7M−8

SNAILにおけるハードウェアデバッガ及びパフォーマンスモニタ*

寺田 純　笹原 正司　天野 英晴†

慶應義塾大学‡

1 はじめに

現在、SSS(Simple Serial Synchronized)型MIN(Multistage Interconnction Network)[1]を用いたマルチプロセッサSNAILを実装中である[3]。SNAILは動作の規則性が高く、ソフトウェアに対するデバッガビリティに優れている。しかし、この特性を活かして、SNAIL上での並列プログラム開発時のデバッグ環境を構築するためには、ハードウェアのサポートが必要である。また、SNAILの開発の目的の第一は、SSS-MINの性能評価にある。このためには、ハードウェアの各部をモニタし、統計を取るパフォーマンスモニタが必要である。ここでは、デバッガとハードウェアモニタの機能を併せ持つシステムの設計と実装を行う。

2 ハードウェアデバッガ及びパフォーマンスモニタ

大規模並列計算機には、基本となるノードとは独立して、他に以下の機能を持つ補助システムがあると非常に便利である。

- システム開発時
  - テストプログラムロード、簡単なモニタ
  - プログラムデバッグ支援
- システム稼働時
  - 故障診断、回復命令
  - 性能モニタ
  - OSを介さないプログラムのデバッグ支援

我々はこのような機能を併せ持つシステムUBA(Universal Backup Architecture)を提案している[2]。UBAは大規模マルチプロセッサを対象としている。このため、マルチプロセッサのクラスタ単位に、デバッガとモニタ用の情報を集計するプローブノードを設け、これらを互いに、ネットワークで接続した構成を持つ。

今回、実装したシステムはUBAシステムにおけるプローブノードに相当する。図1にシステムの概観を示す。

バスを監視すればプロセッサ間交信の全てを把握できる共有バス型マルチプロセッサに比べると、MINを用いたマルチプロセッサは、一般的にはメモリモジュールが多数あるので、交信の全体を把握するのが困難である。しかし、SSS型MINは基本的にはビットシリアル通信で、しかも全パケットが同期して入出力されるため、MINを用いた他のシステムに比べ、モニタが大幅に楽になっている。今回の実装では、Xilinx社のLCA XC3090 5個を用いて、MINのメモリモジュール側からパケットをモニタする。モニタには、加算器付きのカウンタ(図1のPacket Counter)が用いられ、メモリモジュールに対する16本のラインからのパケットに関する情報を集計する。

同期操作については、プロセッサ毎の集計も併せて取るため、MINのプロセッサ側からもモニタを行っている。これも専用のカウンタ(図1:Packet Monitor/Event Counter)を用いて集計している。しかし、16プロセッサの全てを同時にモニタすることは実装面積の点で困難であるため、4プロセッサを選んで測定する構造を取っている。

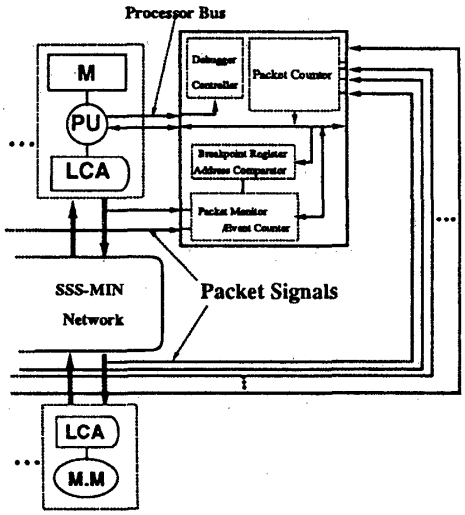

図1: プローブノードの構造と接続法

これに対し、プロセッサの動作のモニタは、ピン数制限の問題から、代表とするマスタプロセッサ1つのみが可能である。

2.1 ハードウェアデバッガ

SNAILは全プロセッサが同一クロックで動作し、パケットの授受、メモリモジュールのリフレッシュ等全てパケット同期信号(フレーム信号)に同期して行なわれる。このため、動作に再現性があり、ソフトウェア開発の上でのデバッガビリティに優れている。

さらに、プローブノードの働きにより以下の機能が提供される。

- 共有メモリに対するブレークポイント
  メモリ側に送られてくるパケットをモニタすることにより、指定された共有メモリに対してアクセスが行われた場合、全てのプロセッサの処理を停止させることができる。この処理は図1中のブレークポイントレジスタと比較器によって実現される。モニタしたパケットとブレークポイントレジスタの値を転送時に比較し、フレームに同期させて、割り込みまたは停止信号を発生させる。
- 1フレーム実行
  1フレーム分の処理を終える毎に全プロセッサに対し割り

*A hardware debugger and a performance monitor for SNAIL
†Jun TERADA, Masashi SASAHARA, Hideharu AMANO
‡Keio University

込みをかけ、停止させる。

2.2 パフォーマンスモニタ

アプリケーションの実行におけるパケットのコンバインの回数やルーティングの失敗の回数等のハードウェアの情報は、単にSNAILとSSS型MINの性能評価に用いられるだけでなく、プログラムの効率を改善するためのデータとして有効である。ここで実装したSNAIL用プローブノードでは、以下のモニタ機能を持つ。

- ネットワーク通過パケットの解析
  ルーティングの失敗回数はネットワークを通過した後でないと判別できないので、パケットに関する解析はネットワークの出口で行う。しかし、combineされたパケットが存在するため、ネットワークに入ってきたパケット数は計測できなくなってしまう。本来はネットワークの入口でも調査すればいいが、最大16プロセッサからなるシステムでは膨大なピン数を必要としてしまう。そこで、ネットワークに入るパケット(IN)はコンバイン機能を付加しないモードでパケット数を測定して求めることにする。このパケット数は以下の式で求めることができる。

  $$IN = OUT - CF$$

  ネットワークの出口で分かるパケットの情報は、ネットワークの出口に到達したパケットの総数(OUT)、ライトのパケット数(WR)、ルーティングに失敗したパケット数(CF)である。リードのパケット数(RD)は以下の式で求めることができる。

  $$RD = IN + CF - WR$$

- combine機能の効果

  combineが起きた回数(CB)も以下の式で表すことができる。

  $$CB = IN - OUT + CF$$

- 同期の回数

  同期の回数はtest&setとfetch&decの回数を調査する。

3 評価

現在、SNAILはブレッドボードが2プロセッサで稼働している。そこで、この2プロセッサでキャッシュとコンバインについての評価をとった。

実装したアプリケーションはN-Queen問題である。これはHopfield型に基づくニューラルネットワークのアルゴリズムを用いて、$N \times N$のチェスの格子に、N個のQueenをお互いにとられることのないように並べる[5]。

今回の実装では、SNAILのローカルメモリにプログラムを置き、共有メモリ上には実際に交換するデータのみが置かれている。2プロセッサでは通常MIN上での衝突や、コンバインの効果が現れ難い。そこで、今回は2つのメモリモジュールをインタリーブせず、共有データを単一メモリモジュール上に置いて実行した。さらに、SNAILは、オンチップキャッシュと共有メモリとの間の高速転送機能を持つが、ここではそれを利用せず、またPrefetch機能も用いていない。Queenの数$N = 32$とした。プロセッサのオンチップキャッシュの、ローカルメモリに対する使用の有無、SSS-MINのコンバイン機能の使用の有無についてアプリケーションの実行時間をモニタ機能により測定した結果を表1に表す。

この結果により次のことがわかる。

- ローカルメモリに対するキャッシュを使わない(つまり相対的にプロセッサが遅い)場合、ネットワーク上での衝突はほとんど起こらないため、コンバインの効果は現れない。
- ローカルメモリに対するキャッシュを使った場合は、ネットワーク上での衝突のため、コンバインを用いないと、実行速度の向上が小さい。しかし、コンバインを用いれば、性能は改善される。

表1: N-Queenの実行時間

	1PU	2PU (コンバインなし)	2PU (コンバイン付き)
cacheなし	77.734	39.630	39.630
cache付き	42.784	39.950	21.822

ただし、この結果は、メモリがインタリーブされないかなり特殊な場合であり、また、オンチップキャッシュとのデータ転送機能、Prefetch命令等の利用により、実際に16プロセッサで稼働した場合、かなり様子が異なるであろうことが予測できる。

4 おわりに

現在、SNAILはブレッドボードが2プロセッサで稼働しており、16プロセッサ用のPCBを実装中である。現在のプローブノードは2プロセッサにのみ対応するものだが、16プロセッサ用に拡張する予定である。また、共有メモリに対するブレークポイント、1フレーム実行をサポートするソフトウェアを含むデバッグ環境を実現する必要もある。

SNAILはメッセージ結合、Prefetch命令他、様々な機能を持つ。ATTEMPT-0[4]上で動作する並列プログラムの移植はきわめて容易であることから、16プロセッサのシステム上で実用的な並列プログラムに関して様々なデータを取り、結果に関して検討を加えていく予定である。

参考文献

[1] H.Amano, L.Zhou, and K.Gaye. SSS(Simple Serial Synchronized)スイッチングアーキテクチャ. 情報処理学会第44回全国大会, 1992.

[2] 山本淳二, 寺沢卓也, 天野英晴. 並列計算機のプログラムのデバッグのためのハードウェアサポート. 情報処理学会研究報告, Vol. 92, No. 97, 1992.

[3] 笹原正司, 寺田純, 小椋里, 周洛, 天野英晴. SSS-MINアーキテクチャに基づくマルチプロセッサプロトタイプSNAIL. 情報処理学会第46回全国大会, 1993.

[4] 鳥居淳, 竹本卓, 天野英晴, 小椋里. バス結合型並列計算機の交信用メモリの性能評価. 情報処理学会論文誌, 1992.

[5] 大和純一 鈴木響太郎, 天野英晴, 武藤佳恭. ニューラルネットワークを用いた最適化アルゴリズムの並列計算機への実装と評価. 電子情報通信学会技術研究報告, 1992.