社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2003－ARC－155　(10)
2003／11／28

SR11000 モデル H1 におけるバリア同期の高速化手法

中村 友洋　高山 恒一　青木 秀貴　松居 昭宏　助川 直伸
(株)日立製作所 中央研究所

共有メモリ型計算機において高い並列実効性能を実現するには，並列処理の起動終結時のバリア同期オーバーヘッドを低減することが重要である。ノードを構成する複数のマイクロプロセッサを一斉にしかも高速に起動させる協調型マイクロプロセッサ機構により高い並列実効性能を達成したスーパーテクニカルサーバ SR8000 の後継シリーズの初代モデルである SR11000 モデル H1 は，キャッシュシステムを利用したソフトウェアによるバリア同期方式により，高速なバリア同期処理を実現することで，高い並列実効性能を達成する。本稿では高速バリア同期方式の概要とその高速化手法について述べ，SR11000 モデル H1 による性能評価結果を紹介する。

Fast Barrier Synchronization Mechanism of SR11000 model H1

Tomohiro Nakamura　Koichi Takayama　Hidetaka Aoki　Akihiro Matsui　Naonobu Sukegawa
Hitachi, Ltd., Central Research Laboratory

In order to realize a high parallel execution performance in a shared memory type parallel computer, it is important to reduce the barrier synchronization overhead. SR11000 model H1 is the first model of the succeeding series of the super technical server SR8000 that attained the high parallel execution performance by the CO-operative Micro-Processors in single Address Space (COMPAS) function which simultaneously and rapidly synchronizes the multiple CPUs in a node. SR11000 model H1 attains a high parallel execution performance by high-speed barrier synchronization processing according to the barrier synchronization mechanism by the software using the cache system. In this paper, the outline and technique of a high-speed barrier synchronization mechanism are described, and the performance evaluation result is introduced.

1　はじめに

1.1　背景と目的

2003 年 5 月に製品発表した SR11000 モデル H1 は，ベクトル・スカラ融合型 HPC（High Performance Computing）マシンとして好評を得た SR8000 シリーズの後継機として開発を進めてきたスーパーテクニカルサーバ SR11000 シリーズの初代モデルである。1.7GHz 動作の POWER4+プロセッサ 16CPU（チップ当たり 2CPU コア×8 チップ）がメインメモリを共有するノードを構成単位として，多段クロスバネットワークにより最大 256 ノードを結合できる SR11000 モデル H1 は，27.8TFlops の最大性能を有し，大規模な科学技術計算にも対応可能である。

前世代モデルである SR8000 シリーズは，科学技術計算向けに擬似ベクトル処理機構，協調型マイクロプロセッサ機構（COMPAS : CO-operative Micro-Processors in single Address Space），多次元クロスバネットワークなどの特長技術を持ったサーバである[7]。SR11000 モデル H1 は，SR8000 シリーズのこれらの特長を継承しつつ，HPC 市場の高い並列実効性能への要求を満たすものとして開発を進めてきた。

そのために，SR11000 モデル H1 では，SR8000 の特長技術のうち特にノード内での並列実効性能に大きな影響を与える協調型マイクロプロセッサ機構の機能継承を最も重要な課題とした。

協調型マイクロプロセッサ機構とは，自動並列化コンパイラによる要素並列化と，それをサポートするプロセッサ間高速同期機構であり，並列実行時のオーバーヘッドとなる並列処理起動終結時のバリア同期処理時間を低減することで高い並列実効性能を達成可能とする。SR8000 シリーズでは，専用のハードウェアによりノード内 8CPU 間でのバリア同期処理時間を 3～4 μ s に押さえることで高い並列実効性能を達成したが，SR11000 モデル H1 においては，高周波数動作の POWER4+プロセッサと高速なキャッシュシステムの活用により，専用ハードウェアを設けることなくバリア同期処理時間を SR8000 と同程度に押さえるバリア同期方式を開発したので，その高速化手法について紹介する。

1.2　本論文の構成

本稿では，まず SR11000 モデル H1 の構成の概要と特長について述べる。次に，特長の中でも特にノード内の並列処理に大きな影響を与える，起動終結時のバリア同期性能の重要性について説明し，キャッシュシステムを利用したソフトウェアによるバリア同期方式により，高速なバリア同期処理を実現する手法を紹介する。最後に，SR11000 モデル H1 による性能評価結果を示して，本手法により高い並列実効性能を達成できることを示す。

2 SR11000 モデル H1

2.1 SR11000 モデル H1 の概要

SR11000 モデル H1 は，大規模科学技術計算向けに高いピーク性能と高速なネットワークを組み合わせて高い実効性能を発揮する SMP(Symmetric Multi Processor)型並列コンピュータである。

SR11000 モデル H1 のシステム構成を図 2-1に示す。高いピーク性能を実現するために，1.7GHz 動作の POWER4+プロセッサを 1 ノードに 16CPU 搭載し，ノード当たり 108.8GFlops のピーク性能を達成した。さらに，高い実効性能を実現するために，オンチップ L2 キャッシュと 256MB に及ぶ大容量 L3 キャッシュを設けた。さらに，これらのキャッシュを活かして，ハードウェアが自動的にメモリを先読みするハードウェアプリフェッチと，自動並列化コンパイラでメモリ先読みを最適制御できるソフトウェアプリフェッチの 2 種類のプリフェッチ機構を用意し，高いデータ転送性能を実現した。ノード間の接続はネットワーク上での転送データの衝突が少ない多段クロスバネットワークで行い，最大 12GB/s(単方向)×2 の性能をもつ。複数のノードに跨る大規模計算に向けて，先代機 SR8000 モデル G1 からのノード性能の向上(約 8 倍)に合わせノード間データ転送性能も約 8 倍向上することで、SR8000 と同等の優れた演算とネットワーク転送の性能バランスを実現している。

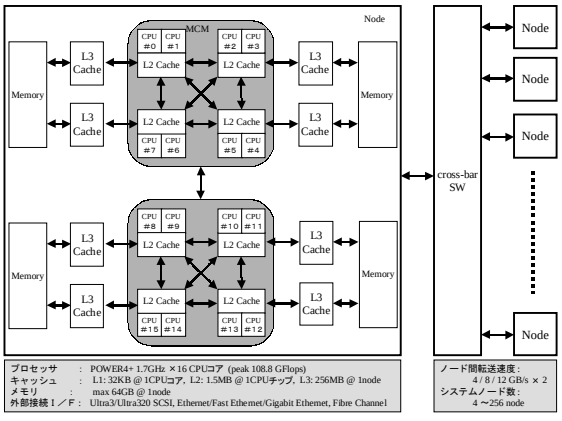

図 2-1 SR11000 モデル H1 のシステム構成

2.2 SR11000 モデル H1 の特長

前世代モデルである SR8000 シリーズの特長を継承しつつ，HPC 市場の高い並列実効性能への要求を満たす SR11000 モデル H1 の特長は次の通りである。

(1) 高スケーラブルアーキテクチャ

CPU 数に対する高いスケーラビリティを実現するノード構成，高性能自動並列化コンパイラと並列実行を高効率に行うための機構，高い CPU 動作周波数に対応した高性能メモリシステム

(2) コンパクトな実装

メインフレームで培った高密度実装技術による世界最高クラスの単位床面積当たりピーク演算性能

(3) 幅広い HPC アプリケーション対応，運用支援機能

ノード性能に見合うネットワーク性能を持ち，幅広い HPC アプリケーションに最適なバランスの取れたシステム構成，ノード稼動時保守などの支援機能

高スケーラブルアーキテクチャを実現する技術として，本稿では高速バリア同期方式を，論文 9)では SR11000 モデル H1 のノード内マルチプロセッサ構成を紹介する。さらに，論文10)では,メモリ負荷の評価手法を提案し，高メモリ負荷ジョブでも安定して高性能が発揮できることを示す。

3 バリア同期処理の高速化手法

3.1 並列処理におけるバリア同期性能影響

SR11000 モデル H1 で実行される大規模科学技術計算では，処理の高速化のために高性能自動並列化コンパイラによりプログラムを SIMD 並列化して実行する。大規模科学技術計算には多くのデータ並列性が存在するため，並列実行方式としては SIMD 方式がとられる。SIMD 並列実行では，逐次実行部分・並列実行部分から構成される単一コードをマルチ CPU で実行する。SIMD 並列実行では逐次実行部分と並列実行部分，もしくは並列実行部分間でのデータの同期をとるためにバリア同期を行いつつ処理を進める。図 3-1に SIMD 並列実行とバリア同期の関係を示す。図 3-1に示す逐次実行部分 S_1, S_2, S_3と並列実行部分 P_1, P_2, P_3 からなるプログラムコードでは，各 CPU 間での実行タイミングを揃えるために，バリア同期ポイントでデータの分配・収集・交換を同期して行う必要がある。これは，バリア同期をはさむことにより，分散共有メモリ上で複数プロセッサからの同一アドレスメモリへのアクセス順序を規定することで，データ依存関係を守るためである。

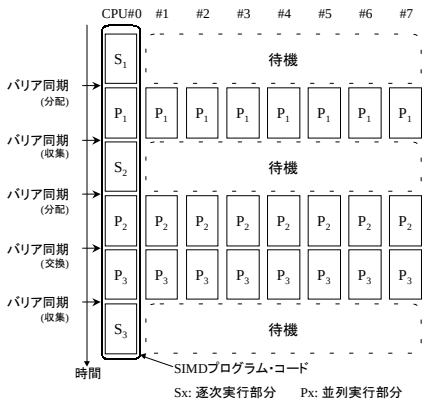

図 3-1 SIMD 並列実行とバリア同期の関係

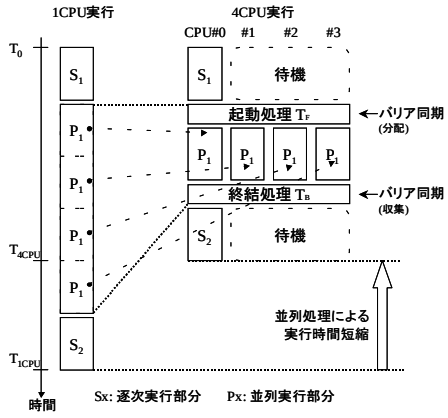

図 3-2 並列実効性能とバリア同期の関係

図 3-1の S_1, P_1, S_2 部分の SIMD 並列実行による実行時間詳細を図 3-2に示す。図 3-2左側は, 1CPU 実行時の実行時間 T_{1CPU} であり, 逐次実行時間 S_1, S_2 と, 並列実行時間 P_1×4 の合計時間となる。図 3-2右側は, 4CPU 実行時の実行時間 T_{4CPU} であり, 逐次実行時間 S_1 および S_2 と, 並列実行時間 P_1 に加えて, バリア同期処理において必要となる起動処理時間 T_F と終結処理時間 T_B の合計時間となる。一般に, n CPU 実行時の実行時間を T_{nCPU} とすると, T_{1CPU} , $T_{n\,CPU}$ は次式で与えられる。

$$T_{1\,CPU} = S_1 + S_2 + P_1 \times n \qquad (1)$$

$$T_{n\,CPU} = S_1 + S_2 + P_1 + T_F + T_B \qquad (2)$$

並列処理において使用する CPU 数に応じた高い性能スケーラビリティを得るには, n CPU 実行時に 1CPU 実行時の n 倍に近い性能となる必要があり, 逐次実行時間 ($S_1 + S_2$) が並列実行時間 P_1 に比べて十分小さいことの他に, バリア同期の起動終結処理時間 ($T_F + T_B$) が並列実行時間 P_1 に比べて十分に小さいことが必要である。逐次実行時間 S_1, S_2 と, 並列実行時間 P_1 はプログラムと単体 CPU 性能で決定されるため, 高い並列実効性能の達成には, 特にバリア同期の起動終結処理時間 ($T_F + T_B$) の短縮が重要である。

図 3-3にプログラムのデータサイズと実効性能の関係が, 同期処理時間の大小により受ける影響を示す。一般に, データサイズがキャッシュにおさまらない場合(キャッシュアウト領域)には, メモリからのデータ転送により演算時間が増大し, 全実行時間を決定する。よってキャッシュアウト領域では, 図 3-3に示すように, バリア同期処理時間に較べて演算時間の比率が高くなり, バリア同期処理時間の大小による全体性能への影響は軽微となる。一方, データサイズがキャッシュにおさまる場合(インキャッシュ領域), 全実行時間に占める並列実行時間の比率がデータサイズに比例して増大するため実効性能は向上するものの, 全実行時間に占める比率では演算時間に較べてバリア同期処理時間の比率が高い。よって, バリア同期処理時間の大小による全体性能への影響は大きくなる。一般に, スカラ型並列計算機で良い実行効率を得るためにはインキャッシュ領域で演算がされるようにプログラムをチューニングすることが必要であり, その際 図 3-3に示す処理性能曲線から, インキャッシュ領域の実効性能に影響の大きいバリア同期処理時間の低減が非常に重要となる。よって, SR11000 モデル H1においても, 高い並列実効性能を達成するためには, プログラムのインキャッシュ化チューニングに加え, バリア同期処理時間の低減が重要である。

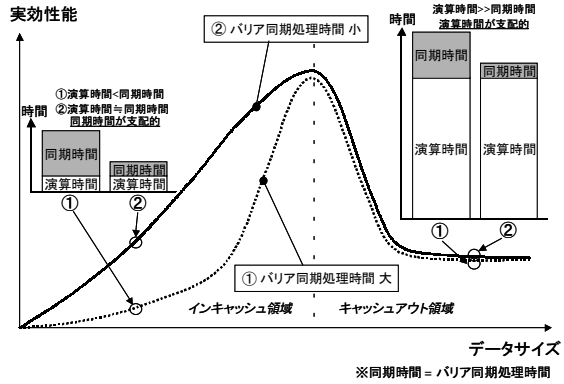

図 3-3 バリア同期性能の実効性能への影響

3.2 ソフトウェアによる高速バリア同期方式の提案

バリア同期処理の実装方法としては, ロック変数を使ったソフトウェアによる方法と専用ハードウェアによる方法がよく知られているが[1], 同期処理時間の短さの点で専用ハードウェアによる方法が優れている。SR8000 では, 独自開発 CPU に同期処理用の専用ハードウェアを実装することにより, ノード内 8CPU 間でのバリア同期処理時間を 3～4 μs としているが, SR11000 モデル H1 では, 高い周波数で動作する POWER4+ プロセッサと高速なキャッシュシステムを用いたソフトウェアによる実装で同程度のバリア同期処理時間を実現することにした。ソフトウェアによる実装は, 専用ハードウェアの開発コストが不要であり, ハードウェア・リソースによる実行制限が無く, 複数の並列処理の同時実行対応が容易であることが利点である。一方, バリア同期性能の面では, ハードウェア実装に較べて劣るため, 一般的に用いられるロック変数を使ったバリア同期処理ではなく, キャッシュシステムを活用した高速な方式を開発した。以下では, SR11000 モデル H1 で利用するソフトウェアによる高速バリア同期(FBS: Fast Barrier Synchronization)方式について説明する。

まず最初に, 通常のロード・ストア命令などを使い高速なバリア同期を実現するための処理フローを図 3-4に示す。この処理フローは, それぞれの CPU が同期フラグと呼

ぶ変数を持ち，同期フラグに各 CPU が処理を完了したバリアポイント番号[1]を保持し，この値が並列処理に参加している CPU 間で一定の条件に達するのを待つことでバリア同期を実現している。一定の条件とは，他の全CPUのバリアポイント番号が，自 CPU のバリアポイント番号と同一か，その次のバリアポイント番号になることである。この処理では，ロック処理が不要で，各 CPU は他の CPU の同期フラグを監視することで同期待ちを行う。

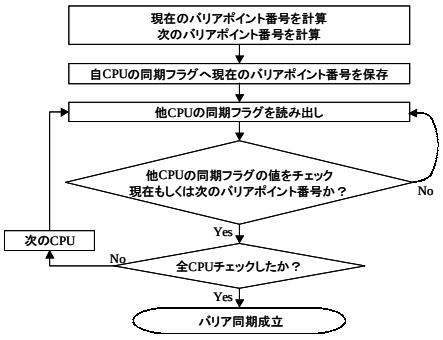

図 3-4 高速バリア同期処理フロー

図 3-4に示した高速バリア同期処理フローを，キャッシュシステムを活用して高速に実行するには，同期フラグのメモリマッピングがポイントとなる。同期フラグのサイズは 1B で十分である。これは，同期フラグに格納するバリアポイント番号について，高速バリア同期処理フローで区別できなければならないバリアポイントの種類が，現在のバリア番号と次のバリア番号，それ以外のバリア番号のたかだか 3 つであるためである。この 1B の同期フラグをノード内の CPU 数である 16 個用意し，図 3-5に示す方式でメモリマッピングすることを考える。図 3-5で Flag[0]～[15]は 16 個の同期フラグ(1B×16)を意味する。

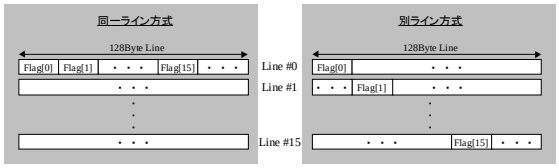

図 3-5 同期フラグのメモリマッピング方式

同一ライン方式は 16 個の同期フラグを1つのキャッシュライン(128B)にまとめる方式である。この場合，キャッシュ上で同期フラグとして占有される量は 1 ライン 128B 分のみであるが，図 3-4に示した処理フローで，各 CPU が同期フラグへ現在のバリアポイント番号を保存する毎に同一のキャッシュラインへのストア処理が発生する。そのため，ストア処理が発生するたびに同期フラグを含むキャッシュラインが無効化され各 CPU のキャッシュから追い出される。特に複数の CPU がほぼ同時刻にバリア同期処理を開始した場合には，キャッシュラインの取り合いが生じてしまい，処理性能が低下する可能性がある。

そこで，この問題を解決する方式が図 3-5右側に示した別ライン方式である。この方式は，各同期フラグを別キャッシュラインにわけることで，1 つのキャッシュラインに対するストア処理は，1 回のバリア同期処理で 1 回のみにでき，複数の CPU 間でキャッシュラインの取り合いが生じないようにした方式である。SR11000 モデル H1 では別ライン方式を採用している。

4 高速バリア同期(FBS)方式の評価

4.1 高速バリア同期(FBS)方式のバリア同期速度評価

最初に，FBS 方式によるバリア同期の起動・終結処理時間を SR11000 モデル H1 実機測定した結果を図 4-1に示す。縦軸は並列処理の起動と終結にかかるバリア同期処理時間の合計時間である。FBS 方式(別ライン)は 8CPU で 2.4μs，16CPU で 4.8μs であり，目標である SR8000 と同程度の処理時間を達成できる。16CPU 時には，ロック処理に較べて別ライン方式は約 5.8 倍高速である。また，同一ライン方式に較べても約 2.8 倍高速であり，特に CPU 数が多い領域でその差は大きい。

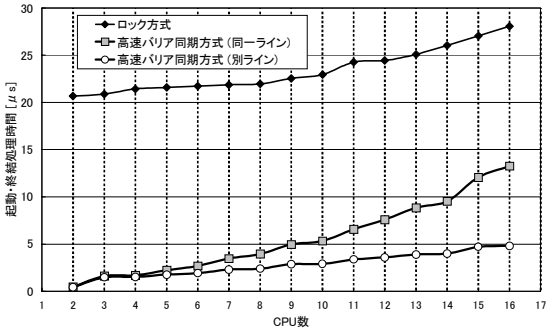

図 4-1 FBS 方式の起動・終結処理時間比較

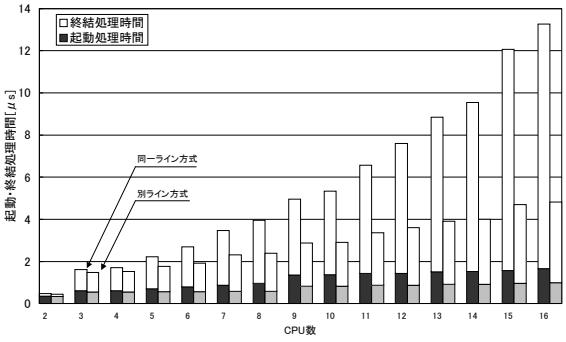

図 4-2 FBS 方式の起動・終結処理時間詳細

[1] プログラム中のバリア同期処理を行う部分をバリアポイントと呼び，複数のバリアポイントを区別するための番号をバリアポイント番号とする。

この原因を分析するために，同一ライン方式と別ライン方式の起動・終結処理時間の詳細を示したグラフが図4-2である。同一ライン方式は CPU 数が多い領域で終結処理時間が大きく増大している。これは全CPUから同一ラインに対するストア処理が集中し，キャッシュラインの取り合いが生じたことを裏付けている。

FBS 方式は，通常各 CPU がキャッシュシステム上で，他の CPU の同期フラグをチェックすることでバリア同期を実現しているため，同期フラグ更新に伴い発生するキャッシュ間データ転送性能がバリア同期性能に影響する。

図 4-3はバリア同期を行うCPUの物理CPUへのマッピング方法とバリア同期性能の関係を示した図である。2分木マッピングは図 4-4左側に示すように2分木方式で使用する物理 CPU 番号を決定し，逐次マッピングは図 4-4右側に示すように番号の若い順に使用する方法である。なお，図 4-4中の物理 CPU 番号は図 2-1中に記載のCPU 番号に対応している。図 4-3から，少ない CPU 数では逐次マッピングの方が起動・終結処理時間が短いことが分かる。これは，8 CPUまではMCMをまたがるデータ転送が生じないためで，さらに2 CPU時には共有L2キャッシュにより処理が1CPU チップ内におさまるためである。

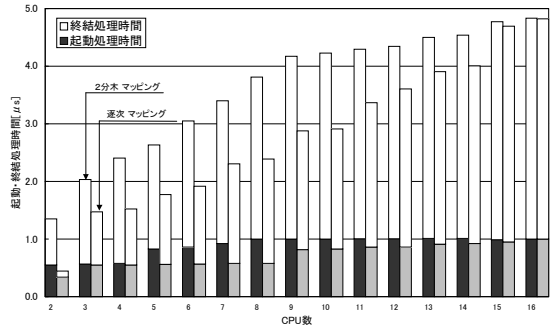

図 4-3 CPU マッピングと起動・終結処理時間の関係

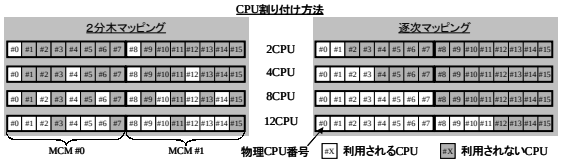

図 4-4 CPU 割付方法

4.2 高速バリア同期(FBS)方式によるアプリケーション高速化の効果

高速なバリア同期処理を実現する FBS 方式による並列実効性能への効果を，基本ループと並列化チューニングレベルの異なる2つのアプリケーションを例に示す。

図 4-5は，科学技術計算でよく使われる行列ベクトル積の基本ループ DAXPY(A(i)=B(i)+s*C(i))について，ループ長と16プロセッサ実行時のインキャッシュ並列実効性能の関係を示したグラフである。図 3-3で示したように，インキャッシュ時にはFBS方式による高速なバリア同期処理の効果が大きく，ループ長の小さい領域ではロック方式に較べて4倍程度，インキャッシュ領域の性能ピーク時にも1.3倍の DAXPY 実効性能が得られる。

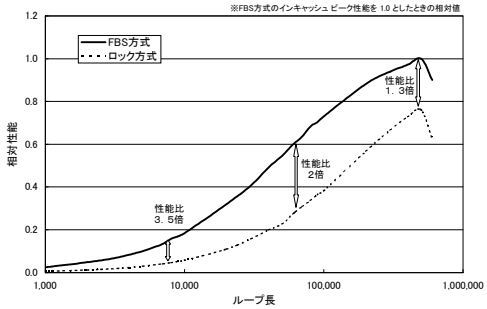

図 4-5 DAXPY (A(i)=B(i)+s*C(i)) インキャッシュ性能

次に 2 つの科学技術計算アプリケーションの並列実効性能を図 4-6，図 4-7に示す。各グラフの縦軸は，ロック方式によるバリア同期処理時の実行時間を 1.0 とした相対実行時間で，各棒グラフは演算時間と同期処理時間を積み上げている。棒グラフ上部の％表示は全実行時間に占める同期処理時間の割合である。横軸はバリア同期処理にロック処理とSR11000 モデル H1 で使用するFBS 方式を使用した場合を 1 セットとし，8CPU 実行(2 分木マッピング，逐次マッピング)と 16CPU 実行の順に 3 セット示した。

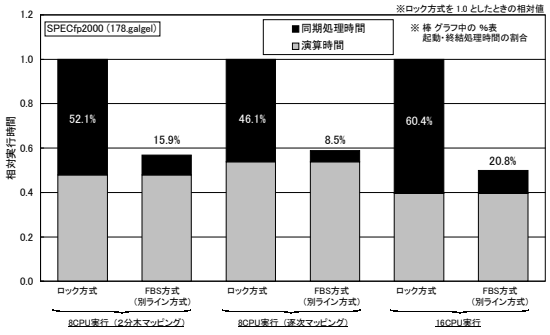

図 4-6 SPECfp2000 (178.galgel)性能比較

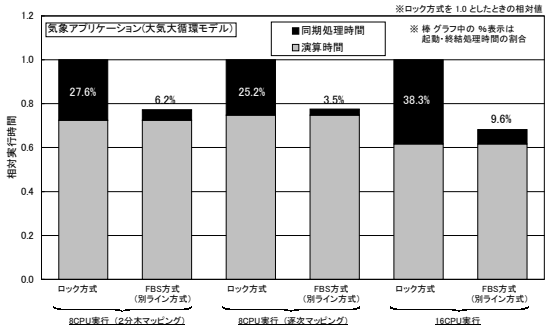

図 4-7 気象アプリケーション性能比較

図 4-6に示した SPECfp2000 (178.galgel)[6]は並列化チューニングを行っていないため同期処理時間が大きく，FBS 方式による高速なバリア同期処理により実行時間が約 1/2 になる。一方，図 4-7に示した気象アプリケーション(大気大循環モデル)は，並列化のためのチューニングを行っているが，バリア同期処理がロック方式に較べて約 5 倍高速な FBS 方式により，実行時間が 2～3 割削減される。同期処理時間の割合も 8CPU 実行で 4％程度，16CPU でも10％未満と並列実効性能を低下させる同期処理の影響を低く押さえており，FBS 方式のバリア同期処理により高い並列実効性能が得られている。

CPU マッピングの相違による8CPU 実行時の性能差は，図 4-6，図 4-7のいずれもバリア同期性能が約 1.6 倍高い(図 4-3参照)逐次マッピングの方が全実行時間に占める同期処理時間の割合が低い。しかし，表 4-1に示すように全実行時間はいずれも2分木マッピングの方が短い。これは，2分木マッピングの場合，CPU 当たりの L2 キャッシュ容量が倍となることで，演算時間が短縮された効果による。キャッシュ容量倍増による効果とバリア同期性能差の全実行時間への効果はアプリケーションおよびノード内ジョブ運用環境により異なる。今回の測定では，環境変数設定により CPU マッピングを変更して行った。

表 4-1 8CPU 実行時の CPU マッピングによる性能比較

	SPECfp2000(178.galgel)		気象アプリケーション	
マッピング方式	2 分木	逐次	2 分木	逐次
演算時間	84.1 %	110.7 %	93.8 %	106.6 %
同期処理時間	15.9 %	10.3 %	6.2 %	3.9 %
全実行時間	100.0 %	121.0 %	100.0 %	110.5 %

注) FBS 方式(別ライン方式)，2分木マッピング時を 100.0%とした相対値

5 関連研究

高い並列実効性能を達成するには，バリア同期処理の高速化が重要であることは広く知られており，ハードウェア実装[4)7)] とソフトウェア実装[2)3)]における方式について多くの研究がなされている。本稿で提案した FBS 方式と同じソフトウェア実装のバリア同期方式では，最も基本的なロック変数を用いた方式の欠点である，共有変数への排他的なアクセスを無くすことが基本的な改善方法である。この方法には，さらにバリア同期に参加する CPU 間に主従関係を規定し，ツリー状に順次バリア同期処理を進めることで CPU 数が増大した場合にも処理時間のオーダーが logN に収まる方式が各種提案されている。

その他，バリア同期処理単体の性能だけでなく，並列化時の各 CPU 間での負荷バランス均等化[8)]や，バリア同期処理を越えた投機処理[5)]により並列実効性能を向上するアプローチも研究されている。

6 まとめと今後の課題

本稿では，協調型マイクロプロセッサ機構により高い並列実効性能を達成したスーパーテクニカルサーバ SR8000 の後継シリーズの初代モデルである SR11000 モデル H1 の概要と特長を紹介し，大きな特長の1つである高い並列実効性能の実現に重要な並列処理の起動終結時のバリア同期処理時間を低減する手法を説明した。

本稿で示した高速バリア同期(FBS)方式は，特別なハードウェアを必要としないキャッシュシステムを活用したソフトウェアによるバリア同期方式であるが，専用ハードウェアを利用した SR8000 シリーズと同等のバリア同期性能を実現可能なことを示した。FBS 方式は，ロック変数を使ったソフトウェアによる方式に較べて，同じソフトウェアベースの方式にもかかわらず約 5.8 倍高速であり，アプリケーションの並列実効性能に影響を及ぼす同期処理時間を大幅に低減し，高い並列実効性能を実現できることを示した。

今後の課題としては，更なるシステム規模の拡大に対応した，ノード間も含む高性能なバリア同期処理手法の検討が挙げられる。

参考文献

1) Jeffrey P. Bradford and Seth Abraham, Efficient Synchronization for Multithreaded Processors, 4th International Symposium On High-Performance Computer Architecture (1998),
2) Mellor-Crummey, J.M. and Scott, M.L. : Algorithms for Scalable Synchronization Shared-Memory Multiprocessors, ACM Trans. on Computer Systems, Vol.9, No.1,pp.21-65 (1991)
3) N.-F. Tzeng and A. Kongmunvattana, Distributed Shared Memory Systems with Improved Barrier Synchronization and Data Transfer, Proc. 11th ACM Int'l Conference on Supercomputing (ICS), pp. 148-155 (7.1997)
4) M. T. O'Keefe and H. G. Dietz. Hardware Barrier Synchronization: Static Barrier MIMD (SBM). In Proc. of the 1990 Int'l Conf. on Parallel Processing (ICPP'90), Volume I, pp.35-42 (8.1990)
5) 松尾 治幸, 大野 和彦, 中島 浩.同期操作に対する投機的メモリアクセス機構 specMEM の改良.情報処理学会論文誌, Vol.43, No. 4, 855-865, (4.2002)
6) SPEC CPU2000, http://www.spec.org/cpu2000/
7) Y.Tamaki, N.Sukegawa, M.Ito, Y.Tanaka, M.Fukagawa, T.Sumimoto and N.Ioki : "Node Architecture and Performance Evaluation of the Hitachi Super Technical Server SR8000", Proc. of 12th International Conference on Parallel and Distributed Computing Systems, pp.487-493 (8.1999)
8) Jack Dongarra, Ian Foster, Geoffrey Fox, William Gropp, Ken Kennedy, Linda Torczon and Andy White : "Sourcebook of Parallel Computing" (2002)
9) 青木秀貴，高山恒一，中村友洋，松居昭宏，助川直伸: "SR11000 モデル H1 のノード構成とスケーラビリティ評価", 情処研報 ARC-155-8 (2003).
10) 松居昭宏，助川直伸，高山恒一，青木秀貴，中村友洋: "SR11000 モデル H1 における高精度性能分析手法", 情処研報 ARC-155-9 (2003).