

Comet-VIA の評価

小林伸治、陣崎 明

新情報処理開発機構 並列分散システム富士通研究室
〒211-8588 川崎市中原区上小田中 4-1-1
E-mail: {koba,zinjin}@flab.fujitsu.co.jp

Gigabit Ethernet 等の高速ネットワークを利用した並列計算クラスタシステムを実現するうえで鍵となるのは高速な IP 通信システムである。しかし、従来の通信アーキテクチャでは 1Gbps 超の IP 通信を行うのは困難である。そこで、我々は高速 IP 通信を実現するために Comet 通信アーキテクチャを提唱している。

本論文では、Comet 通信アーキテクチャのホスト-アダプタインタフェースである Comet-VIA を詳しく説明する。そして、Comet-VIA のプロトタイプ実装の性能を評価し、その結果に基づいてストリームプロセッサを使用した Comet 通信アーキテクチャの性能予測を行った。その結果、Comet 通信アーキテクチャで 1Gbps 超の IP 通信を実現可能であることを確認した。

Comet-VIA, Comet, VIA, IP, ストリームプロセッサ

An Evaluation of Comet-VIA

Shinji Kobayashi, Akira Jinzaki,

Parallel and Distributed Systems Fujitsu Laboratory, RWCP
1-1, Kamikodanaka 4-Chome, Nakahara-ku, Kawasaki, 211-8588 Japan
E-mail: {koba,zinjin}@flab.fujitsu.co.jp

High-speed IP communication system is the key technology to realize parallel computing cluster systems using high-speed networks such as Gigabit Ethernet. However, it is difficult to realize the 1Gbps-over IP communication with the existing communication architecture. For this purpose, we are proposing the Comet communication architecture to realize fast IP communications.

In this paper, we describe the Comet-VIA which is the host-adaptor interface of the Comet communication architecture. Then, we evaluated the performance of the Comet-VIA prototype implementation, and we estimated the performance of the Comet communication architecture using the Stream Processor based on the results. We confirmed that we can realize 1Gbps-over IP communication using the Comet communication architecture.

Comet-VIA, Comet, VIA, IP, Stream Processor

1. はじめに

Gigabit Ethernet や WDM といった高速ネットワーク技術により、1Gbps 超のバンド幅が簡単に得られるようになった。数年前に並列計算クラスタのノード間接続に用いていた専用ネットワークと同程度以上のバンド幅である。このようなネットワークを利用して並列計算クラスタを構築するというのはごく自然な発想である。

並列計算クラスタを構成するネットワークにおいては、低遅延、高バンド幅を実現するために専用のネットワークと専用の通信プロトコルを使用するのが一般的である。現在広く使われている IP (Internet Protocol) は並列計算クラスタには重すぎると考えられてきた。実際、1Gbps のバンド幅を有する Gigabit Ethernet でも通常の UNIX ワークステーションから利用するとネットワーク処理に専念しても数百 Mbps 程度のバンド幅しか利用できないことが多い。ソフトウェアで行うプロトコル処理がボトルネックとなり、最新の高速プロセッサをもってしても 1Gbps 超の IP 通信を実現するのが困難なのである。このような状況では、IP での利用を前提として開発されている高速ネットワークを利用して並列計算クラスタを構築しても期待した性能が得られない可能性がある。

この状況を打破するためには通信アーキテクチャ全体を再検討する必要がある。我々はネットワークアダプタでプロトコル処理を行うインテリジェントアダプタ方式と、ユーザプロセスが直接ネットワークアダプタを操作するユーザレベル通信とを組み合わせ、1Gbps 超の IP 通信を実現する Comet 通信アーキテクチャを提唱している。Comet 通信アーキテクチャを実現すれば、現在インターネット向けに急速に高帯域化しているバックボーンネットワークを IP で利用するなど自由度の高い並列計算クラスタを構築できると考えている。

Comet 通信アーキテクチャのホスト-アダプタインタフェースとして、我々は Comet-VIA を提唱している [1][2]。Comet-VIA はユーザレベル通信の一種である VIA (Virtual Interface Architecture) [3] を基にインターネット向けに拡張した方式である。

本論文では、高速な IP 通信を行ううえでの問題点を指摘し、それを解決するために我々が提唱している

Comet-VIA の方式を説明し、評価を行う。まず第2節では従来の通信アーキテクチャを分析して高速な IP 通信を行う上での問題点を指摘し、それを解決するための Comet 通信アーキテクチャを説明する。第3節では Comet 通信アーキテクチャのホスト-アダプタインタフェースである Comet-VIA 方式の説明を行う。第4節ではプロトタイプ実装を用いた性能評価を行い、開発中の Comet-NP を用いた Comet 通信アーキテクチャの性能予測を行う。

2. コンピュータ通信の高速化

2.1. 従来のアーキテクチャ

これまでのネットワーク通信アーキテクチャはもともと UNIX における IP の実現のなかで発展してきたため、そのアーキテクチャには UNIX の流儀が色濃く反映されている。アプリケーションが用意したデータはカーネルにコピーされた後にカーネル内でプロトコル処理を行ってからネットワークアダプタに渡され、ネットワークに送出される (図 1)。

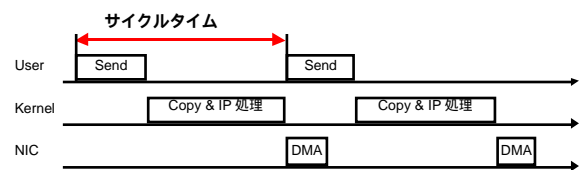


図 1: 従来の送信処理の流れ

データ転送の DMA と他の処理とは並行動作可能である。データ長、プロセッサ速度にも依存するが、一般にはデータコピーやコンテキストスイッチのオーバーヘッドが大きく、図のように DMA 時間よりもソフトウェア処理に要する時間が長くなる。ネットワークアダプタへの DMA 以外をすべてソフトウェア処理しているため並行動作できる部分が少ない。最大バンド幅を決定する 1 パケット当たりのサイクルタイムはユーザレベルおよびカーネルでの処理時間に依存することになる。

従来のアーキテクチャには、ネットワークアダプタが単純な構成で済み、ネットワークアダプタの排他制御もカーネルで一元管理できるといった利点がある。しかし、上述のようにオーバーヘッドが大きく、1Gbps 超のネットワークに対応するのは難しい。たとえば、UltraSPARC-II 450MHz を搭載し Solaris 7 を実行するワークステーションで UDP/IP データグラムをループ

バックインタフェースに対して連続送出する実験を行ったところ、データグラムサイズ 1440Bytes で得られたバンド幅は 40MB/s であった。

BSD/OS で UDP/IP 処理を分析した結果、1000Bytes の UDP/IP データグラムの処理に約 16,000 クロックを要していることがわかった。たとえば 1Gbps のネットワークを使い切るには 1GHz 以上の Pentium が 100%の能力を必要とする (図 2)[4]。

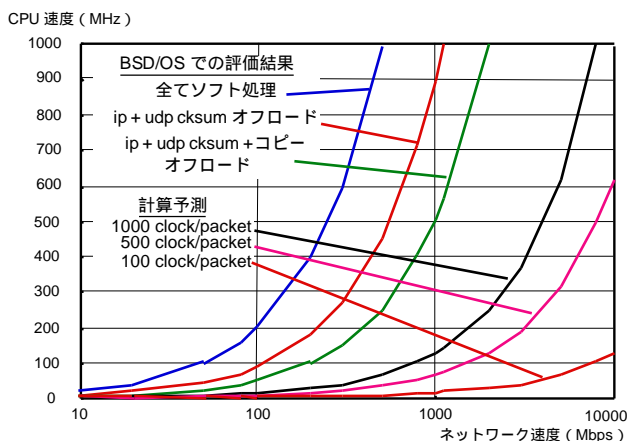


図 2: BSD/OS での UDP/IP 処理性能

受信処理にも同程度のプロセッサ能力を必要とすること、ホストプロセッサが通信処理にかかりっきりでは本来の処理が行なえないことを考慮すると、従来の通信アーキテクチャで 1Gbps を超えるネットワークに対して IP 通信を行うのは非常に困難であることがわかる。

2.2. 通信の高速化

コンピュータ通信を高速化するために従来行われているアプローチを整理する。

• 軽装プロトコル

軽装プロトコルは限られた条件や特定の目的を前提とした軽い通信プロトコルを定義し、これを用いることで処理のオーバーヘッドを削減するアプローチである。XTP[5]、AM[6]、FM[7] などのプロトコルが開発され、利用されているが、Internet Protocol とは異なり相互接続性が低いため、利用範囲が限られる。高速化のアプローチとしてプロトコルそのものを対象とする場合、新プロトコルを標準にするか、標準プロトコルとの相互通信方式を用意する必要がある。

• ユーザレベル通信

ユーザプロセスとカーネルとの間のコンテキストスイッチ、データコピーのオーバーヘッドを削減するために

ユーザプロセスが直接ネットワークアダプタを操作する通信方式をユーザレベル通信と呼ぶ。特に、ユーザアプリケーションのデータ領域から一時的なバッファに一度もコピーすることなく直接ネットワークアダプタに転送する方式を Zero Copy 方式と呼ぶ。

ユーザレベル通信ではユーザプロセスが直接ネットワークインタフェースを操作するため、ユーザプロセス間で排他制御を行う必要が生じる。VIA ではこの問題を解決するためにアダプタが複数のコンテキストを持てるようにし、これらを VI (仮想インタフェース) としてユーザプロセスに提供する。各ユーザプロセスは VI を専有できるので排他制御が不要になる。

ユーザレベル通信ではコンテキストスイッチやデータコピーのオーバーヘッドは削減できるが、プロトコル処理のオーバーヘッドそのものは削減できない。

• インテリジェントアダプタ

ネットワークアダプタにマイクロプロセッサなどの処理機能を装備し、ホストプロセッサが行っていたプロトコル処理をオフロードする方式である。この方式の利点はホストプロセッサの通信処理を削減するばかりでなく、ホストプロセッサ処理とアダプタ処理の処理量を同じ程度に調整することで並行動作の効果を最大限にあげられる点である。その一方で、ホストプロセッサが年々高速化するのに対してアダプタの制御プロセッサはコストの面から左程高速化できないため、結局のところホストプロセッサで処理したほうが高性能になる場合が多いという問題がある。

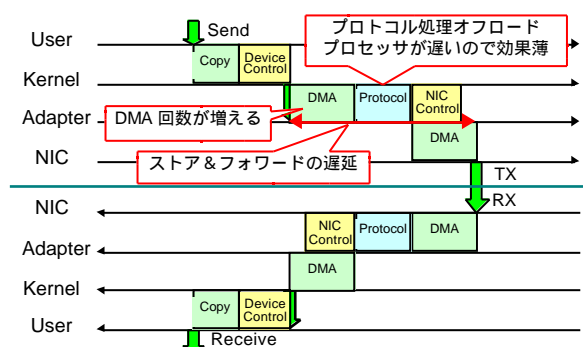


図 3: インテリジェントアダプタの通信処理

以上をまとめると、接続性を考慮する限り軽装プロトコルは問題が多く、ユーザレベル通信などのホストソフトウェアだけによる対処ではかなりの改善が期待できるもののプロトコル処理のオーバーヘッドは残る。インテリジェントアダプタはホストプロセッサの処理を

オフロードすると共にパイプライン処理によるスループットの向上が期待できる反面、アダプタプロセッサの処理能力がネックとなる可能性が高い。

これらの限界を打破するため、我々はユーザレベル通信などのホストソフトウェアの改善とインテリジェントアダプタを合わせた高速化方式として Comet を提案している。アダプタプロセッサが処理ネックとならないように、ネットワーク転送クロックレートでパケット処理を行うための通信処理専用プロセッサを用いる点に特徴がある。

2.3. Comet 通信アーキテクチャ

Comet はインテリジェントアダプタとユーザレベル通信を組み合わせることで高速な処理を可能にする通信アーキテクチャである。通常のインテリジェントアダプタとは異なり、DMA 転送中にパケット処理を行う SP (Stream Processor) [8] を用いることで、従来カーネルで行っていたプロトコル処理を DMA 転送中に行うことができる。Comet 通信アーキテクチャでの送信処理の流れは図 4 のようになる。

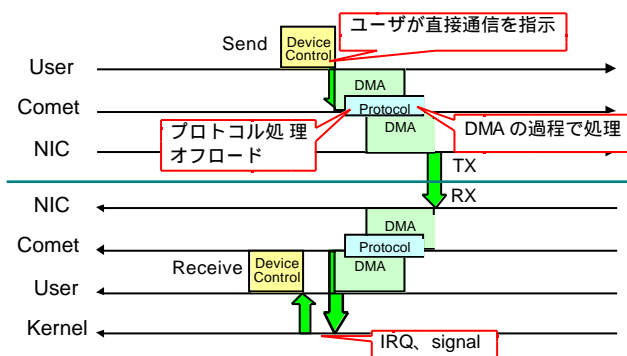


図 4: Comet での通信処理の流れ

ホストプロセッサのソフトウェアで行う必要があるのは送信データの場所と長さをアダプタに通知する程度のごく軽い処理だけになる。図のように各要素が並行動作できるため、サイクルタイムはユーザレベルでの送信処理、DMA 設定時間、DMA 時間の最大時間となる。一般には、パケットが長くなると DMA 時間がサイクルタイムとなり、短いパケットでは DMA 設定時間がサイクルタイムとなる。

従来の通信アーキテクチャと比較してみると、DMA 設定時間または DMA 時間の方がホストプロセッサでのソフトウェア処理よりも短ければ高いバンド幅が得られることになる。ホストプロセッサは年々高速化しているが、アダプタに搭載できる制御プロセッサはそ

れと比較すると低速である。Comet 通信アーキテクチャが有効であることを示すには、アダプタ上の制御プロセッサによる DMA 設定時間や DMA 時間がどの程度になるのかを検討する必要がある。

3. Comet-VIA

3.1. VIA

前節で述べたように、Comet 通信アーキテクチャでは通信処理にカーネルは介在せず、ユーザレベルのアプリケーションが直接ネットワークアダプタとデータを交換する。すなわちユーザレベル通信を行う。

ユーザレベル通信としてはさまざまな方式が提唱されている。なかでも、仮想化したネットワークアダプタ (仮想インタフェース、VI) を各ユーザプロセスが専有する VIA のような方式はモデルが単純で適用範囲も広い (図 5)。

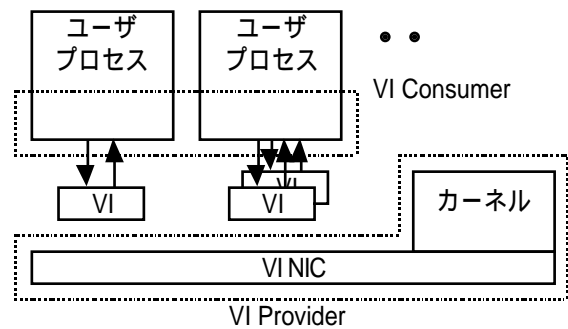


図 5: VIA のモデル

VIA では、ディスクリプタによって制御する DMA エンジンを持つインタフェースとしてネットワークアダプタを仮想化している。各 VI は個別のディスクリプタを格納するキューをメインメモリ上に持ち、キューにディスクリプタを追加したことをアダプタに知らせるドアベルも持つ (図 6)。

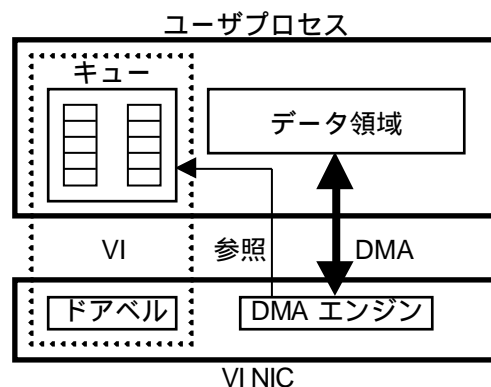


図 6: VIA の VI

3.2. VIA の Internet 拡張

VIA は高速なユーザレベル通信を実現するうえで有効だと考えられるが、Comet が対象としている Internet での通信、すなわち IP 通信に適用するには問題がある。IP はネットワークの階層モデルにおける第 3 層に位置し、その上で様々な通信プロトコルが利用される。また、IP 自体も現在広く利用されている IPv4 (IP version 4) から IPv6 (IP version 6) への移行が始まったところである。しかし、VIA ではユーザアプリケーションとネットワークアダプタとのインターフェースが 1 種類に限られるため、さまざまなプロトコルが利用される IP に柔軟に対応できない。

Comet-VIA はさまざまな通信プロトコルをサポートするために VIA を拡張した方式である。異なるプロトコルを同時に利用できるよう、複数の種類の VI を提供できるように拡張を行った (図 7)。

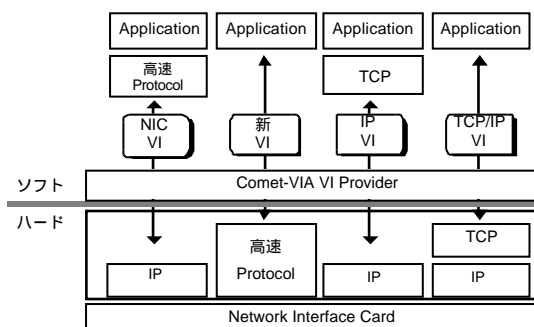


図 7: Comet-VIA のモデル

たとえば、MPI や PVM といった並列計算ライブラリを実現する場合、Comet-VIA では IP 層の処理は Comet ネットワークアダプタが担当する。この処理では各ライブラリが使用するプロセッサノード番号等から IP アドレスに変換する必要が生じる。この変換は単純なテーブル検索で実現できるが、ノード番号と IP アドレスとの対応方式や、どのようにしてノード番号を指定するかはライブラリ毎に異なる。すなわち、IP 層での処理といってもそのために必要な処理は利用するライブラリ毎に異なってくる。オリジナルの VIA のように VI の種類が 1 つしかない場合、ネットワークアダプタはホストから渡されたデータの種類を判別して適切な処理を選択する必要が生じる。一方、Comet-VIA では MPI を IP 通信させるための VI と PVM を IP 通信させるための VI を別々に用意して提供することができる。アプリケーションは VI を要求するときに適切な VI の種類を指定すればよい。ネッ

トワークアダプタは各 VI が作成された時点で、その VI を通過するデータに必要な処理を知ることができる。

3.3. VI の構成

Comet-VIA では、キューとドアベルからなる VI の構成も再検討した。ユーザプロセスから VI NIC に渡す必要がある制御データは、送受信データが存在するアドレスとその長さである。一般には 1 つのパケットが複数のデータ領域に分割されている場合にも対応する必要がある。このため、VIA では DMA 用のディスクリプタをメインメモリ上に用意し、そのディスクリプタのアドレスをドアベルを介して通知する方式を採用している。VI NIC の DMA エンジンはこのディスクリプタを参照して DMA を行う。VI NIC は PCI 等の I/O バス上に置かれるため、この参照は I/O バスを介した読み出しアクセスとなる。マスタが要求を出してからターゲットがデータを準備し、準備が整ってからデータ転送を行うという手順になるため、読み出しアクセスは書き込みアクセスに比べてオーバーヘッドが大きい。PCI のスロット数を増やすためにバスブリッジが存在するとオーバーヘッドはさらに大きくなる。

そこで、Comet-VIA ではメインメモリ上にディスクリプタを用意する代わりに、VI NIC 上の FIFO にデータアドレスやデータ長を書き込む方式を採用した(図 8)。

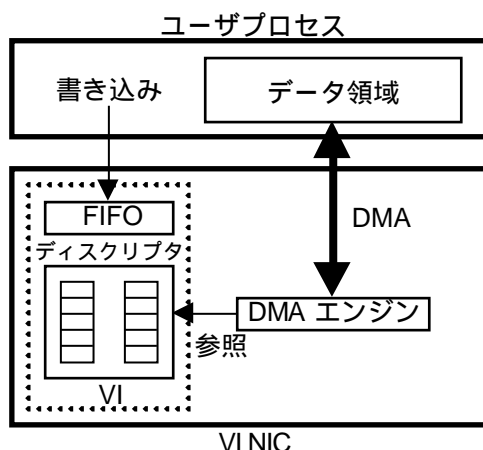
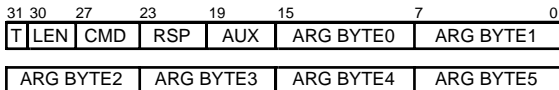


図 8: Comet-VIA の VI

3.4. コマンド体系

前節で述べたように Comet-VIA では VI NIC 上の FIFO にユーザプロセスが書き込むことで操作を行う。送信バッファ、受信バッファ、その他制御情報を受け渡すために図 9 のようなコマンド体系を定義した。FIFO は 32bit 長である。



...
T 0: Command
1: Response
LEN Length (bytes)
CMD Command code
0000: SYNC
0100: SEND_BUF
0101: RECV_BUF
1001: VI_CTL
etc.
RSP Response code
AUX Auxiliary code

図 9: Comet-VIA のコマンド体系

コマンドに対するレスポンスは FIFO を読み出すこと
によって得られる。

例として、送信データ領域を通知する SEND_BUF の
コマンド/レスポンスのフォーマットを図 10 に示す。

| SEND_BUF Command | | Response | |
|----------------------------|--|----------|--|
| T | 0 | T | 1 |
| LEN | 2 (3 for 64bit PCI) | LEN | 2 (3 for 64bit PCI) |
| CMD | 4 | CMD | 4 |
| RSP | 0 | RSP | 0: Normal completion 1: Abnormal completion |
| AUX | 1: Packet head flag 2: Packet tail flag | AUX | 1: Packet head flag 2: Packet tail flag |
| ARG0 | Buffer length 0 | ARG0 | Buffer length 0 |
| ARG1 | Buffer length 1 | ARG1 | Buffer length 1 |
| ARG2 | Buffer address 0 | ARG2 | Buffer address 0 |
| ARG3 | Buffer address 1 | ARG3 | Buffer address 1 |
| ARG4 | Buffer address 2 | ARG4 | Buffer address 2 |
| ARG5 | Buffer address 3 | ARG5 | Buffer address 3 |
| (ARG6..ARG9 for 64bit PCI) | | | |

図 10: SEND_BUF コマンドフォーマット

1 つのパケットが複数のデータ領域から構成される場
合、領域の数だけ SEND_BUF コマンドを発行する。
データ転送が完了すると VI NIC は対応する
SEND_BUF レスポンスを返す。それによってホスト
は領域を解放することができる。

VI の実体である FIFO やディスクリプタは NIC に存
在するが、これらの管理はカーネルが行う。カーネル
はアクティブな VI のリストを管理しており、アプリ
ケーションからの要求に従って NIC に対して VI_CTL
コマンドを発行し、VI 生成等の処理を行う。

たとえば VI の生成は次のような手順で行う。

1. アプリケーションが ioctl でカーネルに VI 生成を
要求。VI の種類を表すコードも渡す。
2. カーネルは未使用の VI 番号を検索し、その VI 番
号と VI の種類コードを引数に VI_CTL コマン
ドを NIC に発行する。
3. NIC は FIFO やディスクリプタを設定し、
VI_CTL レスポンスを返す。

4. カーネルが VI_CTL レスポンスを受け取る。

5. ブロックしていた ioctl を再開する。

このように、VI の生成等の操作にはカーネルが介在
することになるが、データの送受信に比べてこれらの
操作が行われる頻度は非常に少ないためオーバーヘッド
にはならない。

4. 評価

Comet-VIA で実際に高速な IP 通信を実現できるかど
うかを検討する。まず自作のインテリジェントアダプ
タに Comet-VIA のプロトタイプを実装し、性能評価
を行った。その結果を 4.1 節に示す。本実装ではプロ
トコル処理をインテリジェントアダプタの汎用プロセ
ッサで行うため、プロトコル処理の高速化は見込めな
い。しかし、処理の内訳を分析することで Comet-VIA
方式を採用する Comet 通信アーキテクチャの処理時
間を見積もることができる。そこで、4.2 節ではスト
リームプロセッサを使用した Comet 通信アーキテク
チャの性能予測を行う。

4.1. インテリジェントアダプタの性能評価

評価に使用したインテリジェントアダプタは制御プロ
セッサとして Intel i960RN を搭載している。i960RN
のプロセッサは 100MHz で動作する 32bit RISC であ
る。ネットワークは Gigabit Ethernet NIC (Network
Interface Card) を使用している。ホストプロセッサ上
のデータは一旦 i960RN メモリに DMA され、そこで
i960RN が IP 処理等を行った上で Gigabit Ethernet NIC
の DMA を起動し、ネットワークに送られる (図 11)。

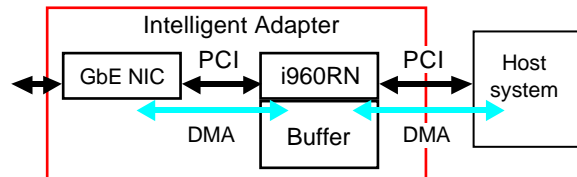


図 11: インテリジェントアダプタ

ホストシステムとしては UltraSPARC 250MHz を使用
した。OS は Solaris 2.6 である。

Comet VIA の処理オーバーヘッドを評価することを目的
として、Illinois Fast Messages (FM) を基にした軽装ブ
ロトコルを IP 上に実現して通信する VI (FM/IP VI)
を実装し、測定を行った。FM/IP VI ではアダプタが
IP 層での処理を担当する。1 つのパケットを送受信し
たときの処理時間の内訳を表 1 に示す。

表 1: 処理時間の内訳

| 処理内容 | | 処理時間 (μsec) |
|------|------------------------|-------------|
| 送信 | Host 送信処理 | 2.11 |
| | Host -> Adapter DMA 起動 | 7.08 |
| | DMA 完了待ち | 5.60* |
| | パケット処理 | 6.00 |
| | GbE NIC 送信 DMA 起動 | 2.52 |
| 受信 | GbE NIC 受信検出 | 2.04 |
| | パケット処理 | 2.20 |
| | Adapter Host DMA 起動 | 4.76 |
| | DMA 完了待ち | 3.40* |
| | Host 受信処理 | 1.16 |

* データ長に依存するが、ここでは最短値

Comet-VIA ではプロトコル処理をアダプタで行うためホスト側に必要な処理が少ない。送信に 2.11 μ秒、受信に 1.16 μ秒を要するだけである。これはパケット長に依存しない。Ethernet の MTU である 1500Bytes で計算すると、710MB/s 程度まではホストプロセッサでの処理がボトルネックとならないことがわかる。

本インテリジェントアダプタはプロトコル処理を 100MHz 動作の i960RN のソフトウェアで行うため、各処理に時間がかかっている。たとえば、送信時の IP ヘッダ、Ethernet ヘッダの生成を行うパケット処理には 6 μ秒かかっている。また、DMA ディスクリプタを設定して起動する処理も時間がかかることがわかる。送信時に i960RN が処理する時間の合計は DMA 完了待ちを除いても 15.6 μ秒となる。1500Bytes をこの時間で処理できたとしても 96MB/s しか実現できない。インテリジェントアダプタ方式ではアダプタ上のプロセッサの処理能力に限界があることがわかる。

DMA 時間はパケットが一定の長さ以下ではほぼ固定の処理時間となり、それ以上ではパケット長に比例した処理時間となる。表 1 に示した時間は固定長の時間である。パケット長に比例する時間はバスのバンド幅と使用率から計算できる。

以上をまとめると、ホストプロセッサの処理はプロトコル処理オフロードにより 5Gbps 以上のネットワークに対応可能となる。1GHz のプロセッサであれば 50% の CPU 利用率で 10Gbps ネットワークを利用可能である。一方、プロトコル処理を行うインテリジェントアダプタは従来のような制御プロセッサを用いる限り高性能を期待できず、1GHz クラスのプロセッサが必要となる。最近発表されているいわゆるネットワークプロセッサ[9][10] や VLIW アーキテクチャによる組込プロセッサ[11][12] は一つの候補となりうる。

4.2. Comet アーキテクチャの性能予測

Comet 通信アーキテクチャでは制御プロセッサとして独自アーキテクチャの SP を利用する。SP はパケットデータを一旦バッファに格納してから処理するストアアンドフォワード方式ではなく、データ転送中に通信プロトコル処理を行う点に特徴がある。現在、送受信用に SP を 1 つずつ搭載したネットワーク処理プロセッサ、Comet-NP を開発中である。Comet-NP の外部バスは 64bit、66MHz の PCI であり、バンド幅は 533MB/s となる。Comet ネットワークアダプタはこの Comet-NP を採用し、みかけ上一度の DMA で IP プロトコル処理を行いながらホストからネットワークにデータ転送を行うことができる (図 12)。

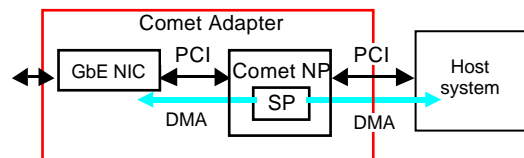


図 12: Comet ネットワークアダプタ

前節で測定したインテリジェントアダプタ方式と比べると DMA が 1 回で済む上、パケット処理の時間が DMA 時間に隠ぺいされる。すなわち、表 1 の『DMA 完了待ち』と『パケット処理』の時間を無くすることができる。

DMA 起動にかかる時間を表 1 の実測値とし、DMA 時間を PCI のバンド幅と使用率から算出すれば Comet-NP を使用した Comet ネットワークアダプタの最大バンド幅を計算できる。PCI バスの使用率を 50% として計算した送信バンド幅を図 13 に示す。

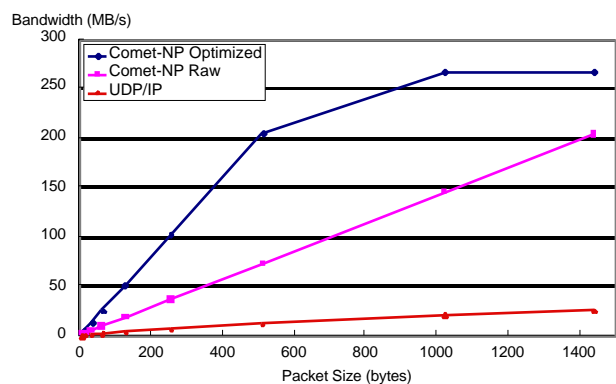


図 13: Comet-NP 性能予測

Comet-NP Raw はホスト側 DMA の設定をソフトウェアで行った場合、Comet-NP Optimized はホスト側 DMA の設定を Comet-NP のハードウェアで行った場

合である。比較対象として、Gigabit Ethernet アダプタを OS のプロトコルスタックから使用した場合の UDP/IP のバンド幅を示した。測定には前節の測定と同様、UltraSPARC 250MHz、Solaris 2.6 を使用した。

OS による UDP/IP は Gigabit Ethernet のバンド幅を使いきれないのに対して、Comet-NP はホスト側 DMA 設定をソフトウェアで行っても Gigabit Ethernet のバンド幅を超える能力を持っていることが分かる。最適化を行い、ホスト側 DMA 設定をハードウェアで行うようにするとさらに性能が上がる。パケット長 1000Bytes 程度以上では PCI バス (533MB/s、使用率 50%) がボトルネックとなる。

以上の結果から、従来の通信アーキテクチャでは困難な 1Gbps 超での IP 通信を Comet 通信アーキテクチャで実現可能であることを確認できた。

5. まとめ

インターネット向けの高速ネットワークを利用して並列計算クラスタを構成するためには高速な IP 通信の実現が鍵であることを述べた。1Gbps を超えるバンド幅で IP 通信を行うのは従来の通信アーキテクチャでは困難であることを指摘し、それを解決する Comet 通信アーキテクチャについて説明した。特に、ホスト-アダプタインタフェースとして我々が提唱している Comet-VIA について詳しく説明し、Comet-VIA が IP で利用されるさまざまな通信プロトコルに対応できる枠組みを提供していることを述べた。自作インテリジェントアダプタと Gigabit Ethernet を用いたシステムに Comet-VIA のプロトタイプを実装して性能を評価し、処理の内訳時間を分析した。最後に、その分析結果に基づいて Comet-NP を使用したネットワークアダプタの性能予測を行い、Comet 通信アーキテクチャで 1Gbps 超の IP 通信を実現可能であることを確認した。Comet 通信アーキテクチャは 10 Gigabit Ethernet、SONET 等の超高速ネットワークや Infiniband など次世代 I/O ネットワークに対応できる。

今後は Comet VIA としては TCP/IP、UDP/IP を実現する VI を開発し、MPI ライブラリを動作させてアプリケーション通信の性能評価を行う予定である。Comet アダプタとしては Comet-NP の開発を進めているが、これを用いて OC48c (2.5Gbps) を用いたクラスタシステムを実現する。

参考文献

- [1] 小林伸治, 陣崎明: Comet における VIA 的インタフェース, 信学技報, Vol. 98, No. 234, CPSY98-63, pp. 23-28, 1998.
- [2] 小林伸治, 陣崎明: Virtual Interface Architecture の Internet 拡張方式, 並列処理シンポジウム JSPP'99, pp. 23-30, 1999.
- [3] Virtual Interface Architecture, <http://www.viarch.org/>
- [4] 陣崎明, 中村修, 村井純: 並列ネットワークサーバ Comet のアーキテクチャとその応用, 信学技報, Vol. 98, No. 234, CPSY98-62, pp. 15-22, 1998.
- [5] Chesson, G.: The Evolution of XTP, *Proc. 3rd Int'l Conference on High Speed Networking*, 1991.
- [6] von Eicken, T., Culler, D. E., Goldstein, S. C. and Schauer, K. E.: Active Messages: a Mechanism for Integrated Communication and Computation, *Proc. 19th Int'l Symp. on Computer Architecture*, 1992.
- [7] Pakin, S., Karamcheti, V. and Chien, A.: Fast Messages (FM): Efficient, Portable Communication for Workstation Clusters and Massively-Parallel Processors, *IEEE Concurrency*, Vol. 5, No. 2, pp. 60-73, 1997.
- [8] 陣崎明: Stream Processor, 並列処理シンポジウム JSPP2000, 2000.
- [9] C-PORT, <http://www.cportcorp.com/>
- [10] IXP12000, <http://developer.intel.com/design/network/IXP12000.htm>
- [11] FR-V, http://www.fujitsu.co.jp/hypertext/Products/Device/CATALOG/AD07/07-00026/index_j.html
- [12] MAJC, <http://www.sun.com/microelectronics/MAJC/>