計算機アーキテクチャ　102－8
（１９９３．１０．２１）

高速通信のためのアーキテクチャ・サポート

根岸　康　　田胡　和哉

日本ＩＢＭ東京基礎研究所

１Ｇｂｐｓを越える、きわめて高速な通信媒体が利用できるようになっている。しかしながら、通信機構のオーバヘッドにより、このような通信媒体の性能を十分に活用できないのが現状である。通信機構の性能の測定によれば、プロトコル処理のみならず、ＤＭＡ機構の制御や、データ・コピーによって大きなオーバヘッドが生じていることが判明している。本稿では、このオーバヘッドを削減するために、通信機構を実現する際に、ＤＭＡ転送機構を使用しない方法と、仮想アドレス空間との間でデータ転送が可能な改良されたＤＭＡ転送機構を利用する方法をあげ、両者の優劣について論ずる。

Architectural Supports for High Performance Communication Systems

Yasushi Negishi　Kazuya Tago

The bandwidth of communication media is growing rapidly. However, computer systems can not make full use of the bandwidth because of the overhead of the communication mechanism. The overhead is caused not only by protocol handling but also by DMA set-up and data copy. One approach to alleviate the overhead is to eliminate the DMA mechanism and another is to improve the DMA mechanism to support data transfer to/from virtual spaces. Comparison of two approaches is stated in this paper.

1．まえがき

光ファイバを用いた高速通信媒体の普及が著しい。最近では、ワークテーションの入出力バスのスループットを越える、１Ｇｂｐｓの通信媒体も利用可能になっている。このような通信媒体を有効に利用するためには、計算機の入出力機構の効率が重要であり、通信制御ソフトウェアのみならず、計算機アーキテクチャも含めた、広範なシステム技術に関する再検討が必要である。逆に、このような通信媒体を有効に利用することができれば、単に従来の形態の通信サービスの性能改善が図れるばかりでなく、クラスタ・システム等の新しい形態のシステムの実現が可能になると期待される。

高性能な通信システムを構築するための技術について検討する。

現在のＬＡＮの通信機構では、アプリケーション・プログラムからみると、通信媒体自体の性能によらず、通信制御ソフトウェアや計算機アーキテクチャ自体の制限のために、スループットでは100M bps、レイテンシでは2m sec.程度が上限になっている。これを、アプリケーション・プログラムの実現に大きな影響を与えずに、１０倍程度改善することを目標とする。

ＵＮＩＸシステムについて、ＬＡＮの通信機構の性能について計測してみると、通信自体の処理のみならず、物理入出力やアプリケーション・プログラムと、オペレーティング・システムの間でのデータ入出力に多くの時間がかかっていることがわかる。以下では、物理入出力の効率改善に焦点をあてて、通信性能に関する議論を行う。

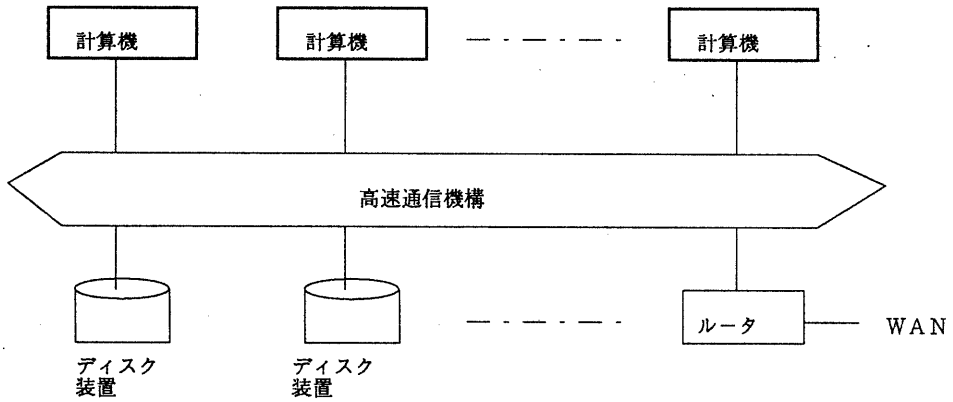


図１　高速通信機構を用いた計算機システム

すでに、通信機構として、ＬＡＮを対象とした通信機構が広く用いられており、その利用技術が急速に発展している。本稿では、このような資産を有効に利用しつつ、さらに、並列処理等新たな分野にも適用可能な、より

2．動機

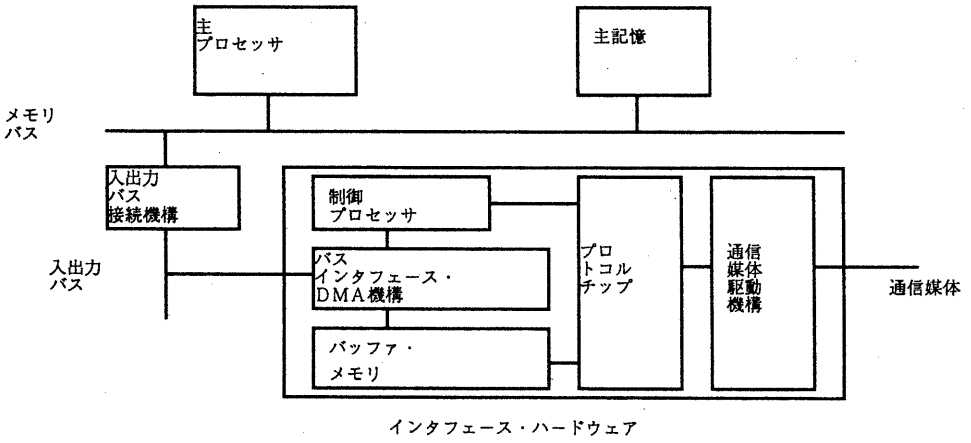
2．1　ニーズ

通信機構は、主に、地理的に離れた地点間でのデータ交換のためと、マルチ・コンピュータを構成するための利用されている。高速通信は、どちらの目的にも重要であるが、ここでは特に、後者について考える。図１に、例を示す。リングやスイッチ構成のネットワークにより、計算機、ディスク等の二次記憶装置、外部ネットワークへのゲートウェイ等が接続されている。全体として、一つのサーバ計算機として利用される。

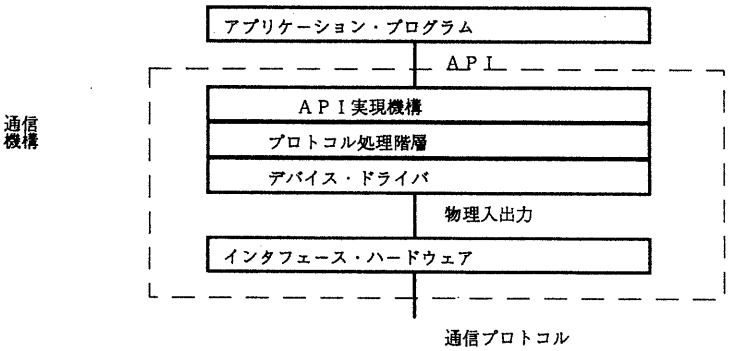
この構成は、数値計算、トランザクション処理、データベース処理、マルチメディア等多くの利用分野を持つ。最近では、ワン・ボード構成の安価な計算機でも、十分な性能が得られるようになっている。このような計算機を多数結合してサーバとすることは、コスト上、大きな利点がある。

数値計算では、通信機構は、主に、計算対象となる配列を計算機間で交換するためのに利用される。すでに、この構成のシステム向きのプログラミング言語や、通信制御のためのミドル・ソフトウェアの開発が進められている。

トランザクション処理では、通信機構は、



（a）　ハードウェア構成



（b）論理構造

図２　通信機構の構造

主に、処理負荷の分散に利用される。また、通信機構は、信頼性の向上に有用である。単一の二次記憶装置を複数の計算機からアクセスできるようにすることにより、一つの計算機に障害が生じても、処理を継続することができる。また逆に、二次記憶装置のミラーリングを実現するためにも有効である。

データベース処理では、その並列化のために、任意の計算機から任意の二次記憶装置にアクセスするために用いられる。

マルチメディア・サーバでは、二次記憶装置と外部ネットワークを直接接続するために用いられる。これによって、特別に高い性能を持たない計算機を用いても、高いスループットを持つメディア・サーバが実現できる。

これらの目的に一般的に適用するためには、計算機間のみならず、計算機と二次記憶装置、計算機と外部ネットワーク、二次記憶装置と外部ネットワーク等、種々の構成要素間で高い性能の通信が実現されている必要がある。

2．2　技術的背景

（1）通信機構の全体構造

通信機構は、通常、図2に示すように、オペレーティング・システム内の制御プログラム、および、通信インタフェース・ハードウェアからなる。通信機構、ソケット等のＡＰＩを通じて、アプリケーション・プログラムから利用される。また、通信機構間の接続は、通信プロトコルによって規定されている．すなわち、ＡＰＩ、及び、通信プロトコルが、通信機構の外部仕様と考えることができる。通信性能は、通信機構の実現方式とともに、これらの仕様によって決まる。本稿では、主に、通信機構の実現について議論し、外部仕様に関する検討は行わない。ＬＡＮにおいて利用されていた仕様をなるべくそのまま利用することにより、アプリケーション・プログラムに与える影響を最小にするよう努力する。

制御プログラムは、ＡＰＩを実現する部分、通信プロトコル階層に対応する状態機械を実現する部分、および、物理入出力を実現するデバイス・ドライバからなる。たとえば、通信プロトコルとしてＴＣＰ／ＩＰを利用する場合は、プロトコル処理機構は、ＴＣＰ階層と、ＩＰ階層にわかれて実現される。

通信インタフェース・ハードウェアは、通信媒体に直接接続されており、媒体の駆動機構、下位プロトコルを実現するプロトコル・チップ、バッファ・メモリ、制御プロセッサ、計算機の入出力バスとのインタフェース等から構成されている。

オペレーティング・システム核内には、通信のためのバッファが割り当てられる。送信時には、ＡＰＩ実現機構を経由して、アプリケーション・プログラムの論理アドレス空間からこのバッファに通信データがコピーされる。通信プロトコル階層部分で、これをフラグメントに分解するとともに、パケット・ヘッダを付加することにより、通信パケットを作成する。作成された通信パケットは、デバイス・ドライバを経由してインタフェース・ハードウェアに送出される。

受信時には、受信パケットは、デバイス・ドライバによって核内のバッファに一旦蓄えられる。プロトコル階層がこれを分析し、また、データグラムの組み立てを行う。作成されたデータは、ＡＰＩ実現機構を経由して、受信したアプリケーション・プログラムの論理アドレス空間に転送される。

（2）通信機構のオーバヘッド

ＬＡＮ通信機構における、制御プログラムのオーバヘッドを測定してみると、図３のよ

探索による受信プロセスの同定処理の時間が大きい。

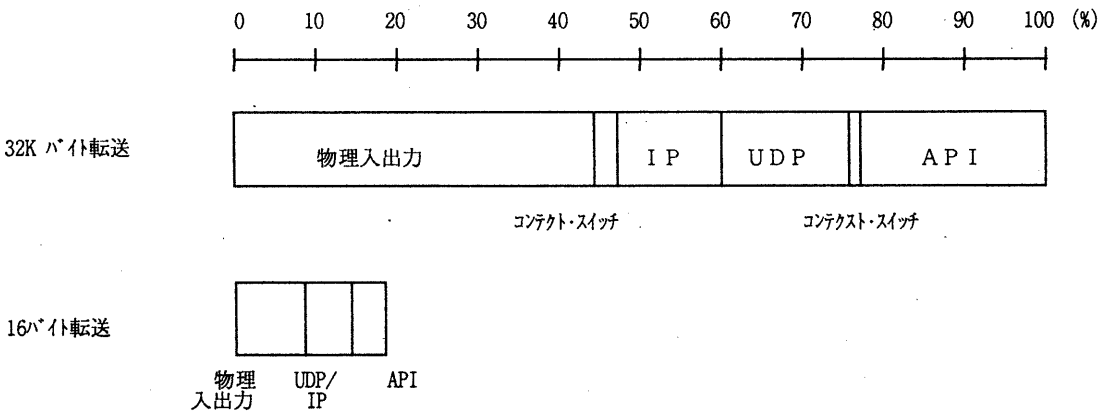


図３　受信処理の処理時間比

うになる。これは、ＵＮＩＸオペレーティング・システムによる、ＵＤＰ／ＩＰプロトコルによるパケットの受信に要する処理の相対時間を表したものである。３２Ｋバイト・データの受信と、16バイト・データの受信の場合について示してある。

物理入出力部分には、ＤＭＡ転送の準備、コマンドの発行、バッファの割り当て等が含まれており、ＤＭＡ転送時間は含まれていない。３２Ｋバイトの例において時間が増大するのは、物理入出力が繰り返し行われることによる。

ＩＰ、および、ＵＤＰが、プロトコル処理に対応する。３２ｋバイト受信時には、この部分は、ＩＰ階層によるデータグラムの組み立て、および、ＵＤＰ階層によるチェックビットの生成によって大半が占められている。１６バイト・データ受信時には、ポート表の

３２ｋバイト受信時のＡＰＩ機構のオーバヘッドは、データ・コピーによって生じている。１６バイト受信時には特定の処理が大半を占めることはない。

全体の時間のなかで、データ・コピー、および、物理入出力時間の占める割合を合わせた部分が大半を占める。これは、通信機構の性能が、プロトコル処理ではなく、オペレーティング・システムや、ハードウェアの設計に大きく左右されていることを示している。この事実が、本稿の動機である。

（３）ハードウェアの構造とオーバヘッド

１）ＤＭＡ

ＤＭＡ機構は、物理アドレス空間と周辺機器の間でデータ転送を行うものであり、入出力に先立って、ページのピンダウン、論理アドレスの物理アドレスへの変換、プロセッサ・キャッシュのフラッシュ等を行っておく必要がある。これらの準備が必要であるため

に、オーバヘッドが生ずる。また、入出力バスとメモリ・バスの間の接続方式によっては、転送の前後にソフトウェアによる接続機構の制御が必要になる場合があり、さらにオーバヘッドとなる。

2）本体プロセッサとインタフェース・ハードウェアの間の同期

物理入出力機構の今一つの問題点は、本体プロセッサとインタフェース・ハードウェアの間の同期である。これは、コマンド、状態語、および、割り込みによって実現されている。これらの機構によるオーバヘッドも、無視しえない。たとえば、割り込みによるコンテクスト・スイッチの影響は、ＲＩＳＣプロセッサの場合には小さくない。また、コマンド語の引き渡しに際して、本体プロセッサとインタフェース・ハードウェアの間で、引き渡し開始、終了の同期のためにポーリングが用いられる場合もある。

3）データ・コピー

通常のワークステーションでは、高速通信媒体のスループットより、データ・コピーのスループットの方が低い。のみならず、データ・コピーのスループットは、プロセッサのクロック周波数の向上の程度ほどには向上しない。データ・コピーは、メモリ・アクセスが主体となるので、メモリのアクセス速度、および、バス転送速度がネックになっていることが知られている。また、最近では、プロセッサの内部ロジックを駆動するクロックは、バスを駆動する外部クロックよりはるかに高い周波数のものを利用するようになっているので、プロセッサの処理速度を基準とした、相対的なデータ・コピー速度は低下する傾向にある。

さらに、データ・コピーは、読み出しサイクルと書き込みサイクルを持つので、通常のＤＭＡより多くのバス・バンド幅を消費する等の問題を持つ。

3．性能改善の試み

通信機構におけるデータ・パスを短縮する方法を検討する。物理入出力効率を改善する方法として、対照的な２つの方法をあげることができる。一つは、ＤＭＡを廃止する方法であり、今一つは、仮想空間へのＤＭＡを行うことができる機構を実現する方法である。前者では、データ転送を主プロセッサで行うことにより、キャッシュ・メモリの更新機能や、論理アドレスの変換が自動的に実現される。後者では、これらを直接実現する新たなハードウェアを実現する。今仮に、前者をＤＭＡ廃止アプローチ、後者を改良ＤＭＡアプローチと名付ける。

3．2　ＤＭＡ廃止アプローチによる設計

（1）基本構造

このアプローチでは、インタフェース・ハードウェア内の、通信媒体の駆動機構が利用するバッファ・メモリを、計算機のメモリ・バスに直接接続する。このバッファはマルチ・ポート構成をとり、通信媒体との間のデータ転送と、主プロセッサからのアクセスが並行して行えるようになっている。このバッファとアプリケーション・プログラムの論理アドレス空間の間で、プロセッサがデータ転送を行う。転送に際して、ページ・フォールトが発生する可能性があるので、バッファ・メモリは十分な大きさを持つ必要がある。すなわち、ページフォール時にページ・インの処理が完了するまで転送を中断しても、他の通信が継続できるだけの容量が必要である。

バッファへのアクセスは、キャッシュ・メモリを経由せずに、直接行うようにする。

プロトコル・チップは、物理フレームが受信されると、内容を解析し、受信のためのバッファ・メモリ上に領域を割り当て、ペイロード部分のみをバッファに転送する。

インタフェース・ハードウェア上に制御プロセッサを配置することは、必ずしも必要ない。制御プロセッサを用いない場合、バッファへの転送が終了した時点で本体プロセッサを起動し、後の処理はすべて本体プロセッサが行う。プロトコル処理はこのバッファを用いて行うことができるので、データ・コピーは、最終的にアプリケーション・プログラムの論理アドレス空間に転送を行う場合にのみ行われる。

（2）制御プロセッサの導入

インタフェース・ハードウェア中に制御プロセッサを設け、これによってプロトコル処理の一部を実現することにより、さらに性能の改善を図ることができる。これによる効果は、主プロセッサからインタフェース・ハードウェアへプロトコル処理の負荷をオフロードすることによるものではなく、本体プロセッサとインタフェース・ハードウェアの間での相互作用の機会が減らせることによる。

受信したパケットのユーザ・データグラムへの組み立てをインタフェース・ハードウェア上で行うことにより、パケット受信ごとに割り込みが発生することを防ぐことができる。また、ＡＣＫパケットの生成をインタフェース・ハードウェア上のプロセッサで行うことにより、ＡＣＫパケットの返送が遅れて送信側で次のパケットの送出ができなくなることによるスループットの低下を防ぐことができる。また、ＡＣＫパケットの送出に要する物理入出力を無くすことができる。

通信の待ち合わせ表、および、プロセス・レディー・キューの一部をインタフェース・ハードウェアのバッファ上に割り当てることにより、本体プロセッサの割り込み処理を無くすことができる。制御プロセッサは、受信したデータを待ち合わせているプロセスを探索し、プロセス・レディー・キューにそのプロセスを登録する。プロセスのスケジューラはこのキューをポーリングによって探索することにより、実行可能なプロセスを得る。スケジュールされたプロセスは、データ・コピーを行った後、ユーザ・モードに遷移する。

３．３　改良ＤＭＡアプローチによる設計

改良ＤＭＡアプローチによる実現では、インタフェース・ハードウェア上のバス・インタフェースに、仮想アドレスの物理アドレスへの変換機構、および、キャッシュ・コヒーレンシを保つ機構を付加する。これを実現するためには、マルチプロセッサの実現と同様に、キャッシュ・コヒーレンスの維持を保証するバス・プロトコルと、アドレス・マッピング・テーブルの変更の同期機構が必要である。これらの部分は、本体プロセッサと同一構造にすることにより、比較的容易に実現できる。

改良ＤＭＡアプローチでは、高速の制御プロセッサを利用することを仮定している。制御プロセッサは、受信したパケットの内容を分析し、ただちに受信プロセスを同定し、転送先を決める。さらに、ＤＭＡ転送機構を起動する。ＤＭＡ転送機構は、パイプライン処理によって物理フレームの受信と並行して、受信プロセスの論理アドレス空間への受信データの転送を行う。

送信時には、通信媒体が使用中である場合に備えて、バッファを用意する。本体プロセッサは、送信コマンドを発行した後、データ転送が終了するのをポーリングによって待ち合わせる。転送に時間がかかる場合には、プロセス・スイッチを行う。

このとき、ページフォールトに対応する必要がある。受信時には、あらかじめ主記憶上のページフレームを確保しておき、ページ・フォールト時にはそちらに切り換えてデータ転送を行い、後にそのページフレームを受信プロセスの論理アドレス空間にリマップすればよい。送信時のページフォールトでは、送信処理を中止する必要がある。

この方式では、アプリケーション・プログラムの論理アドレス空間とインタフェース・ハードウェアの間で通信データを直接やりとりする。そこで、インタフェース・ハードウェア側で転送先を同定する機能が必要であり、このため、制御プロセッサによって、ほぼ通信機構全体を実現する必要がある。これは、制御プロセッサが、本体プロセッサのオペレーティング・システムの論理アドレス空間に直接アクセスし、通信待ちテーブルや、プロセス・レディー・キューを参照できるようにすることにより、実現できる。

4．考察

3．でのべた二つの方法について、優劣を検討してみる。

（1）性能

1）レスポンス

大量データの受信においては、改良DMAアプローチの方が、高いレスポンスが期待できる。その一つの理由は、改良DMAアプローチでは、データの受信、受信プロセスの論理アドレス空間へのデータ転送、および、プロセスのスケジューリングが並行して実行できるからである。これに対して、DMA廃止アプローチでは、後2者を並列して実行することはできない。

一方、このような並列実行が意味を持たない、少量データの受信においては、両者の性能には、原理的には大きな差異がない。

2）スループット

改良DMA方式の方が、より高いスループットが期待できる。これは、DMA転送の方が、データ・コピーに比べてバス負荷が小さいことによる。

（2）実現効率

DMA廃止アプローチの方が、実現が容易である。改良DMAアプローチでは、ハードウェア、ソフトウェアともに、実現が複雑になる。しかしながら、マルチプロセッサ・システムの場合には、プロセッサの一つとして実現することにより、ハードウェア、ソフトウェアともに実現ははるかに容易になる。

5．あとがき

通信性能を向上する方式について検討した。測定によれば、通信制御システムのオーバヘッドの大きな部分が、物理入出力やデータコピーによって生じていることが判明した。これを軽減するための2つのアプローチについて検討した。改良DMAアプローチについては、部分的にプロトタイプを実現した。今後、システム性能の評価について報告したい。また、ここでは、議論の対象としなかった、ＡＰＩについても、検討する予定である。