(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4264866号
(P4264866)
(45)発行日　平成21年5月20日(2009.5.20)　(24)登録日　平成21年2月27日(2009.2.27)

(51)Int.Cl.　　F I
G06F 13/00 (2006.01)　　G06F 13/00 353C
H04L 29/06 (2006.01)　　H04L 13/00 305A

請求項の数 28　(全 35 頁)

(21)出願番号　特願2000-568012(P2000-568012)
(86)(22)出願日　平成10年11月20日(1998.11.20)
(65)公表番号　特表2002-524005(P2002-524005A)
(43)公表日　平成14年7月30日(2002.7.30)
(86)国際出願番号　PCT/US1998/024943
(87)国際公開番号　WO2000/013091
(87)国際公開日　平成12年3月9日(2000.3.9)
審査請求日　平成17年11月21日(2005.11.21)
(31)優先権主張番号　09/141,713
(32)優先日　平成10年8月28日(1998.8.28)
(33)優先権主張国　米国(US)

(73)特許権者　501082978
アラクリテック・インコーポレイテッド
アメリカ合衆国，カリフォルニア州・９５０１２，サン・ノゼ，イースト・ギッシュ・ロード・２３４
(74)代理人　100098062
弁理士　梅田　明彦
(72)発明者　ブーチャー，ローレンス，ビー
アメリカ合衆国，カリフォルニア州・９５０７０，サラトガ，モントルボ・ドライブ・２０６０５
(72)発明者　ブライトマン，スチーブン，イー，ジェイ
アメリカ合衆国，カリフォルニア州・９５１３２，サン・ノゼ，アーレン・コート・３７３３
最終頁に続く

(54)【発明の名称】通信を高速化するインテリジェントネットワークインタフェース装置及びシステム

(57)【特許請求の範囲】
【請求項１】
ハードウエアとプロセッサとプロトコル処理層のスタックとを有するホストコンピュータによりネットワーク通信を行うための方法であって、
ネットワークから前記ホストコンピュータによって、複数のヘッダと、前記ヘッダのプロトコル処理に従って前記ホストコンピュータの行先に配置するためのデータとを含むメッセージパケットを受信する過程と、
前記ハードウエアにより、前記ヘッダの要約を作成することを含めて、前記複数のヘッダをグループとして処理する過程と、
前記データを前記ホストコンピュータの行先に格納するために、前記要約に基づいて、前記プロトコル層によって前記パケットを処理するか、又は前記プロトコル層による処理を回避するかを選択する過程と、
前記データを前記ヘッダなしで前記要約に従って、前記プロトコル層により前記ヘッダを処理することなく、前記ホストコンピュータ上のアプリケーションにより提供される前記行先に転送する過程とを有することを特徴とする方法。
【請求項２】
前記パケットに対応する伝送制御プロトコル（ＴＣＰ）コネクションのための通信制御ブロックを作成する過程と、前記データを前記行先に送出するために、前記要約と前記通信制御ブロックとを整合させる過程とをさらに含むことを特徴とする請求項１に記載の方法。

【請求項３】
前記パケットに対応する伝送制御プロトコル（ＴＣＰ）コネクションのための通信制御ブロックを作成する過程をさらに含み、前記データを前記行先に送出する過程は、前記通信制御ブロックによって前記データを誘導する過程を含むことを特徴とする請求項１に記載の方法。
【請求項４】
前記通信制御ブロックを参照することにより、前記ホストコンピュータから前記ネットワークに第２のメッセージを伝送する過程をさらに含むことを特徴とする請求項３に記載の方法。
【請求項５】
通信プロセッサを備える装置によりネットワークに接続され、プロトコル処理スタックを有するホストコンピュータにより通信を処理するための方法であって、
前記ホストコンピュータによって前記ネットワークから前記プロトコル処理スタックに対応する一連のヘッダとデータとを含むメッセージフレームを受信する過程と、
前記フレームについて発信元及び行先のインターネットプロトコル（ＩＰ）アドレスと発信元及び行先の伝送制御プロトコル（ＴＣＰ）ポートとを決定することを含めて、前記装置によって前記一連のヘッダをバイトのストリームとして解析する過程と、
前記装置により前記データを前記ヘッダなしで、前記ＩＰアドレス及びＴＣＰポートに基づいて、前記プロトコル処理スタックによって前記ヘッダを処理することなく、かつ前記ホストコンピュータによって前記データをコピーすることなく、前記ホストコンピュータ上で実行中のアプリケーションにより制御されるホストメモリ内の行先に送出する過程とを有することを特徴とする方法。
【請求項６】
前記ホストメモリ内の行先に前記ヘッダなしで前記データを送出する前記過程が、前記ホストコンピュータによって、前記フレームを含むメッセージのための通信制御ブロックを作成する過程と、
前記装置に前記通信制御ブロックを格納する過程と、
前記ＩＰアドレス及びＴＣＰポートを前記通信制御ブロックと整合させる過程とをさらに含むことを特徴とする請求項５に記載の方法。
【請求項７】
前記データを前記行先に送出する前に、前記ＩＰアドレス及びＴＣＰポートのハッシュを前記通信制御ブロックを識別するハッシュテーブルと比較する過程をさらに含むことを特徴とする請求項６に記載の方法。
【請求項８】
前記装置を経由して、前記ホストコンピュータから前記ネットワークに送信データを伝送する過程をさらに含み、前記伝送する過程は、リモートホストへのネットワーク転送のために、前記送信データの前にいくつかのプロトコルプロトコルヘッダを同時に付加する過程を含むことを特徴とする請求項５に記載の方法。
【請求項９】
プロセッサとプロトコル層の順次式スタックとを有し、ネットワークインタフェースによりネットワークに接続されたホストコンピュータにより通信を行うための方法であって、該方法は、
前記ネットワークから前記ネットワークインタフェースによって、それぞれにデータ部分と、前記プロトコル層の順次式スタックにより処理した場合には、前記データのための前記ホストコンピュータ内の上位層の行先に前記データ部分を格納させる、関連するヘッダのシーケンスとを含む多数のパケットを有するメッセージを受信する過程と、
前記ネットワークインタフェースによって、前記関連するヘッダなしで、かつ任意のホストＣＰＵへの割り込みを発生することなく、複数の前記データ部分を前記行先に送出する過程とを有することを特徴とする方法。
【請求項１０】

前記データ部分を前記行先に送出する過程の前に、前記プロトコル層のスタックによって前記パケットを処理するか否かを選択する過程をさらに含むことを特徴とする請求項９に記載の方法。

【請求項１１】

前記ヘッダを用いることなく前記データ部分を前記行先に送出する前に、前記ネットワークインタフェースで前記ヘッダを処理する過程をさらに含むことを特徴とする請求項９に記載の方法。

【請求項１２】

前記ネットワークインタフェースを介して前記ホストから前記ネットワークにデータファイルを伝送する過程をさらに含み、前記伝送する過程は、前記データファイルを一連のデータユニットに分割する過程と、前記データユニットの前に前記ヘッダを付加し、それにより一連のネットワークフレームを作成する過程と、任意のホストＣＰＵに割り込みを発生することなく、前記ネットワークフレームを前記ネットワークに配置する過程とを含むことを特徴とする請求項９に記載の方法。

【請求項１３】

ＣＰＵと、メモリと、プロトコル層の順次式スタックとを備え、ネットワークインタフェース装置によりネットワークに接続されたホストコンピュータによって通信を行うための方法であって、

前記ネットワークインタフェース装置によって前記ネットワークから第１のパケットを受信する過程と、

前記ホストコンピュータによって、前記第１のパケットに対応する伝送制御プロトコル（ＴＣＰ）状態を含む通信制御ブロックを作成する過程を含めて、前記第１のメッセージを処理する過程と、

前記通信制御ブロックを前記装置に転送する過程と、

前記ネットワークインタフェース装置によって前記ネットワークから、データとヘッダとを含み、かつ前記通信制御ブロックに対応する第２のパケットを受信する過程と、

前記ネットワークインタフェース装置によって前記ヘッダのＴＣＰ部分を処理することを含めて、前記ネットワークインタフェース装置によって前記ヘッダを処理する過程と、

前記ネットワークインタフェース装置によって前記データを、前記ホストコンピュータにより前記ヘッダを処理することなく、前記プロトコル層のＴＣＰ層より上位にある上位層に、該上位層に適した形で送出し、前記通信制御ブロックを用いて前記送出を誘導する過程とを有することを特徴とする方法。

【請求項１４】

前記通信制御ブロックに対応する第３のパケットを前記装置によって受信する過程と、前記装置から前記メモリに前記通信制御ブロックの制御を渡す過程と、前記ＣＰＵにより前記第３のパケットを処理する過程とをさらに含むことを特徴とする請求項１３に記載の方法。

【請求項１５】

前記データを前記上位層に送出する前に、前記第２のパケットのＩＰアドレスとＴＣＰポートとを前記通信制御ブロックに整合させる過程をさらに含むことを特徴とする請求項１３に記載の方法。

【請求項１６】

前記ホストコンピュータから前記ネットワークに第３のパケットを伝送する過程をさらに含み、該過程は、前記通信制御ブロックを参照し、かつ複数のプロトコル層ヘッダを含む送信ヘッダをホストソースから取得されるデータに付加することにより、前記装置を経由して前記第３のパケットを送出する過程を含むことを特徴とする請求項１３に記載の方法。

【請求項１７】

プロトコル処理装置によりネットワークに接続され、プロトコル処理スタックを有するローカルホストによりネットワーク通信を行うための方法であって、

前記プロトコル処理スタックにより、前記ローカルホストとリモートホストとの間の伝送制御プロトコル（ＴＣＰ）コネクションを定義する通信制御ブロックを作成する過程と、

前記通信制御ブロックを前記プロトコル処理装置に渡し、それにより前記コネクションに関連しかつ前記ネットワークと前記ローカルホストとの間で転送されるメッセージパケットのＴＣＰ処理の制御を渡し、それにより、前記メッセージパケットを前記プロトコル処理スタックではなく、前記プロトコル処理装置によって処理する過程とを有することを特徴とする方法。

【請求項１８】

前記ネットワークと前記ローカルホストとの間で転送されかつ前記コネクションに関連する第２のメッセージパケットが全般に前記プロトコル処理スタックによって処理されるように、前記通信制御ブロックの制御を前記ローカルホストに渡す過程をさらに含むことを特徴とする請求項１７に記載の方法。

【請求項１９】

前記プロトコル処理装置によって、前記ネットワークからメッセージフレームを受信する過程と、

前記プロトコル処理装置によって、前記メッセージフレームのヘッダを解析し、それにより前記メッセージフレームのＴＣＰ発信元ポート及び行先ポートを決定する過程と、

前記ＴＣＰ発信元ポート及び行先ポートを前記通信制御ブロックと比較する過程とをさらに含むことを特徴とする請求項１７に記載の方法。

【請求項２０】

前記プロトコル処理装置によって、前記ネットワークにメッセージフレームを伝送する過程をさらに含み、前記伝送する過程は、前記通信制御ブロックに基づいてヘッダを形成する過程と、前記メッセージフレームの前に前記ヘッダを付加する過程とを含むことを特徴とする請求項１７に記載の方法。

【請求項２１】

ネットワークを経由してリモートホストに接続することができるローカルホストで用いるための装置であって、前記ローカルホストは、前記ローカルホストのアプリケーションと前記リモートホストのアプリケーションとの間のＴＣＰコネクションを定義する通信制御ブロックを作成するプロトコル処理層のスタックを動作させるＣＰＵを含み、

該装置は、

前記ＣＰＵと前記ネットワークとに接続され、前記プロトコル処理層によってネットワークメッセージを処理するか、前記プロトコル処理層を回避し、前記ネットワークと前記ローカルホストアプリケーションとの間で前記ネットワークメッセージに含まれるデータを転送するために前記通信制御ブロックを用いるかを選択するために構成されるプロセッサを有する通信処理機構を備えることを特徴とする装置。

【請求項２２】

前記プロセッサは、前記通信制御ブロックをその前記メッセージの処理に基づいて更新するように構成されることを特徴とする請求項２１に記載の装置。

【請求項２３】

前記通信処理機構は、前記プロセッサに接続され、前記ネットワークから受信されるメッセージパケットの要約を生成するために構成される受信シーケンサを備え、前記メッセージパケットは、前記プロトコル層のスタックに対応する制御情報を含み、前記プロセッサは前記要約を前記通信制御ブロックと比較するように構成されていることを特徴とする請求項２１に記載の装置。

【請求項２４】

前記プロセッサは、前記通信制御ブロックに対応するヘッダを作成し、前記プロトコル処理層のいくつかに対応する制御情報を含み、前記ホストから前記リモートヘッダに前記メッセージを伝送するために前記データの前に前記ヘッダを付加し、前記通信制御ブロックを更新するように構成されることを特徴とする請求項２１に記載の装置。

【請求項２５】
前記通信処理機構は、前記通信制御ブロックに基づいて、前記データに付随する全ゆるヘッダを用いることなく、前記通信処理機構から前記ホストアプリケーションに前記データを送出するためのダイレクトメモリアクセスユニットを備えることを特徴とする請求項２１に記載の装置。
【請求項２６】
前記プロセッサは複数のマイクロプロセッサを備え、前記マイクロプロセッサのうちの少なくとも１つは主に、前記ネットワークから前記ホストによって受信されるメッセージを処理するように構成され、前記マイクロプロセッサのうちの第２のマイクロプロセッサは前記ホストから前記ネットワークに伝送されるメッセージを処理するように構成されることを特徴とする請求項２２に記載の装置。
【請求項２７】
前記マイクロプロセッサは、フェーズをローテーションする際に共有されるハードウエア機能を利用することを特徴とする請求項２６に記載の装置。
【請求項２８】
前記通信処理機構は、複数のキュー記憶ユニットにおいて情報をキューに入れるために構成されるキューマネージャを備え、前記キュー記憶ユニットのうちの少なくとも１つはＳＲＡＭおよびＤＲＡＭを含むことを特徴とする請求項２１に記載の装置。
【発明の詳細な説明】
【０００１】
（技術分野）
本発明は、一般的にコンピュータ又は他のネットワークに関するものであり、得にネットワークに接続されたコンピュータのようなホスト間で通信される情報の処理に関するものである。
【０００２】
（背景技術）
ネットワークコンピューティングの利点はますます明らかとなってきている。パーソナルコンピュータや他のエンドユーザ装置において情報、通信、又は計算能力を個人に提供することがもたらす利便性及び能率のために、イントラネット装置やアプリケーション及びインターネット等のネットワークコンピューティングが急速に成長することになった。
【０００３】
周知のように、大部分のネットワークコンピュータ通信は、ネットワークに接続されたホストコンピュータ間での情報の転送のための階層化ソフトウェアアーキテクチャの支援によって達成される。その層が、情報を処理可能なセグメントに分割することを助け、各層の一般的な機能は、多くの場合開放型システム間相互接続（ＯＳＩ）と呼ばれる国際標準に準拠している。ＯＳＩは７種の処理の層を規定しており、情報がホストによって受信されると、それはエンドユーザに提示できるようにするためにそれらの層の間をわたされる。同様に、ホストからネットワークへの情報の送信では、情報がそれらの７種の処理層を逆の順に通過し得る。１つの層による処理及びサービスの各ステップでは、処理された情報のコピーが行われる。ＴＣＰ／ＩＰ（ＴＣＰは伝送制御プロトコルを表し、ＩＰはインターネットプロトコルを意味する）と呼ばれる、広く実現されている別の参照モデルは、基本的にＯＳＩの７層のうち５層を利用する。
【０００４】
ネットワークは、例えば、それぞれが複数のホストを有する異なるローカルエリアネットワーク（ＬＡＮ）間のイーサネット接続又はインターネット接続のような高速パス、又はホスト間のデータ伝送のための様々な他の既知の手段の何れかを含み得る。ＯＳＩ標準によれば、物理層は各ホストにおいてネットワークに接続され、その物理層がネットワークを介しての生のデータビットの送信及び受信を提供する。データリンク層は、各ホストの物理層によりサービスされ、物理層から受け取ったデータのフレーム分割及び誤り訂正を提供するとともに、受信したホストによって送られた肯定応答フレームの処理を行う。各

ホストのネットワーク層は、各データリンク層によりサービスされ、主としてデータのパケットのサブネットのサイズ及び統合を制御する。

【０００５】

トランスポート層は各ネットワーク層からサービスされ、セッション層は各ホスト内の各トランスポート層によりサービスされる。トランスポート層はそれらの各セッション層からデータを受け取り、そのデータを他のホストのトランスポート層へ送信するためのより小さいユニットに分割する。他のトランスポート層は、各プレゼンテーション層への提示のためそのデータを連結する。セッション層によってホスト間の機能強化された通信の制御が可能となる。プレゼンテーション層はそれらの各セッション層によりサービスされ、各ホスト固有であり得るデータセマンティクス及び構文と、データ表現の標準化された構造との間での変換を行う。データの圧縮及び／または暗号化も、プレゼンテーション層レベルで達成され得る。アプリケーション層は、各プレゼンテーション層によりサービスされ、個々のホストに特有のプログラムと、アプリケーション又はエンドユーザの何れかに対して提示するための標準化されたプログラムとの間の変換を行う。ＴＣＰ／ＩＰ標準は、下位の４層及びアプリケーション層を含むが、セッション層及びプレゼンテーション層の機能は隣接する層に統合している。一般的に、アプリケーション層、プレゼンテーション層、及びセッション層は上位層と定義され、トランスポート層、ネットワーク層、及びデータリンク層は下位層と定義される。

【０００６】

各層の規則及び規定はその層のプロトコルと呼ばれ、かつ各層のプロトコル及び一般的な機能は様々なホストにおいて概ね等しいことから、たとえ同位の層同士が情報を下位の各層を順次転送させることなく直接やり取りしないとしても、異なるホストの同一の層同士で直接行われる通信というものを考えることは有益である。各下位層は、通信される情報を処理を助けるべくその１つ上位の層にサービスする。各層は、処理及び次の層へのサービスの提供のため情報を保存する。一般的に用いられるハードウェア及びソフトウェアアーキテクチャや装置及びプログラムの多様性のために、各層は、介入し得るハードウェア及びソフトウェアの相違とは無関係に、データが適切な形式で目的の行先に到達し得ることを確実なものとする必要がある。

【０００７】

第１のホストから第２のホストに送信するためのデータの準備において、その層のプロトコルに関連する第１のホストの各層で幾つかの制御データが付加されるが、その制御データはそのホストの全ての下位の層において元のデータ（ペイロードデータ）から識別可能なものある。従って、アプリケーション層はペイロードデータにアプリケーションヘッダを付加し、結合されたデータを送信側のホストのプレゼンテーション層に送る。プレゼンテーション層はその結合データを受け取り、それを処理してそのデータにプレゼンテーションヘッダを付加し、別の結合されたデータパケットが生成される。次に、ペイロードデータ、アプリケーションヘッダ、及びプレゼンテーションヘッダの結合によって生成されたデータをセッション層に渡し、セッション層は、そのデータへのセッションヘッダの付加や、生成されたデータの組み合わせのトランスポート層への提示等の必要な処理を行う。この処理は、情報が下位の層に移動する間継続され、それらの各層においてデータにトランスポートヘッダ、ネットワークヘッダ、及びデータリンクヘッダ、及びトレーラが付加され、各処理過程ではデータの転送及びコピーを行い、その後、データをビットパケットとしてネットワークを通して第２のホストに送信する。

【０００８】

受信側のホストは一般的に上述のプロセスの逆を行い、ネットワークからのデータビットの受信に始まり、最下位の（物理）層から最上位の（アプリケーション）層まで順番にヘッダが除去され、データが処理されて、受信側ホストの行先への送信が行われる。受信側ホストの各層は、その層に関連するヘッダのみを認識し、処理する。その層からみて、より上位の層の制御データはペイロードデータに含められており、そこから識別できないからである。受信側ホストがそのデータをその目的の行先に適切な形式で渡すためには、複

数の割り込み、有効な中央処理装置（ＣＰＵ）による処理時間、及び反復されるデータコピー処理も必要であり得る。

【０００９】

上述の階層化プロトコル処理の説明は、主としてこの主題について詳述された大学レベルの教科書、例えばAndrew S. TanenbaumによるComputer Networks, Third Edition (1996) が入手可能であることから、単純化されている。上記文献はこの引用により本明細書の一部とする。その文献に定義されているように、コンピュータネットワークとは、インターネットや、例えばローカルエリアネットワーク（ＬＡＮ）、ワイドエリアネットワーク（ＷＡＮ）、非同期転送モード（ＡＴＭ）、リング又はトークンリング、配線式、無線式、衛星、又は他の異なるプロセッサ間の通信能力を提供するための手段等のインターネットデバイスのような、相互接続された独立して操作されるコンピュータの集合体である。本明細書において、コンピュータは、データを処理するための論理機能とメモリ機能の両方を有する装置を含むものと定義されており、ネットワークに接続されたコンピュータ即ちホストは、それらが異なるオペレーティング装置に従って機能する場合、即ち異なるアーキテクチャを介して通信を行う場合、ヘテロジニアス（異種）であるという。

【００１０】

ネットワークがますます一般的となり、それによってやりとりされる情報がますます複雑化し内容豊富になるにつれて、そのようなプロトコル処理の必要性が高まってきた。ホストのＣＰＵの処理能力の大部分が、プロトコル処理の制御に充てられて、そのＣＰＵが他のタスクを行う能力は低下していると推定される。物理層やデータリンク層のような下位の層の補助のため、ネットワークインタフェースカードが開発されてきた。単に従来型の構成のＣＰＵにより大きい処理能力を付与することによって、プロトコル処理の速度を高めることも可能である。しかし、この解決方法は手際の良いものとは言えず、コストがかかる。しかし、様々なネットワーク、プロトコル、アーキテクチャ、オペレーティング装置、及びアプリケーションによって付与される複雑さによって、種々のネットワークホスト間での通信能力を与えるために大量の処理が要求されることになる。

【００１１】

（発明の開示）

本発明は、通信されるデータの転送の効率及びその処理速度を大幅に高める、ネットワーク通信の処理のための装置を提供する。本発明は、汎用プロセッサ上での階層化プロトコル処理を行うという、長年にわたって利用されてきた実施の形態を問い直すことによって達成されたものである。そのプロトコル処理方法と、その結果として形成されるアーキテクチャでは、例えばＴＣＰ／ＩＰのようなコネクションをベースにした階層化アーキテクチャの層を、所望の位置又はホスト上のバッファとの間で、より直接的にネットワークデータを送受信できる１層のより広い層に効果的にまとめる。この高速化処理はデータの送受信両方のためにホストに提供され、情報の交換に関係するホストの両方がそのような特徴を有している場合も、その一方しかそのような特徴を有していない場合でも、処理能力を向上させる。

【００１２】

前記高速化処理では、その発信元において直接的にメッセージデータにアクセスしたりそれをその目的の行先に直接供給する高速パスを介してメッセージからのデータを処理できるようにする所定のメッセージに対応する制御命令を用いる。この高速パスは、データに付加されるヘッダの従来型のプロトコル処理をバイパスする。高速パスは、ネットワーク通信の処理のために設計された専用マイクロプロセッサを用いて、例えば反復されるコピー処理やＣＰＵへの割り込みのような、従来のソフトウェアの階層処理の遅れ及び問題点を回避している。要するに、前記高速パスは、従来型のネットワークスタックの複数の層において伝統的に見出される状態を、プロトコル層の厳格な区別及び分離を要求する従来の規定とは異なりこれらの層を全て包含する１個のステートマシンで置き換える。ホストは、高速パスコネクション又はメッセージ例外の処理をセットアップするために用いられ得る順次式プロトコル処理スタックを保持している。専用マイクロプロセッサ及びホスト

が、所定のメッセージまたはメッセージの一部分をそのマイクロプロセッサによって処理するか、ホストのスタックによって処理するかを、インテリジェントに選択する。

【００１３】

（発明の最良の実施の形態）

図１は、リモートホスト２２にネットワーク２５によって接続された本発明のホスト２０を示す。本発明によって達成される処理速度の上昇は、既存のホストに容易に無理なく追加されるインテリジェントネットワークインタフェースカード（ＩＮＩＣ）又はホストに統合される通信処理装置（ＣＰＤ）によって提供され得、何れの場合もホストのＣＰＵを大部分のプロトコル処理から解放し、そのＣＰＵによって実行される他のタスクも改善することができる。第１の実施形態のホスト２０は、ＰＣＩバス３３によって接続されたＣＰＵ２８及びＣＰＤ３０を有する。ＣＰＤ３０はダイレクトメモリアクセス（ＤＭＡ）ユニットによって制御されるメモリバッファ及び通信データの処理のために設計されたマイクロプロセッサを有する。例えば、半導体メモリ又はディクスドライブのような記憶機構３５も、関連する制御機構とともにＰＣＩバス３３に接続されている。

【００１４】

更に図２を参照すると、ホストのＣＰＵ２８が記憶機構３５に収容されたプロトコル処理スタック４４を制御しており、前記スタックはデータリンク層３６、ネットワーク層３８、トランスポート層４０、上位層４６、及び上位層インタフェース４２を有する。上位層４６は、使用される特定のプロトコル及び通信されるメッセージに応じて、セッション層、プレゼンテーション層、及び／又はアプリケーション層を意味し得る。上位層インタフェース４２は、ＣＰＵ２８及び関連するあらゆる制御機構と協働して、矢印４８で示すように、上位層４６又は記憶機構３５へのファイルの転送及びそこからのファイルの取り出しを行える。コネクションコンテキスト５０が生成されており、これは、後に説明するように、例えばプロトコルの種類及び各プロトコル層のための発信元及び行先アドレスのような様々なコネクションの特徴を要約する。そのコンテキストは、矢印５２及び５４で示すように、セッション層４２のためのインタフェースとＣＰＤ３０との間でやりとりされ得、ＣＰＤ３０又は記憶機構３５の何れかに通信制御ブロック（ＣＣＢ）として格納される。

【００１５】

ＣＰＤ３０が特定のコネクションを確定するＣＣＢを保持するとき、ネットワークからＣＰＤによって受信されたそのコネクションに関連するデータはそのＣＣＢによって参照され、次に高速パス５８に従って記憶機構３５に直接送信され得、データリンク層３６、ネットワーク層３８、及びトランスポート層４０によって処理される順次式のプロトコルがバイパスされることになる。記憶機構３５からリモートホスト２２へのファイルの送信のようなメッセージの転送は高速パス５８を介しても生じ得、その場合、トランスポート層４０、ネットワーク層３８、及びデータリンク層３６による処理の間に、逐次ヘッダを付加するのではなく、ＣＣＢを参照するＣＰＤ３０によってファイルデータのためのコンテキストが加えられる。ＣＰＤ３０のＤＭＡコントローラが、これらのＣＰＤと記憶機構３５との間の転送を実行する。

【００１６】

ＣＰＤ３０は、それぞれが可能な分離した状態を有する複数のプロトコルスタックを、高速パス処理のための１個のステートマシンにまとめる。この結果、その１個のステートマシンにおいては準備されていない例外条件が生じ得る。これようにしているのは、主としてそのような条件がまれにしか生じず、ＣＰＤ上でそれを処理することはコストに対して殆ど処理能力上の利点が得られないからである。そのような例外は、ＣＰＤ３０又はＣＰＵ２８で生じたものであり得る。本発明の利点として、高速パスＣＣＢ上で生ずる予期しない状況が取り扱われる方式がある。ＣＰＤ３０は、これらのまれな状況を制御ネゴシエーションを介してホストのプロトコルスタック４４にＣＣＢ及び関連するあらゆるメッセージフレームを戻す、即ちフラッシュすることによって処理する。次にその例外条件は、従来の方式でホストのプロトコルスタック４４によって処理される。しばらく後に、通常

は例外条件の処理が終了し、高速パス処理が再開し得るようになった直後に、ホストのスタック４４はＣＣＢをＣＰＤに戻す。

【００１７】

このフォールバック能力によって、ホストのプロトコルの能力に影響を与える機能をＣＰＤネットワークマイクロプロセッサが取り扱うことができるようになるとともに、全体の処理能力に無視できる程度の影響しか与えない程度にまれにしか生じない例外がホストのスタックによって取り扱われることになる。カスタム設計のネットワークプロセッサは、ネットワーク情報の送受信のための独立したプロセッサと、補助及びキューのための別のプロセッサとを備え得る。好ましいマイクロプロセッサの実施形態は、パイプライン型の３個１組の受信、送信、及びユーティリティプロセッサを備えるＤＭＡコントローラがインプリメンテーションに組み入れられ、コントローラに隣接するバッファと長時間記憶機構のような他の位置との間でデータを迅速にやりとりさせるべく、ネットワークマイクロプロセッサと密接に協働する形で機能するものである。ＤＭＡコントローラに論理的に隣接するバッファを設けることによって、ＰＣＩバスに不必要な負荷が加わるのを避けられる。

【００１８】

図３は、本発明によって受信されたメッセージの概略流れ図である。ファイル転送のような大きいＴＣＰ／ＩＰメッセージは、多数の分離された約６４ＫＢの転送でネットワークからホストに受信され得るが、それぞれが多数の約１．５ＫＢのフレーム即ちネットワーク上の転送するためのパケットに分割されてもよい。Ｉｎｔｅｒｎｅｔ Ｐａｃｋｅｔ Ｅｘｃｈａｎｇｅ（ＩＰＸ）上でＳｅｑｕｅｎｔｉａｌ Ｐａｃｋｅｔ Ｐｒｏｔｏｃｏｌ（ＳＰＸ）又はＮｅｔＷａｒｅ Ｃｏｒｅ Ｐｒｏｔｏｃｏｌ（ＮＣＰ）を走らせるＮｏｖｅｌｌ社のＮｅｔＷａｒｅプロトコルスーツは、同様の方式で機能する。高速パスによって取り扱われ得るデータ通信の他の形態は、Ｔｒａｎｓａｃｔｉｏｎ ＴＣＰ（以下Ｔ／ＴＣＰ又はＴＴＣＰという）であり、これは、幾つかのメッセージ初期化ダイアログを介してコネクションを開始させて後のメッセージでデータを転送するのではなく、初期のトランザクションリクエストでコネクションを開始させ、その後データを含む応答がそのコネクションに従って送られるようなＴＣＰのバージョンである。これらのプロトコルによって代表されるあらゆる転送において、各パケットは、従来通り転送されるデータの部分とともに、プロトコル層のそれぞれに対応するヘッダ及びそのメッセージの残りのパケット群に対するそのパケットを位置を決めるためのマーカを有する。

【００１９】

ＣＰＤによってネットワークからメッセージパケット即ちフレームが受信されたとき（４７）、ハードウェアの補助によりそれが初めに検査される。ここでは、様々な層のプロトコルの種類を決定し、関連するチェックサムを確認し、かつこれらの認識事項をステータスワードに要約する（５７）。これらのワードに含められるのは、そのフレームが高速パスデータフローの候補であるか否かの表示である。高速パス候補の選択（５９）は、そのホストがＣＰＤによって取り扱われるそのメッセージコネクションから利益を得られるか否かに基いて行われ、そのパケットが例えばＴＣＰ／ＩＰ又はＳＰＸ／ＩＰＸのような特定のプロトコルを表示するヘッダバイトを有しているか否かの判定が行われる。小さいパーセンテージで存在する高速パス候補でないフレームは、低速パスプロトコル処理のためにホストのプロトコルスタックに送られる（６１）。その後の各高速パス候補についてのネットワークプロセッサ作業は、ＴＣＰ又はＳＰＸ ＣＣＢのような高速パスコネクションがその候補に対して常に現存しているか否か、又はその候補が、例えばＴＴＣＰ／ＩＰトランザクションのためのもののような新たな高速パスコネクションをセットアップするために用いられ得るか否かを決定する。ＣＰＤによって提供される妥当性検査が、或るフレームが高速パスで処理されるか、低速パスで処理されるかに関わらず、処理の高速化をもたらし、誤りのない場合にのみ、検査されたフレームが、たとえ低速パス処理の場合でもホストのＣＰＵによって処理される。

【００２０】

高速パスの候補となるものとしてＣＰＤハードウェアの補助によって決定された受信されたメッセージフレームは全て、ネットワークマイクロプロセッサ又はＩＮＩＣ比較器回路によって調べられ（５３）、それらがＣＰＤによって保持されたＣＣＢに一致するか否かが判定される。そのような一致を確認する際、ＣＰＤは下位層のヘッダを取り除き、ＣＰＤのダイレクトメモリアクセス（ＤＭＡ）ユニットを用いて、残りのアプリケーションデータをフレームから直接その最終行先に送信する（６９）。この操作は、例えばＴＣＰコネクションが既に存在しかつ行先バッファがネゴシエートされた場合には、メッセージパケットを受け取った直後に行われ得るが、初めに初期ヘッダを処理して、この転送のための最終行先アドレスの新たな組を取得することが必要なこともある。後者の場合には、ＣＰＤは、行先アドレスを待つとともに後続のメッセージパケットをキューに入れ、次いでキューに入れられたアプリケーションデータをその行先にＤＭＡ転送する。

【００２１】

ＣＣＢに一致しない高速パス候補を用いて、そのフレームを順次プロトコル処理のためにホストに送信（６５）することによって、新たな高速パスコネクションをセットアップすることができる。この場合、ホストはそのフレームを用いてＣＣＢを生成し（５１）、次いでＣＣＢはＣＰＤにわたされて、そのコネクションにおける後のフレームを制御する。ＣＣＢはＣＰＤにキャッシュされ（６７）、従来通りのソフトウェア階層処理を使用している場合には、処理された全てのプロトコルに対して適切な制御及び状態情報を有する。ＣＣＢはまた、後の関連するメッセージパケット内に含められたアプリケーションレベルのデータを、直接的な使用に利用可能な形態でホストのアプリケーションに直接わたすのを容易にするために用いられる、転送毎の情報のための記憶空間も有している。ＣＰＤは、ホストからそのコネクションのためのＣＣＢを受け取ったとき、コネクション処理の指揮を取る。

【００２２】

図４Ａにより明確に示されているように、リモートホスト２２からメッセージパケットがネットワーク２５を介して受け取られたとき、そのパケットはＣＰＤ３０のハードウェア受信ロジック３２に入り、これがヘッダ及びデータを検査し、ヘッダを構文解析してメッセージパケット及びステータスを特定するワードを生成し、ヘッダ、データ、及びワードをメモリ６０に一時的に格納する。パケットの検査と同時に、受信ロジック３２は、そのパケットが高速パス処理の候補であるか否かを前記ワードを用いて示す。図４Ａは、そのパケットが高速パス候補でない場合を示しており、この場合ＣＰＤ３０は、矢印５６によって示されるように、ホストのＣＰＵによる処理のために、内部バスを通してメモリ６０からデータリンク層３６へ検査されたヘッダ及びデータを送信する。そのパケットは、データリンク層３６、ネットワーク層３８、トランスポート層４０、及びセッション層４２のホストのプロトコルスタック４４によって処理され、次にパケットからのデータ（Ｄ）６３は、矢印６５に示されるように、記憶機構３５に送信され得る。

【００２３】

図４Ｂは、ＣＰＤの受信ロジック３２が、例えばパケットのヘッダからそのパケットがＴＣＰ／ＩＰ、ＴＴＣＰ／ＩＰ、又はＳＰＸ／ＩＰＸメッセージに属すること導き出すことによってそのメッセージパケットが高速パス処理の候補であることを決定する場合を示す。次に、ＣＰＤ３０におけるプロセッサ５５は、高速パス候補を要約するワードがキャッシュ６３に保持されたＣＣＢに一致するか否かを調べるためのチェックを行う。そのパケットに対して一致がみられないときは、ＣＰＤは、ホストのプロトコルスタック４４にメモリ６０から検査されたパケットを処理のために送信する。ホストのスタック４４は、そのパケットを用いて、パケットに関連するメッセージからのデータの行先を見出し、それを予約することを含むメッセージのコネクションコンテキストを生成することができ、そのコンテキストはＣＣＢの形態を取る。本実施例は、高速パス及び非高速パスの候補の両方の処理のために一個の専用ホストスタック４４を用いるが、以下に説明する実施例では、高速パスの候補は、非高速パスの候補とは異なるホストスタックによって処理される。その初めのパケットから幾つかのデータ（Ｄ１）６６が、矢印６６で示すように、所望に

応じて記憶機構３５における行先に送られ得る。次にＣＣＤが、矢印６４に示すように、キャッシュ６２に保存されるＣＰＤ３０を送信する。例えば、ＴＣＰ／ＩＰに代表されるような伝統的なコネクションベースのメッセージの場合は、初めのパケットが、ＣＣＢが生成されてＣＰＤ３０にわたされる前にホスト間で生ずるコネクション初期化ダイアログの一部であり得る。

【００２４】

ここで図４Ｃを参照すると、初めのパケットと同一のコネクションからの後続のパケットが、ＣＰＤ３０によってネットワーク２５から受信されたとき、パケットのヘッダ及びデータは、受信ロジック３２によって検査され、そのヘッダは構文解析されて、メッセージパケットの要約及び対応するＣＣＢを見つけるためのハッシュが生成され、その要約及びハッシュはワードに含められる。そのワードは、パケットとともにメモリ６０に一時的に保存される。プロセッサ５５は、キャッシュ６２に保存された各ＣＣＢとハッシュとの間の一致をチェックし、その一致を見出したときには、矢印７２に示すように、高速パスを介して記憶機構３５における行先に直接データ（Ｄ２）７０を送信し、セッション層４２、トランスポート層４０、ネットワーク層３８、及びデータリンク層３６はバイパスさせる。メッセージからの残りのデータパケットは、ＤＭＡによって直接記憶機構に送信され得、ＣＰＵスタック４４による比較的低速のプロトコル層処理及び反復されるコピー処理は回避される。

【００２５】

図４Ｄは、例えば図４Ｃに示されるような、それに対して高速パスがコネクションが確立されたメッセージが、ＣＰＤによって容易に取り扱われないパケットを有するときの稀な例を取扱うための手順を示す。この場合、そのパケットは、プロトコルスタックによって処理されるべく送信され、プロトコルスタックは、矢印７６で示すように、ＣＰＤとの制御ダイアログを介してキャッシュ６２からのメッセージのためにＣＣＢにわたされ、そのメッセージの処理を引き継がせるためにＣＰＵに信号が送られる。次にプロトコルスタックによる低速パス処理によって、矢印８２に示すように、そのパケットからのデータ（Ｄ３）８０が記憶機構３５に送られることになる。一旦そのパケットが処理され、誤り状態が補正されると、ＣＣＢは制御ダイアログを介してキャッシュ６２に戻され得、そのメッセージの後続のパケットからのペイロードデータは、同様にＣＰＤ３０の高速パスを介して送信され得ることになる。従って、ＣＰＵ及びＣＰＤは協働して、所定のメッセージが高速パスハードウェア処理に従って処理されるか、より一般的なＣＰＵによるソフトウェア処理に従って処理されるのかを決定する。

【００２６】

リモートホスト２２へ供給するためのホスト２０からネットワーク２５へのメッセージの送信も、図５に示すように、ＣＰＵを介した順次式プロトコルソフトウェア処理か、ＣＰＤ３０を介した高速化ハードウェア処理の何れかによって処理され得る。記憶機構３５からＣＰＵ２８によって選択されたメッセージ（Ｍ）９０は、矢印９２及び９６によって示されるように、スタック４４によって処理するためセッション層４２に送信され得る。しかし、コネクションが存在し、かつＣＰＤ３０がそのメッセージのための適切なＣＣＢを既に有している状況では、データパケットはホストスタック４４をバイパスし、ＤＭＡによって直接メモリ６０に送信され得る。このとき、プロセッサ５５は、各データパケットに全ての適切なプロトコル層を含む１個のヘッダを加え、得られたパケットをリモートホスト２２への送信のためネットワーク２５に送る。この高速パス送信によって、たとえ１個のパケットのための処理でも非常に高速化することができ、より大きいメッセージの場合は大幅に高速化する。

【００２７】

高速パスコネクションのためのメッセージは現存しておらず、従って高速パス送信をガイドするための適切な制御及び状態情報と共にＣＣＢを生成することによって利益が得られる。ＴＣＰ／ＩＰ又はＳＰＸ／ＩＰＸにに代表される伝統的なコネクションベースのメッセージの場合、ＣＣＢは、コネクション初期化ダイアログの間に生成される。ＴＴＣＰ／

ＩＰに代表されるような高速コネクションメッセージの場合、ＣＣＢはペイロードデータを送信する同じトランザクションで生成され得る。この場合、ペイロードデータの送信が高速パスコネクションをセットアップするために用いられたリクエストに対する応答であり得る。いずれの場合にも、ＣＣＢは、各プロトコル層に関連するプロトコル及びステータス情報を提供し、これには、ユーザが関与するもの及び１回の情報の転送のための記憶空間が含まれている。ＣＣＢは、プロトコルスタック４４によって生成され、そのプロトコルスタックは、次に、矢印９８によって示されるように、ＣＰＤのコマンドレジスタに書き込むことによってＣＣＢをＣＰＤ３０にわたす。ＣＣＢによってガイドされ、プロセッサ５５は、ホストメモリ３５における発信元からのデータのネットワークフレームサイズにされた一部分を、矢印９９に示すように、ＤＭＡを用いてそれ自身のメモリ６０に送る。次にプロセッサ５５は、適切なヘッダ及びチェックサムをそのデータの一部分の先頭に付加し、得られたフレームを関連するプロトコルの規約に準拠させてネットワーク２５に送信する。ＣＰＤ３０は、全てのデータがその行先に達したという確認応答を受信した後、ＣＰＤは応答バッファに書き込むことによってホスト３５に通知する。

【００２８】

従って、データ通信の高速パス送信はまた、ホストをＣＰＵのフレーム毎の処理から解放させる。データ送信の大部分は、高速パスによってネットワークに送信され得る。入出力の両方が高速パスであることによって、上位層レベル即ちセッション層レベル以上における機能による割り込みが大幅な低減を達成され、かつネットワークマイクロプロセッサとホストとの間の相互作用は上位層が作ろうとする全転送サイズを用いて行われることになる。高速パス通信の場合、割り込みは（その大部分が）上位層メッセージトランザクション全体の始まりと終わりにおいてのみ生じ、各階層の部分及びそのトランザクションのパケットの送受信において割り込みは生じない。

【００２９】

ホスト１５２に対するネットワークインタフェースを提供する、単純化されたインテリジェントネットワークインタフェースカード（ＩＮＩＣ）１５０が図６に示されている。ＩＮＩＣ１５０のハードウェアロジック１７１は、ＩＮＩＣとホストとを接続する周辺バス（ＰＣＩ）１５７でネットワーク１５５に接続されている。この実施例において、ホスト１５２はＴＣＰ／ＩＰプロトコルスタックを有し、それはネットワーク１５５から受け取ったメッセージフレームの順次式ソフトウェア処理のための低速パス１５８を提供する。ホスト１５２のプロトコルスタックは、データリンク層１６０、ネットワーク層１６２、トランスポート層１６４、及びホスト１５２における通信データの発信元又は行先１６８を提供するアプリケーション層１６６を有する。図示されていない他の層、例えばセッション層やプレゼンテーション層もホストスタック１５２に含められ得、発信元又は行先はデータの性質に応じて変わり、実際にはアプリケーション層であり得る。

【００３０】

ＩＮＩＣ１５０は、ホストのプロトコルスタックを含む低速パス１５８に沿って処理するメッセージ、又はホストのプロトコルスタックをバイパスする高速１５９に沿って処理するメッセージとの間の選択を行うネットワークプロセッサ１７０を有する。受信されたパケットのそれぞれは、ＩＮＩＣ１５０に含められたフライバイ（fly by）ハードウェアロジック１７１上で処理され、従って或るパケットの全てのプロトコルヘッダは、プロトコル層の間でデータのコピー、移動、記憶させることなく処理され得る。ハードウェアロジック１７１は、パケットバイトがそのハードウェアを通過するときに、選択されたヘッダバイトを分類することによって１回で所定のパケットのヘッダを処理する。選択されたバイトの処理の結果は、チェックサム検査を含むパケットの要約が生成されるまで、パケットのどの他のバイトを分類するかを決定することの助けとなる。次に受信されたパケットからのデータ及び処理されたヘッダが、そのパケットのヘッダ及びステータスを要約するワードとともにＩＮＩＣの記憶機構１８５に記憶される。ネットワーク記憶機構の構成のために、ＩＮＩＣ１５０はＩＤＥ、ＳＣＳＩ、又は類似のインタフェースを有するディスクドライブのような周辺記憶装置に接続され得、その記憶装置のためのファイルキャッシ

ュはＩＮＩＣ１５０のメモリ１８５上に存在する。そのようなネットワークインタフェースの幾つかは１つのホストのために存在し得、それぞれが関連する記憶装置を有する。

【００３１】

ネットワーク１５５からＩＮＩＣ１５０によって受信されたメッセージパケットのハードウェア処理は、図７により詳細に示されている。受信されたメッセージパケットは、初めにメディアアクセスコントローラ１７２に入り、そのメディアアクセスコントローラはネットワークへのＩＮＩＣのアクセス及びパケットの受信を制御し、かつネットワークプロトコル管理のための統計的情報を提供する。そこから１回に１バイトのデータがアセンブリレジスタ１７４に流れる。アセンブリレジスタ１７４は、この実施例では１２８ビットの幅を有する。そのデータはフライバイシーケンサ１７８によって分類されるが、後に図８を参照してより詳細に説明するように、フライバイシーケンサは、パケットのバイトを、それらが移動している（fly by）ときに調べ、パケットを要約するために用いられるそれらのバイトからステータスを生成する。生成されたそのステータスはマルチプレクサ１８０によってデータと併合され、得られたデータはＳＲＡＭ１８２に格納される。パケット制御シーケンサ１７６は、フライバイシーケンサ１７８を監視し、メディアアクセスコントローラ１７２からの情報を調べ、データのバイトをカウントし、アドレスを生成し、ステータスを送り、かつアセンブリレジスタ１７４からＳＲＡＭ１８２、更にはＤＲＡＭ１８８へのデータの移動を管理する。パケット制御シーケンサ１７６は、ＳＲＡＭコントローラ１８３を介してＳＲＡＭ１８２におけるバッファを管理し、かつデータをＳＲＡＭ１８２からＤＲＡＭ１８８のバッファへ移動させる必要があるときにＤＲＡＭコントローラ１８６への表示も行う。一旦パケットのためのデータの移動が終了し、全てのデータがＤＲＡＭ１８８のバッファに移動すると、パケット制御シーケンサ１７６は、フライバイシーケンサ１７８において生成されたステータスを、パケットデータの先頭に付加するべくＤＲＡＭ１８８のバッファの先頭部及びＳＲＡＭ１８２に動かす。次にパケット制御シーケンサ１７６は、受信バッファディスクリプタを受信キューに入れるようにキューマネージャに要求し、受信キューはプロセッサ１７０にパケットがハードウェアロジック１７１によって処理されたこと、及びその要約されたステータスを通知する。

【００３２】

図８は、フライバイシーケンサ１７８が幾つかの段を有していることを示し、この段のそれぞれは通常パケットヘッダの特定の部分を集中して取扱い、従ってその段に関連するステータスを生成するため特定のプロトコル層を集中して取り扱う。この実施形態では、フライバイシーケンサ１７８が、メディアアクセス制御シーケンサ１９１、ネットワークシーケンサ１９２、トランスポートシーケンサ１９４、及びセッションシーケンサ１９５を有する。より高位のプロトコル層に関連するシーケンサを追加的に提供することもできる。フライバイシーケンサ１７８は、パケット制御シーケンサ１７６及びフライバイシーケンサに所定のバイトがアセンブリレジスタ１７４から利用可能であるか否かを伝えるパケット制御シーケンサによる所定のポインタによってリセットされる。メディアアクセス制御シーケンサ１９１は、バイト０－５を見ることにより、或るパケットが、別のホストではなくホスト１５２を、又は別のホストに加えてホスト１５２をアドレス指定されているかを決定する。パケットのオフセット１２及び１３もメディアアクセス制御シーケンサ１９１によって処理され、例えばそのパケットがイーサネット又は８０２．３であるか否かのタイプフィールドを決定する。そのタイプフィールドがイーサネットである場合、それらのバイトはメディアアクセス制御シーケンサ１９１にそのパケットのネットワークプロトコルの種類も伝える。８０２．３の場合は、それらのバイトは全フレームの長さを表示し、メディアアクセス制御シーケンサ１９１は更にパケット内の８個のバイトをチェックし、ネットワーク層の種類を決定する。

【００３３】

大部分のパケットについて、ネットワークシーケンサ１９２は、受信されたヘッダ長が正しい長さを有していることを検査し、ネットワーク層のヘッダを検査する。高速パス候補について、ネットワーク層のヘッダは、メディアアクセス制御シーケンサ１９１によって

行われた解析からＩＰ又はＩＰＸであることが判っている。例えば、タイプフィールドが８０２．３であり、ネットワークプロトコルがＩＰであると仮定すると、ネットワークシーケンサ１９２はＩＰの種類を決定するためにバイト２２で始まるネットワークヘッダの初めのバイトを解析する。ＩＰヘッダの初めのバイトは、ネットワークシーケンサ１９２によって処理され、そのパケットがどのＩＰの種類に関係しているかを決定する。そのパケットが、例えばＩＰバージョン４に関係するものであると決定することにより、ネットワークシーケンサ１９２による更なる処理が方向付けられ、ネットワークシーケンサはパケットのトランスポートヘッダプロトコルの表示のためＩＰヘッダ内の１０個のバイトに配置されたプロトコルのタイプを調べる。例えば、イーサネット上のＩＰの場合、そのＩＰヘッダはオフセット１４から始まり、プロトコルタイプのバイトは、オフセット２３であり、それはネットワークロジックによって処理され、トランスポート層プロトコルが例えばＴＣＰであるか否かが決定される。通常２０－４０バイトであるネットワークヘッダの長さから、ネットワークシーケンサ１９２が、トランスポート層ヘッダの検査のため、パケットのトランスポート層ヘッダの始まりを決定する。トランスポートシーケンサ１９４は、トランスポート層ヘッダ及びデータのためのチェックサムを生成してもよく、トランスポート層ヘッダ及びデータは、少なくともＴＣＰの場合にＩＰヘッダからの情報を含み得る。

【００３４】

ＴＣＰパケットの例について続けると、トランスポートシーケンサ１９４は、１つにはそのメッセージのためのＴＣＰ発信元及び行先ポートを決定するため、例えばそのパケットがＮｅｔＢｉｏｓか他のプロトコルであるかを決定するために、ヘッダのトランスポート層部分における初めの数バイトの解析も行う。ＴＣＰヘッダのバイト１２が、トランスポートシーケンサ１９４によって処理され、ＴＣＰヘッダの長さが決定され、検査される。ＴＣＰヘッダのバイト１３は、肯定応答（ack）フラグ及びプッシュフラグを除いて、例えばリセットや終了のような、プロセッサにこのパケットを例外として分類させ得る予期しないオプションを表示し得るフラグを含む。ＴＣＰオフセットバイト１６及び１７はチェックサムであり、それはハードウェアロジック１７１によって引き出されたり、格納されたりし、フレームの残りの部分はそのチェックサムに対して検査される。

【００３５】

セッションシーケンサ１９５は、セッション層ヘッダの長さを決定し、セッション層ヘッダは、ＮｅｔＢｉｏｓの場合には４バイトに過ぎず、そのうちの２バイトはＮｅｔＢｉｏｓペイロードデータの長さを通知するが、他のプロトコルの場合にはより長いこともあり得る。セッションシーケンサ１９５は、例えば読み出し又は書き込みとしてメッセージのタイプを分類するためにも用いられ得、そのために高速パスが特に有益なものとなり得る。メッセージの種類に応じて更なる上位層ロジックの処理が、フライバイシーケンサ１７８及びパケット制御シーケンサ１７６のハードウェアロジック１７１によって行われ得る。従って、ハードウェアロジック１７１はインテリジェントに１つのバイトストリームから選択されたバイトを分類することによってヘッダのハードウェア処理を誘導し、そのパケットのステータスが処理の進行中に決定された分類から構築される。一旦パケット制御シーケンサ１７６が全てのパケットがフライバイシーケンサ１７８によって処理されたことを検出すると、パケット制御シーケンサ１７６は、フライバイシーケンサ１７８によって生成されたステータス情報にパケット制御シーケンサ１７６によって生成されたすテータス情報を加え、プロセッサ１７０によるパケットの取扱いにおける便宜のため、そのパケットにそのステータス情報を先頭付加（パケットの先頭部分への付加）する。パケット制御シーケンサ１７６によって生成された追加のステータス情報は、メディアアクセスコントローラ１７２のステータス情報及び発見された全ての誤り、又はアセンブリレジスタ若しくはＤＲＡＭバッファの何れかにおけるデータオーバーフロー、又はそのパケットに関連する他の種々雑多な情報を有する。パケット制御シーケンサ１７６は、キューマネージャ１８４を介して受信バッファキュー及び受信統計的キューにエントリを入れることも行う。

【００３６】
ハードウェアロジック１７１によりパケットを処理することの利点は、従来のような順次ソフトウェアプロトコル処理とは異なり、そのパケットが各プロトコル層ヘッダの処理のために記憶機構に格納されたり、転送されたり、コピーされたり、又は引き出されたりする必要がなく、処理効率を劇的に向上させ、各パケットの処理時間を短縮できる点にある。そのパケットはビットがネットワークから受信されるときのビットレート、例えば１００ＢａｓｅT接続の場合１００メガビット／秒で処理され得る。従って、この速度で受信された６０バイトの長さを有するパケットを分類する時間は、約５マイクロ秒である。ハードウェアロジック１７１でそのパケットを処理し、パケットデータを高速パスを介してそのホスト行先に送信するためにかかる全時間は、６６ＭＨｚのＰＣＩバスを用いた場合、約１６マイクロ秒以下であり得るが、３００ＭＨｚのPentiumII（登録商標）プロセッサによる従来通りのソフトウェアプロトコル処理では、使用中の装置において２００マイクロ秒もの時間がかかかり得る。従って、従来型の順次式ソフトウェアプロトコル処理を用いる高速ＣＰＵの場合と比較して、高速パス１５９によって１桁以上の処理時間の短縮が達成され得、ＣＰＵの割り込みの低減やホストのバスの大域幅の節約によってもたらされる追加の時間短縮を考慮しなくても、ハードウェアロジック１７１及びプロセッサ１７０でプロトコルヘッダを処理することによって劇的な高速化が提供されることを立証している。

【００３７】
プロセッサ１７０は、記憶機構１８５に保持された受信されたメッセージパケットのそれぞれについて、そのパケットが高速パス１５９の候補であるか否かを選択し、そうである場合には、高速パスが既にそのパケットが属するコネクションに対してセットアップされているか否かを調べるためのチェックを行う。これを行うため、プロセッサ１７０は、初めにそのパケットヘッダが高速パス候補のために定義されたプロトコルのものであるか否かを決定するためにヘッダステータスの要約をチェックする。そうでない場合には、プロセッサ１７０は、ＩＮＩＣ１５０におけるＤＭＡコントローラに、そのパケットを低速パス１５８処理のためにホストに送信する命令を送る。メッセージが低速パス１５８処理される場合でも、ＩＮＩＣ１５０は検査及びメッセージの種類の決定のような初期手順を実行し、検査されたメッセージを少なくともホストのデータリンク層１６０に送る。

【００３８】
高速パス１５９の候補の場合、プロセッサ１７０は、そのヘッダステータスの要約がＩＮＩＣに保持されたＣＣＢに一致するか否かを調べるためのチェックを行う。そうである場合には、パケットからのデータは高速パス１５９を通してホストの行先１６８に送信される。高速パス１５９の候補のパケット要約がＩＮＩＣに保持されたＣＣＢに一致しない場合には、そのパケットは低速パス処理のためにホスト１５２に送信されて、そのメッセージのためのＣＣＢが生成される。高速パス１５９を用いることは、断片化したメッセージの場合や他の複雑さのために必要でない、又は望ましいものではないこともある。しかし、大部分のメッセージについては、ＩＮＩＣの高速パス１５９は、メッセージ処理を非常に高速化し得る。従ってＩＮＩＣ１５０は、従来のように所定のパケットの運命を決定するために幾つかのプロトコル層のそれぞれにおいてステートマシンを用いるのと異なり、処理進行中に収集される情報に基いてデータをその行先に直接送信するか否かを決定する１個のステートマシンプロセッサ１７０を提供するものである。

【００３９】
ホスト１５２において受信された表示即ちパケットの処理において、ホストのプロトコルドライバは、その表示が高速パス又は低速パスの何れであるかに基いて処理経路を選択する。ＴＣＰ／ＩＰ又はＳＰＸ／ＩＰＸメッセージは、そのドライバによって形成され、高速パスパケットとの一致をとの一致をとり、かつそれをコネクション行先１６８に導くために、ＩＮＩＣに送られる。ＴＴＣＰ／ＩＰメッセージの場合、ドライバは、初期リクエストパケットの処理から、そのトランザクションのためのコネクションコンテキストを生成することができる。この処理では、メッセージ行先１６８の位置指定を行い、次にその

行先からの応答のための高速パスを提供するためにＣＣＢの形態でＩＮＩＣにそのコンテキストを送る。ＣＣＢは、プロトコル層及びメッセージのパケットに関するコネクション及び状態情報を含む。従ってＣＣＢは、発信元及び行先メディアアクセス制御（ＭＡＣ）アドレス、発信元及び行先ＩＰ又はＩＰＸアドレス、発信元及び行先ＴＣＰ又はＳＰＸポート、タイマーのようなＴＣＰ変数、ウィンドウプロトコルをスライドさせるための受信及び送信ウインドウ、及びセッション層プロトコルを示す情報を有し得る。

【００４０】

ＩＮＩＣにおけるハッシュテーブルにＣＣＢをキャッシュすることにより、そのパケットが高速パス１５９を介して処理され得るか否かを決定するべく入力されるパケットを要約するワードとの迅速な比較処理が行われるようになり、一方全てのＣＣＢが処理のためにＩＮＩＣにも保持される。この比較を高速化する他の方法としては、コンテンツのアドレス指定可能なメモリ（ＣＡＭ）のようなハードウェア補助機構及びＢツリーのようなソフトウェア処理等が挙げられる。ＩＮＩＣマイクロコード又は比較器回路がＣＣＢとの一致を検出すると、ＤＭＡコントローラはＣＰＵによる割り込み、プロトコル処理又はコピー処理を行うことなく、行先１６８にパケットからのデータをおく。受信したメッセージの種類に応じて、そのデータの行先は、ホスト１５２におけるセッション層、プレゼンテーション層、又はアプリケーション層、又はファイルバッファキャッシュであり得る。

【００４１】

図９は、ファイルサーバとして用いられるホスト２０２に接続されたＩＮＩＣ２００を示す。このＩＮＩＣは、一般にファスト・イーサネットとして知られる８０２．３ｕ標準を用いる幾つかのネットワークコネクションのためのネットワークインタフェースを提供する。ＩＮＩＣ２００は、ＰＣＩバス２０５によってサーバ２０２に接続されており、サーバ２０２はＭＡＣ層２１２、ネットワーク層２１５、トランスポート層２１７、及びアプリケーション層２２０を含むＴＣＰ／ＩＰ又はＳＰＸ／ＩＰＸプロトコルスタックを有し、アプリケーション層の上に発信元／行先２２２が設けられているが、前に述べたように、アプリケーション層が発信元／行先であり得る。ＩＮＩＣは、ネットワークライン２１０、２４０、２４２、及び２４４にも接続されており、これらのネットワークラインは好ましくはファスト・イーサネット、ツイストペア、光ファイバー、同軸ケーブル、又は他の種類のラインであって、それぞれ１００Ｍｂ／ｓのデータ転送が可能なものであるが、より高速又はより低速のデータレートも可能である。ネットワークライン２１０、２４０、２４２、及び２４４は、それぞれハードウェア回路の専用の行に接続されており、前記ハードウェア回路はそれぞれのネットワークラインから受信されたメッセージパケットの検査及び要約を行うことができる。従って、ライン２１０は、シーケンサ２５０の第１の水平方向の行に接続され、ライン２４０は、シーケンサ２６０の第２の水平方向の行に接続され、ライン２４２は、シーケンサ２６２の第３の水平方向の行に接続され、ライン２４４は、シーケンサ２６４の第４の水平方向の行に接続される。パケットがその水平方向のハードウェアの行の１つによって検査、要約された後、そのパケットはそのステータス要約とともに記憶機構２７０に格納される。

【００４２】

ネットワークプロセッサ２３０は、ＩＮＩＣ２００に格納されたその要約及び任意のＣＣＢとの比較に基いて、ホストによって処理される低速パス２３１を通してそのパケットを送信するか否かを決定する。大部分のパケットは、そのような順次処理を避けることができ、一致するＣＣＢに従ってサーバ内のデータ行先に高速パス２３７を通してＤＭＡにより送信されるそのデータ部分を有する。同様に、高速パス２３７は、データをネットワーク送信のためにパケットに分割し、全てのヘッダを付加してプロセッサ２３０によって発信元２２２から任意のネットワークラインへ直接データを送信する経路を提供し、同様にＣＰＵによる処理及び割り込みを最小限にする。明示のため、水平シーケンサ２５０のみがアクティブ状態として示されているが、実際にはシーケンサ行２５０、２６０、２６２、及び２６４のそれぞれが、他の全てのシーケンサ行と同時に全二重通信を提供する。専用ＩＮＩＣ２００は、メッセージパケットとともに動作するとき、ソフトウェアプロトコ

ルスタックに従ってそれらのヘッダを順次処理する高度な汎用ホストＣＰＵよりも非常に高速である。

【００４３】

ファイル転送のような大きいメッセージのために最も一般的に用いられるネットワークプロトコルの１つは、ＴＣＰ／ＩＰを通してのサーバメッセージブロック（ＳＭＢ）である。ＳＭＢはファイルが書き込まれるディスク又はプリンタのような特定の操作のために必要なリソースが、その操作が生成されるホストに存在するか、或いは結び付けられているか、又はファイルサーバのようなネットワークに接続された他のホストに存在するかを決定するリダイレクタソフトウェアとともに動作し得る。ＳＭＢ及びサーバ／リダイレクタは、従来はトランスポート層によりサービスされるが、本発明ではＳＭＢ及びリダイレクタが、ＩＮＩＣによりサービスされ得る。この場合には、大きいＳＭＢトランザクションを受け取ったとき、ＩＮＩＣバッファからＤＭＡコントローラによってデータを送信することにより、ホストが取り扱わなければならない割り込みが著しく低減し得る。更に、このＤＭＡは一般的にファイルデイバスキャッシュにあるその最終的な行先にデータを転送する。本発明のＳＭＢ送信は、基本的に上述のＳＭＢの受信を逆方向にたどるものであり、ホストからＩＮＩＣにデータが転送され、バッファに格納されるとともに、関連するプロトコルヘッダがネットワークラインを介してのリモートホストへの転送のためにＩＮＩＣにおけるデータの先頭に付加される。ホストの反復される割り込み無しに、カスタム設計のハードウェアを介しての複数のＴＣＰ／ＩＰ、ＮｅｔＢｉｏｓ、及びＳＭＢプロトコル層及び複数のパケットのＩＮＩＣによる処理によって、ＳＭＢメッセージのネットワークラインへの送信速度を著しく高めることができる。

【００４４】

図１０に示すように、所定のメッセージがホスト２０２によって処理されるか、又はＩＮＩＣ２００によって処理されるかを制御するために、メッセージコマンドドライバ３００がホスト２０２にインストールされ、ホストプロトコルスタック３１０と協働して機能を果たす。コマンドドライバ３００は、メッセージの受信又は送信に介入し、ＣＣＢを生成し、ＩＮＩＣ２００からＣＣＢを送受信することができ、従って処理能力の向上を除いて、ＩＮＩＣの機能はユーザからは透過的なものとなる。また、図面にはＩＮＩＣメモリ３０４及びＩＮＩＣミニポートドライバ３０６が示されており、ＩＮＩＣミニポートドライバはネットワーク２１０から受信したメッセージパケットを、そのパケットが高速パス候補としてのラベル付けされているか否かに応じて、従来通りのプロトコルスタック３１０又はコマンドプロトコルスタック３００の何れかへ誘導することができる。従来通りのプロトコルスタック３１０は、高速パス候補としてラベル付けされておらず、従ってコマンドスタック３００によって処理されないメッセージの従来通りの下位層の処理のために、データリンク層３１２、ネットワーク層３１４、及びトランスポート層３１６を有する。下位層スタック３１０の上に存在するのは上位層３１８であり、これはやりとりされるメッセージに応じて、セッション層、プレゼンテーション層、及び／又はアプリケーション層を表す。コマンドドライバ３００は、同様にデータリンク層３２０、ネットワーク層３２２、及びトランスポート層３２５を有する。

【００４５】

ドライバ３００は、メッセージをネットワーク２１０に送信するために、上位層３１８から送信されたメッセージが、コマンドスタック３００、それに続けてＩＮＩＣ高速パスによって処理されるか、従来通りのスタック３１０によって処理されるかを決定する上位層インタフェース３３０を有する。その上位層インタフェース３３０が、従来通りホストのプロトコルスタックによるプロトコル処理の後のネットワークへの送信を目的とする上位層３１８から適切なメッセージを受け取ったとき、そのメッセージはドライバ３００に送られる。次にＩＮＩＣが、ＩＮＩＣ　ＤＭＡユニットを介しての送信のために、メッセージデータのネットワークサイズにされた一部分を得て、そのデータの一部分の先頭にヘッダを付加し、形成されたメッセージパケットをワイヤを通して送信する。逆に、ＴＣＰ、ＴＴＣＰ、ＳＰＸ、又は類似のメッセージパケットを、高速パスコネクションのセットア

ップにおいて用いられるようにネットワーク２１０から受け取ったとき、ミニポートドライバ３０６は、そのメッセージパケットをコマンドドライバ３００に処理するために転送する。ドライバ３００は、そのメッセージパケットを処理してそのメッセージのためのコンテキストを生成し、ドライバ３０２は、そのコンテキスト及びコマンド命令を、高速パスを通しての同一のコネクションにおいて後のメッセージのデータを送信するためのＣＣＢとして、ＩＮＩＣ２００に戻す。ＩＮＩＣによって、数百のＴＣＰ、ＴＴＣＰ，ＳＰＸ、又は類似のＣＣＢコネクションは、その数に限界なく維持され得るが、ＩＮＩＣキャッシュが一杯になった場合のため、最小使用頻度（ＬＲＵ）アルゴリズムが用いられる。ドライバ３００は、ＣＣＢとしてＩＮＩＣ２００に送られるＴＴＣＰリクエストのためのコネクションコンテキストも生成することができ、これよってそのリクエストに対するＴＴＣＰ応答の高速パス転送が可能となる。高速化されていないプロトコルを有するメッセージは、プロトコルスタック３１０によって従来通りに処理され得る。

【００４６】

図１１は、Ｍｉｃｒｏｓｏｆｔ（登録商標）プロトコルメッセージのためのコマンドドライバソフトウェアのＴＣＰ／ＩＰインプリメンテーションを示す。従来通りのホストプロトコルスタック３５０は、ＭＡＣ層３５３、ＩＰ層３５５、及びＴＣＰ層３５８を有する。コマンドドライバ３６０は、ホストスタック３５０と協働して働き、ネットワークメッセージを処理する。コマンドドライバ３６０は、ＭＡＣ層３６３、ＩＰ層３６６、及びＡｌａｃｒｉｔｅｃ　ＴＣＰ（ＡＴＣＰ）層３７３を有する。従来型スタック３５０及びコマンドドライバ３６０は、ネットワークドライバインタフェース仕様（ＮＤＩＳ）層３７５を共有しており、これはＩＮＩＣミニポートドライバ３０６を相互作用する。ＩＮＩＣミニポートドライバ３０６は、従来通りのホストスタック３５０又はＡＴＣＰドライバ３６０の何れかによって処理するため受信した表示をソートする。ＴＤＩフィルタドライバ及び上位層インタフェース３８０は、同様に、ＴＤＩユーザ３８２からネットワークに送られたメッセージが、コマンドドライバに送られてＩＮＩＣの高速パスに転送されたか、或いはホストスタックによって処理されたかを判定する。

【００４７】

図１２は、典型的なクライアント１９０とサーバ２９０との間のＳＭＢ交換を示す図であり、クライアントとサーバの両方は本発明の通信装置を有し、その通信装置はそれぞれデータの高速パス転送のためのそれらのコネクションを確定するＣＣＢを有する。クライアント１９０は、ＩＮＩＣ１５０、８０２．３準拠のデータリンク層１６０、ＩＰ層１６２、ＴＣＰ層１６４、ＮｅｔＢｉｏｓ層１６６、及びＳＭＢ層１６８を有する。クライアントは、通信処理のための低速パス１５７及び高速パス１５９を有する。同様にサーバ２９０は、ＩＮＩＣ２００、８０２．３準拠のデータリンク層２１２、ＩＰ層２１５、ＴＣＰ層２１７、ＮｅｔＢｉｏｓ層２２０、及びＳＭＢ層２２２を有する。サーバはネットワークライン２４０、２４２、及び２４４に接続されているとともに、クライアント１９０に接続されたライン２１０にも接続されている。サーバも通信処理のための低速パス２３１及び高速パス２３７を有する。

【００４８】

クライアント１９０がサーバ２９０上の１００ＫＢのファイルの読み出しを行おうとしていると仮定すると、クライアントは、ネットワーク２１０を介して読み出しブロック行（ＲＢＲ）ＳＭＢを送信することによって、サーバ２９０上のそのファイルの初めの６４ＫＢを要求することを開始し得る。ＲＢＲコマンドは、例えば７６バイトに過ぎない大きさであり得、従ってサーバ上のＩＮＩＣ２００は、メッセージタイプ（ＳＭＢ）及び比較的小さいメッセージサイズを認識し、かつその７６バイトを高速パスを介してサーバのＮｅｔＢｉｏｓに直接送信する。ＮｅｔＢｉｏｓは、ＳＭＢにデータをわたし、ＳＭＢは読み出しリクエストを処理し、データの６４ＫＢを取り出して、サーバのデータバッファに入れる。次にＳＭＢは、ＮｅｔＢｉｏｓを呼出してそのデータを送信し、ＮｅｔＢｉｏｓはそのデータをクライアントのために出力する。従来型のホストでは、ＮｅｔＢｉｏｓがＴＣＰ出力を呼出し、６４ＫＢをＴＣＰにわたし、ＴＣＰはそのデータを１４６０バイトの

セグメントに分割し、各セグメントをＩＰ及び最終的にＭＡＣ（低速パス２３１）を介して出力する。本発明の場合には、その６４ＫＢのデータは、クライアント－サーバＳＭＢコネクションに関する表示とともにＡＴＣＰドライバに向かい、前記表示はＩＮＩＣによって保持されているＣＣＢを示す。次にＩＮＩＣ２００は、ホストバッファから１４６０バイトのセグメントをＤＭＡ送信し、ＴＣＰ、ＩＰ及びＭＡＣのための適切なヘッダを一度に付加し、かつ完成したパケットをネットワーク２１０（高速パス２３７）上に送信する。ＩＮＩＣ２００は、この処理を６４ＫＢの転送の全てが送信されるまで反復する。通常、クライアントからその６４ＫＢのを受信したという確認応答を受信した後、ＩＮＩＣは残りの３６ＫＢも高速パス２３７によって送信する。

【００４９】

ＩＮＩＣ１５０は、その応答が着信したとき、クライアント１９０上で動作しており、ＩＮＩＣ１５０は、受信された第１のフレームからそのコネクションが受信高速パス１５９処理（ＴＣＰ／ＩＰ，ＮｅｔＢｉｏｓ、ＣＣＢとの一致）であることを認識し、ＡＴＣＰは、その第１フレームを用いてそのメッセージのためのバッファ空間を確保し得る。後者の場合には、フレームのＮｅｔＢｉｏｓ部分の初めの１２８バイトをＡＴＣＰ高速パスを介して、ホストのＮｅｔＢｉｏｓに直接わたすことによって達成され、これによりＮｅｔＢｉｏｓ／ＳＭＢに全てのフレームのヘッダがわたされる。ＮｅｔＢｉｏｓ／ＳＭＢは、これらのヘッダを解析し、リクエストＩＤとの一致によってこれが元の行読み出しコネクションに対する応答であることを認識し、データを置くべき６４Ｋのバッファのリストを ＡＴＣＰにわたす。この段階において、ただ１個のフレームが着信しているが、この処理が生じている間に複数のフレームが着信し得る。クライアントバッファリストがＡＴＣＰにわたされると直ぐに、それはその転送情報をＩＮＩＣ１５０にわたし、ＩＮＩＣ１５０はそれらのバッファに蓄積された任意のフレームデータのＤＭＡ送信を開始する。

【００５０】

図１３は、ＩＮＩＣ２００を単純化して示した図であり、ＩＮＩＣ２００はネットワークインタフェースコントローラとプロトコルプロセッサの機能を一個のＡＳＩＣチップ４００に結合したものである。この実施形態におけるＩＮＩＣ２００は、全二重４チャネル１０／１００メガビット／秒（Ｍｂｐｓ）のインテリジェントネットワークインタフェースコントローラを提供し、このコントローラはサーバアプリケーションのための高速プロトコル処理のために設計されている。特にサーバの用途のために設計されているが、ＩＮＩＣ２００は、パーソナルコンピュータ、ワークステーション、ルータ、又は他のＴＣＰ／ＩＰ、ＴＴＣＰ／ＩＰ、又はＳＰＸ／ＩＰＸプロトコルが用いられているあらゆる他のホストに接続され得る。

【００５１】

ＩＮＩＣ２００は、４つのネットワークライン２１０、２４０、２４２、及び２４４に接続されており、これらのラインは例えばツイストペア、同軸ケーブル、又は光ファイバーのような複数の異なる経路に沿ってデータを転送することができ、そのコネクションはそれぞれ市販の物理層チップ、例えばSEEQ Technology Incorporated, 47200 Bayside Parkway, Fremont, CA94538製のmodel 80220/80221 Ethernet Media Interface Adapterを介してのメディア独立インタフェース（ＭＩＩ）を提供する。そのラインは、好ましくは８０２．３準拠であり、ＩＮＩＣとともに４つの完全なイーサネットノードを構成し、ＩＮＩＣは１０Ｂａｓｅ－Ｔ、１０Ｂａｓｅ－Ｔ２、１００Ｂａｓｅ－ＴＸ、１００Ｂａｓｅ－ＦＸ、及び１００Ｂａｓｅ－Ｔ４とともに将来のインタフェース標準をサポートしている。物理層の特定及び初期化は、ホストドライバ初期化ルーチンによって達成される。ネットワークライン２１０、２４０、２４２、及び２４４とＩＮＩＣ２００との間のコネクションは、ＭＡＣユニットのＭＡＣ－Ａ　４０２、ＭＡＣ－Ｂ　４０４、ＭＡＣ－Ｃ　４０６、及びＭＡＣ－Ｄ　４０８によって制御され、これらのＭＡＣユニットはＩＮＩＣがネットワークライン２１０、２４０、２４２、及び２４４にアクセスするときに必要な制御を行うＭＡＣサブレイヤの基本的な機能を実行するための論理回路を含む。ＭＡＣユニット４０２～４０８は、無差別でマルチキャスト又はユニキャストモードで動作すること

ができ、これによってＩＮＩＣがネットワークモニタとして機能し、ブロードキャスト及びマルチキャストパケットを受信し、かつ各ノードに対して複数のＭＡＣアドレスをインプリメントすることが可能になる。ＭＡＣユニット４０２～４０８は、シンプルネットワーク管理プロトコル（ＳＮＭＰ）のために用いられ得る統計的情報も提供する。

【００５２】

ＭＡＣユニット４０２、４０４、４０６、及び４０８は、それぞれ送受信シーケンサ、ＸＭＴ＆ＲＣＶ－Ａ４１８、ＸＭＴ＆ＲＣＶ－Ｂ４２０、ＸＭＴ＆ＲＣＶ－Ｃ４２２、及びＸＭＴ＆ＲＣＶ－Ｄ４２４にそれぞれ配線４１０、４１２、４１４、及び４１６によって接続されている。各送受信シーケンサは、メッセージフレームがそのシーケンサを通過するとき、処理進行中に幾つかのプロトコル処理ステップを実行し得る。ＭＡＣユニットと組み合わせて、この送受信シーケンサ４１８～４２２は、ハードウェア上のデータリンク層、ネットワーク層、トランスポート層、セッション層、及び使用する場合にはプレゼンテーション層及びアプリケーション層のプロトコルのためにパケットステータスをコンパイルして、従来型の順次式ソフトウェアエンジンと比較してそのようなプロトコル処理のための時間を非常に短縮させることができる。送受信シーケンサ４１０～４１４は、ライン４２６、４２８、４３０、及び４３２によってＳＲＡＭ及びＤＭＡコントローラに４４４に接続されており、ＤＭＡコントローラ４４４はＤＭＡコントローラ４３８及びＳＲＡＭコントローラ４４２を有する。スタティックランダムアクセスメモリ（ＳＲＡＭ）バッファ４４０は、ライン４４１によってＳＲＡＭコントローラ４４２に接続されている。ＳＲＡＭ及びＤＭＡコントローラ４４４は、ライン４４６を介して外部メモリ制御機構４５０と相互作用し、外部メモリバス４５５を介してダイナミックランダムアクセスメモリ（ＤＲＡＭ）バッファ４６０とフレームの送受信を行う。ＤＲＡＭバッファ４６０は、ＩＣチップ４００に隣接して配置される。ＤＲＡＭバッファ４６０は、４ＭＢ、８ＭＢ、１６ＭＢ、又は３２ＭＢとして構成され得、かつ所望に応じてチップ上に配置され得る。ＳＲＡＭ及びＤＭＡコントローラ４４４はライン４６４を介してＰＣＩバスインタフェースユニット（ＢＩＵ）４６８に接続されており、ＢＩＵ４６８はＩＮＩＣ２００とＰＣＩインタフェースバス２５７との間のインタフェースを管理する。６４ビットの多重化ＢＩＵ４６８は、スレーブ及びマスター機能の両方についてＰＣＩバス２５７に対する直接のインタフェースを提供する。ＩＮＩＣ２００は、６４ビット又は３２ビット何れかのＰＣＩ環境において動作し得、何れの構成においても６４ビットのアドレス指定をサポートしている。

【００５３】

マイクロプロセッサ４７０はライン４７２によってＳＲＡＭ及びＤＭＡコントローラ４４４に接続され、かつライン４７５を介してＰＣＩ　ＢＩＵ４６８に接続されている。マイクロプロセッサ４７０はチップ制御ストア４８０に存在するレジスタファイルに命令を送り、チップ制御ストアはＳＲＡＭの書き込み可能なオンチップ制御ストア（ＷＣＳ）及び読み出し専用メモリ（ＲＯＭ）を有し、かつライン４７７によってマイクロプロセッサに接続されている。マイクロプロセッサ４７０は入力されるフレームを処理し、ホストのコマンドを処理し、ネットワークトラフィックを誘導し、かつＰＣＩバストラフィックを誘導し得る、プログラム可能なステートマシンを提供する。３つのプロセッサは、各クロックサイクル毎に１個の命令を発して完了させる３つのレベルのパイプライン型アーキテクチャに、共通のハードウェアを用いてインプリメントされている。受信プロセッサ４８２は主として通信の受信のために用いられ、送信プロセッサ４８４は主として通信の送信のために用いられて、全二重通信を容易に行えるようにしており、一方ユーティリティプロセッサ４８６はＰＣＩレジスタアクセスの監視及び制御を含む様々な機能を提供する。

【００５４】

３つのプロセッサ４８２、４８４、及び４８６に対する命令は、オンチップ制御ストア４８０に存在する。このため３つのプロセッサの機能は容易に定義しなおすことができ、従ってマイクロプロセッサ４７０は所定の環境に適合させることができる。例えば受信機能のために必要な処理の量は、送信機能及びユーティリティ機能の何れかのために要求され

るものを上回っていることがある。この状況では、幾つかの受信機能が送信プロセッサ４８４及び／又はユーティリティプロセッサ４８６によって実行され得る。或いは、追加のレベルのパイプラインを生成して、３つではなく４以上の仮想プロセッサを生成し、その追加のレベルを受信機能専用にすることができる。

【００５５】

ＩＮＩＣ２００は、この実施形態ではＤＲＡＭ４６０におけるテーブルに維持されている最大２５６個のＣＣＢをサポートし得る。しかし、ＳＲＡＭ４４０において、順次検索を保存するためのハッシュ順のＣＣＢインデックスも存在する。一旦ハッシュが生成されるとＣＣＢはＳＲＡＭにキャッシュされ、この実施例では最大１６個のＣＣＢがＳＲＡＭにキャッシュされる。ＳＲＡＭにキャッシュされた１６のＣＣＢの割り当ては、以下に説明するように、最低使用頻度レジスタによって取り扱われる。これらのキャッシュ位置は、送信プロセッサ４８４と受信プロセッサ４８６との間で共有されており、従ってより重い負荷のかかるプロセッサがより多くのキャッシュバッファを使用することができる。また、シーケンサ間で共有される８個のコマンドバッファ及び８個のヘッダバッファも存在する。所定のヘッダ又はコマンドバッファは、所定のＣＣＢバッファに静的にリンクされていない。そのリンクはフレーム毎に動的だからである。

【００５６】

図１４は、パイプライン型マイクロプロセッサ４７０の概要を示しており、このマイクロプロセッサでは受信、送信、及びユーティリティプロセッサのための命令が、クロックのインクリメントＩ、ＩＩ、ＩＩＩに従って３つの交互のフェーズで実行され、そのフェーズはパイプライン段階のそれぞれに対応している。各フェーズは異なる機能を担っており、３つのプロセッサのそれぞれは、各クロックインクリメントの間に異なるフェーズを占める。通常各プロセッサは、制御ストア４８０からの異なる命令ストリームに基いて動作し、それぞれそのフェーズを通してそれ自身のプログラムカウンタ及びステータスを有する。

【００５７】

一般に、パイプライン型マイクロプロセッサの第１命令フェーズ５００は、命令を終了し、その結果を行先のオペランドに格納し、次の命令を取り出し、かつその次の命令を命令レジスタに格納する。第１レジスタセット４９０は、命令レジスタを含む複数のレジスタを提供し、第１レジスタセットのための制御機構セット４９２は第１レジスタセット４９０に対する記憶機構のための制御を提供する。幾つかの項目が制御機構４９２によって変更されることなく第１フェーズを通過し、代わりに第１レジスタセット４９０又はＲＡＭファイルレジスタ５３３に単にコピーされる。第２命令フェーズ５６０は、第１レジスタセット４９０の命令レジスタに格納された命令をデコードし、生成された全てのオペランドを集める命令デコーダ及びオペランドマルチプレクサ４９８を有し、その収集されたオペランドは次に第２レジスタセット４９６のデコードレジスタに格納される。第１レジスタセット４９０、第２レジスタセット４９６、及び第３命令フェーズ６００で用いられる第３レジスタセット５０１は、多くの同一のレジスタを有し、これは図１５Ａ乃至図１５Ｃにより詳細に示されている。命令デコーダ及びオペランドマルチプレクサ４９８は、ＲＡＭファイルレジスタ５３３のデータポート及び２つのアドレスから読み出しを行うことができ、ＲＡＭファイルレジスタは、第１フェーズ５００及び第２フェーズ５６０の両方において動作する。プロセッサ４７０の第３フェーズ６００は、第２レジスタセットからのオペランドに対するＡＬＵ処理を行う論理演算装置（ＡＬＵ）６０２を有し、ＡＬＵは処理の結果を第３レジスタセット５０１に含められた結果レジスタに格納する。スタック交換機構６０８は、レジスタスタックを順序付けしなおすことができ、キューマネージャ５０３はプロセッサ４７０のためのキューを編成することができ、その結果は第３レジスタセットに格納される。

【００５８】

命令は、環状パイプライン５０５によって示されるように第１フェーズ、次にそれに続く第３フェーズをまで継続する。所定のフェーズ内での組み合わせによる遅れを最小限する

ために、命令実行の３つのフェーズにわたって様々な機能が分散されていることに注意されたい。この実施形態の６６ＭＨｚの周波数で、各クロックインクリメントが終了するまでに１５ナノ秒の時間がかかり、３つのプロセッサのそれぞれに対して１個の命令が完了するまで全体で４５ナノ秒かかることになる。循環する命令フェーズは図１５Ａ乃至図１５Ｃにより詳細に示されており、各フェーズが異なる図面において示されている。

【００５９】

詳述すると、図１５Ａは第１レジスタセット４９０及び関連する制御機構４９２を有する第１フェーズ５００の幾つかの特定のハードウェア機能を示す。第２レジスタセット４９２のための制御機構には、ＳＲＡＭ制御機構５０２が含まれ、これはアドレスをロードし、データをＳＲＡＭアドレス及びデータレジスタ５２０に書き込むための論理制御機構である。従って第３フェーズ６００からのＡＬＵ６０２の出力は、ＳＲＡＭ制御機構５０２によってＳＲＡＭアドレス及びデータレジスタ５２０のアドレスレジスタ又はデータレジスタに置かれ得る。ロード制御機構５０４は、同様にファイルコンテキストレジスタ５２２に或るファイルに対するコンテキストの書き込みのための制御を提供し、別のロード制御機構５０６は様々なデータをフリップフロップレジスタ５２５に格納するための制御を提供する。例えば、送られたビットがセットされたか否かを示すＡＬＵ条件コードは、第１フェーズ５００において実行される処理なしで、ＡＬＵ条件コードレジスタ５２８にクロックに応じて入力される。フラグデコード５０８は、例えばロックの設定のようなフラグレジスタ５３０に格納された様々な機能を実行することができる。

【００６０】

ＲＡＭファイルレジスタ５３３は、アドレス及びデータのための１つの書き込みポート及びアドレス及びデータのための２つの読み出しポートを有し、従って２以上のレジスタが一度に読み出され得る。上述のように、ＲＡＭファイルレジスタ５３３は基本的に第１及び第２フェーズをまたいでおり、即ち第１フェーズ５００において書込みが行われ、第２フェーズ５６０において読み出しが行われる。制御ストア命令５１０によって制御ストア４８０から新たなデータが入ってくるため、図面には示されていないが、命令レジスタ５３５に格納された命令をプロセッサがプログラムしなおすことが可能となる。このためのアドレスは、フェッチアドレスレジスタ５３８に格納されたアドレスのうちどのアドレスを取り出すかを決定するフェッチ制御レジスタにおいて生成される。ロード制御機構５１５は、プログラムカウンタのための命令を供給し、プログラムカウンタは制御ストアのためのフェッチアドレスの場合と概ね同様に動作する。３つのレジスタの後入先出スタック５４４は、このフェーズにおいて他の操作を受けずに第１レジスタセットにコピーされる。最後に、デバッグアドレス５４８のためのロード制御機構５１７が所望に応じて含められるが、これによって生じ得る誤りの訂正が可能となる。

【００６１】

図１５Ｂは、ＲＡＭファイルレジスタ５３３からのアドレス及びデータの読み出しを含む、第２マイクロプロセッサフェーズ５６０を示す。スクラッチＳＲＡＭ５６５には、第３フェーズにインクリメントされるまで初めの２つのフェーズを通過するレジスタを有する第１レジスタセットのＳＲＡＭアドレス及びデータレジスタ５２０から書込みが行われる。スクラッチＳＲＡＭ５６５は、命令デコーダ及びオペランドマルチプレクサ４９８によって読み出される。これは、上述のように、スタック５４４、デバッグアドレス５４８、及びＳＲＡＭアドレス及びデータレジスタのような例外を除いて、第１レジスタセットの大部分のレジスタと同様である。命令デコーダ及びオペランドマルチプレクサ４９８は、セット４９０の様々なレジスタ及びＳＲＡＭ５６５を参照して命令をデコードし、次のフェーズにおける処理のためのオペランドを収集する。特に、以下のＡＬＵ６０２に提供するオペランドを決定する。命令デコーダ及びオペランドマルチプレクサ４９８の出力は、ＡＬＵオペランド５７９及び５８２、ＡＬＵ条件コードレジスタ５８０、及びこの実施形態では３２のキューを制御し得るキューチャネル及びコマンド５８７レジスタを含む第２レジスタセット４９６における多数のレジスタに格納される。セット４９６における幾つかのレジスタは、プログラム制御機構５９０、リテラルフィールド５８９、テスト選択５

８４、及びフラグ選択５８５を含む、デコーダ４９８によって実質的にデコードされることなく命令レジスタ５３５から直接ロードされる。第１フェーズ５００のファイルコンテキスト５２２のような他のレジスタは、第２フェーズ５６０のファイルコンテキスト５７７に常に格納されるが、マルチプレクサ５７２によって集められたオペランドとしても取り扱われる。スタックレジスタ５４４は、スタックレジスタ５９４に単にコピーされる。プログラムカウンタ５４０は、このフェーズにおいてインクリメントされ（５６８）、レジスタ５９２に格納される。同様に、オプションのデバッグアドレス５４８もインクリメントされ（５７０）、ロード制御機構５７５は、この時点で、各フェーズにおける誤り制御を可能にするためにパイプライン５０５から供給され得、その結果はデバッグアドレス５９８に格納される。

【００６２】

図１５Ｃは、ＡＬＵ及びキュー操作を含む、第３のマイクロプロセッサフェーズ６００を示す。ＡＬＵ６０２は、加算器、優先エンコーダ、及び他の標準的な論理機能を有する。ＡＬＵの結果は、レジスタであるＡＬＵ出力６１８、ＡＬＵ条件コード６２０、及び行先オペランド結果６２２に格納される。ファイルコンテキストレジスタ６１６、フラグ選択レジスタ６２６、及びリテラルフィールドレジスタ６３０は、単に前のフェーズ５６０からコピーされる。テストマルチプレクサ６０４は、条件ジャンプがジャンプを生ずるか否かを決定するために設けられ、その結果はテスト結果レジスタ６２４に格納される。テストマルチプレクサ６０４は、フェッチ制御機構５１１のような類似の判断とともに、第１フェーズ５００において実行され得る。スタック交換６０８は、スタック５９４からプログラムカウンタをフェッチすることにより、又はプログラムカウンタをそのスタックに入れることによってスタックを上限に移動させ、その結果はプログラム制御機構６３４、プログラムカウンタ６３８、及びスタック６４０レジスタに格納される。ＳＲＡＭアドレスは所望に応じてこのフェーズ６００においてインクリメントされ得る。別のデバッグアドレス６４２のための別のロード制御機構６１０は、このフェーズにおいても誤り制御を可能にするために、この時点でパイプライン５０５から強制的にわたされる。図面では一体に示されているＱＲＡＭ及びＱＡＬＵ６０６は、キューチャネル及びコマンドレジスタ５８７から読み出され、ＳＲＡＭに格納され、キューを再編成し、必要に応じてデータ及びポインタを加えたり取り除いたりして、データのキューを管理し、その結果をテストマルチプレクサ６０４、及びキューフラグ及びキューアドレスレジスタ６２８に送る。従って、ＱＲＡＭ及びＱＡＬＵ６０６は、３つのプロセッサのためのキューを管理する任務、ソフトウェアによってＣＰＵ上で順次実行される従来通りのタスクを想定しており、キューマネージャ６０６は、代わりに高速化した実質的に同時並行のハードウェアキューイングを提供する。

【００６３】

図１６は、キューマネージャ６０６によって管理される３２のハードウェアキューの２つを示しており、そのキューのそれぞれは、ＳＲＡＭヘッド、ＳＲＡＭテール、及びＤＲＡＭボディにおいて情報をキューに入れる能力を有しており、これによって各キューの拡張及び個別の編成が可能となる。従ってＦＩＦＯ７００は、ＳＲＡＭ記憶ユニット７０５、７０７、７０９、及び７１１を有し、それぞれが全部で３２個のバイトに対して８個のバイトを含んでいるが、これらのユニットの数及び処理能力は、実施形態によって変えることができる。同様に、ＦＩＦＯ７０２はＳＲＡＭ記憶ユニット７１３、７１５、７１７、及び７１９を有している。ＳＲＡＭユニット７０５及び７０７は、ＦＩＦＯ７００のヘッドであり、ユニット７０９及び７１１はＦＩＦＯのテールであり、ユニット７１３及び７１５はＦＩＦＯ７０２のヘッドであり、ユニット７１７及び７１９はそのＦＩＦＯのテールである。ＦＩＦＯ７００の情報は、矢印７２２で示すように、ヘッドユニット７０５又は７０７に書き込まれ得、矢印７２５で示すように、テールユニット７１１又は７０９から読み出され得る。しかし特定のエントリは、ヘッドユニット７０５又は７０７から読み書きの両方が行われ得、又はテールユニット７０９又は７４１から読み書きの両方が行われ得、データ転送及びラテンシーが最小限にされている。同様に、ＦＩＦＯ７０２のため

の情報は、一般的に、矢印７３３で示すように、ヘッドユニット７１３又は７１５に書き込まれ、矢印７３９で示すように、テールユニット７１７又は７１９から読み出されるが、それが書き込まれた同一のヘッド又はテールユニットから読み出されることもある。

【００６４】

ＳＲＡＭ　ＦＩＦＯ７００及び７０２は、共にＤＲＡＭ４６０に接続されており、このＤＲＡＭ４６０によって、これらのＦＩＦＯを無限に拡張しＳＲＡＭヘッド及びテールが一杯になった状況を取り扱うことが可能となる。例えば、３２個のキューのうちの、Ｑ（０）とラベル付けされた第１のキューは、矢印７２７で示すように、ＦＩＦＯ７００のヘッド又はテールにキューを入れる代わりにキューマネージャの指示の下で動作するＤＭＡユニットによってＤＲＡＭ４２０においてエントリをキューに入れることができる。ＤＲＡＭ４６０に格納されたエントリは、矢印７３０で示されるように、ＳＲＡＭユニット７０９に戻され、そのＦＩＦＯの長さ及び終了時間が延長される。ＳＲＡＭからＤＲＡＭへの転換は、一般的にＳＲＡＭが一杯になったときのために準備されている。ＤＲＡＭはより低速であり、ＤＭＡ転送によって更なるラテンシーが生ずるからである。従ってＱ（０）は、ＦＩＦＯ７００及びＤＲＡＭ４６０の両方におけるキューマネージャ６００によって格納されたエントリを含み得る。同様に、ＦＩＦＯ７０２に向かう情報は、例えばＱ（２７）に対応するものであり得、矢印７３５で示すように、ＤＭＡによってＤＲＡＭ４６０に転送され得る。より低速のＤＲＡＭ４６０にも関わらず、費用的に有効にキューを作成する能力は、初期化の間にユーザが決定でき、これにより所望に応じてキューのサイズを変更することが可能となる。ＤＲＡＭ４６０にキューイングされた情報は、矢印７３７に示すように、ＳＲＡＭユニット７１７に戻される。

【００６５】

３２のハードウェアキューのそれぞれのステータスは、従来通り図１７に示すように、４つの３２ビットのレジスタのセット７４０に維持され、そこからアクセスされる。各レジスタにおける特定のビットは特定のキューに対応する。そのレジスタは、Ｑ－Ｏｕｔ＿Ｒｅａｄｙ７４５、Ｑ－Ｉｎ＿Ｒｅａｄｙ７５０、Ｑ－Ｅｍｐｔｙ７５５、及びＱ－Ｆｕｌｌ７６０とラベル付けされている。特定のビットがＱ－Ｏｕｔ＿Ｒｅａｄｙレジスタ７５０にセットされた場合、そのビットに対応するキューは読み出される状態となった情報を含み、Ｑ－Ｉｎ＿Ｒｅａｄｙ７５２レジスタにおける同一のビットがセットされることは、そのキューが書込み可能な状態となったことを意味する。同様にＱ－Ｅｍｐｔｙレジスタ７５５における特定のビットを正値にセットすることは、そのビットが空になったことに対応するキューを意味し、Ｑ－Ｆｕｌｌレジスタ７６０の特定のビットが正値にセットされることは、そのビットが一杯になったことに対応するキューを意味する。従ってＱ－Ｏｕｔ＿Ｒｅａｄｙ７４５は、ビット（２７）７５２、ビット（２８）７５４、ビット（２９）７５６、及びビット（３０）を包含するビット（０）７４６～ビット（３１）７４８を含む。Ｑ－Ｉｎ＿Ｒｅａｄｙ７５２は、ビット（２７）７６６、ビット（２８）７６８、ビット（２９）７７０、及びビット（３０）７７２を含むビット（０）７６２～ビット（３１）７６４を含む。Ｑ－Ｅｍｐｔｙ７５５は、ビット（２７）７７８、ビット（２８）７８０、ビット（２９）７８２、及びビット（３０）７８４を含むビット（０）７７４～ビット（３１）７７６を含み、Ｑ－Ｆｕｌｌ７６０は、ビット（２７）７９０、ビット（２８）７９２、ビット（２９）７９４、ビット（３０）７９６を含むビット（０）７８６～ビット（３１）７８８を含む。

【００６６】

ＦＩＦＯ７００に対応するＱ（０）は、フリーのバッファキューであり、全ての利用可能なバッファであるアドレスのリストを保持する。このキューは、マイクロプロセッサ又は他のデバイスがフリーバッファアドレスを必要とするときにアドレス指定され、従って共通に感知できるＤＲＡＭ４６０を含む。従って、フリーバッファアドレスを必要とするデバイスは、Ｑ（０）をチェックし、そのアドレスを得る。ＦＩＦＯ７０２に対応するＱ（２７）は、受信バッファディスクリプタキューである。受信シーケンサによって受信されたフレームが処理された後、そのシーケンサはそのフレームのためのディスクリプタをＱ

（２７）に格納しようとする。このようなディスクリプタの位置がＳＲＡＭにおいて即座に利用可能である場合は、Ｑ－Ｉｎ＿Ｒｅａｄｙ７５０のビット（２７）７６６がセットされる。そうでない場合は、そのシーケンサはキューマネージャがＳＲＡＭからＤＲＡＭへＤＭＡ転送を開始させて、受信ディスクリプタを格納するための空間を空けるのを待たなければならない。

【００６７】

ＳＲＡＭとプロセッサとの間でのキューエントリの移動、受信及び送信シーケンサ間のキューエントリの移動、及びＳＲＡＭとＤＲＡＭ間のキューエントリの移動を管理するキューマネージャの動作は、図１８により詳細に示されている。キューを利用するリクエストとしては、プロセッサリクエスト８０２、送信シーケンサリクエスト８０４、及び受信シーケンサリクエスト８０６がある。そのキューのための他のリクエストは、ＤＲＡＭからＳＲＡＭへのリクエスト８０８及びＳＲＡＭからＤＲＡＭへのリクエスト８１０であり、これはキューのＤＲＡＭ及びＳＲＡＭヘッド又はテール間でデータをやりとりする際にキューマネージャの代わりに作用する。これらの様々なリクエストの何れが次のサイクルにおいてキューマネージャを用いるかの決定は、優先ロジックアービタ８１５によって取り扱われる。高い周波数での動作を可能にするため、キューマネージャはパイプライン型で、レジスタＡ８１８及びレジスタＢ８２０が一時的な記憶機構を提供するとともに、ステータスレジスタ８２２が次の更新までステータスを維持する。キューマネージャはＤＭＡ、受信及び送信シーケンサリクエストのために偶数番目のサイクルを予約し、プロセッサリクエストのために奇数番目のサイクルを予約する。二重ポートＱＲＡＭ８２５は、各キューに関連する変数を格納し、各キューのための変数としては、キューのＳＲＡＭ条件に対応するヘッド書き込みポインタ、ヘッド読み出しポインタ、テール書き込みポインタ、及びテール読み出しポインタと、キューのＤＲＡＭ条件及びキューのサイズに対応するボディ書込みポインタ及びボディ読み出しポインタとが挙げられる。

【００６８】

アービタ８１５が実行される次の処理を選択した後、ＱＲＡＭ８２５の変数がフェッチされ、ＱＡＬＵ８２８によって選択された処理に従って改変され、ＳＲＡＭ読み出しリクエスト８３０又はＳＲＡＭ書込みリクエスト８４０が生成され得る。その変数は更新され、更新されたステータスは、ステータスレジスタ８２２及びＱＲＡＭ８２５に格納される。ステータスはアービタ８１５にも供給されて、以前にリクエストされた処理が完了したことを示す信号を供給して、リクエストの重複を防止する。ステータスレジスタ８２２は、４つのキューレジスタＱ－Ｏｕｔ＿Ｒｅａｄｙ７４５、Ｑ－Ｉｎ＿Ｒｅａｄｙ７５０、Ｑ－Ｅｍｐｔｙ７５５、及びＱ－Ｆｕｌｌ７６０を更新して、アクセスされたキューの新たなステータスを反映するようにする。同様に更新されるのは、ＳＲＡＭアドレス８３３、ボディ書込みリクエスト８３５及びボディ読み出しリクエスト８３８であり、これらはＤＭＡを介してそのキューのためにＳＲＡＭヘッド及びテールとやりとりする。或いは、様々なプロセスが、Ｑ書き込みデータ８４４によって示されるように、キューへの書き込みを行おうとすることもあり得、そのプロセスはマルチプレクサ８４６によって選択され、ＳＲＡＭ書き込みリクエスト８４０にパイプライン処理される。ＳＲＡＭコントローラは、アクセスされたキューのテールの書き込み又はヘッドの読み出し及び確認応答を戻すことによって、読み出し及び書き込みリクエストをサービスする。このようにして、様々なキューが利用され、それらのステータスが更新される。

【００６９】

図１９Ａ乃至図１９Ｃは、ＩＮＩＣキャッシュメモリに維持するコンテキスト又はＣＣＢを選択するために用いられる最低使用頻度レジスタ９００を示す。この実施形態におけるＩＮＩＣは、所定の時間においてＳＲＡＭに最大１６のＣＣＢをキャッシュでき、新たなＣＣＢがキャッシュされるときには、古いものが破棄されなければならないことが多く、破棄されるＣＣＢは、通常、このレジスタ９００に従って最小使用頻度のＣＣＢとして選択されたものである。この実施形態では、最大２５６のＣＣＢのためのハッシュテーブルがＳＲＡＭに維持されており、一方最大２５６の全ＣＣＢがＤＲＡＭに保持されている。

最小使用頻度レジスタ９００は、Ｒ０－Ｒ１５とラベル付けされた１６個の４ビットブロックを有し、各ブロックはＳＲＡＭのキャッシュユニットに対応する。初期化の際、そのブロックは０－１５に番号付けされ、番号０は、最小使用頻度（ＬＲＵ）キャッシュユニットを表すブロックに格納され、番号１５は、最大使用頻度（ＭＲＵ）キャッシュユニットを表すブロックに格納される。図１９Ａは、ＬＲＵブロックＲ０が番号９を保持しており、ＭＲＵブロックＲ１５が番号６を保持している、任意の時点におけるレジスタ９００を示す。

【００７０】

現在ＳＲＡＭに保持されているものとは異なるＣＣＢがキャッシュされたとき、図１９Ａでは、番号９を保持しているＬＲＵブロックＲ０が読み出され、新たなＣＣＢが番号９に対応するＳＲＡＭキャッシュユニットに格納される。番号９に対応する新たなＣＣＢが現時点で最も使用頻度の高いＣＣＢであることから、番号９は、図１９Ｂに示すように、ＭＲＵブロックに格納される。他の番号は全て1個左のレジスタブロックにシフトされ、ＬＲＵブロックに番号１が残る。以前に番号９に対応するＳＲＡＭユニットにキャッシュされたＣＣＢは、よりゆっくりとではあるが、よりコストパフォーマンスの良い形でＤＲＡＭに移されている。

【００７１】

図１９Ｃは、次に使用されるＣＣＢが常にＳＲＡＭにキャッシュされたときの結果を示す。この例では、ＣＣＢが番号１０に対応するＳＲＡＭユニットにキャッシュされており、そのＣＣＢを使用した後、番号１０にＭＰＵブロックに格納される。番号１０以外の最近使用された番号のみ（レジスタブロックＲ９－Ｒ１５）が左側にシフトされ、ＬＲＵブロックには番号１が残る。このようにして、ＩＮＩＣはＳＲＡＭキャッシュに最もアクティブなＣＣＢを維持する。

【００７２】

場合によっては、使用されているＣＣＢが、限定されたキャッシュメモリに保持することが望ましくないものであることがある。例えば、終了したことが判っているコンテキストのためのＣＣＢは、キャッシュせずに、他のキャッシュされたＣＣＢがＳＲＡＭに長く残るようにすることが好ましい。この場合、デキャッシュ可能なＣＣＢを保持するキャッシュユニットを表す番号が、ＭＲＵブロックＲ１５でないＭＲＵブロックＲ０に格納され、従ってデキャッシュ可能なＣＣＢがＬＲＵブロックＲ０に保持された番号に対応するＳＲＡＭユニットにキャッシュされる新たなＣＣＢが使用されるとすぐに置きかえられる。図１９Ｄは、番号８（図１９Ｃにおけるブロック９にあったもの）が、使用され次に終了されるＣＣＢに対応する場合を示す。この場合では、番号８がブロックＲ６から取り除かれ、ＬＲＵブロックＲ０に格納される。次いでブロックＲ９の左側（Ｒ１～Ｒ８）に以前に格納された全ての番号は、1個右側のブロックにシフトされる。

【００７３】

図２０は、最低使用頻度レジスタ９００を操作するために用いられる論理ユニットをいくつか示す。１６個の３又は４入力マルチプレクサ９１０のアレイ（明示のため、そのうちマルチプレクサＭＵＸ０、ＭＵＸ７、ＭＵＸ８、ＭＵＸ９、及びＭＵＸ１５のみが示されている）は、最低使用頻度レジスタ９００の対応する１６個のブロックに供給される出力を有する。例えば、ＭＵＸ０の出力は、ブロックＲ０に格納され、ＭＵＸ７の出力はブロックＲ７に格納される等である。レジスタブロックのそれぞれの値は、対応するマルチプレクサのための入力に接続されており、かつブロック番号のシフトの際に使用するために両隣のマルチプレクサの入力にも接続されている。例えば、Ｒ８に格納された番号は、ＭＵＸ７、ＭＵＸ８、及びＭＵＸ９の入力に供給される。ＭＵＸ０及びＭＵＸ１５のそれぞれは、ただ１つの隣接ブロックしか有しておらず、それらのマルチプレクサのための別の入力がＬＲＵ及びＭＲＵブロックの選択のためにそれぞれ用いられる。ＭＵＸ１５は、４入力マルチプレクサとして図示されており、入力９１５はＲ０に格納された番号を供給する。

【００７４】

１６個の比較器９２０のアレイのそれぞれが、最低使用頻度レジスタ９００の対応するブロックに格納された値を受け取る。各比較器は、ライン９３５を介してプロセッサ４７０からの信号を受け取り、従ってプロセッサ４７０によって送られた一致する番号を有するレジスタブロックが、論理回路９３０に真の値を出力し、一方他の１５個の比較器は偽の値を出力する。論理回路９３０は、マルチプレクサへの入力を選択するためにマルチプレクサのそれぞれに接続された一対の選択ラインを制御し、従ってレジスタブロック番号のシフトを制御する。従って選択ライン９３９はＭＵＸ０を制御し、選択ライン９４４はＭＵＸ７を制御し、選択ライン９４９はＭＵＸ８を制御し、選択ライン９５４はＭＵＸ９を制御し、選択ライン９５９はＭＵＸ１５を制御する。

【００７５】

ＣＣＢが使用されるとき、プロセッサ４７０は、そのＣＣＢが１６個のキャッシュユニットの１つを現在保持されているＣＣＢに一致するか否かを調べるためのチェックを行う。一致が見出された場合は、そのプロセッサは、ライン９３５を通してそのキャッシュユニットに対応するブロック番号、例えば番号１２を有する信号を送る。比較器９２０は、そのライン９３５からの信号とブロック番号とを比較し、比較器Ｃ８はその信号に一致するブロックＲ８に真の出力を供給し、他の全ての比較器は偽の値を出力する。論理回路９３０はプロセッサ４７０からの制御の下で、選択ライン９５９を用いてＭＵＸ１５のためライン９３５からの入ロクを選択し、ＭＲＵブロックＲ１５に番号１２を格納する。論理回路９３０も、ＭＵＸ１５を除いて、ＭＵＸ８及びより高位のマルチプレクサのための対になった選択ラインを通して信号を送り、各マルチプレクサＭＵＸ８及びそれより上位のマルチプレクサへの入力として、１個右側のレジスタブロック（Ｒ９－Ｒ１５）に格納された値を選択することにより、それらの出力を１ブロック左にシフトさせる。ＭＵＸ８の左側のマルチプレクサの出力は一定に選択される。

【００７６】

一方プロセッサ４７０が１６個のキャッシュユニットのなかでＣＣＢについて一致を見出せなかった場合は、プロセッサは、ライン９６６を通してＬＲＵブロックＲ６から読み出しを行い、ＬＲＵブロックに対応するキャッシュを特定し、そのキャッシュに格納されたデータをＤＲＡＭに書き込む。この場合では番号３であるＲ０に格納された番号は、ＭＲＵブロックＲ５における記憶機構のためにＭＵＸ１５への入力９１５としてライン９５９を選択することによって選び出される。他の１５個のマルチプレクサは、それらの各レジスタブロックにそれぞれの右隣のレジスタブロックに格納されていた番号を出力する。

【００７７】

プロセッサが、使用後にキャッシュからＣＣＢを取り除こうとしている状況では、ＭＲＵブロックＲ５ではなく、ＬＲＵブロックＲ０がそのＣＣＢを維持するキャッシュユニットに対応する番号の置き換えのために選択される。ＳＲＡＭから除去するためにＬＲＵブロックＲ０に置かれるＣＣＢに対応する番号（例えばブロックＲ９に保持されている番号１）は、ライン９３５を通してプロセッサ４７０に送られ、それは比較器Ｃ９によって一致を取られる。プロセッサは、論理回路９３０にＭＵＸ０への入力９３５としてライン９３９を選択することにより、番号１をＲ０に入力するように命令する。ＭＵＸ９への選択ライン９５４は、入力としてレジスタブロックＲ８に保持された番号を選択し、従ってＲ８からの番号はＲ９に格納される。Ｒ０～Ｒ９の間の他のレジスタブロックによって保持された番号は、同様に右側にシフトされ、Ｒ９の右側のレジスタブロックにおける番号は一定に維持される。これによって、多くのサイクルにわたって使用されなくなったＣＣＢが限りのあるキャッシュメモリに維持されることがなくなるとともに、それらの特定する番号がＭＲＵからＬＲＵブロックにレジスタブロックを通して動くことになる。

【００７８】

既に述べたように、上述のデータ通信の処理のための装置及びシステムは、大きなコネクションベースのメッセージを処理するために必要な時間を劇的に短縮させる。プロトコル処理速度は、専用に設計されたプロトコル処理ハードウェアによって汎用ＣＰＵが従来のプロトコルソフトウェアを実行する場合と比較して非常に高速化され、かつホストのＣＰ

Ｕへの割り込みも実質的に低減する。これらの利点は、インテリジェントネットワークインタフェースカード（ＩＮＩＣ）を追加することによって既存のホストにもたらすこともできるが、或いはプロトコル処理ハードウェアをＣＰＵに組み込んでもよい。何れの場合も、プロトコル処理ハードウェア及びＣＰＵは、インテリジェントに与えられたメッセージを処理するデバイスを決定し、メッセージの条件に基いてその処理の割り当てを変更することができる。

【図面の簡単な説明】

【図１】　ネットワーク通信を高速化するための通信処理装置を有するホストコンピュータを備えた、本発明の装置の平面図。

【図２】　高速パス、低速パス、及び高速パスと低速パスとの間のコネクションコンテキストの転送を含む、ネットワーク通信の処理における図１のホストの情報の流れを示す図。

【図３】　本発明によるメッセージ受信の流れ図。

【図４】　Ａ図は、低速パスによって処理される初めのメッセージパケットを受信した、図１のホストにおける情報の流れを示す図、Ｂ図は、高速パスによって処理される初めのメッセージパケットを受信した、図１のホストにおける情報の流れを示す図、Ｃ図は、高速パスによって処理される順次式のメッセージパケットを受信した、図４Ｂのホストにおける情報の流れを示す図、Ｄ図は、低速パスに戻すための処理を生じさせる誤りを有するメッセージパケットを受信した、図４Ｃのホストにおける情報の流れを示す図。

【図５】　高速パス又は低速パスの何れかによってメッセージを転送する、図１のホストにおける情報の流れを示す図。

【図６】　ＴＣＰ／ＩＰ処理スタックを有するクライアントに関連するインテリジェントネットワークインタフェースカード（ＩＮＩＣ）の第１の実施形態における情報の流れを示す図。

【図７】　パケット制御シーケンサ、フライバイシーケンサを含む、図６に示すＩＮＩＣの実施形態のためのハードウェアロジックを示す図。

【図８】　ＩＮＩＣによって受信されたときにヘッダバイトを解析する、図７のフライバイシーケンサを示す図。

【図９】　ＴＣＰ／ＩＰ処理スタックを有するサーバに関連するＩＮＩＣの第２の実施形態における情報の流れを示す図。

【図１０】　高速パスのための通信制御ブロックを生成し制御するための、図９にインストールされたコマンドドライバを示す図。

【図１１】　ＮｅｔＢｉｏｓ通信のために構成された図１０のコマンドドライバ及びＴＣＰ／ＩＰスタックを示す図。

【図１２】　図６のクライアントと図９のサーバとの間の通信の交換を示す図。

【図１３】　図９のＩＮＩＣに含められたハードウェア機構を示す図。

【図１４】　各フェーズに１個のプロセッサがある３つのフェーズを含む、図１３のＩＮＩＣに含められた３個１組のパイプライン型マイクロプロセッサを示す図。

【図１５】　Ａ図は図１４のパイプライン型マイクロプロセッサの第１フェーズを示す図、Ｂ図は図１４のパイプライン型マイクロプロセッサの第２フェーズを示す図、Ｃ図は図１４のパイプライン型マイクロプロセッサの第３フェーズを示す図。

【図１６】　図１４のマイクロプロセッサと相互作用し、かつＳＲＡＭ及びＤＲＡＭを備えた複数のキュー記憶ユニットを示す図。

【図１７】　図１６のキュー記憶ユニットのための、１組のステータスレジスタを示す図。

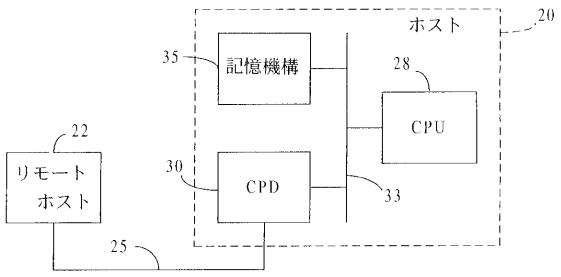
【図１８】　図１６及び図１７のキュー記憶ユニット及びステータスレジスタと相互作用する、キューマネージャを示す図。

【図１９】　Ａ図はキャッシュメモリの割り当てのために用いられる最低使用頻度レジスタの様々な動作段階の１つを示す図、Ｂ図はキャッシュメモリの割り当てのために用いられる最低使用頻度レジスタの様々な動作段階の１つを示す図、Ｃ図はキャッシュメモリの

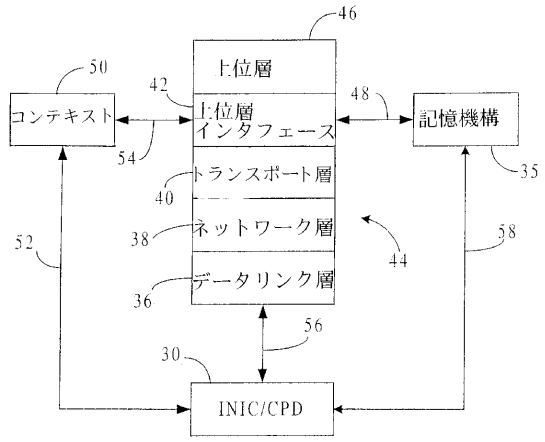
割り当てのために用いられる最低使用頻度レジスタの様々な動作段階の１つを示す図、Ｄ図はキャッシュメモリの割り当てのために用いられる最低使用頻度レジスタの様々な動作段階の１つを示す図。

【図２０】　図１９Ａ乃至図１９Ｄの最低使用頻度レジスタを操作するための用いられるデバイスを示す図。

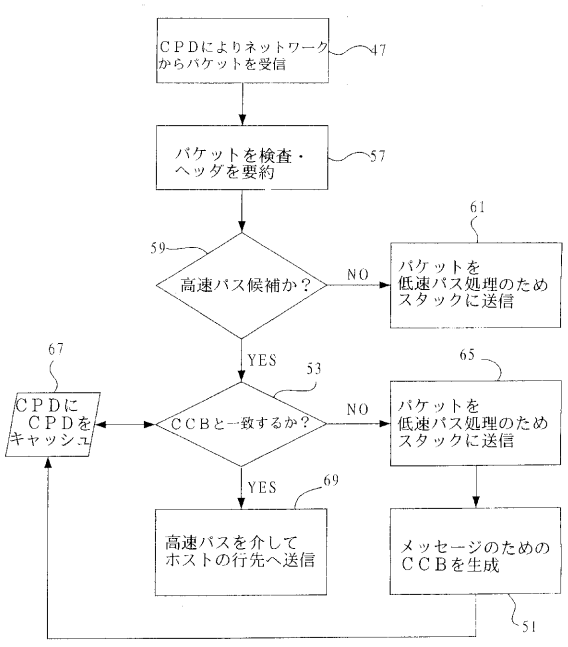
【図１】



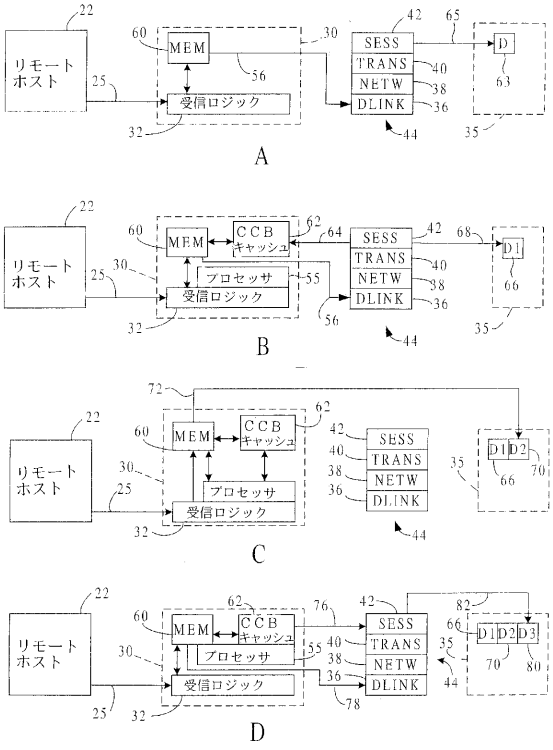
【図２】



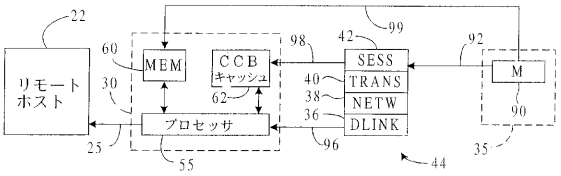
【図３】



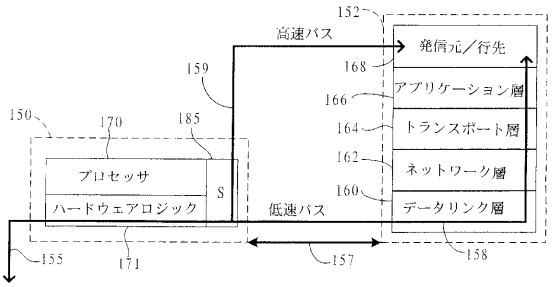
【図４】


リモートホスト
MEM
受信ロジック
SESS
TRANS
NETW
DLINK
D
A
CCBキャッシュ
プロセッサ
D1
B
D1D2
C
D1D2D3
D

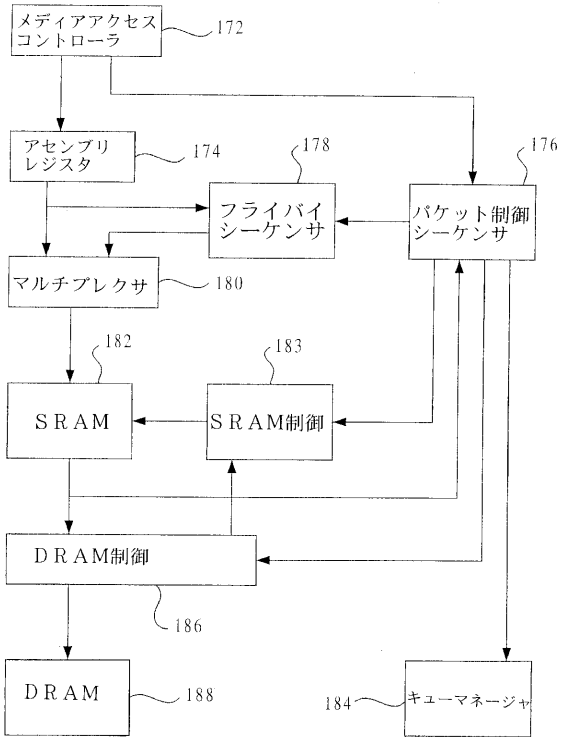
【図５】


リモートホスト
MEM
CCBキャッシュ
プロセッサ
SESS
TRANS
NETW
DLINK
M

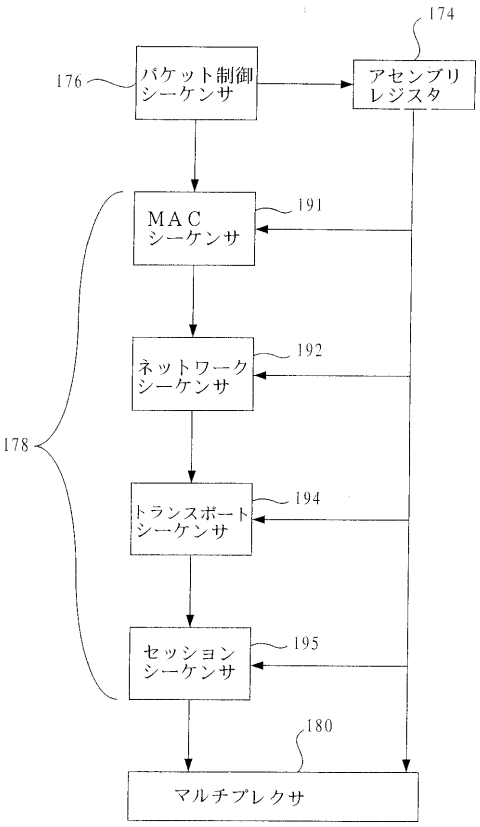
【図６】


高速バス
発信元／行先
アプリケーション層
トランスポート層
ネットワーク層
データリンク層
プロセッサ
ハードウェアロジック
S
低速バス

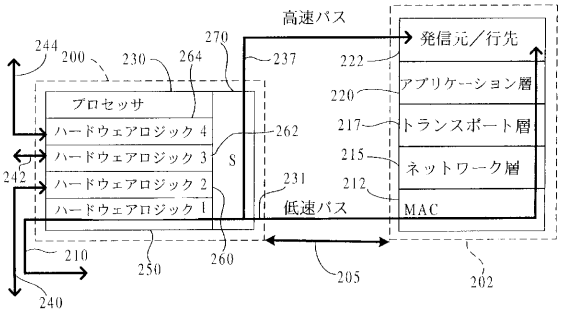
【図７】


メディアアクセスコントローラ
172
アセンブリレジスタ
174
フライバイシーケンサ
178
パケット制御シーケンサ
176
マルチプレクサ
180
SRAM
182
SRAM制御
183
DRAM制御
186
DRAM
188
キューマネージャ
184

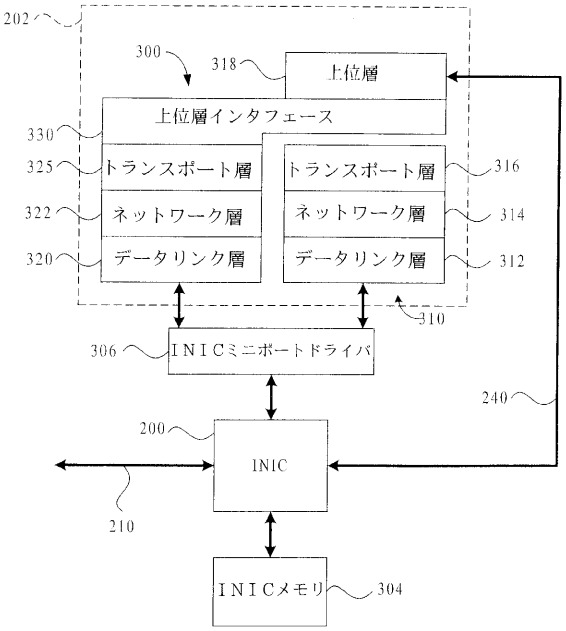
【図８】


パケット制御シーケンサ
176
アセンブリレジスタ
174
MACシーケンサ
191
ネットワークシーケンサ
192
トランスポートシーケンサ
194
セッションシーケンサ
195
178
マルチプレクサ
180

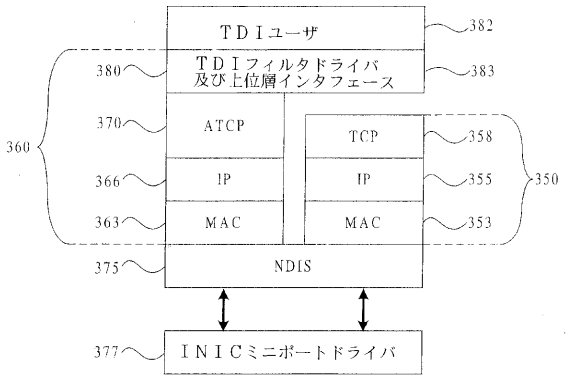
【図９】



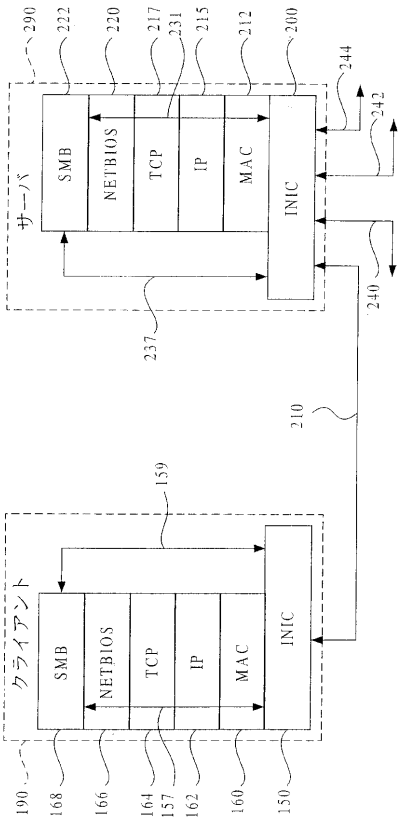
【図１０】



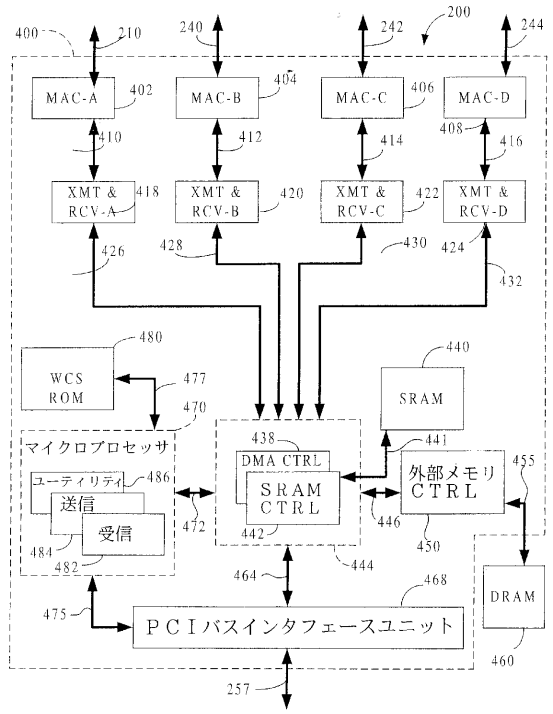
【図１１】



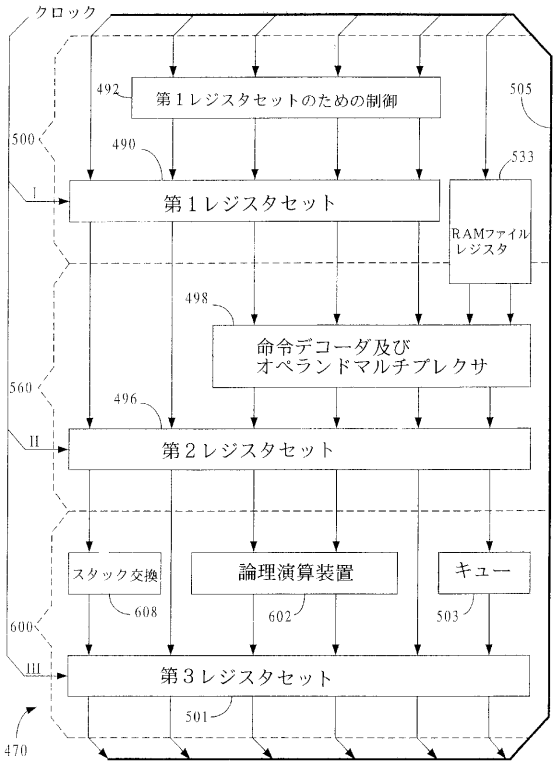
【図１２】



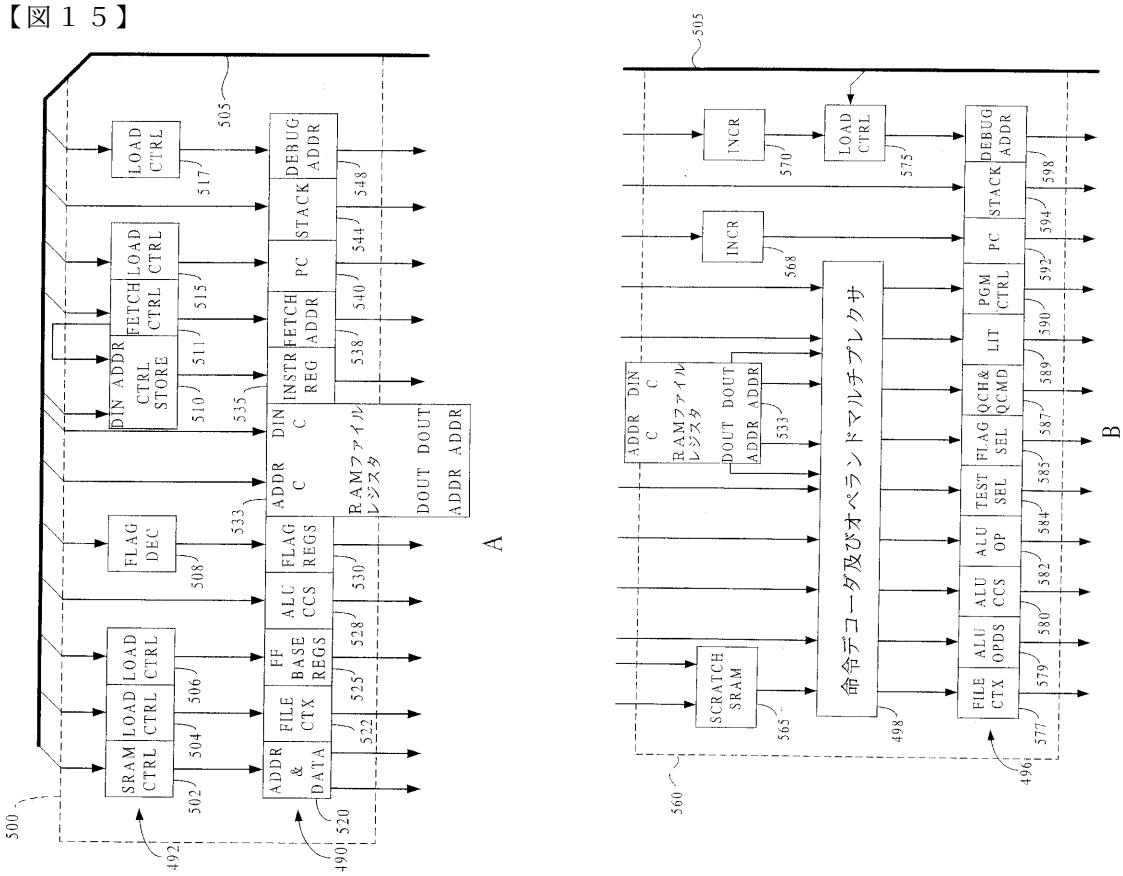
【図１３】

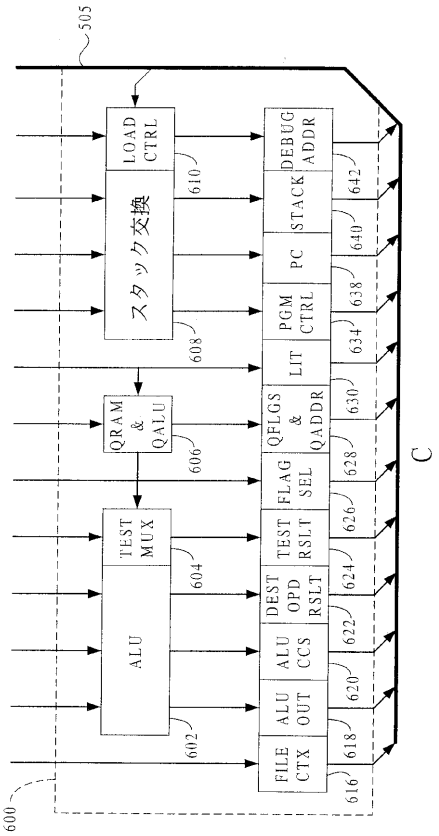

400
210
240
242
200
244
MAC-A
402
MAC-B
404
MAC-C
406
MAC-D
408
410
412
414
416
XMT & RCV-A
418
XMT & RCV-B
420
XMT & RCV-C
422
XMT & RCV-D
424
426
428
430
432
480
WCS ROM
477
470
マイクロプロセッサ
ユーティリティ
486
送信
受信
484
482
472
438
DMA CTRL
SRAM CTRL
442
441
SRAM
440
外部メモリ CTRL
446
455
450
444
464
468
DRAM
460
475
ＰＣＩバスインタフェースユニット
257

【図１４】

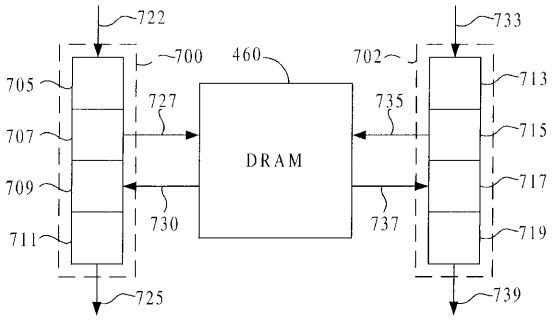

クロック
492
第１レジスタセットのための制御
505
500
490
533
I
第１レジスタセット
RAMファイルレジスタ
498
命令デコーダ及びオペランドマルチプレクサ
560
496
II
第２レジスタセット
スタック交換
608
論理演算装置
602
キュー
503
600
III
第３レジスタセット
501
470

【図１５】

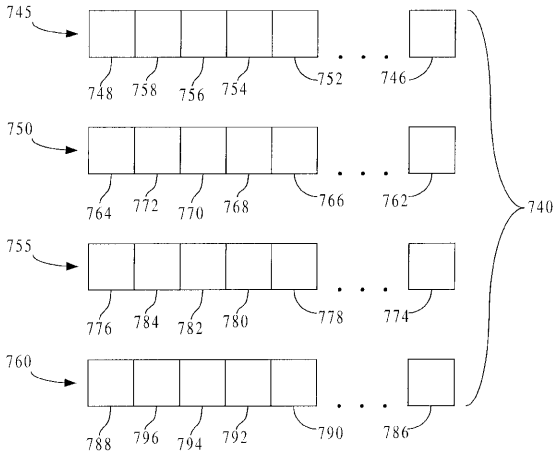

505
LOAD CTRL
517
DEBUG ADDR
548
STACK
544
PC
FETCH CTRL
515
FETCH ADDR
540
LOAD CTRL
DIN ADDR CTRL STORE
511
INSTR REG
538
510
535
ADDR C
RAMファイルレジスタ
DOUT DOUT
ADDR ADDR
533
FLAG DEC
508
FLAG REGS
530
ALU CCS
528
LOAD CTRL
FF BASE REGS
525
506
LOAD CTRL
FILE CTX
522
504
SRAM LOAD CTRL
ADDR & DATA
520
502
500
492
490
A
INCR
570
LOAD CTRL
575
INCR
568
ADDR DIN C
RAMファイルレジスタ
DOUT DOUT
ADDR ADDR
533
命令デコーダ及びオペランドマルチプレクサ
SCRATCH SRAM
565
498
DEBUG ADDR
598
STACK
594
PC
592
PGM CTRL
590
LIT
589
QCH& QCMD
587
FLAG SEL
585
TEST SEL
584
ALU OP
582
ALU CCS
580
ALU OPDS
579
FILE CTX
577
496
560
B



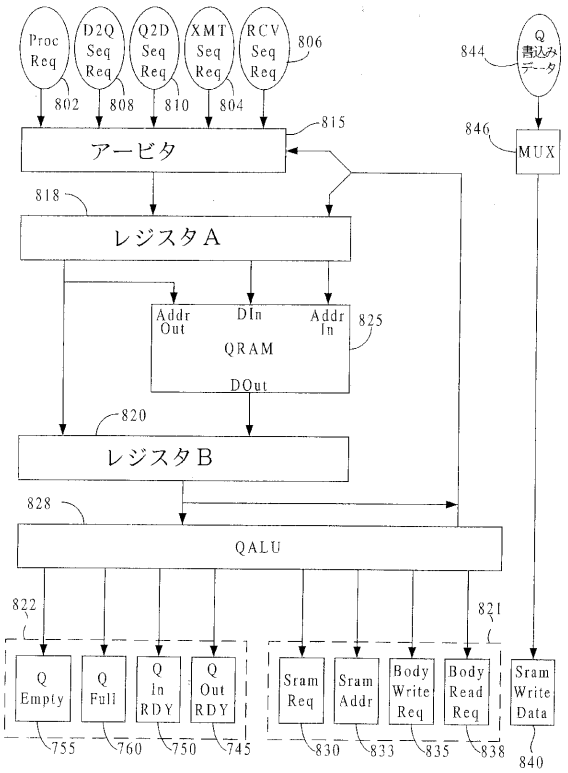
【図１６】



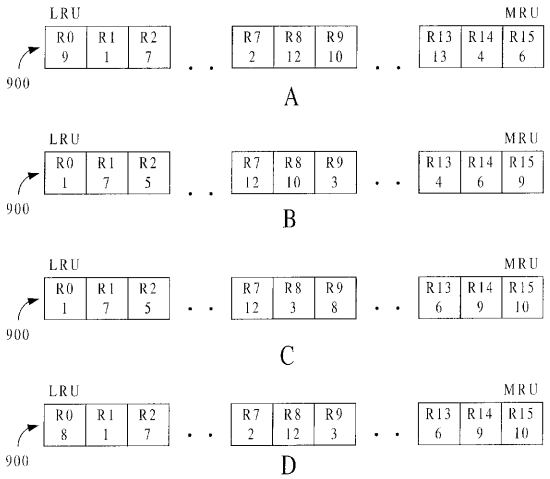
【図１７】



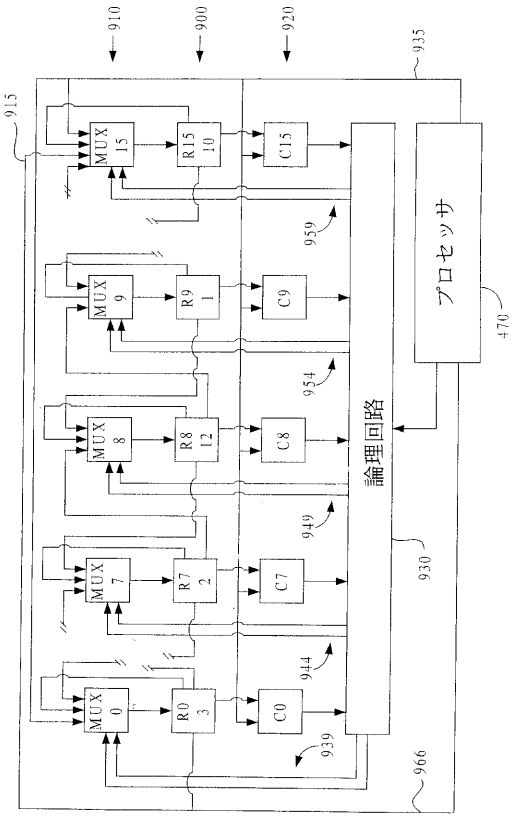
【図１８】



【図１９】



【図２０】


910
900
920
935
915
MUX 15
R15 10
C15
959
MUX 9
R9 1
C9
954
MUX 8
R8 12
C8
949
MUX 7
R7 2
C7
944
MUX 0
R0 3
C0
939
論理回路
930
プロセッサ
470
966

フロントページの続き

(72)発明者　クラフト，ピーター，ケイ
アメリカ合衆国，カリフォルニア州・９４１１４，サンフランシスコ，ヘンリー・ストリート・１５６
(72)発明者　ヒギン，デヴィッド，エー
アメリカ合衆国，カリフォルニア州・９５０７０，サラトガ，ロス・アラモス・ドライブ・１７８８０
(72)発明者　フィルブリック，クライヴ，エム
アメリカ合衆国，カリフォルニア州・９５１２５，サンノゼ，ロイコット・ウェイ・１１７０
(72)発明者　スター，ダリル
アメリカ合衆国，カリフォルニア州・９５０３５，ミルピタス，フォルソム・コート・４４６

審査官　小曳　満昭

(56)参考文献　特開平０２－２３８５４４（ＪＰ，Ａ）
特開平０９－１２８３１４（ＪＰ，Ａ）
特開平０３－２５０９４６（ＪＰ，Ａ）
特開平０２－１３７５５５（ＪＰ，Ａ）
特開昭６２－０３８０６０（ＪＰ，Ａ）
特開平０１－２５６２４８（ＪＰ，Ａ）
向井伸一郎，ＬＡＮカード周辺ハードウェア動作概要と機能評価，インターフェース，ＣＱ出版株式会社，１９９２年　４月　１日，第18巻，第4号，p.223～237
陣崎　明　外３名，大規模広域並列分散システムの実現を目指す超高速インターネットの構想，電子情報通信学会技術研究報告ＣＰＳＹ９７－６１，社団法人電子情報通信学会，１９９７年８月２０日，第97巻，第226号，p.75～82

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 13/00、
H04L 12/00-12/26、12/50-13/18、
29/00-29/12