計算機アーキテクチャ 102－5
（1993．10．21）

トレース駆動型アーキテクチャシミュレータによる多段キャッシュを持つ共有バス結合型MPの性能評価

高橋正人、前田慎司、宮田裕行、菅隆志

三菱電機(株)情報システム研究所

多段キャッシュを持つ4CPU共有バス結合型マルチプロセッサにおいて、各種アーキテクチャパラメータの変化が、OSを含むシステム全体の性能に与える影響を、トレース駆動型シミュレーションに基づいて評価した。作業負荷として、UNIX OSの上での標準的ベンチマークの一つであるSPEC SDMトレースの一部を用いた。この結果、今回の実験条件のもとで、以下のことが明らかになった。(1)キャッシュコヒーレンシプロトコルは、無効化型の性能が放送型より優れ、Berkeley,Illinoisが最適であった。(2)2次キャッシュのマッピング方式は、ダイレクトマップから2-wayセットアソシエイティブにすることで性能向上が見られたが、それ以上のセット数の増加は効果が薄かった。(3)ダイレクトマップの2次キャッシュに対する小容量のvictim cache付与は、セットアソシエイティブ化を越える効果は観測されなかったが、セットアソシエイティブ化が困難な場合の選択肢となりうる可能性が示唆された。2次キャッシュの1.6%という小容量のvictim cache付与で、2-way化による性能向上のほぼ1/2に相当する効果がみられた。(4)2次キャッシュ容量の拡大と、バス幅の拡張が性能面にもたらすインパクトを、プロセッサ利用率とバス利用率の値として定量的に与えた。

A Performance Evaluation on Shared-bus Multiprocessor Systems with Multi-level Cache using a Trace-driven Simulator

Masato Takahashi, Shinji Maeda, Hiroyuki Miyata, Takashi Kan

Mitsubishi Electric Corporation Computer & Information Systems Laboratory

5-1-1, Ofuna, Kamakura, Kanagawa, 247, Japan

This paper represents a performance evaluation on shared-memory shared-bus type multiprocessors using a trace-driven simulator. Our purpose is to identify the reasonable selection of several architectural parameters of the multiprocessors to avoid bus saturation and achieve high performance. Under the workload of SPEC SDM, we simulated multiprocessor systems that consists of four RISC chips with two level private caches and found following results: (1) Out of six cache coherency protocols, Illinois and Berkeley showed the best performance. (2) On associativity of the second cache, there exists considerable performance gain when the mapping is changed from direct-mapped to 2-way associativity. (3)When victim cache is added to the direct-mapped second cache, performance gain was observed. Although the gain does not reached to that of 2-way associativity, it is suggested that victim cache can be an alternative where 2-way associativity is hard selection for some reason. (4)The impact of expansion of 2nd cache capacity and bus width on system performance is examined.

1. はじめに

マルチプロセッシングの領域において、共有メモリ型マルチプロセッサは広く受け入れられている。この型のマルチプロセッサは単一アドレス空間を提供するためプログラミングし易く、ユーザにデータ分割の問題に直面させることがない[Robi91]。また、プログラミングにおけるほとんどの標準的言語とコンパチブルであり、ユニプロセッサシステムのプログラムをなんら変更することなく稼働させることが可能である。この理由のため、共有メモリ型マルチプロセッサは、メッセージパッシングシステムや、分散メモリシステムよりも広くユーザに受け入れられてきている。

共有メモリ型マルチプロセッサのなかでも、共有バス方式は、バスベースのユニプロセッサの自然な拡張と考えられ、比較的素直な実装方法である。しかし、このタイプのシステムはメモリアクセスとバスアクセスの飽和というボトルネックの可能性を抱えている。特に、プロセッサ性能とメモリ性能とのギャップが大きくなっている近年の状況下では、バス飽和の問題はますます深刻なものとなっており、設計によってはシステムの複数のプロセッサがほとんど有効に稼働できない状況も発生する。

この問題への対策として、各プロセッサにプライベートキャッシュを持たせ、メモリおよびバスアクセスを低減させる場合が多い。この時、プライベートキャッシュは、容量の大きさと、高速な動作という2つの要請を満たすため、小容量だが高速の1次キャッシュと、比較的低速だが大容量の2次キャッシュを備えることが通常行なわれる。またこれに伴って、キャッシュ間のデータの整合性維持の問題が発生するため、なんらかのキャッシュコヒーレンシプロトコルによっても整合性を維持しなければならない[高橋義89]。

ここで問題となるのは、キャッシュ容量や、マッピング方式、ラインサイズを、バスおよびメモリの性能とプロセッサの速度との関連において、どのように設計すれば、システムのバス飽和を避けられ、システムに与えられたプロセッサを十分効果的に稼働させることができるかということである。また、そのアーキテクチャにおいてオーバーヘッドの少ないキャッシュコヒーレンシプロトコルの採用も重要となる。これらは設計段階で熟慮されるべき課題である。

このような背景をもとに、本稿では、次のような範囲について我々の行なったトレース駆動型シミュレーションの結果を報告する。対象は、Motorola 社 88100 RISCプロセッサ[Moto88]4台から構成される共有バス結合共有メモリ型マルチプロセッサシステムである。1次キャッシュはオンチップであり、各8KBのデータキャッシュと命令キャッシュを持つ。システムの使用目的は、UNIX OSを稼働するワークステーションとし、通常4ユーザ程度が同時にプログラム開発や文書処理を行なうものとする。バスは、近年 IEEE で標準化が策定されている Futurebus+[Fbus91] 相当の性能のものを用いるものとする。

この前提で構成されるシステムにおいて、スヌーピングキャッシュコヒーレンシプロトコル、2次キャッシュの容量、ラインサイズ、マッピング方式、バス幅がシステム全体への性能へ与える影響について検討する目的で、トレース駆動型シミュレーションを行なった。スヌーピングキャッシュコヒーレンシプロトコルの評価はいくつかの先行研究があるがその多くは CAD トレース[Egge88][Egg89a]や、人工的に合成されたトレース[Arch86]が用いれることが多い。本稿では共有バス結合共有メモリマルチプロセッサ上において稼働されることが比較的多いUNIX OSトレースを用い、UNIX OSの稼働を目的としたマルチプロセッサシステムの評価という面でより直接的な評価を与えたい。

同時に、次のような評価も本稿ではおこなう。一般に2次キャッシュがダイレクトマップの場合、コンフリクトミスに基づいたバスアクセスの増大が予想される。が、これは2次キャッシュをセットアソシエイティブにすることによって軽減されるが、この方策は一般にコストが高い。そこで、ユニプロセッサにおける1次キャッシュでのコンフリクトミスの低減策として近年提案された victim cache[Norm90] という概念の、マルチプロセッサシステムの2次キャッシュへの適用を考え、その場合のシステム性能向上への効果を検討する。

2. 評価フロー

トレースの採取は、トレース採取ツールを用いてソフトウエア的によって行なった。その後、採取されたトレースを用いて、トレース駆動型シミュレーションによるマルチプロセッサアーキテクチャの評価を行なった。以下にその各工程について述べる。

2.1トレース採取

トレース採取ツールにはg88[Bedi90]を用いた。g88はMotorola 社の RISCプロセッサ 88100をエミュレートし、その上で OS およびアプリケーションを動作させることができる。また4プロセッサまでのマルチプロセッサ構成をシミュレート可能であり、この際プロセッサ別のメモリ参照トレースが採取可能である。この機能を用いて、4台の88100を共有バス結合した構成で、UNIX OSを稼働させ、次に示す作業負荷を与えたときの各プロセッサのメモリ参照列をトレースとして採取した。

作業負荷として、UNIXが動作するマルチプロセッサ

システム における標準的ベンチマークとなりつつあるSPEC SDM sdet[Case92]の21スクリプトの中から、SPEC SDM sdetの挙動の典型的なものと考えられる4スクリプト(script0, script3, script7, script10)を並行実行させた。これにより、UNIX OS 上で、プログラム開発ないしドキュメント作成作業を4ユーザが同時に行なっている状況の作業負荷を実現している。トレース長は4プロセッサ全体で約350MBである。

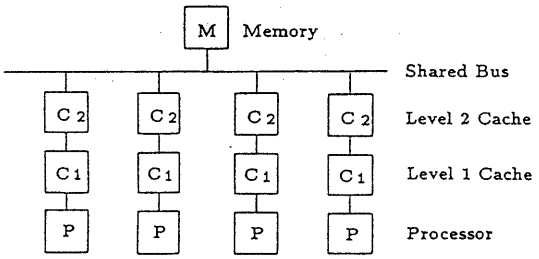

図 1: 評価対象のマルチプロセッサ構成

2.2 マルチプロセッサアーキテクチャシミュレーション

このトレースを与えることにより、MP アーキテクチャシミュレータが、プロセッサ、キャッシュ、バス、メモリの前後関係や相互作用などを、タイミング情報を含んでシミュレーションを行なう。ここでは、次の各項目が指定できる。(1)6つのスヌーピングキャッシュコヒーレンシプロトコルから1つ。内訳は、Berkeley, Illinois, Firefly, Dragon, WriteOnce, Synapseの各プロトコル [Arch86]。(2) キャッシュ容量、マッピング方式、ラインサイズ等のキャッシュパラメータ。(3) バス幅、転送様式、アービトレーションコスト等のバスパラメータ。(4) メモリクセスの際のメモリウエイトや、バスにおける無効化信号や、放送信号などのトランザクションコストに代表されるタイミング情報等。

これらのパラメータに基づく シミュレーションの結果、各プロセッサにおけるプロセッサ稼働率やバス使用率、プログラム実行時間などの性能指標を出力する。

今回評価の対象としたのは、Motorola社88100、4プロセッサ共有バス共有メモリ構成。各々のプロセッサは2段のプライベートキャッシュを備える。図1に評価対象のマルチプロセッサ構成の概念図を示す。

1次キャッシュはセパレートキャッシュで、8KB のデータキャッシュと8KBのインストラクションキャッシュを持つ。ラインサイズはともに32B。マッピング方式は4-way associativeである。

2次キャッシュは統合型キャッシュで、容量は変数とし32KBから8MBまでの各値を検討した。ラインサイズは64Bを中心に、32B、128Bを検討した。マッピング方式は、ダイレクトマップおよび2から8までのset associativityの効果も検討した。

キャッシュコヒーレンシプロトコルには、今回のような構成において最も有効であるスヌーピング方式から6種類のプロトコル[高橋義89]を検討した。無効果型プロトコルとして、Berkeley, Illinois, Synapse, Write-onceの各プロトコルを検討した。放送型プロトコルとして、Dragon, Fireflyを検討した。

バス幅は8Bから64Bまでの値を検討した。

タイミング情報に関しては、1次キャッシュのヒットは、パイプライン化されているため0サイクルとした。2次キャッシュのヒットは1サイクル。バストランザクションに関しては、キャッシュ間データ転送が 13 サイクル、メモリキャッシュ間データ転送が 18 サイクル、信号無効化が3サイクル、更新内容放送が18サイクル等とした。アービトレーションコストは6サイクルとし、バス使用中にもバックグラウンドでアービトレーションが可能な機構を備えるものとした。これらの仕様は、C-busII[Coro92] あるいは Futurebus+[Fbus91]といった近年のバス仕様を反映させたものである。

3. 評価結果

3.1スヌーピングキャッシュコヒーレンシプロトコル

スヌーピングキャッシュコヒーレンシプロトコルはこれまでいくつが提案されてきた[高橋義89]。大別して、書き込み時無効化型と、書き込み時放送型に大きく分類される。ここでは、バス結合共有メモリ型マルチプロセッサに最も適しているスヌーピング方式のなかから主要な6プロトコルを取り上げ、先に述べたマルチプロセッサシステムへの適合性を評価する。

ここでの目的は、今回使用されたトレースにおいて、(1) 無効化型と放送型の性能比較と各々の特性の検討 (2) 無効果型において方略の全く異なるBerkeley ProtocolとIllinois Protocolの性能比較(3)無効化型および放送型においてキャッシュラインサイズが性能に与える影響の検討、とする。

3.1.1 無効果型と放送型の性能特性条件

2次キャッシュについて、容量は256KB、ラインサイズは64B、ダイレクマップキャッシュを対象とした。バスについてバス幅は8Bとした。

結果

図2に各々のプロトコルについてアクセスタイプ別に2次キャッシュのヒット率を示す。データストアのヒット率は放送型プロトコルが無効化型プロトコルを上回っている。これは、無効化型では、被共有データへの書き込みが生じた際に他キャッシュに存在するコピーを無効化するため、無効化された側のプロセッサで再びそのブロックにアクセス要求があった時に、キャッシュミスとなることを反映していると考えられる。

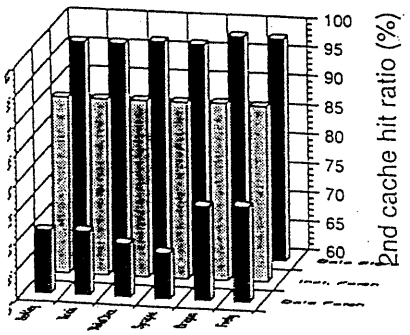

図2: キャッシュコヒーレンシプロトコル別の
2次キャッシュにおけるヒット率
(奥から Data Store, Inst Fetch, Data Fetch の各値。
左から Berkeley, Illinois, WriteOnce, Synapse, Dragon, Firefly の各値)

一方放送型においては、その逆に更新された情報が放送されるため、被共有データに関するヒット率は向上する。放送型におけるこのようなキャッシュヒット率の向上は望ましいことではあるが、これは被共有データへの書き込み毎に発生する頻繁なバス放送によってはじめて維持されているものと考えられる。実際、図3の各プロトコルにおける平均バス使用率を見ると、放送型のバス使用率は90%以上と極めて高くなっていることが認められる。一方無効化型のバス使用率は50%程度にとどまっている。

次に総合的な性能指標としてのプログラム実行所要時間とその内訳を図4に示す。この図からは、放送型の実行時間は、無効化型の実行時間に比較してて、より延長していること、また、その主な原因はビジー状態のバスが解放されるまで待たされているプロセッサの空転にあることがわかる。つまり無効化型にくらべて放送型は、多量のバストランザクションのためバス飽和を生じており、これにより各プロセッサからのバス要求はバス解放待ち待ちを余儀なくされている。このた無効型において60～70%程度を示すプロセッサ稼働率は、放送型においては20～30%まで低下している。

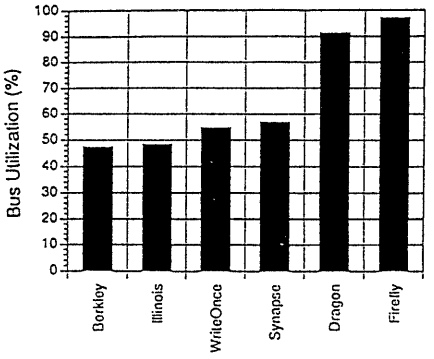

図3: キャッシュコヒーレンシプロトコル別バス使用率

3.1.2 Berkeley と Illinois:Dirty-Shared状態の有無による性能への影響

無効化型は放送型に比べて一般によい性能を示したが、無効化型の中でも性能的には Berkeley, Illinois と、Synapse, Writeonce との2グループに分かれた。このグループ間の性能差は、前者がもつキャッシュ間転送機能を後者は持たないことが主因であると考えられる。

Berkeley と Illinois はキャッシュ間転送機能を持ち、プロトコル的にも比較的類似点が多い。その反面、両者は Dirty Shared 状態のデータの扱いについて全く相反する方略をとっている。前者は不要な書き戻しを可能な限りさける方略をとることで効率化を図るが、リプレイス時の書き戻しは避けれない。後者は Dirty データの

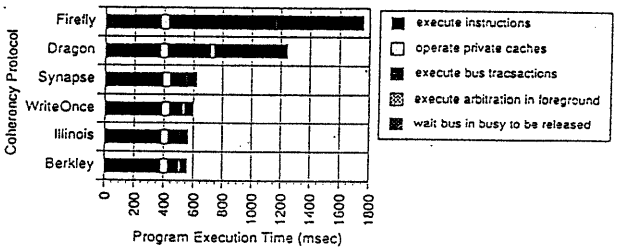

図4: 各キャッシュコヒーレンシ別単位時間あたりプロセッサの稼働状況比較

キャッシュ間転送時にはつねに同時にメモリ書き戻しをおこなうという機構を採用している、この両者の機構のうちのどちらが性能面で優れるものであるかに興味が持たれた。

結果は図4にみられるようにごく接近した性能をしめした。両者の戦略は性能面に関してのみ言えば優劣付け難いものであることが示唆された。このことを踏まえ、またFuturebus+等に代表される近年の標準的バスにおける実装の容易さを考慮して、以下の評価において特に断わらない場合はIllinoisを用いることとする。

3.1.3 キャッシュラインサイズが無効果型、放送型の性能に与える影響

条件

2次キャッシュのラインサイズが、無効化型プロトコルと、放送型プロトコルの性能に与える影響について検討する。無効化型の代表として、Illinoisを、放送型の代表としてDragonを用いた。2次キャッシュ容量は256KBに固定した。検討するラインサイズの下限は1次キャッシュのラインサイズと同じ32Bとし、64B、128Bを検討した。バス幅は8Bを仮定した。

結果

図5に示すように、無効化型では性能はラインサイズ64Bを中心として山型を示した。これは、小さなラインサイズでは、連続したデータの取得に無駄が生じ、大きなラインサイズにおけるfalse sharingにもとづく性能低下によるものと考えられる。

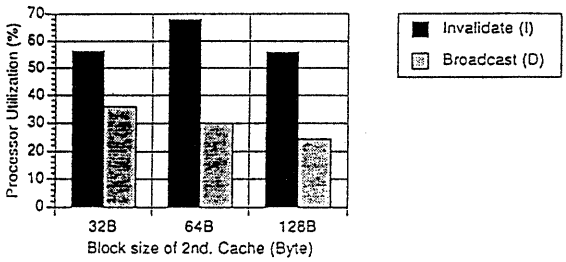

図5:コヒーレンシプロトコルと2次キャッシュラインサイズとプロセッサ稼働率

放送型の性能は、ラインサイズの増加にともなってなって単調減少している。これは、頻繁に放送を行なう放送型において、ラインサイズの拡大にともない、一回の放送にかかるバスサイクルが延長することが、バス競合に拍車をかけることによると考えられた。

表:1 コヒーレンシプロトコルと2次キャッシュラインサイズと性能指標

	Invalidate (Illinois)			Broadcast (Dragon)		
line size	Te(msec)	PU(%)	BU(%)	Te(msec)	PU(%)	BU(%)
32B/line	578.0	66.2	49.4	1062.9	35.9	85.5
64B/line	564.3	68.0	48.5	1244.1	30.1	91.4
128B/line	584.6	65.9	53.9	1495.0	24.4	95.8

Te:Execution Time PU: Processor Utilization BU:Bus Utilization

3.2 2次キャッシュのマッピング方式

条件

2次キャッシュをセットアソシエイティブにした場合の効果を検討する。2次キャッシュの容量は256KB、ラインサイズ64Bに、バス幅8Bを仮定した

結果

図6のプロセッサ稼働率に示されるように2次キャッシュをダイレクトマップから2-wayセットアソシエイティブにした場合の性能の伸びがそれ以外の場合に比べて顕著であった。図7に示されるバス使用率の低下の傾向は、ダイレクトマップキャッシュにおける conflict miss の低減の重要性を示唆していると考えられる。

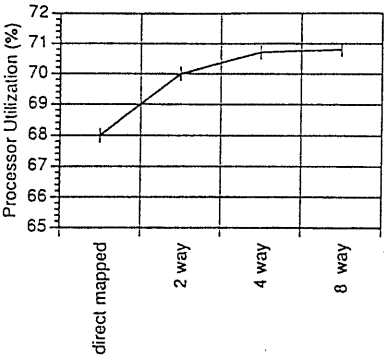

図6: 2次キャッシュのマッピング方式とプロセッサ稼働率

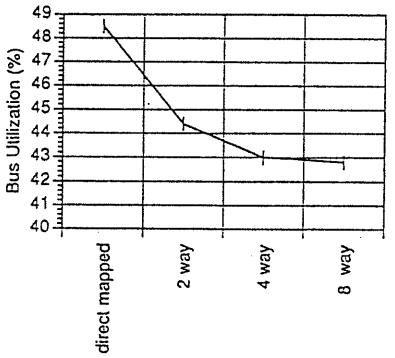

図7:2次キャッシュのマッピング方式とバス使用率

表 2: 2次キャッシュのマッピング方式と性能指標

	associativity of 2nd cache direct_mapped	2-way	4-way	8-way
BU (%)	48.5	44.4	43.0	42.8
PU (%)	68.0	70.0	70.7	70.8
Te (msec)	564.3	555.2	552.7	552.1

Te:Execution Time PU: Processor Utilization BU:Bus Utilization

3.3 victim caching効果

victimcacheとは、ダイレクトマップキャッシュにおいて頻繁に生じているconflict missに注目し、conflict missによる性能低下を軽減することを目的として提案されたものである[Norm90]。ダイレクトマップキャッシュと、より高次のキャッシュとの間に、小容量のセットアソシエイティブなキャッシュを付与し、ダイレクトマップにおいて置換されたエントリを保持させる。後にプロセッサからそのエントリの要求があり、メインキャッシュがミスした際には、その小容量キャッシュがそのエントリを保持していればそれを提供するという機構である。[Norm90]は、ダイレクトマップ1次キャッシュにおいて3から5エントリという、ごく小容量のvictim cache を付与することにより、コンフリクトミスを大きく低減させることが可能であることを複数のトレースについてのシミュレーションに基づく性能評価により示した。

本来この提案は、高性能なチップをもつユニプロセッサシステムにおける1次キャッシュを対象として提案されたものである。われわれは、このアイデアを、共有バス結合型マルチプロセッサにおける2次キャッシュにおいて応用し効果の検討を試みた。

その理由は次のようなものである。近年の共有結合型マルチプロセッサでは、2段のプライベートキャッシュを備えることが常識的となってきている。高速で小容量の1次キャッシュと、低速だが、容量の比較的大きな2次キャッシュとの組合せにより、性能のボトルネックとなるバスへのアクセス要求をできるだけ低減しようと試みることが多い。この状況にあって、2次キャッシュには、ダイレクトマップがコスト面から用いられることが多く、そのため、conflict miss の増加が懸念される。2次キャッシュにおけるミスの増大は直ちにバス負荷に直結するため重要である。このような conflict miss を避けるために、セットアソシエイティブキャッシュにすることも考えられるが、victim cache の付与によって得られる効果の程度によっては、victim cache による方法が、その代替策になることも考えられる。こ

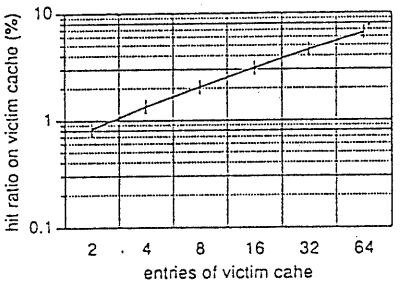

図 8: victim cache エントリ数と victim cache のヒット率

のような考えに基づいて、victim cacheの効果検討を行なった。

条件

256KBのダイレクトマップ2次キャッシュ。2, 4, 8, 16, 64エントリを持つvictim cache を付与した結果を検討した。これは容量では、64エントリのvictim cache は2次キャッシュの1.56%の容量という小さなものである。victim cache のヒットにおけるデータの読み出しないし書き込みコストには1サイクルを仮定した。victim cache は統合型キャッシュ、フルアソシアティブとした。

結果

64エントリのvictim cacheにより、1.5%のプロセッサ稼働率の改善が得られた。これは、後述の2次キャッシュ容量の検討に照らし合わせると、2次キャッシュの40%の増量に相当する。また、ダイレクトマッピングキャッシュから、2-way セットアソシエイティブキャッシュにした場合の性能向上のほぼ1/2に相当する。この効果が2次キャッシュの1.56%の容量という小さなcacheによって達成されたところが興味深い。それだけ、conflict miss が重要であることを示唆しているといえる。

表 3: victim cach エントリ数と性能指標

	victim cache entries without	2	4	8	16	32	64
Hit Ratio(%)	--	0.83	1.35	2.06	3.11	4.60	6.62
BU (%)	48.5	48.1	47.9	47.5	47.0	46.4	45.5
PU (%)	68.0	68.2	68.3	68.4	68.7	69.0	69.5
Te (msec)	64.3	563.1	563.7	562.4	560.2	558.6	557.7

Te:Execution Time PU: Processor Utilization BU:Bus Utilization

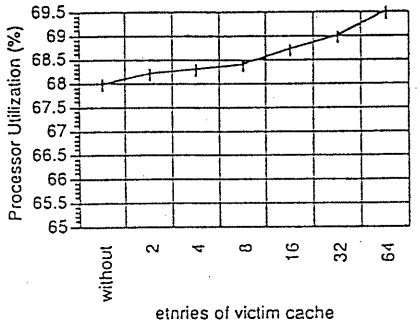

図 9: victim cache エントリ数と プロセッサ稼働率

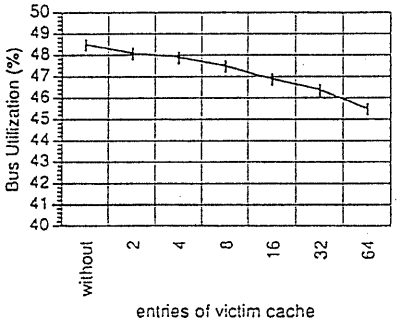

図 10: victim cache エントリ数と バス使用率

3.4 2次キャッシュ容量拡大と、バス幅拡大が性能にもたらすインパクトの評価

設計段階において、キャッシュ容量とバス幅は重要な選択項目であるが、これらの要因が相互の関係において性能向上に実際どの程度の効果をもつかを実装前に把握することは困難である。そこで各要因が性能向上にもつ効果を定量的にを明らかにするために、シミュレーションにより検討した。

条件

2 次キャッシュ容量は 32KB から 8MB までを用いた。バス幅は、8Bから2次キャッシュラインサイズの64Bまで検討し、64Bにおいてはさらにメモリウエイトの少ないメモリを用いた構成を検討した(図中には64BW+Mとして表記する)。

結果

プロセッサ稼働率として図11、バス使用率として図12のような結果を得た。図 11 のプロセッサ稼働率から判断すると、キャッシュ容量の拡大は、ほぼ 1MB のあたりで性能向上が頭うちになっており、1MB あるいはそれをやや上回る容量の2次キャッシュを装備することが効率的であろうと考えられた。また図12から、バス幅として、2次キャッシュラインサイズと同じ64B を採用する場合の性能面での利得は考慮に値するものであると考えられる。

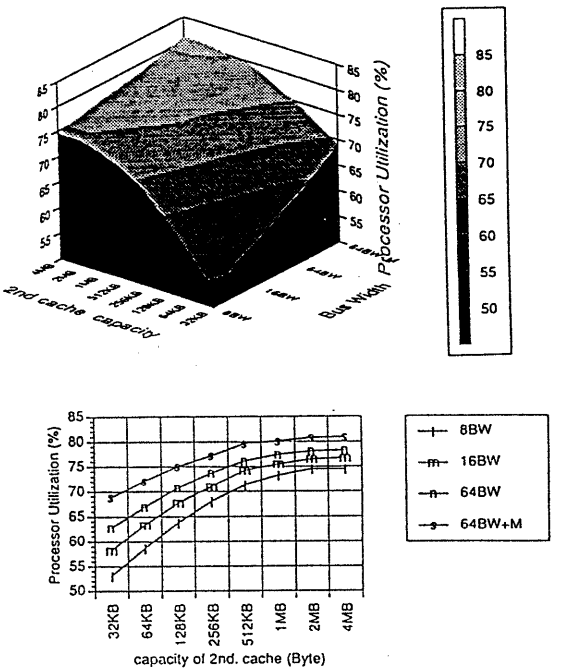

図 11:キャッシュ容量とバス幅とプロセッサ稼働率

4. まとめ

トレース駆動型シミュレーションを用いて、多段キャッシュを持つ共有バス結合型RISC4CPUマルチプロセッサの各種アーキテクチャパラメータを評価した。作業負荷として、UNIX OSの上での標準ベンチマークの一つであるSPECSDMトレースの一部を用いた。この結果今回の実験条件のもとで、以下のことが明らかになった。

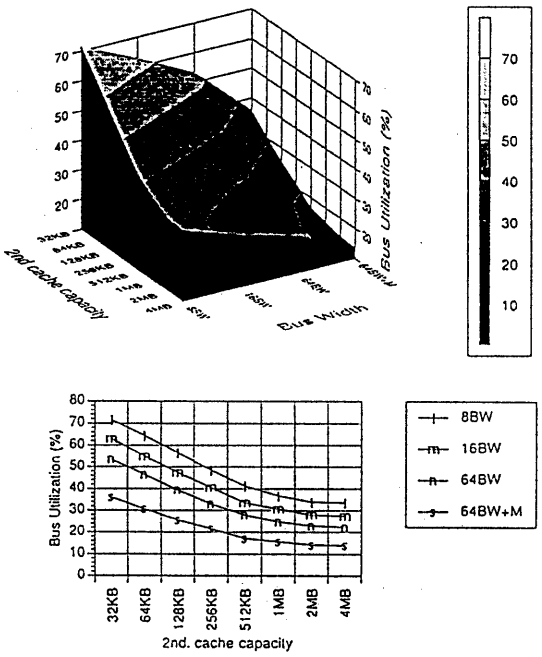

図 12:キャッシュ容量およびバス幅とバス使用率

(1)キャッシュコヒーレンシプロトコルは、無効化型の性能が放送型よりすぐれ、Berkeley、Illinoisが最適であった。(2)放送型はキャッシュラインサイズの増加にともない性能が劣化する傾向が見られた。無効化型は64B/lineにおいて最適な性能を示した。(3)2次キャッシュのセットアソシエイティブ化はダイレクトマップから2-wayにすることで性能向上が見られたが、それ以上のセット数増加は効果が薄かった。(4)ダイレクトマップ2次キャッシュに対する小容量のvictim cache付与は、セットアソシエイティブ化を越える効果は観測されなかったが、セットアソシエイティブ化が困難な場合の選択肢となりうる可能性が示唆された。2次キャッシュのほぼ1.6%の容量しかないvictim cache付与で、2-way化にした場合の性能向上のほぼ1/2に相当する効果が観測された。(5)2次キャッシュ容量の拡大と、バス幅の拡張が性能面にもたらすインパクトを、プロセッサ稼働率とバス利用率の値として定量的に与えた。

今回のトレースは、UNIX OS上での典型的なマルチユーザマルチタスクを反映するように配慮されており、今後、よりデータを集めることにより検討を深めてゆく計画である。さらに、今後スケーラブルな共有メモリマルチプロセッサシステムとして、階層バス/キャッシュ構成[Wils87]、インタリーブバス構成[Tung91]、クラスタ構成における性能評価を行なっていくことを計画している。

参考文献

[Robi91]Popular and Parallel, M. Robinson, BYTE,June 1991, pp.21-226

[高橋義 89]高橋義造,、並列処理機構、第5章バス結合型マルチプロセッサ、丸善、1989

[Moto88]Motorola Inc., MC88100 User's manual, 1988

[Fbus91] IEEE Standard for Futurebus+ Physical Layer and Profile Specification, IEEE Std 896.2-1991

[Norm90]Norman P. Jouppi, Improving Direct-Mapped Cache Performance by the Addition of a Small Fully-Associative Cache and Prefetch Buffers, IEEE 1990, pp.364-373

[Case92]Brian Case, Microprocessor Report -- Updated SPEC Benchmarks Released, Sept. 16, 1992, pp. 14-19

[Coro92] C-busII Specification Revision1.1, Corollary,Inc., Califoria.1992

[Bedi90]Robert Bedicheck, Some Efficient Architecture Simulation Techniques,USENIX -- Winter 1990 pp.53-63

[Arch86]J. Archibald and J. L.Baer, Cache Coherence Protocols: Evaluation Using a Multiprocessor Simulation Model, ACM Trans. on Comp. Sys., Nov., 1986, pp.273-298

[Egg88]S. Eggers and R.Kats, A Characterization of Sharing in Parallel Programs and Its Application to Coherency Protocol Evaluation, Proc. 15th ISCA, 1988, PP.378-382

[Egge89a] Eggers and R.Kats, The Effects of Sharing on the Cache and Bus Performance of Parallel Programs, Proc. ASPLOS III Conference, 1989, pp.257-270

[Egge89b]S. Eggers and R.Kats, Evaluating the Performance of Four Snooping Cache Coherency Protocols, Proc. 16th ISCA, 1989, pp.2-15

[Tung91]Cheng-Hsien Tung, On Large Scale Shared-Memory Multiprocessor Systems, Proc. of the 4th ISMM/IASTED Int. Conf. Parallel and Distributed computing and systems, Washington D.C., Oct. 91, pp.139-143

[Wils87]Andrew W. Wilson Jr., Hierarchical Cache/Bus Architecture for Shared Memory Multiprocessors, Proc. 14th Int. Symp. Computer Architecture,87, pp.244-252