並列シミュレーションマシン

3N-1

松下 智† 中田登志之† 梶原信樹† 浅野由裕‡ 小池誠彦†

†日本電気(株)、‡日本電気技術情報システム開発(株)

はじめに

現在、並列回路シミュレーション、並列同時故障シミュレーションなどを実行する汎用並列シミュレーションマシンのアーキテクチャ・ソフトウェアについて研究している[1]~[3]。これらのシミュレーションでは、問題自体が大規模でランダム・スパース性があったり、演算に際し頻繁に条件判断を行なうためベクトル型よりむしろMIMD型の並列マシンの方が向いていると考えられる。

本稿では、本マシンのハードウェア・ソフトウェアの設計思想について紹介したあと、システムの性能評価について報告する。

1 システム・アーキテクチャ

本システム上で実行されるアプリケーションは、モジュール分割法を用いた回路シミュレーション[2]や故障リスト分割法による並列同時故障シミュレーション[3]などの様に、基本アルゴリズムがProducer - consumerモデルに基づいていて、各々のプロセッサ間のデータ転送量が比較的少ないものを対象にしている。

本システムの構成を図1に示す。本システムは、64台のプロセッサ(以下pe)から構成される。peは、クラスタバスにより8台ずつ結合されクラスタを構成する。クラスタ間は蓄積型多段結合網で結合される。クラスタにはさらに、クラスタ間結合網をアクセスするためにネットワークアダプタ(以下nada)、および、ネットワークプロセッサ(以下nwp)が接続される。peには、MC68020(20MHZ), 4MBのメモリ、MC68882・weitekの浮動小数点アクセラレータ(以下fpa)を搭載している。nwpはfpaを除いてはpeと同様の構成を持つ。nadaは、ネットワークアクセスを管理するハードウェアであり、nwpに対しメッセージ通信ポートを提供する。さらにnadaは、バスを監視しクラスタ間のwriteアクセスを検知しwriteを代行するオートマトンも含む。各々のプロセッサにあるローカルメモリには、システムで単一なグローバルアドレス空間が与えられており、2ポートメモリとして他のプロセッサからアクセスできる。

本システムの対象とするシミュレーションでは、基本的にほとんどのプロセッサ間のアクセスはwriteアクセスとして実現することができる。ゆえに、複雑なプロトコルを要するクラスタ間のreadアクセスは、ソフトウェアで行なう。これに対し、クラスタ間のwriteアクセスは、ハードウェア化されパイプライン化されている。

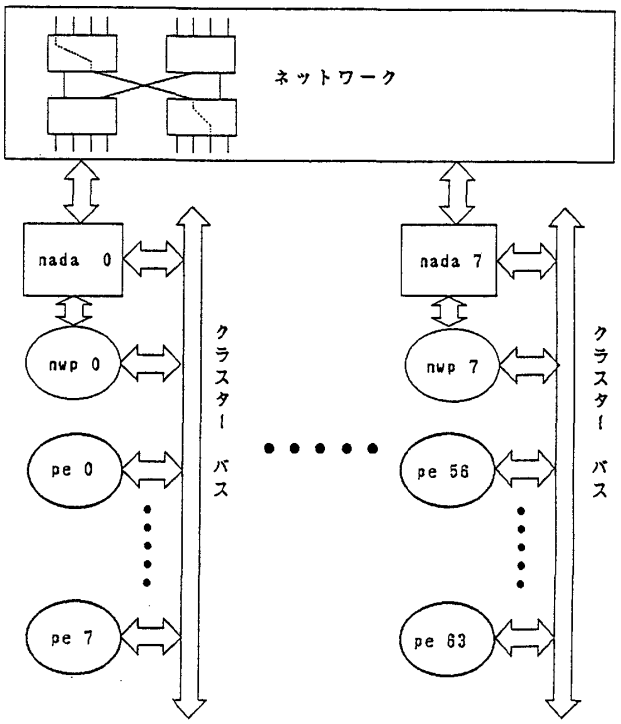

図1. システム構成

すなわち、クラスタ間writeしたdataがメモリに実際に書き込まれるまで5.2μSを要するが、パイプライン化されているため、表1に示すようにクラスタ内アクセスと遜色ないアクセス速度800nSが得られる。ただし、ネットワークからバスへのアクセス変換のためデータ量が多い場合パイプの停滞が起こり転送間隔は2.8μSとなる。

本システムは、DMAチャネルによってホストと接続される。

2 ソフトウェア設計思想

本システムの様なヘテロジニアスな通信環境の上に、汎用なシステムカーネルを構築する必要から、ソフトウェアは図2の様に階層構造をとった。また、階層化により、ソフトウェアの信頼性・稼働性の向上、システム立ち上げ期間の短縮を目指

Parallel Simulation Machine

Satoshi MATSUSHITA†, Toshiyuki NAKATA†, Nobuki KAJIHARA†, Yoshihiro ASANO‡, and Nobuhiko KOIKE†

†NEC Corporation, ‡NEC Scientific Information System Development Ltd.

した。さらに、この階層構造の上に効率のよいカーネルを作成することを目標とした。

3　カーネルソフトウェア

プロセッサ間で生じる非同期事象を統合的に取り扱うために、最下層であるメッセージ交換機構の上にvectored rpcを構築した。これは、いわば引数付きプロセッサ間ソフトウェア割り込みである。vectored rpcは、ベクタ番号によって排他型と非排他型に分けており排他制御および例外処理へ自然な形で対応しうる。この層を設けることで、ツリー状で効率の良いブロードキャスト、アプリケーションの起動、同期型および非同期型のリモートプロシジャコール、abortなどのプリミティブを系統的かつ簡潔に実現している。

デバッグ環境として、各peにデバッグモニタを登載した。nwpも含め６４台以上のプロセッサに端末を接続しハードウェア、カーネルのデバッグを行なうのは困難である。これに対し、メッセージ機構の上に簡潔な仮想端末機構を構築することで、デバッグの早い段階から特定のpeのモニタにホストからコマンドを投入できるようになった。仮想端末機能では、メッセージの作成・分解をホスト側で行うことで、ホストとのインタフェースをプログラム転送と一本化し、他の部分も、同一方針で簡潔かつ高信頼性化が達成された。

基本ソフトウェアの機能をまとめると

a)peの診断　b)シミュレータのロード、配布

c)シミュレーション結果のホストへの格納

d)エラー情報、ロギング情報のホストへ転送

e)相互排除、rpc、プロセッサ間同期の実現

f)クラスタ間リード　g)デバッグ環境の実現

などが挙げられる。

4　クラスタ間リード

peが他のクラスタのpeのメモリをリードした場合、リードを行なったクラスタのnadaが、そのpeに対してバスエラーを引き起こす。モニタのバスエラーハンドラは、スタックフレームを解析してリード要求のメッセージを生成し、そのクラスタのnwpを介してアクセスすべきメモリのあるクラスタのnwpへメッセージを送り、そのnwpによりリード処理が代行される。結果は、クラスタ間writeで要求したpeに直接返される。

5　クラスタ間リード・ｒｐｃの性能評価

表１に示す様に、本システムでは、クラスタ間readのオーバヘッドは極めて大きい。

これは、バスエラースタックフレーム生成のオーバヘッドを初め、ハングアップ防止のタイムアウト機構、メッセージのキューイングの排他制御、及び、nwpのメッセージハンドリングのオーバヘッドなどが原因である。

表1　メモリアクセスのサイクルタイム

ローカルメモリ	クラスタ内	クラスタ間ライト	クラスタ間リード
250nS	750nS	800nS (パイプライン)	240μS

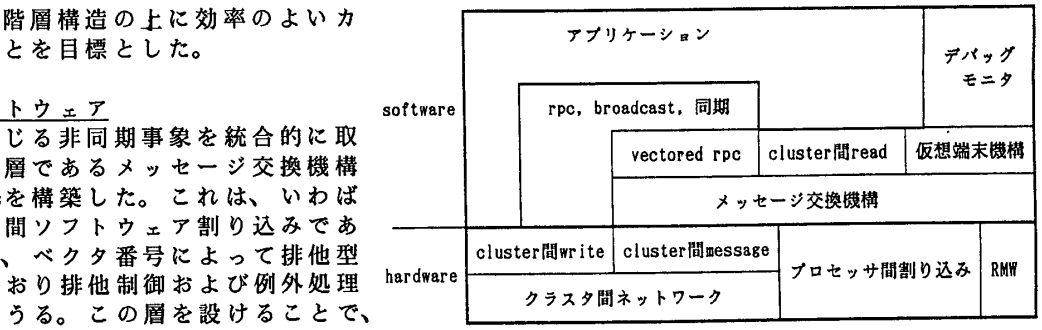

図２．ソフトウェア階層概念図

しかし、本システムが対象としているアプリケーションで生じるプロセッサ間のデータアクセスのほとんどは、シミュレーションの各フェーズの終了時のバーストアクセスであり、またシミュレーション開始時にデータ領域を特定できる静的なアクセスであることから、アルゴリズムの変更によって、これらをwriteアクセスに置き換えることができ、性能の低下を抑えることができる。

rpcについては、１引数のstab関数を同期型rpcで呼んだ場合を実測し、１回の呼び出しに640μSを要するとの値を得た。従って、3回以上(Ｎ回とする)のクラスタ間readを、1つのrpcによって起動されるwriteに変更する事で速度向上が図られる。その場合、速度向上比は、

85.7 / (1 + 228.6 / N)) --- (1)　となる。

このようなrpcを用いた変更は、前述の性能向上法に比べアルゴリズム的な変更が小さく、実際的であろう。

6　おわりに

本システムでは、シミュレーションプログラムに適合した、高並列化向けアキテクチャを採用している。これをソフトウェア的に補い疑似的な共有メモリ環境を提供している。これにより、逐次マシンで動作しているアプリケーションのアルゴリズムをあまり改造する事なく並列化・実装した後、性能の向上、さらには、高並列マシンに適応したアルゴリズムの開発・検証をするといったインクリメンタルなソフトウェア作成が可能である。

また、カーネルが様々なプリミティブを提供する事で汎用に近い並列環境が提供された。今後、ソフトウェアのチューニングを行なったうえ、アプリケーションまで含めたシステムの評価を行なうと同時に、さらに汎用化を押し進めていく。

参考文献

[1] N.Koike and K.Ohmori "MAN-YO : A Special Purpose Parallel Machine for Logic Design Automation", proc. 1985 ICPP, pp.583-590

[2] 小池ほか, "並列回路シミュレータプロトタイプのアーキテクチャ", 情処第３５回全国大会予稿集６Ｃ－７

[3] 中田ほか, "統合ＤＡ用専用並列マシンMAN-YOにおける並列同時故障シミュレータ", 情処第３６回全国大会予稿集５Ｙ－８