアドレス変換バッファの構成方式を決定することができた。

なお，今回の評価はESPプログラムを対象としたものであったが，今後KL1プログラムに関しても評価を行うことを計画している。

参考文献

[1] H. Nakashima and K.Nakajima, Hardware Architecture of the Sequential Inference Machine: PSI-II. *Proc. of 4th IEEE Symp. on Logic Programming* (1987).

[2] K. Taki, M. Yokota, A. Yamamoto, H. Nishikawa, S. Uchida, H. Nakashima and A. Mitsuishi, Hardware Design and Implementation of the Personal Sequential Inference Machine (PSI). *Proc. of the Intl. Conf. on Fifth Generation Computer Systems 1984* (1984).

[3] 中島 浩, 三石 彰純, 中島 克人, 瀧 和男：パーソナル逐次型推論マシンPSIの性能評価 —実行速度と各部の性能について—, 情報処理学会論文誌, Vol. 28, No. 12 (1987).

[4] E. Tick, Prolog Memory-Reference Behavior. *Technical Report No.85-281, Computer Systems Lab., Stanford Univ.*, (1985).

[5] S. Uchida, K. Taki, K. Nakajima, A. Goto and T. Chikayama, Research and Development of the Parallel Inference System in the Intermediate Stage of the FGCS Project. *Proc. of the Intl. Conf. on Fifth Generation Computer Systems 1988* (1988).

[6] Y. Takeda, H. Nakashima, K. Masuda, T. Chikayama and K. Taki, A Load Balancing Mechanism for Large Scale Multiprocessor Systems and Its Implementation. *Proc. of the Intl. Conf. on Fifth Generation Computer Systems 1988* (1988).

[7] Y. Kimura and T. Chikayama, An Abstract KL1 Machine and its Instruction Set. *Proc. of 4th IEEE Symp. on Logic Programming* (1987).