(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4002336号
(P4002336)
(45)発行日　平成19年10月31日(2007.10.31)　(24)登録日　平成19年8月24日(2007.8.24)

(51)Int.Cl.		F I		
H04L 12/44	(2006.01)	H04L	12/44	300
H04L 12/66	(2006.01)	H04L	12/66	Z
G06F 13/00	(2006.01)	G06F	13/00	353B

請求項の数 4　(全 135 頁)

(21)出願番号　特願平9-361536
(22)出願日　平成9年12月26日(1997.12.26)
(65)公開番号　特開平10-210070
(43)公開日　平成10年8月7日(1998.8.7)
審査請求日　平成16年12月24日(2004.12.24)
(31)優先権主張番号　774602
(32)優先日　平成8年12月30日(1996.12.30)
(33)優先権主張国　米国(US)

(73)特許権者　591030868
コンパック・コンピューター・コーポレーション
COMPAQ COMPUTER CORPORATION
アメリカ合衆国テキサス州７７０７０，ヒューストン，ステイト・ハイウェイ　２４９，２０５５５
２０５５５ State Highway 249, Houston, Texas 77070, United States of America
(74)代理人　100089705
弁理士　社本　一夫

最終頁に続く

(54)【発明の名称】ネットワーク・スイッチのためのマルチポート・ポーリング・システム

(57)【特許請求の範囲】

【請求項１】

ネットワーク・スイッチであって、

ネットワーク・デバイスとの間でデータを受信および送信するように構成された複数のネットワーク・ポートと、

前記複数のネットワーク・ポートと結合され、前記複数のネットワーク・ポート間のデータの流れを制御するスイッチ・マネジャと、

個々のポートでのデータ使用可能度を判定するように動作するロジックと

を備え、

前記複数のネットワーク・ポートの各ネットワーク・ポートがポート状態ロジックと関連し、該ポート状態ロジックは、対応するネットワーク・ポートがネットワーク・デバイスからデータを受信したかについて、および、対応するネットワーク・ポートがネットワーク・デバイスへ送信するデータを受信するための使用可能なスペースを有するかについてを示す状態信号を供給するように構成されることと、

前記状態信号を受信する前記複数のネットワーク・ポートの各ネットワーク・ポートの前記ポート状態ロジックを周期的にポーリングするように構成されたポーリング・ロジックと、

前記複数のネットワーク・ポートの各ネットワーク・ポートに対する前記状態信号を示す値を記憶するメモリと

を特徴とするネットワーク・スイッチ。

【請求項２】
請求項１に記載のネットワーク・スイッチであって、
前記ポーリング・ロジックは、問い合わせ信号を周期的にアサートするため、および前記複数のネットワーク・ポートの各ネットワーク・ポートから送信状態信号及び受信状態信号を受信するためのロジックを含み、
前記複数のネットワーク・ポートの各ネットワーク・ポートの前記ポート状態ロジックは、前記問い合わせ信号を受信し、対応するネットワーク・ポートが前記スイッチ・マネジャからデータを受信するための場所を有するかを示す送信状態信号をアサートするため、および、前記対応するネットワーク・ポートがネットワーク・デバイスからデータを受信したかを示す受信状態信号をアサートするためのロジックを含む、
ことを更に特徴とするネットワーク・スイッチ。
【請求項３】
請求項２に記載のネットワーク・スイッチであって、
複数のマルチポート・デバイスを特徴とし、該複数のマルチポート・デバイスの各々は、前記複数のネットワーク・ポートのサブセットを実施するように動作し、且つその各々は、前記問い合わせ信号を受信するため、および、前記複数のマルチポート・デバイスの各マルチポート・デバイスの前記複数のネットワーク・ポートの前記サブセットの各ネットワーク・ポートの状態を示す、対応する多重化された送信状態信号及び対応する多重化された受信状態信号を供給するための、ポート状態ロジックを含むことと、
前記ポーリング・ロジックは、前記複数のネットワーク・ポートの各ネットワーク・ポートの状態を判定するために、前記複数のマルチポート・デバイスから、複数の多重化された送信状態信号及び複数の多重化された受信状態信号を受信するように構成される、
ことを更に特徴とするネットワーク・スイッチ。
【請求項４】
請求項３に記載のネットワーク・スイッチであって、前記複数のマルチポート・デバイスの各マルチポート・デバイスは、前記複数のネットワーク・ポートの４つまでのネットワーク・ポートを受け入れるカッド・カスケード・マルチポート・デバイスを備える、ネットワーク・スイッチ。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、ネットワーク用デバイスの分野に関連し、特に、ネットワーク・スイッチのためのマルチポート・ポーリング・システムに関連する。
【０００２】
【従来の技術】
ファイル及びリソースを共用するための、又は２つ又はそれ以上のコンピュータ間の通信を可能にするための多くの異なる種類のネットワーク及びネットワーク・システムがある。ネットワークは多種の特徴及び機能、例えば、メッセージ容量、ノードが分散される範囲、ノード又はコンピュータのタイプ、ノードの関係、トポロジー的又は論理的及び／又は物理的なレイアウト、ケーブル型及びデータ・パケット・フォーマットを基にしたアーキテクチャ又は構造、アクセス可能性その他を基にして分類できる。例えば、ネットワークのレンジ（範囲）は、例えば、建物のオフィス又は床の内部のローカル・エリア・ネットワーク（ＬＡＮ）や、大学の構内や町や州を横切って広がる広域ネットワーク（ＷＡＮ）や、国境などを横切って広がるグローバル・エリア・ネットワーク（ＧＡＮ）のような、ノードが分散される距離を言及する。
【０００３】
ネットワークの構造は、一般に、用いられるケーブル又は媒体（メディア）及びメディアのアクセス、及びメディアを通して送信されるデータのパケットの構造を言及する。多種の構造が一般的であり、それらは、１０メガビット／秒（Ｍｂｐｓ）（例えば、１０Ｂａｓｅ－Ｔ、１０Ｂａｓｅ－Ｆ）で動作するための同軸のねじられたペアの又は光ファイバ

・ケーブルを用いるイーサネット又は１００Ｍｂｐｓ（例えば、１００Ｂａｓｅ－Ｔ、１００Ｂａｓｅ－ＦＸ）で動作する高速イーサネットを含む。ＡＲＣｎｅｔ（Attached Resource Computer Network、アタッチト・リソース・コンピュータ・ネットワーク）は、２．５Ｍｂｐｓで動作する同軸のねじられたペアの又は光ファイバ・ケーブルを用いる比較的安価なネットワーク構造である。トークン・リング（Token Ring）・トポロジーは、１～１６Ｍｂｐｓの間で動作するために特別のＩＢＭのケーブル又は光ファイバ・ケーブルを用いる。もちろん、多くの他の種類のネットワークが知られていて、使用可能である。

【０００４】

一般に各ネットワークは２つ以上のコンピュータを含み、それらはしばしばノード又はステーションと呼ばれ、それらは、ノード間でデータをリレーするため、送信するため、反復するため、変換するため、フィルタリングするためその他のための多種の他のネットワーク・デバイス及び選択されたメディアを通じて結合される。用語「ネットワーク・デバイス」は、一般に、コンピュータ、それらのネットワーク・インターフェース・カード（ＮＩＣ）、及び、幾つかの例として、リピータ、ブリッジ、スイッチ、ルータ、ブルータなどのネットワーク上の多種の他のデバイスを言及する。所与の通信プロトコルに従って動作するネットワークは、一つ以上のリピータ（中継器）、ブリッジ又はスイッチを用いることによって広げることができる。リピータは、物理層で機能し、受信した各パケットを各ポートに再送信するハードウェアデバイスである。ブリッジは、ＯＳＩ参照モデルのデータ・リンク層で動作し、各ネットワーク・セグメント上のパケットをフィルタリングして不要なパケットの伝搬量を減少することによって効率を向上させる。

【０００５】

ネットワーク・スイッチはマルチポート・ブリッジと機能的に類似であるが、より効率的であり、ネットワーク中でネットワーク・トラフィックを導くための、幾つかの類似のネットワークに結合するための複数のポートを含む。ネットワーク・スイッチは、バスをまたいだポートに結合され、ポート間のデータフロー（データの流れ）を制御するためのスイッチング・マトリクス又はそれと同様のものを含む。スイッチング・マトリクスは、ポートがネットワーク・デバイスからデータを受信したときに、及びポートが送信のためにデータを受信するために使用可能であるときに、何らかの判定をしなければならない。

【０００６】

【発明が解決しようとする課題】

ネットワーク・スイッチのポートの受信状態及び送信状態を判定するための効率的なシステムを提供することが望まれる。

【０００７】

【課題を解決するための手段】

本発明に従うネットワーク・スイッチのためのマルチポート・ポーリング・システムは、複数のネットワーク・デバイス中の通信を可能にし、ネットワーク・デバイスからデータを受信するため及びネットワーク・デバイスにデータを送信するための複数のネットワーク・ポートを含む。各ポートは、対応するポートがネットワーク・デバイスからデータを受信したかを、及び対応するポートがネットワーク・デバイスに送信するためにデータを受信するための使用可能なスペースを有するかを示すポート状態信号を供給するためのポート状態ロジックを含む。ネットワーク・スイッチは、ポート間のデータの流れを制御するためのスイッチ・マネジャを更に含む。スイッチ・マネジャは、状態信号を受信するために各ポートのポート状態ロジックを周期的にポーリング（ポール）するためのポーリング・ロジックと、各ポートに対する状態信号を示す値を記憶するためのメモリとを含む。このようにして、すべてのポートがポーリングされ、各ポートの受信及び送信状態がメモリに維持される。これによって、仲裁（アービトレーション）及びコントロール・ロジックを容易にする。このロジックは、メモリを見直して、ソース・ポートからいつデータを受信するか、及び送信のために受信したデータをいつ１つ又はそれ以上の宛て先ポートに供給するかを決定する。

【０００８】

ポーリング・ロジックは、好適には、問い合わせ（query）信号を周期的にアサートするため、及びネットワーク・ポートの各々から送信状態信号及び受信状態信号を受信するためのロジックを含む。更に、各ポートのポート状態ロジックは、問い合わせ信号を受信するため、ポートがスイッチ・マネジャからデータを受信するための場所を有するかを示すために送信状態信号をアサートするための、及び、ポートがネットワーク・デバイスからの受信したデータを有するかを示すために受信状態信号をアサートするためのロジックを含む。このように、ポーリング・ロジックは、問い合わせ信号を周期的にアサートし、複数のポートを同時にポーリングを行うために複数の送信状態信号及び受信状態信号を受信する。

【０００９】

例示する特定的な実施例において、ネットワーク・スイッチは幾つかのマルチポート・デバイスを含み、その各々はネットワーク・ポートの２又はそれ以上を実施するためのものであり、各々はポート状態ロジックを含む。各マルチポート・デバイスに対するポート状態ロジックは、問い合わせ信号を受信し、そのポートの各々の状態を示す対応する多重化された送信状態信号及び対応する多重化された受信状態信号を供給する。即ち、ポーリング・ロジックは、マルチポート・デバイスから複数の多重化された送信状態信号及び複数の多重化された受信状態信号を受信する。好適には、各マルチポート・デバイスは、４ポートまで含むことができるカッド・カスケード（quad cascade）・マルチポート・デバイスである。各マルチポート・デバイスのためのポート状態ロジックは、そのポート間で集中化しても又は分散しても何れでもよいことに留意されたい。

【００１０】

ネットワーク・スイッチのメモリは、何れのポートがネットワーク・デバイスへ送信するためのデータを受信するためのスペースを有することを示したかを示すためのプログラマブル送信リストと、何れのポートがネットワーク・デバイスからデータを受信したことを示したかを示すためのプログラマブル受信リストとを記憶する。ポーリング・ロジックは、状態信号を監視するため及び送信リストを周期的に更新するための送信状態マシンを含む。更に、ポーリング・ロジックは、状態信号を監視するため及び受信リストを周期的に更新するための受信状態マシンを含む。好適には、送信リストはポートの各々に対する送信アクティブ・ビットを含み、送信状態マシンは、対応するネットワーク・ポートがデータを受信するためのスペースを有することを示すときに、対応する送信アクティブ・ビットをセットする。対応する送信アクティブ・ビットは、対応するネットワーク・ポートに送信するためのデータが供給されたときにクリアされる。更に、受信リストはポートの各々に対する受信アクティブ・ビットを含み、受信状態マシンは、対応するネットワーク・ポートがネットワーク・デバイスからの受信したデータを有することを示すときに、対応する受信アクティブ・ビットをセットする。対応する受信アクティブ・ビットは、データが対応するネットワーク・ポートから読み出されたときにクリアされる。

【００１１】

送信リストはポートの各々に対する送信優先順位（プライオリティ）カウントを含み、送信状態マシンは、対応するネットワーク・ポートがデータを受信するためのスペースを有することを示すときに、対応する送信優先順位カウントを更新する。受信リストはポートの各々に対する受信優先順位カウントを含み、受信状態マシンは、対応するネットワーク・ポートがネットワーク・デバイスからの受信したデータを有することを示すときに、対応する受信優先順位カウントを更新する。優先順位カウントは、好適には、先着順サービス（ＦＣＦＳ）優先順位スキーム、又は所定の重み（ウエイト）ファクタ優先順位スキームを基にする。ひとたび送信及び／又は受信優先順位カウントがポートに割り当てられると、カウントは、そのポートにサービスが行われるまで優先順位を維持するためにマスクされる。

【００１２】

仲裁及びコントロール・ロジックは、送信リスト及び受信リストを見直し、ポートにサービスを行い、対応する送信アクティブ・ビット及び受信アクティブ・ビットをクリアする

。仲裁及びコントロール・ロジックは、優先順位カウントを基にして最高の優先順位を有するポートを判定し、適当な転送動作を行うための送信及び受信ロジック部を含む。

【００１３】

好適な実施例において、ネットワーク・ポートの各々は、ネットワーク・デバイスへの送信のためのデータを記憶するための送信バッファと、ネットワーク・デバイスから受信したデータを記憶するための受信バッファとを含む。ポートの各々のポート状態ロジックは、送信バッファが少なくとも所定のバス転送サイズと等しい使用可能スペース量を有することを示す送信状態信号をアサートするための送信状態ロジックと、受信バッファがネットワーク・デバイスから少なくともバス転送サイズと等しいデータ量を受信したことを示す受信状態信号をアサートするための受信状態ロジックとを更に含む。

【００１４】

本発明に従うネットワーク・スイッチは、１つ又はそれ以上のネットワーク・プロトコルに従うデータ・パケットを送信及び受信するための複数のネットワーク・デバイスを含むネットワーク・システムにおいて好適に用いられる。ネットワーク・スイッチは、データ・パケットを転送するためにネットワーク・デバイスの１つ又はそれ以上に結合するための複数のポートを含む。ネットワーク・スイッチは、ポートの各々の受信及び送信状態を連続的に判定するポーリング・システムを含み、ポートの各々は、対応する受信状態信号を供給することよって、及びそのポートの送信状態を示す対応する送信信号を供給することによって、問い合わせ信号に応答する。

【００１５】

本発明に従うポーリング・システムが、ネットワーク・スイッチのポートの受信状態及び送信状態を決定するための効率的なシステムを提供することが理解される。

【００１６】

【発明の実施の態様】

図１を参照すると、本発明に従って構成されたネットワーク・スイッチ１０２を含むネットワーク・システム１００の簡略図が示されている。ネットワーク・スイッチ１０２は、それぞれが適当なメディア・セグメント１０８を介して“Ａ”ネットワーク１０６の１つと結合およびこれと交信する１つまたは複数の“Ａ”ポートを含む。各メディア・セグメント１０８は、よった対のワイヤ・ケーブル、光ファイバ・ケーブルその他のような、ネットワーク・デバイスを接続するための任意のタイプの媒体である。ポート１０４は、ネットワーク・スイッチ１０２とネットワーク１０６の各々との間における双方向通信またはデータ・フローを可能ならしめる。このような双方向データ・フローは、例えば半二重モードあるいは全二重モードのような、いくつかのモードのいずれか１つのモードに従う。図１に示すように、“ｊ”＋１までのネットワーク１０６が存在し、それぞれにＡネットワーク（Ａ－ＮＥＴＷＯＲＫ）０、Ａネットワーク１、・・・、Ａネットワークｊという名称が付与されており、各ネットワーク１０６は、それぞれＡポート（Ａ－ＰＯＲＴ）０、Ａポート１、・・・、Ａポートｊという名称が付与されているｊ＋１個のポート１０４のうち対応する１つを介してネットワーク・スイッチ１０２に結合する。ネットワーク・スイッチ１０２は、対応する数までのネットワーク１０６に結合すべく任意の数のポート１０４を含むことができる。本明細書で説明する実施例において、ｊは２４までのネットワーク１０６との結合のための全部で２４のポートに対するために２３に等しい整数である。本明細書においては、これらのポートを一括してポート１０４と呼ぶか、あるいは個別にポート（ＰＯＲＴ）０、ポート１、ポート２、・・・、ポート２３と呼称する。

【００１７】

同様に、ネットワーク・スイッチ１０２はさらに、それぞれが適当なメディア・セグメント１１４を介して“Ｂ”ネットワーク１１２に結合およびこれとインタフェースする１つまたは複数の“Ｂ”ポート１１０を含む。また、各メディア・セグメント１１４は、よった対のワイヤ・ケーブル、光ファイバ・ケーブルその他のような、ネットワーク・デバイスを接続するための任意のタイプの媒体である。ポート１１０もまた双方向型であり、ネットワーク・スイッチ１０２とネットワーク１１２との間におけるデータ・フローを、ポ

ート１０４に関する上述の説明と同様に可能ならしめる。本明細書で説明する実施例において、それぞれにＢネットワーク（Ｂ－ＮＥＴＷＯＲＫ）０、Ｂネットワーク１、・・・、Ｂネットワークｋという名称が付与されている“ｋ”＋１までのネットワーク１１２との結合に備えて“ｋ”＋１の数のポート１１０が存在し、個別にＢポート（Ｂ－ＰＯＲＴ）０、Ｂポート１、・・・、Ｂポートｋと呼称する。ネットワーク・スイッチ１０２は、対応する数までのネットワーク１１２に結合すべく任意の数のポート１１０を含むことができる。本明細書に示す特定的な実施例において、Ｋは４つまでのネットワーク１１２との結合のための全部で４個のポート１１０のために、３に等しい整数である。“Ａ”タイプのポートおよびネットワークは、“Ｂ”タイプのポートおよびネットワークと異なるネットワーク・プロトコルおよび／または速度で動作する。本明細書に示す特定的な実施例において、ポート１０４およびネットワーク１０６はイーサネット（Ethernet）プロトコルに従い１０メガビット／秒（Ｍｂｐｓ）で動作し、一方、ポート１１０およびネットワーク１１２はイーサネットのプロトコルに従って１００Ｍｂｐｓで動作する。本明細書では、Ｂポート０、Ｂポート１、・・・、Ｂポート３を総称してポート１１０とし、個別にはそれぞれポート２４、ポート２５、・・・、ポート２７と呼称する。

【００１８】

ネットワーク１０６および１１２は、データの入力あるいは出力のために１つまたは複数のデータ・デバイスもしくはデータ端末装置（ＤＴＥ）、あるいは１つまたは複数のデータ・デバイスを接続するために任意のタイプのネットワーク・デバイスを含む。このように、Ａネットワーク０やＢネットワーク・１などのようないずれのネットワークも、それぞれ１つまたは複数のコンピュータ、ネットワーク・インタフェース・カード（ＮＩＣ）、ワークステーション、ファイル・サーバ、モデム、プリンタ、あるいはリピータ、スイッチ、ルータ、ハブ、集信装置といったネットワーク内でのデータの受信や送信のための他のデバイスを含むことができる。例えば図１に示すように、いくつかのコンピュータ・システムあるいはワークステーション１２０、１２２および１２４は、Ａネットワークｊの対応するセグメント１０８に結合されている。コンピュータ・システム１２０、１２２および１２４は相互に、あるいはネットワーク・スイッチ１０２を介して他のネットワークの他のデバイスと通信することができる。そこで各ネットワーク１０６および１１２は１つまたは複数のセグメントを介して結合された１つまたは複数のデータ・デバイスを表し、ネットワーク・スイッチ１０２がネットワーク１０６および１１２のいずれかの中の何れか２つまたはそれ以上のデータ・デバイスの間でデータの転送を行う。

【００１９】

ネットワーク・スイッチ１０２は、ポート１０４および１１０の各々に結合されたデータ・デバイスから情報を受け取り、その情報を他のポート１０４および１１０のいずれかのものまたは複数のものへルーティングする（送る）動作を一般的に行う。ネットワーク・スイッチ１０２はまた、同じネットワーク内のデータ・デバイスに対してのみと意図された、１つのネットワーク１０６または１１２内の１つのデータ・デバイスから受信した情報をドロップ（落とす）するか、さもなくば無視することによって情報のフィルタリングを行う。データあるいは情報はパケットの形になっているが、各データ・パケットの形はそのネットワークがサポートしているプロトコルによって異なる。パケットは予め定義されたバイトのブロックであり、通常ヘッダ、データ、およびトレーラから成り、特定のパケットの形式はそのパケットを生成したプロトコルによって決まる。ヘッダは、一般に、宛て先のデータ・デバイスを識別する宛先アドレス、およびパケットの発信元であるデータ・デバイスを識別するソース・アドレスを含み、普通これらのアドレスは業界内での一意性を保証するメディア・アクセス・コントロール（ＭＡＣ）アドレスである。１つの宛て先デバイスに対して意図されたパケットを、ここではユニキャスト（unicast）・パケットという。さらに、ヘッダはグループ（ＧＲＯＵＰ）ビットを含み、このビットは、そのパケットが複数の受信先デバイスに向けられたマルチキャスト（multicast）又はブロードキャスト（ＢＣ）・パケットであるかを表示する。もしグループ・ビットがロジック１（１）にセットされていれば、それはマルチキャスト・パケットであると考慮され、も

し宛先アドレスのビットがすべてロジック１（１）にセットされていれば、そのパケットはＢＣパケットである。しかし、本発明の目的上、マルチキャストおよびＢＣパケットを同等に扱い、以降はＢＣパケットと呼称する。

【００２０】

図２を参照すると、ネットワーク・スイッチ１０２のさらに詳細なブロック図が示されている。示した実施例において、ネットワーク・スイッチ１０２は、６つの類似のカッド・コントローラあるいはカッド・カスケード（ＱＣ）・デバイス２０２を含み、それぞれが４つのポート１０４を組み込んでいる。ＱＣデバイス２０２は、単一の特定用途向けＩＣ（ＡＳＩＣ）パッケージへ統合して、あるいは示されているような個別の集積回路（ＩＣ）チップとして、任意の所望の形で実施することができる。示した実施例において、各ポート１０４は半二重方式により１０Ｍｂｐｓで動作し、合計スループットが全二重で１ポートあたり２０Ｍｂｐｓとなる。その結果、６つのＱＣデバイス２０２がすべて全二重方式で動作すれば合計で４８０Ｍｂｐｓとなる。各ＱＣデバイス２０２は、好適には、ＱＣ／ＣＰＵバス２０４に結合したプロセッサ・インタフェース、および高速バス（ＨＳＢ）２０６に結合したバス・インタフェースを含む。ＨＳＢ２０６は、データ部２０６ａおよび各種の制御及び状態信号２０６ｂを含む。ＨＳＢ２０６は、毎秒１ギガビット以上のデータを転送する３２ビット、３３メガヘルツ（ＭＨｚ）のバスである。

【００２１】

ＨＳＢ２０６およびＱＣ／ＣＰＵバス２０４はさらに、イーサネット・パケット・スイッチ・マネジャ（ＥＰＳＭ）２１０に結合される。ＥＰＳＭ２１０の実施について、本発明はなんら特定の物理的または論理的制約を課していないが、示されている実施例ではＡＳＩＣとして実施される。ＥＰＳＭ２１０はさらに、データおよびアドレス部２１４ａと制御信号２１４ｂを含む３２ビットのメモリ・バス２１４を介してメモリ２１２に結合される。メモリ２１２は、好適には、特定の用途で必要に応じて任意に増設が可能ではあるが、４から１６メガバイト（ＭＢ）のダイナミック・ランダム・アクセス・メモリ（ＤＲＡＭ）を含んでいる。ＥＰＳＭ２１０は、動作が約６０ナノ秒（ｎｓ）の高速ページ・モード（ＦＰＭ）のシングル・インライン・メモリ・モジュール（ＳＩＭＭ）、拡張データ出力（ＥＤＯ）モードのＤＲＡＭ　ＳＩＭＭ、あるいは同期モードのＤＲＡＭ　ＳＩＭＭを含む、メモリ２１２の実施のための少なくとも３つの異なったタイプのＤＲＡＭのうちのいずれか１つをサポートする。同期ＤＲＡＭは、一般に、６６ＭＨｚデータ速度又は１秒あたり２６６ＭＢのバースト・データ速度を達成するために、６６ＭＨｚのクロックを必要とする。ＥＤＯ　ＤＲＡＭは、３３又は６６ＭＨｚのいずれかのクロックで動作できるが、いずれのクロック速度においても３３ＭＨｚ、または１秒あたり１３３ＭＢの最大データ・バースト・データ速度を達成する。ＦＰＭ　ＤＲＡＭもまた３３又は６６ＭＨｚのクロックで動作が可能であるが、３３ＭＨｚクロックで１６ＭＨｚ又は１秒あたり６４ＭＢの最大バースト速度を達成し、６６ＭＨｚクロックで２２ＭＨｚ又は１秒あたり８８ＭＢのバースト速度を達成する。

【００２２】

メモリ・バス２１４は、メモリ・データ・バスＭＤ［３１：０］、データ・パリティ信号ＭＤ＿ＰＡＲ［３：０］、行および列（カラム）アドレス信号ＭＡ［１１：０］、ライト（書き込み）・イネーブル信号ＭＷＥ＊、ＦＰＭ　ＤＲＡＭ及びＥＤＯ　ＤＲＡＭの行信号又は同期ＤＲＡＭのチップ選択のいずれかであるバンク選択信号ＲＡＳ［３：０］＊／ＳＤ＿ＣＳ＊［３：０］、ＦＰＭ及びＥＤＯの列信号または同期ＤＲＡＭのＤＱＭであるメモリ・バイト制御信号ＣＡＳ［３：０］＊／ＳＤ＿ＤＱＭ［３：０］、同期ＤＲＡＭのみへの行信号ＳＤ＿ＲＡＳ＊、同期ＤＲＡＭのみへの列信号ＳＤ＿ＣＡＳ＊、シリアル入力ＳＩＭＭ／ＤＩＭＭ存在検知信号ＰＤ＿ＳＥＲＩＡＬ＿ＩＮ、およびパラレル入力ＳＩＭＭ／ＤＩＭＭ存在検知信号ＰＤ＿ＬＯＡＤ＊を含む。

【００２３】

ＨＳＢ２０６は、サンダー（Thunder）ＬＡＮ（ＴＬＡＮ）ポート・インタフェース（ＴＰＩ）２２０に結合され、これがさらにデータ及びアドレス信号２２２ａおよび関連の制

御及び状態信号２２２ｂを含む周辺コンポーネント相互接続（ＰＣＩ）バス２２２に結合される。ＰＣＩバス２２２は４つのＴＬＡＮ２２６に結合され、これは任意の様式で実施される。ＴＬＡＮ２２６は、それぞれがポート１１０の１つを組み込んでいる、テキサス・インストルメンツ社（Texas Instruments, Inc.）（ＴＩ）製のＴＮＥＴＥ１００　ＴｈｕｎｄｅｒＬＡＮ™（サンダーＬＡＮ、登録商標）　ＰＣＩ　Ｅｔｈｅｒｎｅｔ™（イーサネット、登録商標）コントローラが好適である。ＥＰＳＭ２１０に対して、ＴＰＩ２２０は４つのポートをインタフェースするために、別のＱＣデバイス２０２と同様にＨＳＢ上で動作する。従って、ＥＰＳＭ２１０には実際上７つのカッド・ポート・デバイスが「見える」。ＰＣＩバス２２２に関しては、ＴＰＩ２２０が、標準ＰＣＩバスのエミュレーションを、通常ＰＣＩのメモリ・デバイスとインタフェースするＴＬＡＮ２２６の適切な動作に必要な程度まで、行う。従って、ＰＣＩバス２２２は完全にＰＣＩに従順である必要がない。ＰＣＩバス２２２は、ＣＰＵ２３０をローカルのＲＡＭ２３４、ローカルのフラッシュＲＡＭ２３６および必要であればシリアル・ポート・インタフェース２３８に結合するためのローカル・プロセッサ・バス２３２に結合されているプロセッサ又は中央処理装置（ＣＰＵ）２３０に結合される。シリアル・ポート・インタフェース２３８は、ＵＡＲＴまたは同等のものが望ましい。示した実施例においては、ＣＰＵはインテル社（Intel）製の３２ビット、３３ＭＨｚのｉ９６０ＲＰ　ＣＰＵであるが、ＣＰＵ２３０は他の適切なプロセッサでも構わない。

【００２４】

ＣＰＵ２３０は、通常ネットワーク・スイッチ１０２のパワーアップでＴＰＩ２２０およびＥＰＳＭ２１０の初期設定とコンフィギュレーションの処理を行う。また、ＣＰＵ２３０は統計情報の監視及び収集を行い、さらに動作時にはネットワーク・スイッチ１０２の各種デバイスの機能を管理及び制御する。さらにまたＣＰＵ２３０は、メモリ２１２内のハッシュ・テーブル・データをＥＰＳＭ２１０を通じて更新する。しかし、ＥＰＳＭ２１０は、メモリ２１２へのアクセスを制御し、ＤＲＡＭのリフレッシュ・サイクルを実行し、それによってＣＰＵ２３０によるリフレッシュ動作が不要となる。このように設計されていなければ、ＣＰＵ２３０は各リフレッシュ・サイクルの実行におよそ６～８バス・サイクルを要することになり、これは貴重なプロセッサ・リソースを消費することとなる。ＣＰＵ２３０はまた、様々な目的のための付加的なネットワーク・ポートとして機能し、従って本明細書ではポート（ＰＯＲＴ）２８として言及する場合がある。このように、ポート１０４、１１０、およびＣＰＵ２３０は、それぞれポートポート０～ポート２８を集合的に含むものである。

【００２５】

ＣＰＵ２３０はさらに、アドレス及びデータ部２１８ａおよび関連の制御及び状態信号２１８ｂを含むＣＰＵバス２１８を介してＥＰＳＭ２１０に結合される。アドレス及びデータ部２１８ａは、アドレスとデータ信号間で多重化されていることが望ましい。特定的には、ＣＰＵバス２１８は、アドレス／データ・バスＣＰＵ＿ＡＤ［３１：０］、ＣＰＵ２３０からのアドレス・ストローブＣＰＵ＿ＡＤＳ＊、データ・バイト・イネーブルＣＰＵ＿ＢＥ［３：０］、リード／ライト選択信号ＣＰＵ＿ＷＲ＊、バースト最終データ・ストローブＣＰＵ＿ＢＬＡＳＴ＊、データ・レディ信号ＣＰＵ＿ＲＤＹ＊、および少なくとも１つのＣＰＵ割り込み信号ＣＰＵ＿ＩＮＴ＊を含む。本開示において、データまたはアドレス信号の他の通常の信号名は正のロジックを表し、その信号はハイ又はロジック１のときアサートされるとみなされ、後尾にアステリスク（＊）が付加された信号名は負のロジックを示し、その信号はロー又はロジック０のときにアサートされるとみなされる。各信号の機能的な定義は一般に直接的であって、普通はその信号名で判断され得る。

【００２６】

図３は、４つのポート１０４の実施のための例示的なＱＣデバイス２０２のブロック図であり、このデバイスは２４ポート、ポート０～ポート２３を実施するために６つ複製される。特定のデバイスを１つ挙げれば、ＬＳＩロジック社（LSI Logic Corporation）（ＬＳＩ）製のＬ６４３８１カッド・カスケード・イーサネット（Quad Cascade Ethernet）

・コントローラ・デバイスがある。これよりグレードの高いデバイスとして、やはりＬＳＩ製のＱＥ１１０カッド・カスケード・イーサネット・コントローラ・デバイスがあり、これは本明細書で説明しているような付加的機能および能力を備えている。しかし留意すべきは、本発明はポート１０４の実施をなんら特定のデバイスに限定しているものではない。示した実施例において、各ＱＣデバイス２０２はポート１０４のそれぞれに対してイーサネット・コア３００を含み、イーサネット・コア３００は完全な同期型であって、メディア・アクセス・コントローラ、マンチェスタ・エンコーダ／デコーダ、およびよった対／ＡＵＩ（接続機構インタフェース（Attachment Unit Interface））トランシーバを含む。各イーサネット・コア３００は、対応するセグメント１０８上の結合されているネットワーク１０６との双方向データ通信を可能とし、それぞれが対応する１２８ビット受信ＦＩＦＯ（先入れ先だし（First-In, First-Out））３０２および１２８ビット送信ＦＩＦＯ３０４と結合している。各イーサネット・コア３００は、さらに統計カウンタ３０６のブロックと結合しており、統計カウンタ３０６の各ブロックは、オンチップ・メインテナンス用に２５のカウンタを含む。統計カウンタ３０６の各ブロック内のカウンタは、シンプル・ネットワーク・マネジメント・プロトコル（Simple Network Management Protocol）（ＳＮＭＰ）の要件に見合うのが望ましい。ＦＩＦＯ３０２および３０４の各々は、さらに、各ＱＣデバイス２０２とＥＰＳＭ２１０の間での双方向データ・フローを可能とするためにＨＳＢ２０６に結合しているバス・インタフェース・ロジック３０８に結合される。各ＱＣデバイス２０２は、ソース・アドレス挿入、フレーム・チェック・シーケンス（ＦＣＳ）挿入、衝突時の即時再送信、バス転送サイズ、および送信バッファ・スレッショルド・サイズといったコンフィギュレーションをプログラミング可能（プログラマブル）とするために、コンフィギュレーション及びコントロール（制御）・ロジック３１０を含む。

【００２７】

コンフィギュレーション及びコントロール・ロジック３１０と、統計カウンタ３０６の各ブロックと、ＦＩＦＯ３０２、３０４はＱＣ／ＣＰＵバス２０４に結合される。ＥＰＳＭ２１０は、ＣＰＵバス２１８とＱＣ／ＣＰＵバス２０４との間に別のインタフェースを提供する。このようにして、ＣＰＵ２３０は、各ＱＣデバイス２０２の各々、従ってポート１０４の各々に対し、そのアクティビティを初期設定、構成（コンフィギュレーション）、監視（モニタ）、および修正すべく完全なアクセスを得る。ＱＥ１１０カッド・カスケード・イーサネット・コントローラ・デバイスは、もし背圧（バックプレッシャ（backpressure））指示の受信が間に合うならば、受信されていたパケットを終了するためのジャミング・シーケンス（jamming sequence）をアサートするために背圧指示を検知するために、コンフィギュレーション及びコントロール・ロジック３１０間に付加的な接続３２０を含む。背圧指示はＨＳＢ２０６上で実行される背圧サイクルが望ましいが、背圧指示を示すために別の信号又はそれと同様のものを用いるなど、いくつかの方法の任意のものを用いることができる。

【００２８】

ここで、ジャミング・シーケンスは「早い」又は適時だと考えられるポートで受信中のデータ・パケットの最初の６４バイトの間に送信すべきであるという点に留意されたい。最初の１６バイト（４つのＤＷＯＲＤ）は、後述するハッシュ・ルックアップ手順がＥＰＳＭ２１０によって実行される前に要求される。最初の１６バイトがおよそ１３マイクロ秒（μｓ）で転送されるように、各データ・ビットはイーサネット１０Ｂａｓｅ－Ｔを約１００ｎｓの速度で転送される。６４バイトがおよそ５１μｓの間に受信され、それによって、ネットワーク・スイッチ１０２は、受信された最初の１６バイトを転送し、ハッシュ手順を行い、背圧サイクルを実行し、最終的にジャミング・シーケンスをアサートするために、約３８μｓ有する。ハッシュ・ルックアップは完了するのに約１～２μｓ要するので、ほとんど常に、適時（タイムリー）にジャミング・シーケンスを送信するために十分な時間がある。しかし、ジャミング・シーケンスをタイムリーにアサートできるという保証はない。そのため、スレッショルド違反条件に起因してパケットを落とす（ドロップす

る）可能性がある。もし背圧サイクル遅れて実行されると、そのポートは背圧サイクルを拒否し、ネットワーク・スイッチ１０２はそのパケットを受け取れなければそのパケットをドロップする。スレッショルド条件が早期の指示であり、従ってメモリがパケットを格納するために使用可能であり得るため、ネットワーク・スイッチ１０２はそのパケットを受け取れる。

【００２９】

もし背圧サイクルがタイムリーに実行され、もしポートが半二重モードで動作していれば、コンフィギュレーション及びコントロール・ロジック３１０は示されたポート１０４のイーサネット・コア３００の１つへ衝突コマンドを応答的にアサートする。衝突コマンドを受け取るイーサネット・コア３００は、ジャミング・シーケンスをアサートし、そのポート１０４が受信しているパケットを終了させる。もし背圧サイクルが６４バイト・ウインドウ内に実行されるならば、ポートは、ＨＳＢ２０６上でアボート信号ＡＢＯＲＴ＿ＯＵＴ＊をアサートすることによって、そのポートに背圧サイクルが実行される旨をＥＰＳＭ２１０に示す。もし背圧サイクルが６４バイト・ウインドウの外側であり、従って時間内にアサートされなければ、ＡＢＯＲＴ＿ＯＵＴ＊信号はアサートされず、ＥＰＳＭ２１０はそのパケットをドロップする。背圧アサートの試行が失敗すれば、ほとんどの場合ＥＰＳＭ２１０はそのパケットをドロップする。最高の能率を達成するためにはドロップされるパケットはできるだけ少ない方がよいが、ドロップされたパケットは最終的に送信側のデータ・デバイスにおける高いネットワーク・レベルで検知され、従ってネットワーク・システム１００の全体的な動作には致命的なものとならない。送信側のデバイスはパケットのドロップを検知し、そのドロップされたパケットを含む１つ又はそれ以上の数のパケットを再送信する。

【００３０】

バス・インタフェース・ロジック３０８は、後に詳述するように、ＨＳＢ２０６上で同時のリード及びライト・サイクルを実現するために、リード・ラッチ３２４およびライト・ラッチ３２６を含んでいることが望ましい。これらのラッチは、第１のクロック（ＣＬＫ＿１）信号の特定のサイクルでＨＳＢ２０６上にアサートされたＰＯＲＴ＿ＮＯ［１：０］信号をラッチする。ＣＬＫ＿１信号は、ＨＳＢ２０６にとっての主クロックであり、示した実施例においては通常およそ３０～３３ＭＨｚで動作する。ＣＬＫ＿１信号は主クロックであるので、以降本明細書では単にＣＬＫ信号と呼称する。第２のクロック信号ＣＬＫ＿２もメモリ２１２とのインタフェースに使用され、ＣＬＫ信号の周波数の２倍（２Ｘ）又は約６０～６６ＭＨｚで動作する。

【００３１】

図４は、図３に示す特定のカッド・カスケード・デバイス２０２の信号の図解である。これらの信号は、ＱＣバス２０４と関連のプロセッサ・インタフェース信号、４つのポート１０４に関連のネットワーク・インタフェース信号、状態信号、クロック及びテスト信号、ＨＳＢバス２０６に関連のバス・インタフェース信号、およびその他種々の信号を含む、いくつかの機能およびバスのセクションに分けられる。

【００３２】

ＱＣバス２０４に関しては、ＥＰＳＭ２１０は、データ信号ＰＤＡＴＡ［１５：０］を通じて、ＱＣデバイス２０２のレジスタおよびカウンタ３０６、３１０とデータの読み書きを行う。ＲＥＡＤ＊信号は書き込み動作に対してはハイにアサートされ、読み出し動作に対してはローにアサートされる。ＱＣデバイス２０２内の特定のレジスタは、ＡＤＲＳ［５：０］信号にアサートされたアドレスによって決定される。アドレス・ストローブ信号ＡＤＲＳ＿ＳＴＲＯＢＥ＊がいくつかのチップ選択信号ＣＨＩＰ＿ＳＥＬＥＣＴｍ＊の対応する１つとともにアサートされると、ＱＣデバイス２０２はＡＤＲＳ信号をラッチする。信号名に付けられた小文字の“ｍ”は、一般に１つの特定のタイプに属する複数の信号を意味する。例えば、６つの別々のＣＨＩＰ＿ＳＥＬＥＣＴ［５：０］＊信号があり、その場合それぞれの信号は６つのＱＣデバイス２０２のそれぞれ１つに別個にアクセスするためのものである。信号ＰＲＥＡＤＹ＊は、要求されたデータがラッチされるＣＬＫ信号

の立ち上がり後のライト・サイクル中に、ＣＬＫ信号の１サイクルに対してＱＣデバイス２０２によってローにアサートされる。リード・サイクルについては、ＱＣデバイス２０２が、データをＰＤＡＴＡバス上に置いた後の１ＣＬＫサイクルに対してＰＲＥＡＤＹ＊をローにアサートする。

【００３３】

図５は、ＱＣデバイス２０２のプロセッサ・リード・サイクルを図解する例示的なタイミング図であり、図６は、プロセッサ・ライト・サイクルを図解する例示的なタイミング図である。図７は、ＱＣデバイス２０２のプロセッサ・バースト・リード・アクセス・サイクルを図解する例示的なタイミング図である。これらのタイミング図はいずれもあくまで例示であって、特定のタイミングや特定の信号特性などを示すものではなく、一般的な相関性を図解するものである。

【００３４】

図４に戻り、これを参照する。ネットワーク・インタフェース信号は、負および正の衝突スレッショルド信号、衝突参照信号、信号中のシリアル・データ、負および正のマンチェスタ符号化データ信号、正および負のデータ・スレッショルド信号、データ・スレッショルド参照信号、正および負のプリエンファシス（Pre-emphasis）信号、および各ＱＣデバイス２０２の［３：０］で表される４ポートの各々に対するよった対／ＡＵＩモード選択信号を含む。各ＱＣデバイスはＣＬＫ信号を受信し、ポート１０４が使用する８０、２０および１０ＭＨｚの内部クロック信号を生成するための２０ＭＨｚのクロック信号を受信するＣＬＯＣＫ＿２０ＭＨＺ入力を有する。各イーサネット・コア３００は、対応するセグメント１０８で発生する衝突を検知し、イーサネットのＣＳＭＡ／ＣＤ（キャリア検知多重アクセス／衝突検出（Carrier Sense Multiple Access/Collision Detect））法に従ってジャミング・シーケンスを送信する。

【００３５】

ＨＳＢ２０６に関連するバス・インタフェース信号については、ＱＣデバイス２０２がＡＢＯＲＴ＿ＯＵＴ＊信号をアサートして１つのパケット全体をアボートする。ＥＰＳＭ２１０は、アボート信号ＡＢＯＲＴ＿ＩＮ＊をアサートして現在のバス・サイクルをアボートする。１つの実施例においては、ＱＣデバイス２０２は、ＥＰＳＭ２１０がＨＳＢ２０６上で背圧サイクルを実行することによって受信しているパケットをアボートできるように考案されたＱＥ１１０デバイスである。この特定のタイプの背圧機能は、１つのポートで受信中の１つのパケットの拒否を可能とする「パケット毎（パケット・バイ・パケット）」あるいは動的な「ポートごと」の背圧である。Ｌ６４３８１デバイスは、本明細書で後に詳述する自動挿入フレーム・チェック・シーケンス信号（ＡＩ＿ＦＣＳ＿ＩＮ＊）を含む。ＱＥ１１０デバイスはＡＩ＿ＦＣＳ＿ＩＮ＊信号を信号ＦＢＰＮ＊と置換する。この信号はＡＩ＿ＦＣＳ＿ＩＮ＊信号と同じ機能を遂行するために使用されるが、背圧サイクルおよびエンハンスト・パケット・フラッシュ（enhanced packet flush）を示すためにも用いられる。本明細書で説明しているように、動的背圧を実施するために使用できる代替方法が多数存在することは言うまでもない。特に、ＥＰＳＭ２１０は、背圧要求サイクルを実行するためにリード・サイクル中にＦＢＰＮ＊信号をアサートする。もしＡＢＯＲＴ＿ＯＵＴ＊信号がリード・サイクルのデータ・フェーズの間に対応するＱＣデバイス２０２によってアサートされると、その背圧「要求」はそのＱＣデバイス２０２に認められたことになり、これがジャミング・シーケンスをアサートしてそのパケットをアボートする。もしＡＢＯＲＴ＿ＯＵＴ＊信号がアサートされないと、ＥＰＳＭ２１０はそのパケットをドロップする。

【００３６】

ＥＰＳＭ２１０は、ＱＣデバイス２０２およびＴＰＩ２２０のすべてに対して状態ストローブ信号ＳＴＲＯＢＥ＊をアサートし、その各々は、ＳＴＲＯＢＥ＊信号がＣＬＫ信号の立ち上がりでアサートされてサンプリングされるときに、信号ＰＫＴ＿ＡＶＡＩＬｍ＊およびＢＵＦ＿ＡＶＡＩＬｍ＊上で多重化された様式でその４つのポート１０４又は１１０（ＴＰＩ２２０の場合）の状態で応答する。或る動作に対しては別のポートとして働く、

各ＱＣデバイス２０２に対する別個の信号、ＴＰＩ２２０に対して１組及びＣＰＵ２３０に対して類似の組、がある。特にＰＫＴ＿ＡＶＡＩＬｍ＊およびＢＵＦ＿ＡＶＡＩＬｍ＊信号は、ＱＣデバイス２０２用に信号ＰＫＴ＿ＡＶＡＩＬ［５：０］＊およびＢＵＦ＿ＡＶＡＩＬ［５：０］＊と、ＴＰＩ２２０用にそれぞれＰＫＴ＿ＡＶＡＩＬ［６］＊およびＢＵＦ＿ＡＶＡＩＬ［６］＊とも呼ばれる信号ＴＰＩ＿ＰＫＴ＿ＡＶＡＩＬ＊およびＴＰＩ＿ＢＵＦ＿ＡＶＡＩＬ＊と、ＣＰＵ２３０に対応するＰＫＴ＿ＡＶＡＩＬ［７］＊およびＢＵＦ＿ＡＶＡＩＬ［７］＊ともそれぞれ呼ばれる信号ＰＣＢ＿ＰＫＴ＿ＡＶＡＩＬ＊およびＰＣＢ＿ＢＵＦ＿ＡＶＡＩＬ＊との、１つの信号タイプについて全部で８つの信号を含む。

【００３７】

このように、ＨＳＢ２０６は最初ＱＣデバイス２０２が４つのポート、ポート０～ポート３にアクセスするための信号ＰＫＴ＿ＡＶＡＩＬ［０］＊およびＢＵＦ＿ＡＶＡＩＬ［０］＊を含み、ＨＳＢ２０６は次のＱＣデバイス２０２が次の４つのポート、ポート４～ポート７にアクセスするための信号ＰＫＴ＿ＡＶＡＩＬ［１］＊およびＢＵＦ＿ＡＶＡＩＬ［１］＊を含み、と以下同様で、ＴＰＩ２２０はポート、ポート２４～ポート２７にアクセスするための信号ＰＫＴ＿ＡＶＡＩＬ［６］＊およびＢＵＦ＿ＡＶＡＩＬ［６］＊を含み、ＥＰＳＭ２１０はＣＰＵ２３０に対する内部信号ＰＫＴ＿ＡＶＡＩＬ［７］＊およびＢＵＦ＿ＡＶＡＩＬ［７］＊を含む。ＣＬＫ信号のそれぞれのサイクルで分離される４つのポートに対応する各信号に、最高４ビットが多重化される。

【００３８】

ＳＴＲＯＢＥ＊信号に応答して、バス・インタフェース・ロジック３０８は、それぞれのポートに対する対応する各送信ＦＩＦＯ３０４にデータを格納するスペースが十分あるかどうかを表示するＢＵＦ＿ＡＶＡＩＬ［５：０］＊信号のそれぞれのものに４つの状態ビットを多重化するためのポート状態ロジック３０３を含む。ポート状態ロジック３０３は、図示されている４つのポートのすべてに対して集中化するか、又はポート間に分散するかの何れかである。空きスペースの判定は、ＣＰＵ２３０によって１６、３２あるいは６４バイトにコンフィギュレーションされるのが望ましい、バス転送フィールド・サイズ（ＴＢＵＳ）を格納するバス・インタフェース・ロジック３０８内のコンフィギュレーション・レジスタに従う。同様に、ＳＴＲＯＢＥ＊信号に応答して、ＴＰＩ２２０は、後述するその内部の送信ＦＩＦＯのそれぞれに、ポート２４～ポート２７の各々に対するＴＬＡＮ２２６の対応するものに対するデータを格納するスペースが十分あるかどうかを示すために、ＢＵＦ＿ＡＶＡＩＬ［６］＊信号に４つの状態ビットを多重化するための、ＨＳＢ２０６に結合している類似したポート状態ロジック８２０（図３１）を含む。ＣＰＵ２３０あるいはポート２８については、ＥＰＳＭ２１０内のＰＣＢ４０６（図１１）が、ＥＰＳＭ２１０の内部の内部ＰＣＢ送信ＦＩＦＯにＣＰＵ２３０に対するデータを格納するための使用可能なスペースがあるかどうかを表示するためにＢＵＦ＿ＡＶＡＩＬ［７］＊信号に１つの状態ビットをアサートする。

【００３９】

同様に、ＳＴＲＯＢＥ＊信号に応答して、各ＱＣデバイス２０２内のバス・インタフェース・ロジック３０８のポート状態ロジック３０３は、それぞれのポートに対するその受信ＦＩＦＯ３０２の各々に、ＨＳＢ２０６上におけるバス転送のための受信したデータを送信するために十分なデータがあるかどうかをＴＢＵＳの値によって表示するＰＫＴ＿ＡＶＡＩＬ［５：０］＊信号のそれぞれのものに４つの状態ビットを多重化する。同様に、ＴＰＩ２２０は、その内部の受信ＦＩＦＯがＨＳＢ２０６上における転送のためにそれぞれのポート２３～ポート２７から十分なデータを受信したかどうかを表示するＰＫＴ＿ＡＶＡＩＬ［６］＊信号に４つの状態ビットを多重化する。ＣＰＵ２３０については、ＥＰＳＭ２１０内のＰＣＢ４０６が、ＥＰＳＭ２１０の内部ＰＣＢ受信ＦＩＦＯがＨＳＢ２０６バス転送のためにＣＰＵ２３０から十分なデータを受信したかどうかを表示するＰＫＴ＿ＡＶＡＩＬ［７］＊信号に１つの状態ビットをアサートする。

【００４０】

図８は、ＱＣデバイス２０２およびＴＰＩ２２０のバッファ状態問い合わせを図解する例示的なタイミング図であり、ＥＰＳＭ２１０によるＳＴＲＯＢＥ＊信号のアサートと各ＱＣデバイス２０２の応答、ＴＰＩ２２０のアサートするそれぞれのＰＫＴ＿ＡＶＡＩＬｍ＊およびＢＵＦ＿ＡＶＡＩＬｍ＊信号をを含む。図８におけるポート０、ポート１、ポート２、およびポート３は、特定のＱＣデバイス２０２の４つのそれぞれのポートあるいはＴＰＩ２２０である。ＰＣＢ４０６は、そのポートが４つのフェーズすべてでアクティブになっていることを除けば、その応答は同様である。ＳＴＲＯＢＥ＊信号はレベル・トリガされ、従ってＣＬＫ信号の最初の立ち上がりでローにサンプリングされる。ここで、図８のタイミング図はあくまでも例示であって、特定のタイミングや特定の信号特性などではなく、一般的な相関性を図解するものであることに留意されたい。例えば、ＳＴＲＯＢＥ＊信号は周期的であり、示した実施例の動作においては典型的には１ＣＬＫサイクルを超える間ローにアサートされる。

【００４１】

図４に戻り、これを参照する。信号ＰＯＲＴ＿ＢＵＳＹ＊は、それぞれのポートが半二重モードで送信中であるか受信中であるか、あるいはそのポートがいつ全二重モードで送信しているかを表示するために使用される。リード・データ信号ＲＥＡＤ＿ＯＵＴ＿ＰＫＴ［５：０］＊はＥＰＳＭ２１０にアサートされて、それぞれのＱＣデバイス２０２に対し、それぞれの受信ＦＩＦＯ３０２からのデータをデータ信号ＤＡＴＡ［３１：０］上に置くことを通知する。同様にして、ライト・データ信号ＷＲＩＴＥ＿ＩＮ＿ＰＫＴ［５：０］＊はＥＰＳＭ２１０にアサートされて、それぞれのＱＣデバイス２０２に対し、データ信号ＤＡＴＡ［３１：０］からそれぞれの送信ＦＩＦＯ３０４にデータを取り出すことを通知する。さらに、類似の信号ＰＣＢ＿ＲＤ＿ＯＵＴ＿ＰＫＴ＊、ＰＣＢ＿ＷＲ＿ＩＮ＿ＰＫＴ＊、およびＴＰＩ＿ＲＥＡＤ＿ＯＵＴ＿ＰＫＴ＊、ＴＰＩ＿ＷＲＩＴＥ＿ＩＮ＿ＰＫＴ＊がそれぞれＴＰＩ２２０およびＣＰＵ２３０用に含まれる。すべてのリードおよびライト信号は、集合的にそれぞれＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊およびＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号と呼称する。ＰＯＲＴ＿ＮＯ［１：０］ビットは、どの特定のポート１０４がＨＳＢ２０６上で実行されるサイクルに対してアドレスされているかを表示する。

【００４２】

信号ＳＯＰ＊は、パケットの先頭又はヘッダがＨＳＢ２０６上に転送されたときにパケットの開始（Start Of Packet）を示す。ＡＩ＿ＦＣＳ＿ＩＮ＊信号は、一般にＳＯＰ＊およびＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号の１つとともに外部のデバイスにアサートされ、これにより、（ＱＣデバイス２０２の１つの実施に対して）Ｌ６４３８１デバイスが自動的にパケット内のデータからＣＲＣ（巡回冗長検査（Cyclic Redundancy Check））値を計算し、そのＣＲＣをパケットのＦＣＳフィールドに挿入するようにする。ＱＥ１１０デバイスは付加的な機能のために、前述したように、ＡＩ＿ＦＣＳ＿ＩＮ＊信号をＦＢＰＮ＊信号と置換する。ＥＯＰ＊信号は、ＨＳＢ２０６上でデータ・パケットの最後のデータ転送が転送されたときにパケットの終了（End Of Packet）を表示する。ＢＹＴＥ＿ＶＡＬＩＤ［３：０］＊信号は、ＤＡＴＡ（データ）信号上の現在のワードにおいてどのバイトが有効であるかを表示する。通常１つのデータ・パケットはＨＳＢ２０６上での１回での転送には大き過ぎ、従って各バス・サイクルではＴＢＵＳ値に等しいか又はこれより少ない量のデータが転送されることに留意されたい。

【００４３】

各ＱＣデバイス２０２が４つのポートのそれぞれを１０Ｂａｓｅ－Ｔイーサネット・ポートとして動作させる点が理解できる。また、ＥＰＳＭ２１０がＱＣバス２０４を介してＱＣデバイス２０２のすべてのレジスタに読み書きのアクセスができるということが理解できる。さらに、ＥＰＳＭ２１０はＨＳＢ２０６を介して受信ＦＩＦＯ３０２のすべてからデータを読み取り、送信ＦＩＦＯ３０４のすべてにデータを書き込む。

【００４４】

図９は、ＨＳＢ２０６上での同時リード及びライト・サイクルを図解する例示的なタイミング図である。このタイミング図の一番上にサイクルのタイプを示しており、２つの同時

リード及びライト・サイクルが順次実行される。ＣＬＫ、ＣＬＫ＿２、ＳＴＲＯＢＥ＊、ＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊、ＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊、ＰＯＲＴ＿ＮＯ［１：０］、ＤＡＴＡ［３１：０］、およびＡＢＯＲＴ＿ＯＵＴ＊信号をこのタイミング図のＹ軸（すなわち縦軸）に書いて示し、それに対して時間をＸ軸（すなわち横軸）に書いている。同時リード及びライト・サイクルには２種類があって、それらは特定の構成に依存して実行される。最初の一般的なタイプの同時サイクルについて、ＱＣデバイス２０２がラッチ３２４および３２６を含むＱＥ１１０デバイスで実施される場合は、なんら追加的な策を要せず同時リード及びライト・サイクルが実行される。これに代わって、もしＱＣデバイス２０２がＬ６４３８１デバイスで実施される場合、外部のラッチおよび選択ロジック（示さず）が追加され、ＰＯＲＴ＿ＮＯ信号がＨＳＢ２０６上でアサートされたとき、これをラッチする。２番目の特殊なタイプの同時リード及びライト・サイクルは、何も補強せずＬ６４３８１デバイスで実行される。ただし、それはＰＯＲＴ＿ＮＯ信号が同じであるときのみ且つＱＣデバイス２０２が異なるときのみに限られる。

【００４５】

ＥＰＳＭ２１０は、例えばリード、ライト、同時リード及びライト、背圧などといった、実行すべきサイクルのタイプを決定する。リード・サイクルは一般にＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊信号の１つのアサートによって指示され、ライト・サイクルは通常ＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号の１つのアサートによって指示される。同時リード及びライト・サイクルは、ＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊信号とＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号の同時のアサートによって指示される。ＥＰＳＭ２１０は、例えば、後に詳述するように両ポートともカットスルー（ＣＴ）モードで動作すべくコンフィギュレーションされている場合のみのような、特定の条件の下で２つのポート間で同時リード及びライトを行う。

【００４６】

同時サイクルの期間中、ＥＰＳＭ２１０は３番目のＣＬＫサイクルの始まりでＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊信号の１つをローにアサートしてＱＣデバイス２０２の１つまたはＴＰＩ２２０を指示し、３番目のＣＬＫサイクル中にＰＯＲＴ＿ＮＯ［１：０］信号上に当該のポート番号をアサートして、アサートされた特定のＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊信号で識別されるＱＣデバイス２０２の４ポートのうちの１つを指示する。特定のＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊信号で識別されるＱＣデバイス２０２は、３番目のＣＬＫサイクルにおいてＰＯＲＴ＿ＮＯ［１：０］信号をラッチし、読み出される特定のポートを判断する。例えば、ＱＣデバイス２０２を実施するＱＥ１１０デバイスは、ＰＯＲＴ＿ＮＯ［１：０］信号をラッチするリード・ラッチ３２４を用いて構成される。また、ＴＰＩ２２０は同様のリード・ラッチ８１９ｂ（図３１）を含み、これは、もしＲＥＡＤ＿ＯＵＴ＿ＰＫＴ［６］＊信号で指示されていれば、３番目のＣＬＫサイクルにおいてＰＯＲＴ＿ＮＯ［１：０］信号をラッチする。あるいは、もしＱＣデバイス２０２の機能遂行に用いられるデバイスがＬ６４３８１デバイスであれば、外部のラッチがこの目的に使用される。この時点で、識別されたポート０～ポート２７の特定のポートがＨＳＢ２０６上でリード・サイクルのソース・ポートとして指示されている。

【００４７】

ＥＰＳＭ２１０は、次に４番目のＣＬＫサイクルの始めでＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号の１つをローにアサートして、ＱＣデバイス２０２の同じ又は他のものまたはＴＰＩ２２０を指示し、４番目のＣＬＫサイクル中にＰＯＲＴ＿ＮＯ［１：０］信号上に適当なポート番号をアサートし、アサートされた特定のＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号で示されるデバイスの４ポートのうち１つを指示する。特定のＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号で識別されるＱＣデバイス２０２は、４番目のＣＬＫサイクルにおいてＰＯＲＴ＿ＮＯ［１：０］信号をラッチし、書き込まれる特定のポートを判断する。例えば、ＱＣデバイス２０２の機能を実施するＱＥ１１０デバイスは、第４のＣＬＫサイクルにおいてＰＯＲＴ＿ＮＯ［１：０］信号をラッチするためのライト・ラッチ３２６を用いて構成される。また、ＴＰＩ２２０は、もしＷＲＩＴＥ＿ＩＮ＿ＰＫＴ［６］＊信号で指示されたならば、４番目のＣＬＫサイクルにおいてＰＯＲＴ＿ＮＯ［１：０］信号をラッチするための

同様のライト・ラッチ８１９ｂを含む。このようにして、ポート０～ポート２７の他のいずれかのポートがＨＳＢ２０６上のライト・サイクルの宛て先ポートとして指示され、そのライト・サイクルは指示されたばかりのリード・サイクルと同時に実行される。ソース・ポートと宛て先ポートは、同一のＱＣデバイス２０２上か、ＴＰＩ２２０の２つのポート間か、異なるＱＣデバイス２０２間のいずれに存在し得る。しかし、示した実施例においては、同時リード及びライト・サイクルは、ＱＣデバイス２０２のポート１０４の１つとＴＰＩ２２０のポート１１０の１つとの間では、データ転送の速度が違うために実行されない。

【００４８】

ＣＬＫ信号の次のサイクルで、パケット・データはＨＳＢ２０６を介して転送、あるいはソース・ポートから読み出され、直接に宛て先ポートに書き込まれ、その際ＥＰＳＭ２１０あるいはメモリ２１２には格納されない。データ転送は、実施例によって異なるが幾つかのバイトを転送するためにサイクル５、６、７、および８で実行される。例えば、Ｌ６４３８１デバイスに関しては６４バイトまでが転送され、ＱＥ１１０デバイスでは２５６バイトまでが転送される。データ転送に４つのＣＬＫサイクルを示しているが、送るべきデータの量によっては１、２あるいは４のＣＬＫサイクルで転送される場合もあり得る。新しパケットに関しては、最初に通常のリード・サイクルが実行されてソースおよび宛て先のＭＡＣアドレスがＥＰＳＭ２１０に供給され、これが後に詳述するハッシュ手順を実行し、もし既知であれば、宛て先ポート番号を決定する。受信先（宛て先）ポート番号が分かり、そしてもし宛て先ポートが１つだけであれば、必要に応じてパケットの残存部分のいずれかの部分あるいは全部について、同時リード及びライト動作を実行することができる。

【００４９】

もしＰＯＲＴ＿ＮＯ信号が同じであるが、２つの異なったポート間であり、従って２つの異なったＱＣデバイス２０２間であるならば、特殊なタイプの同時リード及びライト・サイクルが実行される。図９ではこのケースも図解しているが、サイクル全体を通してＰＯＲＴ＿ＮＯ信号が不変のままであるという点が例外である。ＰＯＲＴ＿ＮＯ信号が変わらないので、ラッチ３２４、３２６は不要である。従って、このタイプの同時サイクルは２つの異なるＬ６４３８１デバイス間で、外部にラッチや選択ロジックを必要とせずに実行することができる。ＥＰＳＭ２１０は、送信元（ソース）と宛て先のポート間でＰＯＲＴ＿ＮＯ信号が等しいこと、および２つの異なったＱＣデバイス２０２が用いられることを判断してから、説明したように同時サイクルを実行する。

【００５０】

図９に図解されているように、２回目の同時リード及びライト転送は６番目のＣＬＫサイクルで発生し、ＰＯＲＴ＿ＮＯ［１：０］信号が７番目、８番目および９番目のサイクルにおいて、それぞれ、リード・モード、リード・ポート番号、およびライト・ポート番号でアサートされる。それに応答して、ＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊信号は７番目のＣＬＫサイクルに対してデアサート（de-assert）される。同様に、ＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号は８番目のＣＬＫサイクルに対してデアサートされる。この２回目の同時サイクルは、同一データ・パケットの続きのない連続したデータを供給するための最初の同時サイクルの続きか、あるいはまったく異なったデータ・パケットの開始のいずれかである。同一パケットの連続したデータについては、ソースおよび宛て先のポートは同じである。しかし、ソース・ポートまたは宛て先ポートあるいはその両方は、異なるパケットのデータを転送する２回目の同時サイクルでは同一のものではないこともある。

【００５１】

図１０は、ＨＳＢ２０６上で同時リード及びライト・サイクルを実行する手順を示すフローチャートである。最初のステップ３３０で、ＥＰＳＭ２１０は、ソース・ポートと宛て先ポートの間でのＨＳＢ２０６上での同時リード及びライト・サイクルが実行可能かどうかを判断する。ＥＰＳＭ２１０は、それから次のステップ３３２で、ソース・ポートを識別するための適当な信号をアサートする。これは、ＨＳＢ２０６上でＰＯＲＴ＿ＮＯ信号

を用いてソースまたは「リード」ポートの番号をアサートすることによって、及び適当なＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊信号をアサートすることによって行われる。次のステップ３３４では、識別されたソース・ポート・デバイスがその識別（アイデンティフィケーション）信号を検知もしくは格納する。ラッチを伴わない特殊な同時サイクルでは、ＱＣデバイス２０２がＨＳＢ２０６上でＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊信号を検知し、続いてＰＯＲＴ＿ＮＯ信号を検知して、リード・サイクルの準備を開始する。ラッチを用いる一般的な同時サイクルでは、指示されたＱＣデバイス２０２あるいはＴＰＩ２２０がステップ３３４でリード・ポート番号をラッチし、リード・サイクルの準備を開始する。

【００５２】

次のステップ３３６では、ＥＰＳＭ２１０は宛て先ポートを識別するための適当な信号をアサートする。特殊な同時サイクルでは、ＥＰＳＭ２１０は適当なＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号をアサートし、同じＰＯＲＴ＿ＮＯ信号を維持する。一般の場合では、ステップ３３６において、ＥＰＳＭ２１０はまた、ＨＳＢ２０６上に宛て先または「ライト」ポート番号を適当なＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号とともにアサートする。続くステップ３３８では、識別された宛て先ポート・デバイスがその識別信号を検知もしくは格納する。ラッチを伴わない特殊な同時サイクルでは、示されたＱＣデバイス２０２がＨＳＢ２０６上でＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号を検知し、続いてＰＯＲＴ＿ＮＯ信号を検知して、ライト・サイクルの準備を開始する。一般的な場合では、指示されたＱＣデバイス２０２あるいはＴＰＩ２２０が、ステップ３３８、で宛て先またはライト・ポート番号をラッチする。最後に、同時リード及びライト・サイクルのステップ３４０で、ここで指示されたソース・ポートがＨＳＢ２０６上にデータを送出し、指示された宛て先ポートがＨＳＢ２０６からデータを読み取る。

【００５３】

同時リード及びライト動作は、パケット・データの各転送にただ１つのバスしか必要としないため、最速タイプのデータ転送サイクルである。後に詳述するように、通常のＣＴモードの動作では少なくとも２回の転送が必要である。すなわち、１つはソース・ポートからＥＰＳＭ２１０へ、そしてもう１つはＥＰＳＭ２１０から宛て先ポートへの転送であって、これは同じデータに対してＨＳＢ２０６上で２つの別のサイクルが必要となる。同時リード及びライト・サイクルは、ＨＳＢ２０６上で同一のデータについて１回で直接の転送を要し、それによりＨＳＢ２０６の帯域幅が増大する。その他、幾つかの暫定的なＣＴや蓄積転送（ＳｎＦ）モードを含むより遅いモードもあり、その場合、パケット・データはメモリ２１２に書き込まれてから宛て先ポートに転送される。

【００５４】

次に図１１を参照すると、ＥＰＳＭ２１０の簡略なブロック図で、データの流れとコンフィギュレーション・レジスタを図解している。ＥＰＳＭ２１０は、ＨＳＢコントローラ・ブロック（ＨＣＢ）４０２、メモリ・コントローラ・ブロック（ＭＣＢ）４０４、およびプロセッサ制御ブロック（ＰＣＢ）４０６という３つの主要セクションを含む。ＱＣインタフェース４１０はＨＳＢ２０６をＥＰＳＭ２１０のＨＣＢ４０２に結合する。ＱＣインタフェース４１０の他側には１組のバッファ、すなわちＦＩＦＯ４１２が結合されており、これらのＦＩＦＯ４１２には受信ＦＩＦＯ、送信ＦＩＦＯ、および本明細書で後に詳述するカットスルーＦＩＦＯが含まれる。ＦＩＦＯ４１２の他側（図１２のＣＴバッファ５２８を除く）は、ＭＣＢインタフェース４１４を介してＭＣＢ４０４に結合されており、そのＭＣＢインタフェース４１４は適当なバス４２０を介してＭＣＢ４０４内のＨＣＢインタフェース４１８に結合されている。ＨＣＢインタフェース４１８はさらにメモリ・インタフェース４２２に結合され、メモリ・インタフェース４２２はメモリ・バス２１４を介してメモリ２１２に結合される。メモリ・インタフェース４２２はさらにＰＣＢインタフェース４２４の一側に結合されており、そのＰＣＢインタフェース４２４の他側は適当なＭＣＢバス４２８を介してＰＣＢ４０６内のＭＣＢインタフェース４２６の一側に結合されている。ＭＣＢインタフェース４２６の他側は１組のＦＩＦＯ４３０の一側に結合されており、ＦＩＦＯ４３０がさらにＰＣＢ４０６内のＣＰＵインタフェース４３２に結合

されている。ＣＰＵインタフェース４３２はＱＣ／ＣＰＵバス２０４およびＣＰＵバス２１８に結合される。ＣＰＵインタフェース４３２はさらにＰＣＢ４０６内の第２の組のＦＩＦＯ４３４の一側に結合されており、ＦＩＦＯ４３４の他側はＱＣ／ＨＣＢインタフェース４３６に結合されている。ＱＣ／ＨＣＢインタフェース４３６の他側は適当なＨＣＢバス４３８を介してＱＣインタフェース４１０に結合されている。

【００５５】

ＰＣＢ４０６とＣＰＵ２３０に関連するＨＣＢバス４３８のＰＣＢ＿ＢＵＦ＿ＡＶＡＩＬ＊、ＰＣＢ＿ＰＫＴ＿ＡＶＡＩＬ＊、ＰＣＢ＿ＲＤ＿ＯＵＴ＿ＰＫＴ＊、およびＰＣＢ＿ＷＲ＿ＩＮ＿ＰＫＴ＊信号は、それぞれ、ＢＵＦ＿ＡＶＡＩＬｍ＊、ＰＫＴ＿ＡＶＡＩＬｍ＊、ＲＥＡＤ＿ＯＵＴ＿ＰＫＴｍ＊、およびＷＲＩＴＥ＿ＩＮ＿ＰＫＴｍ＊信号に含まれていることに留意されたい。示した本実施例において、ＨＣＢバス４３８はＨＳＢ２０６と類似しており、本質的にはＥＰＳＭ２１０内のＨＳＢ２０６の内部バージョンである。ＰＣＢ４０６は、ＨＣＢ４０２に対してポート１０４のそれぞれおよびＴＰＩ２２０と同様の働きをする。このようにして、ＣＰＵ２３０はＰＣＢ４０６の動作を通じて、ＨＣＢ４０２に対する追加的なポート（ＰＯＲＴ２８）として動作する。

【００５６】

ＣＰＵインタフェース４３２はバス４４２を介してレジスタ・インタフェース４４０に結合され、レジスタ・インタフェース４４０はさらにレジスタ・バス４４４に結合される。レジスタ・バス４４４はＨＣＢ４０２内の１組のＨＣＢコンフィギュレーション・レジスタ、およびＭＣＢ４０４内の１組のＭＣＢコンフィギュレーション・レジスタ４４８に結合される。このようにして、ＣＰＵ２３０は、ＣＰＵインタフェース４３２とレジスタ・インタフェース４４０を介し、ＨＣＢコンフィギュレーション・レジスタ４４６およびＭＣＢコンフィギュレーション・レジスタ４４８の両方のレジスタの初期設定とプログラミングを行う。

【００５７】

ＭＣＢコンフィギュレーション・レジスタ４４８は、ポートおよびメモリ２１２に関連する相当な量のコンフィギュレーション情報の格納に使用される。例えば、ＭＣＢコンフィギュレーション・レジスタ４４８は、各ポートが学習（ＬＲＮ）状態か転送（ＦＷＤ）状態かブロック（閉じた）（ＢＬＫ）状態か聴取（ＬＳＴ）状態か又はディスエーブル（ＤＩＳ）状態かを示すポート状態情報、メモリ・セクタ情報、メモリ・バス２１４のバス使用情報、ドロップされたパケットの数、ハッシュ・テーブル定義、メモリ・スレッショルド、ＢＣスレッショルド、もしあれば機密保護ポートのアイデンティフィケーション、メモリ制御情報、ＭＣＢ割り込みソース・ビット、割り込みマスクビット、ポーリング・ソース・ビットなどを含む。

【００５８】

ＥＰＳＭ２１０の説明では、ＣＰＵ２３０はコンフィギュレーションおよび制御の目的で、ＱＣデバイス２０２およびメモリ２１２にアクセスできることを述べている。ＥＰＳＭ２１０とのＨＳＢ２０６を用いる主たるデータ・フローはＦＩＦＯ４１２とメモリ２１２を通じてのものであるが、ＨＳＢ２０６とＣＰＵ２３０との間でも、ＨＣＢバス４３８およびＥＰＳＭ２１０の関連するＦＩＦＯ及びインタフェースを介したデータ・フローも発生する。

【００５９】

次に図１２を参照すると、ＨＣＢ４０２の詳細なブロック図が示されている。ＨＣＢバス４３８はＰＣＢ４０６にインタフェースするためのＨＳＢ２０６の内部バージョンであり、そこでバス２０６と４３８を一括してＨＳＢ２０６と呼称する。ポーリング・ロジック５０１は、ＨＳＢ２０６、１組のローカル・レジスタ５０６、およびＨＣＢコンフィギュレーション・レジスタ４４６に結合されている。ポーリング・ロジック５０１は、ＣＬＫ信号を受信し、ポート１０４、１１０およびＰＣＢ４０６を問い合わせるべくＱＣデバイス２０２およびＴＰＩ２２０へのＳＴＲＯＢＥ＊信号を周期的にアサートする。ポーリング・ロジック５０１は、ＱＣデバイス２０２およびＴＰＩ２２０からの多重化されたＰＫ

Ｔ＿ＡＶＡＩＬｍ＊およびＢＵＦ＿ＡＶＡＩＬｍ＊信号をモニタする。ここで、各ＱＣデバイス２０２およびＴＰＩ２２０は、前述したように、その４つのポート１０４、１１０の状態をそれぞれ供給する。ＴＰＩ２２０はＰＫＴ＿ＡＶＡＩＬ［６］＊およびＢＵＦ＿ＡＶＡＩＬ［６］＊信号で応答し、ＰＣＢ４０６はＰＫＴ＿ＡＶＡＩＬ［７］＊およびＢＵＦ＿ＡＶＡＩＬ［７］＊信号で応答する。

【００６０】

ポーリング・ロジック５０１は受信（ＲＸ）ポーリング状態マシン５０２を含み、これでＰＫＴ＿ＡＶＡＩＬｍ＊信号を見直し（リビューし、review）、レジスタ５０６内の受信リスト（RECEIVE LIST）５０９を更新する。同様に、ポーリング・ロジック５０１は送信（ＴＸ）ポーリング状態マシン５０３を含み、これでＢＵＦ＿ＡＶＡＩＬｍ＊信号を見直し、レジスタ５０６内の送信リスト（TRANSMIT LIST）５１０を更新する。もしＨＣＢコンフィギュレーション・レジスタ４４６におけるＷＴＰＲＩＯＲＩＴＹフラグがＣＰＵ２３０によってセットされれば、ＲＸポーリング状態マシン５０２およびＴＸポーリング状態マシン５０３の両方は、ＨＣＢコンフィギュレーション・レジスタ４４６内の１組のウエイト・ファクタ（WEIGHT FACTORS）５０８使用して、後に詳述するように、それぞれ受信リスト５０９および送信リスト５１０をプログラミングする。ＨＣＢコンフィギュレーション・レジスタ４４６はまた１組のＣＴ＿ＳＮＦレジスタ５０７を含んでおり、これがＣＰＵ２３０にプログラミングされ、対応するポートがソース・ポートあるいは宛て先ポートである場合、所望される動作モードをＣＴとＳｎＦとの間で決定する。

【００６１】

レジスタ５０６は、ラッチ、フリップ・フロップ、スタティックＲＡＭ（ＳＲＡＭ）、ＤＲＡＭデバイスなどのような、ＥＰＳＭ２１０の実施に従っての任意の様式で実施され、複数の状態および制御（コントロール）のレジスタ又はバッファを含む。受信リスト５０９は、各ポートの相対的受信状態（ステータス）および優先度（優先順位）を示す複数のレジスタ値を含む。同様に、送信リスト５１０は、各ポートの相対的送信ステータスおよび優先度を示す複数のレジスタ値を含む。ＰＲカウント（RPCOUNT）・レジスタ５１１ａは、各ポートが外部のネットワーク・デバイスからパケット・データを受信したとき、その受信ポートに相対的な受信優先順位を割り当てるためにＲＸポーリング状態マシン５０２によって使用されるＰＲカウント（RPCOUNT）番号を格納している。もしくは、ＲＸポーリング状態マシン５０２はウエイト・ファクタ５０８からの対応するウエイト（重み）・ファクタを使用する。同様に、ＴＰカウント（TPCOUNT）・レジスタ５１１ｂは、ポートによって外部のネットワーク・デバイスへ送信できるパケット・データがあり、ポートが送信のためのデータを収容可能なとき、そのポートに相対的な送信優先順位を割り当てるためにＴＸポーリング状態マシン５０３によって使用されるＴＰカウント（TPCOUNT）番号を格納する。もしくは、ＴＸポーリング状態マシン５０２はウエイト・ファクタ５０８からの対応するウエイト・ファクタを使用する。相対的アービトレーション・カウント番号ＲＸニューカウント（RXNEWCNT）、ＲＸＡＣＴカウント（RXACTCNT）、ＴＸニューカウント（TXNEWCNT）およびＹＸＣＴカウント（TXCTCNT）は、それぞれ、レジスタＲＸニューカウント５１１ｃ、ＲＸＡＣＴカウント５１１ｄ、ＴＸニューカウント５１１ｅおよびＴＸＣＴカウント５１１ｆに格納される。

【００６２】

ＨＣＢ４０２は、レジスタ５０６および４４６内のデータを調べてＨＳＢ２０６上で実行されたサイクルのタイプを判断するために結合されたアービトレーション・ロジック５０４を含む。ＨＳＢコントローラ５０５は、ＥＰＳＭ２１０とＨＳＢ２０６との間のデータ・フローをコントロールするために、ＨＳＢ２０６上で実行される各サイクルを実行及び制御する。ＨＳＢコントローラ５０５は、状態ビットを変更するためにレジスタ５０６に結合される。ＨＳＢコントローラ５０５は、各サイクルのタイプのアイデンティフィケーションをアービトレーション・ロジック５０４から受け取る。アービトレーション・ロジック５０４は、新パケット受信（ＲＸ　ＮＷ）アービタ５１３、受信アクティブ（ＲＸＡＣＴ）アービタ５１４、新パケット送信（ＴＸ　ＮＷ）アービタ５１５および送信カッ

トスルー（ＴＸ　ＣＴ）アービタ５１６の全部で４つのデータ・アービタに結合されたメイン（MAIN）・アービタ５１２を含む。メイン・アービタ５１２は、一般に、ＲＸ　ＮＷアービタ５１３、ＲＸ　ＡＣＴアービタ５１４、ＴＸ　ＮＷアービタ５１５、およびＴＸ　ＣＴアービタ５１６の間で選択を行い、各アービタは調停（仲裁）を行って次のサイクルを決める。メイン・アービタ５１２は、必要に応じて条件に適ったいずれかの優先順位スキームを用いる。例えば、示した実施例においては、メイン・アービタ５１２はラウンドロビン優先順位スキームを採用する。

【００６３】

ＦＩＦＯ４１２は任意の望ましい様式で実施される。示した実施例においては、２つの受信バッファ、ＲＸ　ＢＵＦ５２０および５２２でＲＸ　ＦＩＦＯを実現しており、データは１つのバッファへの書き込み中に他のバッファから読み出され、また、その逆も行われる。また、２つの送信バッファ、ＴＸ　ＢＵＦ５２４および５２６が用意されており、ＲＸ　ＢＵＦ５２０および５２２と同様に動作する。ＦＩＦＯ４１２は、少なくとも１つのカットスルー・バッファ、ＣＴ　ＢＵＦ５２８も含む。ＲＸ　ＢＵＦ５２０および５２２は両方とも６４バイトのバッファであり、それぞれが両方向のデータ・フローを実現するためにＨＳＢ２０６との双方向データ・インタフェース、およびＲＸ　ＭＣＢインタフェース５３０を介してＭＣＢ４０４にデータを送るための単向インタフェースを含む。ＴＸ　ＢＵＦ５２４および５２６は両方とも６４バイトのバッファであり、ＨＳＢ２０６とＴＸ　ＭＣＢインタフェース５３１との間に結合されている。ＴＸ　ＢＵＦ５２４および５２６は、ＴＸ　ＭＣＢインタフェース５３１を介してＭＣＢ４０４からデータを受け取り、データをＨＳＢ２０６に送る。ＣＴ　ＢＵＦ５２８は６４バイトのバッファであり、ＨＳＢ２０６との双方向インタフェースを有する。ＦＩＦＯコントロール・ブロック５２９は、ＦＩＦＯ５２０、５２２、５２４、および５２６のデータ・フローを制御するため、及びＲＸ　ＭＣＢインタフェースおよびＴＸ　ＭＣＢインタフェース５３０、５３１を介してアサートされた特定の状態信号を検知するため、及び後に詳述するように、レジスタ５０６内の特定のビットをセットするために、レジスタ５０６、ＨＳＢコントローラ５０５、ＲＸ　ＢＵＦ５２０と５２２、ＴＸ　ＢＵＦ５２４と５２６、ＣＴ　ＢＵＦ５２８、ＲＸ　ＭＣＢインタフェース５３０およびＴＸ　ＭＣＢインタフェース５３１に結合している。

【００６４】

バス４２０は、ＲＸ　ＭＣＢインタフェース５３０、ＴＸ　ＭＣＢインタフェース５３１、ハッシュ要求ロジック（ハッシュ・リクエスト・ロジック）及びＭＣＢインタフェース（HASH REQ LOGICと呼ぶ）５３２、および送信アービタ要求ロジック（ＴＸ　ＡＲＢリクエスト・ロジック）及びＭＣＢインタフェース（TX ARB REQ LOGICと呼ぶ）５３３を介してＨＣＢ４０２をＭＣＢ４０４にインタフェースするための複数のデータおよび制御信号を含む。ＨＳＢコントローラ５０５は、ポート０～ポート２８の１つからのそれぞれの新しいパケットのヘッダをＲＸ　ＢＵＦ５２０と５２２の１つに、及びＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ５３２にコピーする。ヘッダは、サイズが少なくとも３つのＤＷＯＲＤ（それぞれ３２ビット）すなわち９６ビットであり、ソースと宛て先の両方のＭＡＣアドレスを含む。ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ５３２は、ＭＣＢ４０４によって実行されるハッシュの手順を要求し、適当なビットをレジスタ５０６にセットする。このハッシュ手順は、パケットに対して取られる適当な動作を決定するために行われる。

【００６５】

示した実施例において、新しパケットのヘッダを受け取った後、ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ５３２はＭＣＢ４０４へ信号ＨＡＳＨ＿ＲＥＱ＊をアサートし、ＨＡＳＨ＿ＤＡ＿ＳＡ［１５：０］信号上に４８ビットのＭＡＣの宛て先およびソース・アドレスおよび８ビットのソース・ポート番号を多重化する。ＭＣＢ４０４はＨＡＳＨ＿ＲＥＱ＊信号を検知し、ハッシュ手順を実行し、そしてＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ５３２へ信号ＨＡＳＨ＿ＤＯＮＥ＊をアサートする。ＭＣＢ４０４は、状況が許せば、信号ＨＡＳＨ＿ＤＳＴＰＲＴ［４：０］、ＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］、及びＨＡＳＨ＿ＢＰ＊もアサー

トする。ＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］信号は次の４種類の結果の１つを表示する。すなわち、それらは、パケットをドロップするための００ｂ＝ＤＲＯＰ＿ＰＫＴ（ｂは２進数を示す）、ブロードキャスト（同報通信）（ＢＣ）パケットに対しての０１ｂ＝ＧＲＯＵＰ＿ＢＣ、宛て先ポートが未知であり、従ってＢＣパケットであるという１０ｂ＝ＭＩＳＳ＿ＢＣ、および単一の宛て先ポートへのユニキャスト・パケットを示す１１ｂ＝ＦＯＲＷＡＲＤ＿ＰＫＴである。もしＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］＝ＦＯＲＷＡＲＤ＿ＰＫＴであれば、ＨＡＳＨ＿ＤＳＴＰＲＴ［４：０］信号が、そのパケットの宛て先ポートを指定する２進数のポート番号とともにアサートされる。ＨＡＳＨ＿ＢＰ＊信号は、もし背圧がイネーブルとなっていて適用可能であれば、ＭＣＢ４０４が判断したメモリ２１２におけるスレッショルド・オーバーフロー状態に起因して、背圧を示すためにアサートされる。

【００６６】

一定のしきい（スレッショルド）値が、メモリ２１２全体に対して、特定のタイプのパケット（例えばＢＣパケット）に対して、及びポートごとに設定される。しきい値に達したとき、従ってメモリ２１２にもう１つのパケットを入れるとスレッショルド条件を侵すことになる場合、そのパケットのドロップの如何はネットワーク・スイッチ１０２が決定する。送信側のデバイスは最終的にそのパケットがドロップされたことを検知し、そのパケットを再送信する。或るスレッショルド条件に違反があった場合、もし背圧がイネーブルとなっていてソース・ポートが半二重モードで動作していれば、ＨＡＳＨ＿ＢＰ＊信号がアサートされる。

【００６７】

ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ５３２はＨＡＳＨ＿ＢＰ＊信号を検知して、例えばソース・ポートと宛て先ポートが同じかのように、ＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］＝ＤＲＯＰ＿ＰＫＴであるかどうかを判断する。もしＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］＝ＤＲＯＰ＿ＰＫＴであれば、そのパケットはドロップされるべきものであるから、それ以上の動作は不要である。もしＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］とＤＲＯＰ＿ＰＫＴが等しくなければ、ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ５３２はＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］＝ＦＯＲＷＡＲＤ＿ＰＫＴであるかどうかを判断し、そのパケットはＣＴ　ＢＵＦ５２８を介してＣＴモードで転送されることになり、可能性としてメモリ２１２が避けられる。もし宛て先ポートが使用中（ビジー）であるか、またはもしＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］がパケットのドロップあるいは転送を指示しなければ、ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ５３２が、データを受信するポートに対して背圧サイクルを実行するようＨＳＢコントローラ５０５に指示する。

【００６８】

ＳｎＦ動作の間、ＥＰＳＭ２１０は、パケットのいずれかの部分を宛て先ポートへ送信する前に、パケット全体を受信してメモリ２１２に格納する。パケットの受信が完了後であって、もし宛て先ポートが既知であれば、そのパケットは、使用されている特定のアービトレーション・スキームに従って、可能なときに宛て先ポートに送られる。ＣＴ動作を適用する場合、両方のポートがＣＴ＿ＳＮＦレジスタ５０７内でＣＴモードにプリセットされ、両ポートが同一速度で動作し、そして宛て先ポートのＴＢＵＳ設定がソース・ポートのＴＢＵＳ設定と比べて等しい又は大きい。１００Ｍｂｐｓのイーサネット・ポート、ポート２４～ポート２７の実現にＴＬＡＮ２２６を使用するここに示した特定の実施例において、ＴＬＡＮは送信に先立ってパケット全体のサイズが必要であるため、ポート２４～ポート２７についてＣＴモードは実行されない。また、示した実施例ではＴＢＵＳの値が等しいことが要件である。本発明は、これら様々な設計上の問題には制約されない。ＣＴモードでの動作中、ＥＰＳＭ２１０は指示された宛て先ポートに対して、もしこれがビジーでなければ、データを送信するために、データを適当なＱＣデバイス２０２に供給する。パケット・データはメモリ２１２には転送されず、ソース・ポートと宛て先ポートの間のＦＩＦＯ４１２を通じて緩衝格納（バッファ記憶）される。

【００６９】

もし受信したパケットの先頭で受信先のポートがビジーであれば、データは暫定的（interim）ＣＴ動作モードに従って、ソース・ポートと宛て先ポートの間のメモリ２１２内でバッファされる。しかし、パケット部は宛て先ポートによる送信に直ちに使用可能であって、宛て先ポートへの転送に、パケット全体の受け取り完了を待つ必要がない。安全対策として、暫定的ＣＴ動作モードを無効とし、その特定のパケットのための動作を次のパケットのＳｎＦモードに切り替えるメカニズムが適用できる。

【００７０】

ＣＴモードでのパケット転送中に、例えば宛て先ポートの停止のような何らかの理由で宛て先ポートがそれ以上のデータの受信をできなくなった場合、動作はミッドパケット（mid-packet）暫定ＣＴモードに切り替えられる。ミッドパケット暫定ＣＴモードの間、ＦＩＦＯ４１２内のパケット・データはメモリ２１２へ送られ、その後宛て先ポートがさらにデータを受信することができるときにパケット・データがそのポートに送られる。他の後続の受信されたパケットが、同じ停止したポートによる送信のために他のポートによって受信され、これら後続のパケットはそのポートに対する対応する送信チェーン内に入れられるので、ミッドパケット暫定ＣＴモードに切り替えられたパケットの残りの部分は順序の適性化を意図してその送信チェーンの先頭に置かれることに留意されたい。

【００７１】

もう１つのモードは適応（アダプティブ）ＳｎＦモードと呼ばれる。パケットがＣＴ動作モードで転送されている間、ＣＰＵ２３０は、ポート１０４、１１０、およびＰＣＢ４０６のいずれか１つまたはそれ以上に「ラント（runt）」、「オーバーラン」、「ジャバー（jabber）」、遅刻衝突（レイト・コリージョン、late collision）、ＦＣＳエラーなどの誤りが相当回数発生するかどうかを判定するために、それらのアクティビティの監視及び追跡を行っている。ラントはデータが一定の最少量に満たないパケットで、示した本実施例におけるその最小サイズは６４バイトである。オーバーランはデータが一定の最多量より多いパケットで、イーサネット標準に従って示されている本実施例におけるその最大サイズは１５１８バイトである。ジャバーはサイズが最大サイズ（イーサネットではの１５１８バイト）を超えており、無効なＣＲＣ（巡回冗長検査、（Cyclic Redundancy Check））値が入っているパケットである。通常、このような誤りのあるパケットはドロップされ、システム内に伝播されることはない。適応ＳｎＦモードについては、もしポート１０４がＣＴモードで動作していて、このような誤りの発生がＣＰＵ２３０が判断するところでは頻繁であると、ＣＰＵ２３０は誤りが訂正または除去されるまで、そのポートにプリセットされているモードをＣＴ動作からＳｎＦ動作に切り替える。各ＴＬＡＮ２２６のポート１１０の動作も同様であるが、パケット・データがＴＰＩ２２０を通じてＨＳＢ２０６を介してＥＰＳＭ２１０に入り、送信の前にメモリ２１２に格納されるという点が異なる。ＴＰＩ２２０は、実際上ＰＣＩバス２２２とＨＳＢ２０６との間のブリッジとして動作する。ＴＬＡＮ２２６が外部のネットワークにパケットを送信する前にはパケット全体の長さが必要であり、従って、各パケットはＴＬＡＮ２２６の１つによって再送信される前に、そのパケットが受信されてその全体がメモリ２１２に格納される。さらに、ＱＣデバイス２０２による送信用にＴＬＡＮ２２６が受け取るデータ、およびＴＬＡＮ２２６による送信のためにＱＣデバイス２０２が受け取るデータは、示した実施例におけるデバイス２０２と２２６の間の速度の差が大きいため、ＳｎＦモードで処理されてメモリ２１２に格納される。

【００７２】

ＲＸ　ＭＣＢインタフェース５３０は、パケット・データがＲＸ　ＢＵＦ５２０、５２２の１つに入っていてメモリ２１２への転送準備完了状態にあるとき、ＭＣＢ４０４へＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊信号をアサートする。パケット・データはＨＣＢ４０２から転送され、メモリ・データ出力バスＭｅｍＤａｔａＯｕｔあるいはＭＤＯ［３１：０］を介してＭＣＢ４０４へ転送される。スタティック信号ＭＥＭ＿ＥＤＯは、メモリ２１２のタイプがＥＤＯか同期ＤＲＡＭであればアサートされ、ＦＰＭ　ＤＲＡＭであればアサートされない。ＲＸ　ＭＣＢインタフェース５３０は、ＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊信号を適宜

にアサートしている間、他のいくつかの信号もアサートする。特に、ＲＸ　ＭＣＢインタフェース５３０は、１ＣＬＫサイクルに対してＲＸ＿ＳＲＣ＿ＤＳＴ［４：０］信号上にソース・ポート番号を多重化し、続いて、ＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊信号をアサートしているときに、次のＣＬＫサイクルの間に、もし既知であれば、宛て先ポート番号を多重化する。また、ＲＸ　ＭＣＢインタフェース５３０は、選択されたＲＸ　ＢＵＦ５２０、５２２内の、信号ＲＸ＿ＣＮＴ［５：０］上のＤＷＯＲＤの数（マイナス１ＤＷＯＲＤ）をアサートする。

【００７３】

ＲＸ　ＭＣＢインタフェース５３０は、もしデータがパケットの始まりであれば信号ＲＸ＿ＳＯＰ＊をＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊信号とともにアサートし、もしデータがそのパケットの終わりであれば信号ＲＸ＿ＥＯＰ＊をＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊信号とともにアサートする。ＲＸ　ＭＣＢインタフェース５３０は、パケットがＣＴモードで転送中であるが、暫定ＣＴやミッドパケットＣＴモードの場合のようにメモリ２１２で緩衝格納されていれば、信号ＲＸ＿ＣＵＴ＿ＴＨＲＵ＿ＳＯＰ＊を信号ＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊およびＲＸ＿ＳＯＰ＊とともにアサートする。特に、もし（！ＲＸ＿ＣＵＴ＿ＴＨＲＵ＿ＳＯＰ＊　＆　！ＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊　＆　！ＲＸ＿ＳＯＰ＊）であれば暫定ＣＴ（全パケット）が指示され、もし（！ＲＸ＿ＣＵＴ＿ＴＨＲＵ＿ＳＯＰ＊　＆　！ＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊　＆　ＲＸ＿ＳＯＰ＊）であれば暫定ＣＴミッドパケットが指示される。ＲＸ　ＭＣＢインタフェース５３０は、もし宛て先アドレスが未知であり、そしてパケットがＢＣパケットであれば、信号ＲＸ＿ＭＩＳＳ＿ＢＣ＊をＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊およびＲＸ＿ＳＯＰ＊信号とともにアサートする。ＲＸ　ＭＣＢインタフェース５３０は、もしヘッダ内でＧＲＯＵＰ（グループ）ビットがセットされていれば、従って、また。パケットがＢＣパケットであれば、信号ＲＸ＿ＧＲＯＵＰ＿ＢＣ＊をＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊およびＲＸ＿ＳＯＰ＊信号とともにアサートする。ＲＸ　ＭＣＢインタフェース５３０は、信号ＲＸ＿ＥＮＤ＿ＢＹＴＥ［１：０］をＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊およびＲＸ＿ＥＯＰ＊信号とともにアサートし、パケット内の最終バイトのバイト・レーン（lane）を示す。

【００７４】

ＲＸ　ＭＣＢインタフェース５３０は、ソース・ポートが送信中にＡＢＯＲＴ＿ＯＵＴ＊信号をアサートしてパケット内における誤りの検知と表示を行えば、信号ＲＸ＿ＥＲＲＯＲ＊をＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊およびＲＸ＿ＥＯＰ＊信号とともにアサートする。ＦＩＦＯオーバーラン、ラント・パケット、オーバーサイズのパケット、フレーム・チェック・シーケンス（ＦＣＳ）・エラー、あるいはフェーズ・ロックト・ループ（ＰＬＬ）エラーの検知のように、各種のエラー状況がポート１０４、１１０によってチェックされる。もしＲＸ＿ＥＲＲＯＲ＊信号がアサートされると、ネットワーク・スイッチ１０２は、パケットがＳｎＦモードで転送中であれば、そのパケットをドロップする。

【００７５】

ＭＣＢ４０４は、アサートされたＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊信号を検知した後、そして前述のようにＲＸ＿ＰＫＴ＿ＡＶＡＩＬ＊信号でアサートされた関連の信号をラッチした後に、ＨＣＢ４０２へのＲＸ＿ＡＣＫ＊信号をアサートする。ＭＣＢ４０４は、次のＤＷＯＲＤのデータを受け取れる状態に入ったときＲＸ＿ＳＴＢ＊信号をアサートする。ＭＣＢ４０４は、ＨＣＢ４０２がデータを要求する可能性があると判断したとき、信号ＲＸ＿ＰＫＴ＿ＣＯＭＰＬＥＴＥ＊をアサートする。とりわけＭＣＢ４０４は、ＣＴモードのパケットに対してＨＣＢ４０２によってアサートされたＲＸ＿ＳＯＰ＊信号を検知した後にＲＸ＿ＰＫＴ＿ＣＯＭＰＬＥＴＥ＊信号をアサートする。またＭＣＢ４０４は、ＳｎＦモードのパケットに対してＨＣＢ４０２によってアサートされたＲＸ＿ＥＯＰ＊信号を検知した後にＲＸ＿ＰＫＴ＿ＣＯＭＰＬＥＴＥ＊信号をアサートする。ＭＣＢ４０４は、ＳｎＦパケットに関してＲＸ＿ＥＲＲＯＲ＊信号がアサートされていた場合（ＲＸ＿ＳＯＰ＊信号とともにアサートされていないＲＸ＿ＣＵＴ＿ＴＨＲＵ＊信号で示される状態）には、ＲＸ＿ＰＫＴ＿ＣＯＭＰＬＥＴＥ＊信号をアサートしない。ＭＣＢ４０４は、ＭＣＢ４

０４が判断したメモリ２１２におけるオーバーフロー状態に起因してパケットがドロップされた場合、ＲＸ＿ＰＫＴ＿ＣＯＭＰＬＥＴＥ＊信号の代わりにＨＣＢ４０２へ信号ＲＸ＿ＰＫＴ＿ＡＢＯＲＴＥＤ＊をアサートする。

【００７６】

ＴＸ　ＡＲＢ　ＲＥＱ　ＬＯＧＩＣ５３３は、使用可能な宛て先ポートによる送信のためのデータのメモリ２１２からの取り出し要求を、アービトレーション・ロジック５０４から受け取る。この要求は、一般にＴＸ　ＮＷアービタ５１５から出される。ＴＸ　ＡＲＢ　ＲＥＱ　ＬＯＧＩＣ５３３は、応答してＭＣＢ４０４へ送信要求信号ＴＸ＿ＡＲＢ＿ＲＥＱ＊をアサートし、一方、信号ＴＸ＿ＡＲＢ＿ＰＯＲＴ［４：０］上に宛て先ポート番号を、及び信号ＴＸ＿ＡＲＢ＿ＸＳＩＺＥ［２：０］に各データ部の最大転送長をアサートする。ＴＸ　ＢＵＦ５２４および５２６について、最大転送長は、０００ｂ＝１６バイト、００１ｂ＝３２バイト、０１０ｂ＝６４バイト、０１１＝１２８バイト、および１００＝２５６バイトとして定義される。ＭＣＢ４０４はこれらの値をラッチし、ＴＸ　ＡＲＢ　ＲＥＱ　ＬＯＧＩＣ５３３へ肯定応答（アクノレッジ）信号ＴＸ＿ＡＲＢ＿ＡＣＫ＊をアサートする。ＭＣＢ４０４は、要求されたデータをメモリ２１２から取り出し、そのデータをＴＸ　ＢＵＦ５２４、５２６の１つに書き込む。

【００７７】

データはメモリ・データ入力バスＭｅｍＤａｔａＩｎまたはＭＤＩ［３１：０］を介してＨＣＢ　４０２内のＴＸ　ＢＵＦ　５２４、５２６へ転送される。ＴＸＭＣＢインタフェース５３１は、ＴＸ　ＢＵＦ　５２４および５２６のいずれかがＭＣＢ４０４からのデータの受け取りに使用可能であるとＦＩＦＯ制御ブロック５２９が判断したとき、ＴＸ＿ＢＵＦ＿ＡＶＡＩＬ＊をアサートする。ＭＣＢ４０４は、使用可能なＴＸ　ＢＵＦ　５２４あるいは５２６に格納すべくＨＣＢ４０２のＴＸ　ＭＣＢインタフェース５３１によるサンプリングの対象となるデータが存在するとき、ストローブ信号ＴＸ＿ＳＴＢ＊をアサートする。ＭＣＢ４０４は、データの特性を識別するためにＴＸ＿ＳＴＢ＊と同時にいくつかの信号もアサートする。特に、ＭＣＢ４０４はＴＸ＿ＳＴＢ＊信号とともに信号ＴＸ＿ＳＯＰ＊をアサートし、メモリ２１２からデータの始めを検出する。ＭＣＢ４０４はＴＸ＿ＳＴＢ＊信号とともにＴＸ＿ＡＩＦＣＳ＊信号をアサートし、送信元ポートがＣＰＵ　２３０を指示しているＰＣＢ４０６であるかどうかを判断する。ＭＣＢ４０４はＴＸ＿ＳＴＢ＊信号とともに信号ＴＸ＿ＣＮＴ［５：０］上の２進数をアサートする。ここで、ＴＸ＿ＣＮＴ［５：０］は選択したＴＸ　ＦＩＦＯに書き込むＤＷＯＲＤの数（マイナス１ＤＷＯＲＤ）を表す。ＭＣＢ４０４はＴＸ＿ＳＴＢ＊信号とともに信号ＴＸ＿ＥＯＰ＊をアサートし、メモリ２１２からパケットの終わりを検出する。ＭＣＢ　４０４はＴＸ＿ＥＯＰ＊およびＴＸ＿ＳＴＢ＊信号とともにバッファ・チェーン終結信号ＴＸ＿ＥＯＢＣもアサートし、メモリ２１２内に特定の受信先ポートに宛てたデータが無くなったかどうかを確認する。ＭＣＢ　４０４はＴＸ＿ＥＯＰ＊およびＴＸ＿ＳＴＢ＊信号とともにエンド・バイト信号もアサートしてパケット内の最終バイトのバイト・レーンを示す。

【００７８】

ＢＣパケットについては、ＭＣＢ４０４がＭＤＩ［３１：０］信号上のＢＣビットマップをアサートしつつ信号ＢＣ＿ＰＯＲＴ＿ＳＴＢ＊をアサートする。ＦＩＦＯ制御ブロック５２９は、ＢＣ＿ＰＯＲＴ＿ＳＴＢ＊信号がアサートされたことを検知し、ＭＤＩ［３１：０］信号をラッチして結果を内部のＢＣＢＩＴＭＡＰ［２８：０］レジスタ内に格納する。ＦＩＦＯ制御ブロック５２９は、ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　５１０内のメモリ・ビット配列ＴＸＭＥＭＣＹＣ［２８：０］のビットを設定するときにＢＣＢＩＴＭＡＰレジスタ内の値を用いる。

【００７９】

図１３は、レジスタ５０６に属するいくつかのレジスタの図解である。ＣＴ＿ＳＮＦレジスタ５０７は、プログラミング可能な送信元ポート・モードのビット配列ＳＲＣ　ＣＴ＿ＳＮＦ［２８：０］を含み、各ビットはそれぞれポートＰＯＲＴ２８からＰＯＲＴ０の１つに対応しており、対応するポートが送信元ポートである場合にＣＴとＳｎＦ間における

動作モードを指定するためＣＰＵ　２３０によってプログラミングされる。特に、特定のポートにＳＲＣ　ＣＴ＿ＳＮＦビットがセットされている場合、そのポートが送信元ポートとして機能するときの動作モードとしてはＣＴモードが望ましい。ＳＲＣ　ＣＴ＿ＳＮＦビットがクリアされている場合は、そのポートが送信元ポートとして機能するときの動作モードとしてはＳｎＦモードが望ましい。同様に、ＣＴ＿ＳＮＦレジスタ５０７は、プログラミング可能な受信先ポート・モードのビット配列ＤＥＳＴ＿ＣＴ＿ＳＮＦ［２８：０］を含み、各ビットはそれぞれポートＰＯＲＴ２８からＰＯＲＴ０の１つに対応しており、対応するポートがユニキャスト用の受信先ポートである場合にＣＴとＳｎＦ間における動作モードを指定するためＣＰＵ２３０によってプログラミングされる。ＣＴモードは、送信元と受信先の両方のポートがＣＴ＿ＳＮＦレジスタ５０７でＣＴモードに指定されている場合にのみ望ましい。

【００８０】

ＲＥＣＥＩＶＥ　ＬＩＳＴ　５０９は、対応する受信優先権カウントを格納する複数のレジスタを含む。優先権カウントはＲＸＰＯＲＴＢＵＦｘ［４：０］カウントと呼ばれ、“ｘ”はポート番号である。最高３２のポートに優先権を割り当てるために、示した実施例においては各ＲＸＰＯＲＴＢＵＦｘ［４：０］カウントは５ビットである。ＲＥＣＥＩＶＥ　ＬＩＳＴ　５０９は対応するポート・マスクビット配列ＲＸＰＲＴＭＳＫ［２８：０］を含み、それぞれのＲＸＰＲＴＭＳＫビットは初めにロジック０、すなわち優先権がまだ割り当てられていないとき、そしてそれぞれのＰＫＴ＿ＡＶＡＩＬｍ＊信号がその後アサートされたときにＲＸポーリング・ステート・マシン５０２によってセットされる。そのとき、ＲＸポーリング状態マシン５０２は対応するＲＸＰＯＲＴＢＵＦｘレジスタ内に優先権番号を割り当てる。割り当てられた優先権番号は、そのポートがサービスされるまで有効となっている。ＲＸＰＲＴＭＳＫがセットされている間、ＲＸポーリング状態マシン５０２は対応するＰＫＴ＿ＡＶＡＩＬｍ＊信号のその後のアサートをマスクして、それ以上の要求を無視する。ＨＳＢコントローラ５０５は、新しいパケットの最初の転送以外、それぞれのポートからそのパケットを転送するすべてのリード・サイクル期間中、ＲＸＰＲＴＭＳＫビットをクリアする。ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ　５３２は、もしそのパケットがＳｎＦの動作モードで転送すべきものであれば、最初のリード・サイクル転送期間中、ＲＸＰＲＴＭＳＫビットをクリアする。ＨＳＢコントローラ５０５は、もしそのパケットがＣＴの動作モードで転送されるものであれば、受信先ポートへの最初のライト・サイクル転送期間中、ＲＸＰＲＴＭＳＫビットをクリアする。

【００８１】

ＲＥＣＥＩＶＥ　ＬＩＳＴ　５０９はインキュー・ビットの配列ＲＸＩＮＱＵＥ［２８：０］を含み、各ビットは対応するＲＸＰＲＴＭＳＫビットがセットされたときにセットされる。それぞれのＲＸＩＮＱＵＥビットは優先権の値が有効であるか否か、そしてもし有効であればその対応するポートがアービトレーション・ロジック５０４による調停に委ねられるべきものであるかどうかを表示する。ＲＸＩＮＱＵＥビットは、それぞれのポートが新しいパケットあるいはＳｎＦパケットの続きを転送するための次のポートとして指定されるべくＭＡＩＮアービタ５１２に付託されたとき、アービトレーション・ロジック５０４内のアービタによってクリアされる。

【００８２】

ＲＥＣＥＩＶＥ　ＬＩＳＴ　５０９は、それぞれのポートがメモリ２１２内にデータを受信すべきかどうかを示すメモリ・ビット配列ＲＸＭＥＭＣＹＣ［２８：０］を含む。これは、ＳｎＦモード、暫定ＣＴモード、および暫定ミッドパケットＣＴモードでの動作時に行われる指示である。ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ　５３２は、ＳｎＦモードまたは暫定ＣＴモードが決定したときに対応するＲＸＭＥＭＣＹＣビットをセットする。ＭＡＩＮアービタ５１２は、ミッドパケット暫定ＣＴモードのパケットについて、もし受信先ポートが通常のＣＴモードの動作中に使用可能なバッファのスペースを表示しなければＲＸＭＥＭＣＹＣビットをセットする。ＨＳＢコントローラ５０５は、それぞれのポートについて、転送データの最終リード・サイクルでＲＸＭＥＭＣＹＣビットをクリアする。

【００８３】
ＲＥＣＥＩＶＥ　ＬＩＳＴ　５０９は、それぞれのポートが通常のＣＴ動作モードでデータ・パケットを転送しているかどうかを表示するアクティブのＣＴのビット配列ＲＸＡＣＴＣＹＣ［２８：０］を含む。ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ　５３２は、ＣＴモードのパケットについて対応するＲＸＡＣＴＣＹＣビットをセットする。ＨＳＢコントローラ５０５は、対応するポートに関し、最終パケット・データのリード・サイクルでＲＸＡＣＴＣＹＣビットをクリアする。ＭＡＩＮアービタ５１２は、ビットがＣＴモードを指示すべくセットされていて、ＭＡＩＮアービタ５１２がそのパケットをミッドパケット暫定ＣＴモードのパケットに変更する場合にＲＸＡＣＴＣＹＣビットをクリアする。

【００８４】
ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　５１０は、対応する送信優先権カウントを格納する複数のレジスタを含む。優先権カウントはＴＸＰＯＲＴＢＵＦｘ［４：０］カウントと呼ばれ、“ｘ”はポート番号である。最高３２のポートに優先権を割り当てるために、示した実施例においては各ＴＸＰＯＲＴＢＵＦｘ［４：０］カウントは５ビットである。ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　５１０は対応するポート・マスクビット配列ＴＸＰＲＴＭＳＫ［２８：０］を含み、それぞれのＴＸＰＲＴＭＳＫビットは初めにロジック０、すなわち優先権がまだ割り当てられていないとき、そしてそれぞれのＢＵＦ＿ＡＶＡＩＬｍ＊信号がその後アサートされたときにＴＸポーリング状態マシン５０３によってセットされる。そのとき、ＴＸポーリング状態マシン５０３は対応するＴＸＰＯＲＴＢＵＦｘレジスタ内に優先権番号を割り当てる。割り当てられた優先権番号は、そのポートがサービスされるまで有効となっている。ＴＸＰＲＴＭＳＫがセットされている間、ＴＸポーリング状態マシン５０３は対応するＢＵＦ＿ＡＶＡＩＬｍ＊信号のその後のアサートをマスクして、それ以上の要求を無視する。ＨＳＢコントローラ５０５は、新しいパケットの最初の転送以外、それぞれのポートからそのパケットを転送するすべてのリード・サイクル期間中、ＴＸＰＲＴＭＳＫビットをクリアする。ＨＳＢコントローラ５０５は、受信先ポートに対するパケット・データ転送のすべてのライト・サイクル期間中、ＴＸＰＲＴＭＳＫビットをクリアする。

【００８５】
ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　５１０は待ちインキュー・ビットの配列ＴＸＩＮＱＵＥ［２８：０］を含み、各ビットは対応するＴＸＰＲＴＭＳＫビットがセットされたときにセットされる。それぞれのＴＸＩＮＱＵＥビットは優先権の値が有効であるか否か、そしてもし有効であればその対応するポートがアービトレーション・ロジック５０４による調停に委ねられるべきものであるかどうかを表示する。ＴＸＩＮＱＵＥビットは、それぞれのポートが新しいパケット、あるいはＳｎＦパケットの続きを転送するための次のポートとして指定されるべくＭＡＩＮアービタ５１２に付託されたとき、アービトレーション・ロジック５０４内のアービタによってクリアされる。

【００８６】
ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　５１０は、それぞれのポートがメモリ２１２から受け取ったデータを送信すべきかどうかを示すメモリ・ビット配列ＴＸＭＥＭＣＹＣ［２８：０］を含む。これは、ＳｎＦモード、暫定ＣＴモード、および暫定ミッドパケットＣＴモードでの動作時に行われる指示である。ＦＩＦＯ制御ブロック５２９は、ＨＣＢ　４０２からデータを受け取った後ＭＣＢ４０４によるＲＸ＿ＰＫＴ＿ＣＯＭＰＬＥＴＥ＊信号のアサートに応答して、１つまたは複数のＴＸＭＥＭＣＹＣビットをセットする。ユニキャストのパケットについては、ＴＸＭＥＭＣＹＣビットが１つのみセットされる。ＢＣパケットについては、ＦＩＦＯ制御ブロック５２９がそのＢＣＢＩＴＭＡＰレジスタによってセットすべきＴＸＭＥＭＣＹＣビットを決定する。ＳｎＦモードのパケットに関しては、パケット全体がメモリ２１２内への格納のためにＭＣＢ４０４に転送された後でＴＸＭＥＭＣＹＣビットがセットされる。ＣＴモードのパケットについては、ミッドパケット暫定モードＣＴパケットを含め、ＭＣＢ４０４へのデータの最初のデータ転送中にＴＸＭＥＭＣＹＣビットがセットされる。ＨＳＢコントローラ５０５は、それぞれのポートへの転送デー

タの最終ライト・サイクルでＴＸＭＥＭＣＹＣビットをクリアする。これはＭＣＢ　４０４が、メモリ２１２内にはそのポートに対するデータが無くなった旨を示すＴＸ＿ＥＯＢＣ＊信号をアサートした場合も同じで、ＴＸＭＥＭＣＹＣビットがクリアされる。

【００８７】

ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　５１０は、ＲＸ　ＢＵＦ　５２０、５２２の１つにＣＴの動作モードでそれぞれの受信先ポートへ直接送信すべきデータがあるかどうかを表示するＣＴのビット配列ＴＸＣＴＣＹＣ［２８：０］を含む。ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ　５３２は、最初のパケット・データの転送で対応するＴＸＣＴＣＹＣビットをセットする。ＨＳＢコントローラ５０５は、対応する受信先のポートに対するデータ転送の最終のライト・サイクルでＴＸＣＴＣＹＣビットをクリアする。

【００８８】

ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　５１０は、それぞれのポートがＣＴ動作モードでデータ・パケットを転送しているかどうかを表すアクティブのＣＴのビット配列ＴＸＡＣＴＣＹＣ［２８：０］を含む。ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ　５３２は、そのパケットがＣＴモードで転送すべきものであると判断すれば、対応するＴＸＡＣＴＣＹＣビットをセットする。ＦＩＦＯ制御ブロック５２９は、パケットがＣＴモードからミッドパケット暫定ＣＴモードに変更されるとき、メモリ２１２への格納のためのＭＣＢ　４０４への最初のデータ転送の間にＴＸＡＣＴＣＹＣビットをクリアする。ＨＳＢコントローラ５０５もパケットの最終転送でＴＸＣＴＣＹＣビットをクリアする。

【００８９】

ＷＥＩＧＨＴ　ＦＡＣＴＯＲＳ５０８は、ポートＰＯＲＴ０～ＰＯＲＴ２８のそれぞれのについてポート・ウエイト・ファクタの配列ＰＯＲＴＷＴｘ［４：０］を含む。“ｘ”は特定のポート番号を表す。ＰＯＲＴＷＴウエイト・ファクタは一意であって、ポートの優先権をユーザがプログラミングできるように、ユーザによって予めプログラミングされていることが望ましい。受信および送信の動作にそれぞれの異なったウエイト・ファクタを定義することができるが、示した実施例においては、受信と送信の両方のケースについて、各ポートに同じウエイト・ファクタを割り当てている。

【００９０】

図１４は、ＲＸ受信ポーリング状態マシン５０２の受信ポーリング動作を表す状態図である。ＲＸ受信ポーリング状態マシン５０２の主たる機能は、ＰＫＴ＿ＡＶＡＩＬｍ＊信号のモニタ、優先権カウントＲＸＰＯＲＴＢＵＦｘの割り当て、およびＲＥＣＥＩＶＥ　ＬＩＳＴ　５０９内のＲＸＰＲＴＭＳＫビットの設定である。状態間の移り変わりは、ＣＬＫ信号の遷移もしくはサイクルおよびＳＴＲＯＢＥ＊信号の状態に基づいている。最初に、パワーアップとコンフィギュレーションで受信優先権カウント番号ＲＰＣＯＵＮＴはゼロに設定され、ＲＸポーリング状態マシン５０２は初期アイドリング状態５５０に入る。また、ＰＫＴ＿ＡＶＡＩＬｍ＊信号に対応するＲＸＩＮＣＣＮＴＢＹ［７：０］論理ビットがクリアされる。ＲＸポーリング状態マシン５０２は、ＳＴＲＯＢＥ＊信号がアサートされない間、すなわちＳＴＲＯＢＥ＊信号がハイ、つまりロジック１のときは状態５５０に留まっている。ＳＴＲＯＢＥ＊信号がロウにアサートされたとき、動作は１ＣＬＫ待ち状態（ＲｘＰｏｌｌＷａｉｔ）５５２に移る。

【００９１】

サンプリングで検知されたＳＴＲＯＢＥ＊信号のアサートに応答して、ＱＣデバイス２０２、ＴＰＩ　２２０、およびＰＣＢ　４０６は、ひとつのＣＬＫサイクル後にそれぞれＰＫＴ＿ＡＶＡＩＬｍ＊信号、言い換えればＰＫＴ＿ＡＶＡＩＬ［７：０］＊信号の対応する１つをアサートする。このようにして、動作はひとつのＣＬＫサイクル後に状態５５４に進み、それぞれのＰＫＴ＿ＡＶＡＩＬ［７：０］＊信号のポーリングを開始する。動作は状態５５４から状態５５６に入り、それから状態５５８さらに状態５６０へとＣＬＫ信号の経時的なサイクルに追従して移る。動作は状態５６０から状態５５４へ戻り、ＳＴＲＯＢＥ＊信号がアサートされている間はこのループを継続する。しかし、ＳＴＲＯＢＥ＊信号は周期的であり、１ＣＬＫサイクル間抑止され、そして次の３ＣＬＫサイクル間再ア

サートされるのが望ましい。こうして、もしＳＴＲＯＢＥ＊信号がステップ５６０でディアサートされると動作は状態５５０に戻る。状態５５４、５５６、５５８、および５６０のそれぞれにおいて、初期アービトレーション・カウント論理演算が実行すべき論理演算が残存するか否かを判断するＲＰＣＯＵＮＴ番号との比較におけるＲＸＮＥＷＣＮＴおよびＲＸＡＣＴＣＮＴの増分に基づいて実行される。

【００９２】

もしステップ５５４において初期アービトレーション・カウント論理演算が真であれば、それぞれのＱＣデバイス２０２およびＴＰＩ　２２０の最初のポート、およびＰＣＢ　４０６について１～９と呼称する９回の論理演算が実行される。ここで、最初の８動作はＰＯＲＴ０、ＰＯＲＴ４、ＰＯＲＴ８、ＰＯＲＴ１２、ＰＯＲＴ１６、ＰＯＲＴ２０、ＰＯＲＴ２４、およびＰＯＲＴ２８にそれぞれ対応する。８つのポート論理演算１～８の各々について、ＰＫＴ＿ＡＶＡＩＬｍ＊信号の対応する１つが対応するＲＸＰＲＴＭＳＫビットと比較されて要求を受容するかどうかが決定される。ＲＸＰＲＴＭＳＫビットが予めセットされていない場合にあり得る事象であるが、もし１つのポートについて要求が受け付けられると、そのポートにＲＸＰＯＲＴＢＵＦｘ優先権番号が割り当てられる。また、対応するＲＸＰＲＴＭＳＫビットがロジック１にセットされてポートからのそれ以上の要求をマスクし、そして対応するＲＸＩＮＣＣＮＴＢＹビットがロジック１にセットされる。９番目の論理演算はＲＰＣＯＵＮＴの増分に実行される。

【００９３】

ＰＯＲＴ０について、もしＰＫＴ＿ＡＶＡＩＬ［０］＊信号がアサートされないか、あるいはもしＲＸＰＲＴＭＳＫ［０］がロジック１に等しいと、優先権が既に設定されているのであって、それはＰＯＲＴ０がサービスされるまで変更されることはない。しかし、もしＰＫＴ＿ＡＶＡＩＬ［０］＊信号がロウにアサートされ、かつＲＸＰＲＴＭＳＫ［０］がロジック０であれば、対応する優先権カウントＲＸＰＯＲＴＢＵＦ０はＷＴＰＲＩＯＲＩＴＹフラグがウエイト・ファクタに従って優先権を表示している場合、対応するウエイト・ファクタＲＸＰＯＲＴＷＴ０に等しく設定される。しかし、もしＷＴＰＲＩＯＲＩＴＹフラグが偽であれば、ＲＸＰＯＲＴＢＵＦ０はＲＰＣＯＵＮＴに等しくセットされる。そして、ＲＸＰＲＴＭＳＫ［０］およびＲＸＩＮＣＣＮＴＢＹビットが両方ともロジック１にセットされる。ＲＸＰＲＴＭＳＫ［０］マスクをセットすれば、ＰＯＲＴ０のさらなるポーリング要求を受け付けることになる。ＲＸＩＮＣＣＮＴＢＹビットはＰＫＴ＿ＡＶＡＩＬ［０］＊信号に対応しており、状態５５４における残りの論理演算に用いられてＰＯＲＴ０に優先権の値が設定されたことを表示する。

【００９４】

ＰＯＲＴ４に対応する２番目の論理演算において、もしＰＫＴ＿ＡＶＡＩＬ［１］＊信号がアサートされないか、あるいはもしＲＸＰＲＴＭＳＫ［４］がロジック１に等しいと、優先権が既に設定されているのであって、それはＰＯＲＴ４がサービスされるまで変更されることはない。しかし、もしＰＫＴ＿ＡＶＡＩＬ［１］＊信号がロウにアサートされ、かつＲＸＰＲＴＭＳＫ［４］がロジック０であれば、対応する優先権カウントＲＸＰＯＲＴＢＵＦ４はＷＴＰＲＩＯＲＩＴＹフラグがウエイト・ファクタに従って優先権を表示している場合、対応するウエイト・ファクタＲＸＰＯＲＴＷＴ４に等しく設定される。しかし、もしＷＴＰＲＩＯＲＩＴＹフラグがが偽であれば、優先権カウントＲＸＰＯＲＴＢＵＦ４はＲＰＣＯＵＮＴプラスＲＸＩＮＣＣＮＴＢＹ［０］くセットされる。このようにして、もしＷＴＰＲＩＯＲＩＴＹが偽であれば、ＲＸＰＯＲＴＢＵＦ４にはＰＯＲＴ０に優先権の値が設定ささされていない場合に優先権番号としてＲＰＣＯＵＮＴが割り当てられ、ＰＯＲＴ０に優先権番号が設定されている場合はＲＰＣＯＵＮＴ＋１の優先権番号が与えられる。これによって、ＰＯＲＴ０とＰＯＲＴ４に同じ優先権番号が割り当てられないことが保証される。ＲＸＰＲＴＭＳＫ［４］はそれからロジック１にセットされ、さらなるポーリング要求は無視される。このようにして、各ポートに割り当てられる優先権番号は、そのポートに予め決められたウエイト・ファクタであるか、もしくは優先権番号はＲＰＣＯＵＮＴに加えてより小さいポート番号と同時に割り当てられた優先権番号を持ってい

るポートの数である。

【００９５】

次の６つの論理演算は２番目の論理演算と同様である。ＰＣＢ　４０６に対応する８番目の論理演算において、もしＰＫＴ＿ＡＶＡＩＬ［７］＊信号がロウにアサートされていないか、あるいはもしＲＸＰＲＴＭＳＫ［２８］がロジック１に等しいと優先権が既に設定されているのであって、それはＰＣＢ　４０６がサービスされるまで変更されることはない。しかし、もしＰＫＴ＿ＡＶＡＩＬ［１］＊信号がロウにアサートされていて、かつＲＸＰＲＴＭＳＫ［２８］がロジック０であれば、対応するＰＣＢ　４０６の優先権カウントＲＸＰＯＲＴＢＵＦ２８はＷＴＰＲＩＯＲＩＴＹフラグがウエイト・ファクタに従って優先権を表示している場合、対応するウエイト・ファクタＲＸＰＯＲＴＷＴ２８に等しく設定される。しかし、もしＷＴＰＲＩＯＲＩＴＹフラグがが偽であれば、優先権カウントＲＸＰＯＲＴＢＵＦ２８はＲＰＣＯＵＮＴプラスＲＸＩＮＣＣＮＴＢＹ［６：０］の「ビット合計」に等しくセットされる。ＲＸＩＮＣＣＮＴＢＹ［６：０］の「ビット合計」は、その前に７回のポート論理演算において割り当てられた優先権番号の値の数である。従って、ＰＣＢ　４０６に与えられる優先権番号は予め決められているウエイト・ファクタか、もしくはその優先権番号はＲＰＣＯＵＮＴに加えてより小さいポート番号と同時に割り当てられた優先権番号を持っているポートの数である。９番目の論理演算は状態５５４で実行され、ＲＰＣＯＵＮＴを状態５５４において優先権が割り当てられたポートの数に等しいＲＸＩＮＣＣＮＴＢＹ［７：０］のビット合計だけ増分する。この演算により、状態５５６で実行される１組の論理演算のためにＲＰＣＯＵＮＴが増分されることが保証される。

【００９６】

例えば、ＰＫＴ＿ＡＶＡＩＬ［７］＊信号の最初の多重化されたビットに関連するすべてのポート、すなわちＰＯＲＴ０、ＰＯＲＴ４、ＰＯＲＴ８、ＰＯＲＴ１２、ＰＯＲＴ１６、ＰＯＲＴ２０、ＰＯＲＴ２４、およびＰＯＲＴ２８が状態５５４で同時に要求を出し、ＲＰＣＯＵＮＴが最初から０のままで、前に設定されていて対応するようなＲＸＰＲＴＭＳＫビットが存在せず、そしてＷＴＰＲＩＯＲＩＴＹが偽であれば、状態５５４において対応する優先権カウントＲＸＰＯＲＴＢＵＦｘ（ｘ＝０、４、８、１２、１６、２０、２４、および２８）に対し、それぞれ優先権番号０、１、２、３、４、５、６、および７が割り当てられる。それからＲＰＣＯＵＮＴが８に等しくセットされる。別の例として、サービスを要求しているポートがＰＯＲＴ４、ＰＯＲＴ１２、およびＰＯＲＴ２０のみの場合、もしＷＴＰＲＩＯＲＩＴＹが偽でＲＰＣＯＵＮＴが３に設定されていれば、優先権カウントＲＸＰＯＲＴＢＵＦｘ（ｘ＝４、１２、２０）にそれぞれ０、１、および２の優先権番号が割り当てられる。ビット合計の演算によって、複数のポートが同時にサービスを要求しているとき、各ポートに一意の番号を与えられることが保証される。このようにして、優先権番号は先着順、すなわちＦＣＦＳ（Ｆｉｒｓｔ－Ｃｏｍｅ，　Ｆｉｒｓｔ－Ｓｅｒｖｅｄ）の優先権スキームに従って割り当てられるが、同時割り当ての処理には特定の順序が予め決められる。

【００９７】

状態５５６、５５８、および５６０における論理演算は、もし初期アービトレーション・カウント論理演算が真で、それぞれのＱＣデバイス２０２およびＴＰＩ　２２０の２番目のポート、つまり　ポートＰＯＲＴ１、ＰＯＲＴ５、ＰＯＲＴ９、ＰＯＲＴ１３、ＰＯＲＴ１７、　ＰＯＲＴ２１、およびＰＯＲＴ２５に関連するＰＫＴ＿ＡＶＡＩＬ［６：０］＊信号に基づいた８つの論理演算が実行され、そして状態５５４の８番目の論理演算がＣＰＵ２３０へのポートＰＯＲＴ２８について繰り返されれば、状態５５４での論理演算と同様である。状態５５８において、それぞれのＱＣデバイス２０２およびＴＰＩ２２０の３番目のポート、つまり　ポートＰＯＲＴ２、ＰＯＲＴ６、ＰＯＲＴ１０、ＰＯＲＴ１４、ＰＯＲＴ１８、　ＰＯＲＴ２２、およびＰＯＲＴ２６に関連する７つの論理演算がＰＫＴ＿ＡＶＡＩＬ［６：０］＊信号に基づいて実行され、状態５５４の８番目の論理演算がＣＰＵ　２３０へのポートＰＯＲＴ２８について繰り返される。状態５６０において、そ

れぞれのＱＣデバイス２０２およびＴＰＩ２２０の４番目のポート、つまり　ポートＰＯＲＴ３、ＰＯＲＴ７、ＰＯＲＴ１１、ＰＯＲＴ１５、ＰＯＲＴ１９、　ＰＯＲＴ２３、およびＰＯＲＴ２７に関連する７つの論理演算がＰＫＴ＿ＡＶＡＩＬ［６：０］＊信号に基づいて実行され、状態５５４の８番目の論理演算がＣＰＵ２３０へのポートＰＯＲＴ２８について繰り返される。状態５５６、５５８、および５６０のそれぞれにおいて、最後の論理演算が実行されて前述と同様にＲＰＣＯＵＮＴがＲＸＩＮＣＣＮＴＢＹビットのビット合計だけ増分される。

【００９８】

図１９は、ＴＸ送信ポーリング状態マシン５０３の送信ポーリング動作を表す状態図である。ＴＸ送信ポーリング状態マシン５０３の動作はＲＸ受信ポーリング状態マシン５０２の動作と同様で、状態５５０、５５２、５５４、５５６、５５８、および５６０にそれぞれ相似の状態５６１、５６２、５６４、５６６、５５８、および５７０を含む。しかし、ＴＰＣＯＵＮＴがＲＰＣＯＵＮＴに代わり、初期アービトレーション・カウント論理演算は実行すべき論理演算が残存するかどうかを判断するＴＰＣＯＵＮＴ番号との比較におけるＴＸＮＥＷＣＮＴおよびＴＸＡＣＴＣＮＴの増分に基づいて実行される。ＢＵＦ＿ＡＶＡＩＬｍ＊信号がＰＫＴ＿ＡＶＡＩＬｍ＊信号に代わり、ＴＸＰＲＴＭＳＫビットがＲＸＰＲＴＭＳＫビットに代わる。また各ポートの等式では、ＴＸＰＲＴＭＳＫビットとＴＸＭＥＭＣＹＣ、ＴＸＣＴＡＣＴＣＹＣ、およびＴＸＣＴＣＹＣビット配列の対応するビットに基づいた論理項との論理積が求められる。特に、ＥＰＳＭ　２１０またはメモリ２１２内に受信先のポートが送信すべきデータがある場合にのみ当該ポートに優先権が割り当てられるよう、ＴＸＭＥＭＣＹＣ、ＴＸＣＴＡＣＴＣＹＣ、およびＴＸＣＴＣＹＣビット配列の論理和が求められる。さらに、ＴＸＰＯＲＴＢＵＦｘ優先権カウントがＲＸＰＯＲＴＢＵＦｘ優先権カウントに代わり、ＴＸＰＯＲＴＷＴウエイト・ファクタがＲＸＰＯＲＴＷＴウエイト・ファクタに代わり、そしてＴＸＩＮＣＣＮＴＢＹビットがＲＸＩＮＣＣＮＴＢＹビットに代わる。このようにして、各ポートおよびＰＣＢ　４０６の表示はＳＴＲＯＢＥ＊信号に応答してＢＵＦ＿ＡＶＡＩＬ＊信号のそれぞれ１つによるものとなり、ＴＸポーリング状態マシン５０３はＴＰＣＯＵＮＴを用い、ＦＣＦＳあるいはウエイト・ファクタに基づいて優先権番号を割り当て、それに応じて優先権を設定する。

【００９９】

要求しているポートの各々に対する優先権の割り当て、および対応するポーリング・マスクビットの設定に備えてポーリング・ロジック５０１が周期的あるいは連続的にＳＴＲＯＢＥ＊信号をトグルし、ポート１０４、１１０、およびＰＣＢ　４０６のそれぞれのＰＫＴ＿ＡＶＡＩＬｍ＊およびＢＵＦ＿ＡＶＡＩＬｍ＊信号を監視する機能は高い評価に値する。割り当てられる優先権は、もしＷＴＰＲＩＯＲＩＴＹが真であれば予めプログラミングされているウエイト・ファクタに基づくか、あるいはもしＷＴＰＲＩＯＲＩＴＹが偽であればＦＣＦＳに基づく。与えられた優先権は、そのポートがサービスされるまでそのままに留まっている。後述するように、最終的にそのポートはサービスを受け、そのマスクビットはクリアされる。

【０１００】

アービタ５１３～５１６は、いくつかのアービトレーション・スキームの１つに基づいてポート１０４、１１０、およびＰＣＢ　４０６間における選択を行う。ここで、特定のアービトレーション・スキームをユーザがプログラミングすることも可能である。最初はラウンドロビン法であって、これによりポートがＰＯＲＴ１、ＰＯＲＴ２、．．．、ＰＯＲＴ２８といったような任意の順序でチェックされるか、あるいはその順序はＰＯＲＴＷＴｘレジスタ内に予めプログラミングされているＷＥＩＧＨＴ　ＦＡＣＴＯＲＳ　５０８で選択される。示した実施例においては、ラウンドロビン法による割り当てにＷＥＩＧＨＴ　ＦＡＣＴＯＲＳが用いられており、それぞれのＲＸＰＯＲＴＢＵＦｘおよびＴＸＰＯＲＴＢＵＦｘカウントにプログラミングされている。ＲＸ　ＮＷアービタ５１３はＲＸＮＥＷＣＮＴ優先権番号を用いてこれを増分し、ＲＸ　ＡＣＴアービタ５１４はＲＸＡＣＴＣＮＴ優先権番号を用いてこれを増分し、ＴＸ　ＮＷアービタ５１５はＴＸＮＥＷＣＮＴ優

先権番号を用いてこれを増分し、ＴＸ　ＣＴアービタ５１６はＴＸＣＴＣＮＴ優先権番号を用いてこれを増分する。ラウンドロビン法では、ＲＸアービタ５１３および５１４は、それぞれＲＸＩＮＱＵＥ［］の値を調べてサービスを要求しているアクティブな受信ポートの存在如何を確認し、それからそのそれぞれの優先権番号（ＲＸＮＥＷＣＮＴ、ＲＸＡＣＴＣＮＴ）をアクティブなポートのＲＸＰＯＲＴＢＵＦｘカウント内の値と比較して次にサービスされるべきポートの有無を確認する。また、ＴＸアービタ５１５、５１６は、それぞれＴＸＩＮＱＵＥ［］の値を調べてサービスを要求しているアクティブな送信ポートの存在如何を確認し、それからそのそれぞれの優先権番号（ＴＸＮＥＷＣＮＴ、ＴＸＣＴＣＮＴ）をアクティブなポートのＴＸＰＯＲＴＢＵＦｘカウント内のカウント値と比較して次にサービスされるべきポートの有無を確認する。ＷＥＩＧＨＴ　ＦＡＣＴＯＲＳは特定の順序を決めるので、ポートはラウンドロビンの方式で序列される。

【０１０１】

２番目のアービトレーション・スキームはＦＣＦＳであり、その場合ＷＴＰＲＩＯＲＩＴＹは偽であって、ポートはＲＸＰＯＲＴＢＵＦｘおよびＴＸＰＯＲＴＢＵＦｘ優先権番号で表されているサービスを要求した順序でサービスを受ける。ＦＣＦＳにおける動作は、前述したようにＲＸＰＯＲＴＢＵＦｘおよびＴＸＰＯＲＴＢＵＦｘカウントがＲＰＣＯＵＮＴおよびＴＰＣＯＵＮＴの値に従ってプログラミングされる点を除き、ラウンドロビンの動作と同様である。この場合、ＲＸアービタ５１３および５１４は、それぞれＲＸＩＮＱＵＥ［］の値を調べてサービスを要求しているアクティブな受信ポートの存在如何を確認し、それからそのそれぞれの優先権番号（ＲＸＮＥＷＣＮＴ、ＲＸＡＣＴＣＮＴ）をアクティブなポートのＲＸＰＯＲＴＢＵＦｘカウント内の値と比較して次にサービスされるべきポートの有無を確認する。また、ＴＸアービタ５１５、５１６は、それぞれＴＸＩＮＱＵＥ［］の値を調べてサービスを要求しているアクティブな送信ポートの存在如何を確認し、それからそのそれぞれの優先権番号（ＴＸＮＥＷＣＮＴ、ＴＸＣＴＣＮＴ）をアクティブなポートのＴＸＰＯＲＴＢＵＦｘカウント内のカウント値と比較して次にサービスされるべきポートの有無を確認する。ＲＰＣＯＵＮＴおよびＴＰＣＯＵＮＴは特定の順序を決めるので、ポートはＦＣＦＳの方式で序列される。

【０１０２】

もう１つのスキームはウエイト優先権スキームであり、その場合ＷＴＰＲＩＯＲＩＴＹは真であって、ＲＸＰＯＲＴＷＴｘおよびＴＸＰＯＲＴＷＴｘ番号がＲＸＰＯＲＴＢＵＦｘおよびＴＸＰＯＲＴＢＵＦｘレジスタの対応する１つにコピーされ、優先権の決定に使用される。しかし、ＲＸアービタ５１３、５１４はＲＸ　ＨＩＧＨ　ＰＲＩＯＲＩＴＹ番号から優先権を決め、ＴＸアービタ５１５、５１６はＴＸ　ＨＩＧＨ　ＰＲＩＯＲＩＴＹ番号から優先権を決定する。ＲＸ　ＨＩＧＨ　ＰＲＩＯＲＩＴＹ番号は、アクティブな受信ポートのＲＸＰＯＲＴＢＵＦｘカウント内における最高の優先権番号（すなわち１番小さい数）を識別することによって決定される。ここで、アクティブな受信ポートはＲＸＩＮＱＵＥの値で判断される。同様に、ＴＸ　ＨＩＧＨ　ＰＲＩＯＲＩＴＹ番号は、アクティブな受信ポートのＴＸＰＯＲＴＢＵＦｘカウント内における最高の優先権番号（すなわち１番小さい数）を識別することによって決定される。ここで、アクティブな受信ポートはＴＸＩＮＱＵＥの値で判断される。このようにして、ウエイト・ファクタが最高のアクティブな（すなわちサービスを要求している）ポートが毎回選択されて加重優先権割り当て法の機能が遂行される。

【０１０３】

ＲＸ　ＮＷアービタ５１３は、ポートＰＯＲＴ０～ＰＯＲＴ２８で受信されたすべての新しいパケット・ヘッダのデータおよびＳｎＦモードのパケット・データの続きを処理し、そのデータはＲＸ　ＢＵＦ　５２０、５２２のいずれか１つに転送される。ＲＸ　ＮＷアービタ５１３は、ＲＸＮＥＷＣＮＴ番号を更新し、ＲＥＣＥＩＶＥ　ＬＩＳＴ　５０９をチェックして受信決定基準に合致しているポートはＰＯＲＴ０～ＰＯＲＴ２８のいずれであるかを判断する。ＲＸ　ＮＷアービタ５１３の受信決定基準に適合するポートは、そのＲＸＩＮＱＵＥビットがアサートされていて、そのＲＸＡＣＴＣＹＣビットがアサートさ

れていないポートである。ＲＸ　ＮＷアービタ５１３の受信決定基準として、さらにＲＸＩＮＱＵＥとＲＸＭＥＭＣＹＣビットの両方がアサートされているポートも含まれる。ＲＸ　ＮＷアービタ５１３は、その受信決定基準を満たしている複数のポート間で、選択した前述のアービトレーション・スキームに従って調停を行う。１つのポートを選択してサイクルを定義した後、ＲＸ　ＮＷアービタ５１３はＭＡＩＮアービタ５１２に対してリード・サイクルを１回実行すべく要求する。ＲＸ　ＮＷアービタ５１３がＭＡＩＮアービタ５１２によって次に選択されたとき、ＲＸＮＷアービタ５１３はサービスを受けるべく選択されたポートのＲＸＩＮＱＵＥビットをクリアする。このプロセスをＲＸ　ＮＷアービタ５１３は連続的に繰り返す。

【０１０４】

ＴＸ　ＣＴアービタ５１６は、ＲＸ　ＢＵＦ　５２０、５２２の中のデータを受信先のポートへ通常のＣＴ動作モードで転送する。ＴＸ　ＣＴアービタ５１６は、ＴＸＮＥＷＣＮＴ番号を更新し、ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　５１０をチェックして送信決定基準に合致しているポートはＰＯＲＴ０～ＰＯＲＴ２８のいずれであるかを判断する。ＴＸ　ＮＷアービタ５１６の送信決定基準に適合するポートは、それぞれのＴＸＩＮＱＵＥおよびＴＸＣＴＣＹＣビットがアサートされているポートである。ＴＸ　ＣＴアービタ５１６は、その送信決定基準を満たしている複数のポート間で、選択した前述のアービトレーション・スキームに従って調停を行う。１つのポートを選択してサイクルを定義した後、ＴＸ　ＣＴアービタ５１６は選択したＲＸ　ＢＵＦ　５２０、または５２２からデータを選ばれた受信先ポートへ送信すべく、ＭＡＩＮアービタ５１２に対してライト・サイクルを１回実行するよう要求する。ＴＸ　ＣＴアービタ５１６がＭＡＩＮアービタ５１２によって次に選択されたとき、ＴＸ　ＣＴアービタ５１６はサービスを受けるべく選択されたポートのＴＸＩＮＱＵＥビットをクリアする。このプロセスをＴＸ　ＣＴアービタ５１６は連続的に繰り返す。

【０１０５】

ＲＸ　ＡＣＴアービタ５１４は、（ＲＸ　ＮＷアービタ５１３が処理する）新しいパケットの１回目のリード・サイクルを除き、後続のパケット・データを通常のＣＴ動作モードで動作している送信元のポートからＣＴ　ＢＵＦ　５２８へ転送する。ＲＸ　ＡＣＴアービタ５１４は、ＲＸＡＣＴＣＮＴ番号を更新し、ＲＥＣＥＩＶＥ　ＬＩＳＴ　５０９をチェックしてその受信決定基準に合致しているポートはＰＯＲＴ０～ＰＯＲＴ２８のいずれであるかを判断する。ＲＸ　ＡＣＴアービタ５１４の受信決定基準に適合するポートは、そのＲＸＩＮＱＵＥおよびＲＸＡＣＴＣＹＣビットがアサートされており、そのＲＸＭＥＭＣＹＣビットがアサートされていないポートである。ＲＸ　ＡＣＴアービタ５１４は、その受信決定基準を満たしている複数のポート間で、選択した前述のアービトレーション・スキームに従って調停を行う。１つのポートを選択してサイクルを定義した後、ＲＸ　ＡＣＴアービタ５１４は選択された送信元ポートからＣＴ　ＢＵＦ　５２８へデータを転送すべく、ＭＡＩＮアービタ５１２に対してリード・サイクルを１回実行するよう要求する。ＲＸ　ＡＣＴアービタ５１４がＭＡＩＮアービタ５１２によって次に選択されたとき、ＲＸ　ＡＣＴアービタ５１４はサービスを受けるべく選択されたポートのＲＸＩＮＱＵＥビットをクリアする。このプロセスをＲＸ　ＡＣＴアービタ５１４は連続的に繰り返す。

【０１０６】

ＭＡＩＮアービタ５１２は、ＣＴ　ＢＵＦ　５２８へのＣＴモードの各リード・サイクルに続いて、ＣＴ　ＢＵＦ　５２８内のデータをＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ　５３２に指示される受信先ポートへ転送するためのライト・サイクルを１回実行する。ＭＡＩＮアービタ５１２は、ＲＸ　ＡＣＴアービタ５１４にＣＴデータをＣＴ　ＢＵＦ　５２８へ転送させる前に受信先のポートが使用中かどうかをチェックする。ＭＡＩＮアービタ５１２は、もし受信先ポートが使用中であることを確認すれば、それぞれのＲＸＭＥＭＣＹＣビットをセットし、送信元ポートのそれぞれのＲＸＡＣＴＣＹＣビットをクリアし、送信元と受信先のポートの動作モードをミッドパケット暫定ＣＴモードに変更する。

【０１０７】
ＴＸ　ＮＷアービタ５１５は、ＴＸ　ＢＵＦ　５２４および５２６のいずれかから、データをＨＳＢ　２０６へＳｎＦの動作モードで転送する。ＴＸ　ＮＷアービタ５１５は、ＴＸＮＥＷＣＮＴ番号を更新し、ＴＲＡＮＳＭＩＴ　ＬＩＳＴ５１０をチェックしてその送信決定基準に合致しているポートはＰＯＲＴ０～ＰＯＲＴ２８のいずれであるかを判断する。ＴＸ　ＮＷアービタ５１５の送信決定基準に適合するポートは、それぞれのＴＸＩＮＱＵＥおよびＴＸＭＥＭＣＹＣビットがアサートされており、それぞれのＴＸＡＣＴＣＴＣＹＣビットがアサートされていないポートである。ＴＸ　ＮＷアービタ５１５は、その送信決定基準を満たしている複数のポート間で、選択したアービトレーション・スキームに従って調停を行う。１つのポートを選択して、ＴＸ　ＢＵＦ　５２４および５２６のいずれかから選択された受信先ポートへのライト・サイクルを定義した後、ＴＸ　ＮＷアービタ５１５はＭＡＩＮアービタ５１２に対してライト・サイクルを実行するよう要求する。ＴＸ　ＮＷアービタ５１５がＭＡＩＮアービタ５１２によって次に選択されたとき、ＴＸ　ＮＷアービタ５１５はサービスを受けるべく選択されたポートのＴＸＩＮＱＵＥビットをクリアする。このプロセスをＴＸ　ＮＷアービタ５１５は連続的に繰り返す。
【０１０８】
次に図２４を参照する。ＥＰＳＭ　２１０内のＭＣＢ　４０４の詳細ブロック図である。ＭＣＢ構成レジスタ４４８は図２４に示されていないが以下に説明されており、ここで解説する多数の機能ブロックにより、必要に応じて適切な理解が得られる。ＭＣＢ　４０４は、バス４２０を介してＭＣＢインタフェース４１４に結合されているハッシュ・コントローラ６０２を含む。ハッシュ・コントローラ６０２は、メモリ２１２から取り出されたデータを格納するハッシュ・キャシュ・テーブル６０３をオプションとして含む。ハッシュ・キャシュ６０３を使用すれば、メモリ２１２から最近取り出されたデータに対する速いアクセスが可能となり、最近アクセスされた情報を取り出す場合に、もう一度メモリ・サイクルを実行する必要が無くなる。ハッシュ・コントローラ６０２は、バス６１０を介して４入力アドレス・マルチプレクサ（ｍｕｘ）６３０の１つの複線入力に結合されたアドレス／長さ／状態、ＡＤ／ＬＮ／ＳＴ（ＡＤｄｒｅｓｓ／ＬｅＮｇｔｈ／ＳＴａｔｕｓ）出力を含む。ＡＤ／ＬＮ／ＳＴ出力は、メモリ２１２のアドレス、バースト・サイクルを実行すべきか否かを決定するトランザクションの長さ、およびリード／ライト（Ｒ／Ｗ）信号、バイト・イネーブル、ページ・ヒット信号、ロック信号といった種々の状態信号を定義する。ＤＲＡＭ要求／許可／ストローブ／制御、ＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ（ＤＲＡＭ　ＲｅＱｕｅｓｔ／ＧｒａｎＴ／ＳＴｒｏＢｅ／ＣｏｎＴｒｏＬ）制御６２８は、ＤＲＡＭメモリ・アービタ６３８およびハッシュ・コントローラ６０２のＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ入力に結合されている。ｍｕｘ　６３０の出力はＤＲＡＭメモリ・コントローラ６３６のＡＤ／ＬＮ／ＳＴ入力に供給され、ＤＲＡＭメモリ・コントローラ６３６はメモリ・バス２１４を介して、さらにメモリ２１２に結合されている。ハッシュ・コントローラ６０２は、ＤＲＡＭメモリ・コントローラ６３６からデータ・バス６１８を介してデータを受け取るためのデータ入力（ＤＩＮ）を持っている。
【０１０９】
ＲＸ　ＨＣＢインタフェース６０１は、ＭＤＯ［３１：０］信号を含むバス４２０に結合されており、４入力データ・マルチプレクサ（ｍｕｘ）６３２の１番目の複線入力にバス６２０を介してデータを供給するためのデータ出力（ＤＯＵＴ）を含む。ここでｍｕｘ６３２は、その出力をＤＲＡＭコントローラ６３６のＭｅｍＤａｔａＯｕｔ入力に供給する。ＲＸ　ＨＣＢインタフェース６０１は、ＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６２８のストローブおよび制御信号を受け取るためのＳＴＢ／ＣＴＬ入力を含む。ＲＸコントローラ６０４はバス４２０に結合されており、マルチプレクサ６３０の２番目の入力にバス６１２を介して結合されているＡＤ／ＬＮ／ＳＴ出力を持っている。ＲＸコントローラ６０４は、ｍｕｘ　６３２の２番目の入力にバス６２２を介して結合されているデータ出力ＤＯＵＴ、バス６１８に結合しているデータ入力ＤＩＮ、静的ＲＡＭ（ＳＲＡＭ）６５０関連のＳＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６５４を受け取るためのＳＲＡ

Ｍ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ入力、およびＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６２８を受け取るためのＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ入力を持っている。

【０１１０】

ＴＸ　ＨＣＢインタフェース６０５は、ＭＤＩ［３１：０］信号を含むバス４２０に結合されており、バス６１８に結合されているデータ入力ＤＩＮとＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６２８のストローブおよび制御信号を受け取るＳＴＢ／ＣＴＬ入力を持っている。ＴＸコントローラ６０６はバス４２０に結合されており、ｍｕｘ　６３０の３番目の入力にバス６１４を介して供給されるＡＤ／ＬＮ／ＳＴ出力、ｍｕｘ　６３２の３番目の入力にバス６２４を介して結合されているデータ出力ＤＯＵＴ、バス６１８に結合されているデータ入力ＤＩＮ、ＳＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６５４を受け取るためのＳＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ入力、およびＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６２８を受け取るためのＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ入力を持っている。ＰＣＢインタフェース４２４は、マルチプレクサ６３０の４番目の入力にバス６１６を介して結合されているＡＤ／ＬＮ／ＳＴ出力、マルチプレクサ６３２の４番目の入力にバス６２６を介して結合されているデータ出力ＤＯＵＴ、バス６１８に結合されているデータ入力ＤＩＮ、ＳＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６５４を受け取るためのＳＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ入力、およびＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６２８を受け取るためのＤＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ入力を持っている。

【０１１１】

ハッシュ・コントローラ６０２、ＲＸコントローラ６０４、ＴＸコントローラ６０６、ＰＣＢインタフェース４２４、ＲＸ　ＨＣＢインタフェース６０１、およびＴＸ　ＨＣＢインタフェース６０５は、それぞれＳＴＢ信号を用いてデータ・フローを同期させるが、ＳＴＲＯＢＥ＊信号のアサートで、データがいつリード・サイクルに有効であるか、あるいはデータがいつライト・サイクルに取り出されるかを判断する。ＣＴＬ信号は、例えばデータ・サイクルの完了時を表示する信号のような種々の制御信号である。

【０１１２】

ＤＲＡＭアービタ６３８はさらに、メモリ制御信号（ＭＥＭＣＴＬ）でＤＲＡＭコントローラ６３６に結合し、マルチプレクサ制御信号（ＭＵＸＣＴＬ）をマルチプレクサ　６３０、６３２の選択入力に供給する。ＭＥＭＣＴＬ信号は、一般に各メモリ・サイクルの開始と終了を表示する。このように、ハッシュ・コントローラ６０２、ＲＸコントローラ６０４、ＴＸコントローラ６０６、およびＰＣＢインタフェース４２４は、それぞれの要求信号をアサートすることによって、メモリ２１２に対してメモリ・サイクルを実行するためにＤＲＡＭコントローラ６３６へのアクセスの調停を行う。ＤＲＡＭアービタ６３８は要求信号を受け取って、要求しているデバイス６０２、６０４、６０６、および４２４の１つに対応する許可（ＧＴ）信号をアサートすることにより、そのデバイスに対してアクセスを許可する。いったんアクセスが許可されると、ＤＲＡＭアービタ６３８はマルチプレクサ６３０および６３２へのＭＵＸＣＴＬ信号をアサートし、デバイス６０２、６０４、６０６、および４２４のうち選択された１つが必要に応じてメモリ・サイクルを実行すべくＤＲＡＭコントローラ６３６に対するアクセスを可能とし、そしてＭＥＭＣＴＬ信号の１つがアサートされてＤＲＡＭコントローラ６３６に対しサイクルの開始を示す。ＤＲＡＭコントローラ６３６は、ＭＥＭＣＴＬ信号の１つをアサートまたは抑止してメモリ・サイクルの完了を示す。

【０１１３】

ハッシュ・コントローラ６０２は、ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ　５３２と交信してハッシュ手順を実行し、ＨＡＳＨ　ＲＥＱ　ＬＯＧＩＣ　５３２に格納されている新しいパケット・ヘッダの処理方法を決定する。ハッシュ・コントローラ６０２は、アサートされたＨＡＳＨ＿ＲＥＱ＊信号を検知し、ＨＡＳＨ＿ＤＡ＿ＳＡ［１５：０］信号から送信元および受信先のメディア・アクセス制御（ＭＡＣ）信号を取り出し、ＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］を判定するために、そしてもし受信先のポート番号がメモリ２１２内に予

め格納されていれば、それをＨＡＳＨ＿ＤＳＴＰＲＴ［４：０］上に供給するためにハッシュ手順を実行する。ＲＸコントローラ６０４およびＲＸ　ＨＣＢインタフェース６０１は、ＲＸ　ＢＵＦ　５２０、５２２からのデータを制御し、メモリ２１２へ転送する。ＴＸコントローラ６０６およびＴＸ　ＨＣＢインタフェース６０５は、主としてメモリ２１２からのデータを制御し、ＴＸ　ＢＵＦ　５２４、５２６へ転送する。ＰＣＢインタフェース４２４によって、ＣＰＵ　２３０はメモリ２１２、およびＳＲＡＭ　６５０のメモリ内のデータにより直接的にアクセスすることができる。

【０１１４】

トローラ６０４およびＲＸ　ＨＣＢインタフェース６０１は、ＲＸ　ＢＵＦ　５２０、５２２からのデータを制御し、メモリ２１２へ転送する。ＴＸコントローラ６０６およびＴＸ　ＨＣＢインタフェース６０５は、主としてメモリ２１２からのデータを制御し、ＴＸ　ＢＵＦ　５２４、５２６へ転送する。ＰＣＢインタフェース４２４によって、ＣＰＵ　２３０はメモリ２１２、およびＳＲＡＭ　６５０のメモリ内のデータにより直接的にアクセスすることができる。

【０１１５】

ＳＲＡＭ　６５０はＳＲＡＭコントローラ６５２に結合しており、ＳＲＡＭコントローラ６５２はさらにＲＸコントローラ６０４、ＴＸコントローラ６０６、およびＰＣＢインタフェース４２４にバス６５３を介して結合している。ＳＲＡＭアービタ６５１は、制御信号ＳＣＴＬでＳＲＡＭコントローラ６５２に結合しており、さらにＰＣＢインタフェース４２４によるＳＲＡＭ　６５０へのアクセスを制御するためにＳＲＡＭ　ＲＱ／ＧＴ／ＳＴＢ／ＣＴＬ信号６５４、およびＤＲＡＭアービタ６３８によるＤＲＡＭコントローラ６３６へのアクセス制御と同様に、ＴＸコントローラ６０６およびＲＸコントローラ６０４にバス６５３を介して結合している。

【０１１６】

ＭＣＢ　４０４は、本明細書で後に詳述するように、パケット制御レジスタおよびその他のデータを格納するＳＲＡＭ　６５０を含む。パケット制御レジスタは、ポートごとのＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮ、ポートごとのＴＲＡＮＳＭＩＴ　ＰＡＣＫＥＴ　ＣＨＡＩＮ、およびメモリ２１２の空きメモリ・セクタのＦＲＥＥＰＯＯＬ　ＣＨＡＩＮへの１組のポインタを含む。パケット制御レジスタは、さらにネットワーク１０２内におけるパケット・データの流れの制御を可能とする制御情報やパラメータを含む。メモリ２１２は、隣接した同一サイズの複数のセクタで編成されているパケット・メモリ・セクションを含む。これらのセクタは、初期にはアドレス・ポインタ、あるいは同様な手段で相互にリンクされてをＦＲＥＥＰＯＯＬ　ＣＨＡＩＮ形成している。ポートからパケット・データが受け取られると、これらのセクタはＦＲＥＥＰＯＯＬ　ＣＨＡＩＮから取り出され、そのポートのＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮに追加される。さらにそのパケットは、それが送信時に送られるべき１つまたは複数の受信先のポートの１つまたは複数のＴＲＡＮＳＭＩＴ　ＰＡＣＫＥＴ　ＣＨＡＩＮにリンクされる。バス６５３によって、ＲＸコントローラ６０４、ＴＸコントローラ６０６、およびＣＰＵインタフェース４３６はメモリ２１２内のデータのパケット・チェーンへのポインタを含んでいるパケット制御レジスタにアクセスすることができる。

【０１１７】

ＤＲＡＭコントローラ６３６は、メモリ２１２内のデータを保持するためのメモリ・リフレッシュ・ロジック６６０を含む。リフレッシュ・ロジック６６０は、メモリ・バス２１４に結合されているＦＰＭ　ＤＲＡＭ、ＥＤＯ　ＤＲＡＭ、あるいは同期ＤＲＡＭのような各種のメモリのタイプに従って動作する順応性を備えている。このようにして、ＣＰＵ　２３０はリフレッシュの機能が不要となり、動作能率およびパフォーマンスが向上する。ＭＣＢ構成レジスタ４４８内にある１０ビットのメモリ・リフレッシュ・カウンタ（ＭＲＣ）は、リフレッシュ要求間のクロック・サイクルの数を定義する。その期間は１５．２６μｓに等しいかそれより短いことが望ましい。既定値は２０８ｈであり、“ｈ”は１６進数を示すが、これによって３０　ｎｓのＣＬＫサイクルでのリフレッシュ期間はおよ

そ１５．６０μｓとなる。ＭＲＣカウンタはタイムアウトでＤＲＡＭアービタ６３８への信号ＲＥＦＲＥＱをアサートし、ＤＲＡＭアービタ６３８はＤＲＡＭコントローラ６３６へのＭＥＭＣＴＬ信号の１つをアサートし、メモリ・リフレッシュ・ロジック６６０に対しリフレッシュ・サイクルを実行するよう指示する。ＭＣＢ構成レジスタ４４８は、メモリ２１２のメモリのタイプ、速度、および構成を定義するメモリ制御レジスタ（ＭＣＲ）を含む。例えば、ＭＣＲの２ビットはメモリのタイプがＦＰＭ、ＥＤＯ、および同期ＤＲＡＭのいずれであるかを表す。別の１ビットは、メモリの速度が５０および６０　ｎｓのいずれであるかを示す。その他のビットは、選択したタイプのＤＲＡＭの特定のモードを定義し、パリティ・エラーのような誤りも表示する。

【０１１８】

次に図２５を参照する。ＰＣＢ　４０６の詳細ブロック図である。ＣＰＵバス２１８がＣＰＵインタフェース４３２の中のＣＰＵインタフェース・ロジック７００に結合しており、ＣＰＵインタフェース・ロジック７００は、さらにＱＣ／ＣＰＵバス２０４とインタフェースするためにバス７０１を経由してＱＣ／ＣＰＵインタフェース７０２に結合している。ＣＰＵインタフェース・ロジック７００は、ＦＩＦＯ　４３０内の１６バイトの受信バッファＲＸ　ＢＵＦ　７０６にデータを供給し、これがＭＣＢバス４２８上のデータをアサートする。ＭＣＢバス４２８は、ＣＰＵインタフェース・ロジック７００にデータを供給すべく、これもまたＦＩＦＯ　４３０内にある１６バイトの送信バッファＴＸ　ＢＵＦ　７０８にデータを入れる。ＭＣＢインタフェース４２６はＣＰＵインタフェース・ロジック７００とＭＣＢバス４２８との間のデータの流れを制御する。ＣＰＵインタフェース・ロジック７００は、バス信号７０３でＲＸ　ＢＵＦ　７０６、ＴＸ　ＢＵＦ　７０８、およびＭＣＢインタフェース４２６と結合している。

【０１１９】

ＣＰＵインタフェース・ロジック７００は、バス４４２を介してレジスタ・インタフェース４４０に結合されており、レジスタ・インタフェース４４０によってＥＰＳＭ　２１０内の他の構成レジスタにアクセスが可能となる。ＣＰＵインタフェース・ロジック７００は、割り込みレジスタ、構成レジスタ、パケット情報レジスタ、メモリ関連のレジスタ、設定／状態レジスタ、インタフェース／監視（モニタ）レジスタ、統計レジスタ、モード・レジスタ、アービトレーション・レジスタなどのような、ＣＰＵ　２３０の入出力（Ｉ／Ｏ）空間を定義する。

【０１２０】

ＣＰＵ　２３０は、パワーアップとコンフィギュレーションの間にＰＣＢレジスタ７０４内の初期値ないしは既定値をプログラミングする。例えば、ＣＰＵ　２３０はＰＣＢレジスタ７０４内のＰＯＲＴ　ＳＰＥＥＤ　ＲＥＧＩＳＴＥＲのプログラミングを行うが、これは各ポートの速度を定義するビットマップである。示した実施例では、１０または１００　ＭＨｚである。また、ＰＯＲＴ　ＴＹＰＥ　ＲＥＧＩＳＴＥＲも定義されるが、これはＱＣとＴＬＡＮ間のポートのタイプを定義するビットマップである。普通、これらのレジスタは動作中に変更されることはないが、必要に応じて再プログラミングすることもできる。

【０１２１】

ＰＣＢレジスタ７０４のその他のレジスタは動作中に使用される。例えば、ＰＣＢレジスタはＩＮＴＥＲＲＵＰＴ　ＳＯＵＲＣＥレジスタおよびＰＯＬＬＩＮＧ　ＳＯＵＲＣＥレジスタを含む。ＩＮＴＥＲＲＵＰＴ　ＳＯＵＲＣＥレジスタは１組の割り込みビット、ＭＣＢ＿ＩＮＴ、ＭＥＭ＿ＲＤＹ、ＰＫＴ＿ＡＶＡＩＬ、ＢＵＦ＿ＡＶＡＩＬ、ＡＢＯＲＴ＿ＰＫＴ、およびＳＴＡＴ＿ＲＤＹを含む。ＰＫＴ＿ＡＶＡＩＬおよびＢＵＦ＿ＡＶＡＩＬ割り込みビットは、ＰＣＢ＿ＰＫＴ＿ＡＶＡＩＬ＊およびＰＣＢ＿ＢＵＦ＿ＡＶＡＩＬ＊信号にそれぞれ対応する。少なくとも１つのＣＰＵ＿ＩＮＴ＊信号がＣＰＵ　２３０に用意され、このＣＰＵ＿ＩＮＴ＊信号がアサートされたときＣＰＵ　２３０がＩＮＴＥＲＲＵＰＴ　ＳＯＵＲＣＥレジスタを読み取って割り込み元を特定する。ＭＣＢ＿ＩＮＴ割り込みビットは、割り込みがＭＣＢ　４０４内で発生したことをＣＰＵ　２３０に知らせ

る。ＭＥＭ＿ＲＤＹ割り込みビットは、要求されたメモリ２１２のデータがＦＩＦＯ ４３０内に存在することをＣＰＵ ２３０に通知する。ＰＫＴ＿ＡＶＡＩＬ割り込みビットは、ＣＰＵ ２３０が処理すべきパケット・データが存在することをＣＰＵ ２３０に通知する。ＢＵＦ＿ＡＶＡＩＬ割り込みビットは、ＣＰＵ ２３０がパケット・データを送るために使用するバッファ・スペースがあることをＣＰＵ ２３０に通知する。ＡＢＯＲＴ＿ＰＫＴは、ＡＢＯＲＴ＿ＩＮ＊信号がアサートされたことをＣＰＵ ２３０に通知する。ＳＴＡＴ＿ＲＤＹ割り込み信号は、要求されたＱＣデバイス２０２からの統計情報がＦＩＦＯ ４３０内に存在することをＣＰＵ ２３０に通知する。ＰＯＬＬＩＮＧ ＳＯＵＲＣＥレジスタは、割り込みがマスクされてポーリング方式が適用されている場合の、各割り込みビットのコピーを含む。

【０１２２】

ＣＰＵインタフェース・ロジック７００は、ＦＩＦＯ ４３４内の６４バイトの受信バッファＲＸ ＢＵＦ ７１０にデータを供給し、これがＨＣＢバス４３８上のデータをアサートする。ＦＩＦＯ ４３４内の送信バッファＴＸ ＢＵＦ７１２は、ＣＰＵインタフェース・ロジック７００にデータを供給すべく、ＨＣＢバス４３８から受信データを受け取る。ＣＰＵインタフェース・ロジック７００は、バス信号７０５でＲＸ ＢＵＦ ７１０、ＴＸ ＢＵＦ ７１２、およびＱＣ／ＨＣＢインタフェース４３６と結合されている。ＱＣ／ＨＣＢインタフェース４３６は、ＣＰＵインタフェース・ロジック７００、ＲＸおよびＴＸ ＢＵＦ ７１０、７１２、およびＨＣＢバス４３８と結合しており、ＨＣＢ ４０２とＰＣＢ ４０６との間のデータ転送を制御する。

【０１２３】

図２６は、ＣＰＵインタフェース７００の詳細ブロック図である。ＣＰＵ制御／状態信号２１８ｂは制御ロジック７１３にアサートされる。制御ロジック７１３は、ＣＰＵトラッカ状態マシン７１７およびオルターネット・メモリ・コントロール状態マシン７１８と結合している。ＣＰＵバス２１８のアドレス／データ・ポーション２１８ａは多重化されたバスであり、ＰＣＢ ４０６の他の部分からのデータがＣＰＵ ２３０へのＣＰＵアドレス／データ・ポーション２１８ａ上でアサートされるべく、バス・イネーブル・ロジック７１６に供給される。ＣＰＵ ２３０はアドレス復号／要求生成ロジック７１４をアサートし、そのロジック７１４は複数の要求信号をＣＰＵトラッカ状態マシン７１７およびオルターネット・メモリ・コントロール状態マシン７１８を含むＰＣＢ ４０６の他の部分に供給する。１組のＣＰＵ情報ラッチ７１５はＣＰＵ ２３０からアドレスおよびデータを受け取り、本明細書で後に詳述するように、ＰＣＢ ４０６の他の部分へのラッチされたアドレスおよびラッチされたデータをアサートする。ＣＰＵサイクルを監視し、制御するために、ＣＰＵ制御信号がアドレス復号／要求生成ロジック７１４、ＣＰＵトラッカ状態マシン７１７、およびオルターネット・メモリ・コントロール状態マシン７１８間に供給される。

【０１２４】

図２７は、ＱＣ／ＨＣＢインタフェース・ロジック７０２の詳細ブロック図である。ＱＣ／ＨＣＢインタフェース・ロジック７０２は、ＣＰＵ ２３０とＱＣデバイス２０２との間で、例えばＣＰＵ ２３０の３２ビットとＱＣデバイス２０２の１６ビット間のフォーマット変換のような、一般に比較的トランスペアレントなインタフェースを実現するように動作する。ＲＥＧＩＳＴＥＲ ＲＥＱＵＥＳＴ信号がアドレス復号／要求生成ロジック７１４からＣＰＵトラッカ状態マシン７１７に供給され、ＣＰＵトラッカ状態マシン７１７は、１６ビットと３２ビット間のフォーマット変換のためにディスアセンブリ／アセンブリ状態マシン７２２に結合されている。ディスアセンブリ／アセンブリ状態マシン７２２は、バス７０１を介してＣＰＵインタフェース７００と、およびＱＣ／ＣＰＵバス２０４を介してＱＣデバイス２０２とそれぞれインタフェースするために、１組のデータ、アドレス、制御信号ドライバ／レシーバ７２４に結合している。統計バッファ７２６は、ＱＣ／ＣＰＵバス２０４から統計データおよびその他の情報を受け取り、そのデータをバス７０１を介してＣＰＵインタフェース７００に供給する。ＳＴＡＴＩＳＴＩＣＳ ＲＥＱ

ＵＥＳＴ信号が、アドレス復号／要求生成ロジック７１４からディスアセンブリ／アセンブリ状態マシン７２２とＱＣ／ＣＰＵバス状態マシン７３０に結合している、スタティスティクス・リクェスト状態マシン７２８に供給される。ＱＣ／ＣＰＵバス状態マシン７３０はさらに、ディスアセンブリ／アセンブリ状態マシン７２および１組のデータ、アドレス、制御信号ドライバ／レシーバ７２４に結合している。このようにして、ポート１０４の統計およびその他の情報収集、さらにポート１０４の構成変更のために、ＣＰＵ２３０はデータの流れやＨＳＢ　２０６の動作を妨げることなくＱＣデバイス２０２に対して比較的完全で独立したアクセスができるようになっている。

【０１２５】

ＣＰＵ２３０は、ＰＣＢレジスタ７０４内のＱＣ　ＳＴＡＴＩＳＴＩＣＳ　ＩＮＦＯＲＭＡＴＩＯＮレジスタに書き込むことによってＥＰＳＭ　２１０に対しＱＣデバイス２０２から統計および状態情報を取り出すよう要求する。ＣＰＵ　２３０は、ＱＣデバイス２０２の１つに対応する番号、ポート番号、指定したポートに関する開始レジスタの番号、および指定したポートについて読み取るべきレジスタの数を供給して統計情報を要求する。図２７に示されるように、ＱＣ　ＳＴＡＴＩＳＴＩＣＳ　ＩＮＦＯＲＭＡＴＩＯＮレジスタへの書き込みによってＱＣ　ＳＴＡＴＩＳＴＩＣＳ　ＲＥＱＵＥＳＴ信号がアサートされる。統計リクエスト状態マシン７２８は、１組のデータ、アドレス、制御信号ドライバ／レシーバ７２４を経由してＱＣ／ＣＰバス２０４上で指示された要求を行う。ＣＰＵインタフェース７００は、当該のＣＨＩＰ　ＳＥＬＥＣＴｍ＊信号を用いて１つか複数の当該のＱＣデバイス２０２に対して必要なリード・サイクルを実行し、その情報を統計バッファ７２６に書き込む。

【０１２６】

要求されたデータがすべて取り出されて統計バッファ７２６に格納されると、ＣＰＵインタフェース７００はＰＣＢレジスタ内のＰＯＬＬＩＮＧ　ＳＯＵＲＣＥレジスタのＳＴＡＴ＿ＲＤＹビットを更新し、ＩＮＴＥＲＲＵＰＴ　ＳＯＵＲＣＥレジスタ内のＳＴＡＴ＿ＲＤＹ割り込みビットをセットする。ＥＰＳＭ２１０はＣＰＵ２３０へのＣＰＵ＿ＩＮＴ＊信号をアサートし、ＣＰＵ２３０はこれに応答してＩＮＴＥＲＲＵＰＴ　ＳＯＵＲＣＥレジスタを読み取り、割り込み元を特定する。もし割り込みがマスクされていれば、ＣＰＵ２３０はポーリング・ルーチン中にＰＯＬＬＩＮＧ　ＳＯＵＲＣＥレジスタのＳＴＡＴ＿ＲＤＹビットを検知する。このようにして、ＣＰＵ２３０は要求が割り込みか、割り込みがマスクされていればポーリングのメカニズムによって完了したことを判断する。もしポーリング・メカニズムを適用するのであれば、プログラミングによってＳＴＡＴ＿ＲＤＹ割り込みを必要に応じてマスクすることができる。ＣＰＵ２３は、応答方式により１つまたは連続した複数のプロセッサ・サイクルで、統計バッファ７２６から統計情報をすべて取り出す。ＣＰＵバス２１８上のプロセッサ・サイクルは標準どおりのプロセッサ・バス・サイクルでよいが、大量データの転送にはバーストのサイクルが望ましい。

【０１２７】

勿論いくつかの別の実施形態が企図される。第１の別の実施形態においては、ＣＰＵ２３０がＱＣデバイス２０２のいずれかに対応する番号を供給するだけで、ＥＰＳＭ２１０が応答してＱＣデバイス２０２のすべてのポートのすべてのレジスタ３０６のデータを全部収集する。第２の代案実施例においては、ＣＰＵ　２３０がグローバルな統計情報の要求を出すだけで、すべてのＱＣデバイス２０２のすべてのレジスタ３０６の情報が収集される。しかし、ＣＰＵ２３０は一回につきポート１０４の１つだけの情報を必要とする点に留意する。

【０１２８】

ＣＰＵ２３０が、ＥＰＳＭ　２１０に対するただ１回の要求でポート１０４のいずれに関する統計情報をも、すべて取り出すことができることは高い評価に値する。特に、要求を出す場合はＱＣ　ＳＴＡＴＩＳＴＩＣＳ　ＩＮＦＯＲＭＡＴＩＯＮレジスタが１つのコマンドでＣＰＵ２３０によって書き込まれる。その後ＣＰＵ２３０は、ＱＣデバイス２０２からの応答の待機に拘束されることなく、自由に他のタスクを実行に移ることができる。

その代わりにＥＰＳＭ　２１０がＱＣ／ＣＰＵバス２０４を介して個々の統計読み取り要求を実行し、全部のデータを収集する。ＣＰＵ２３０に対する通知が割り込み信号、もしくはポーリング・メカニズムによって行われ、ＣＰＵ２３０は要求したすべての情報を取り出すことができる。その結果、ＣＰＵ２３０のプロセッサ時間の使用効率が向上する。

【０１２９】

図２８は、ＣＰＵインタフェース７００とＭＣＢ　４０４間のインタフェースの詳細ブロック図である。アドレス復号／要求生成ロジック７１４からのメモリ要求信号が、アドレス生成ロジック７４６およびＦＩＦＯ状態／割り込み生成ロジック７４２に結合しているＦＩＦＯアクセス状態マシン７４０に供給される。ＲＸ　ＢＵＦ　７０６およびＴＸ　ＢＵＦ　７０８を含むＦＩＦＯブロック７４８が、アドレス生成ロジック７４６とＦＩＦＯ状態／割り込み生成ロジック７４２に結合している。アドレス生成ロジック７４６およびＦＩＦＯ状態／割り込み生成ロジック７４２は、バス７０３を介してＣＰＵインタフェース７００と、およびＭＣＢバス４２８を介してＭＣＢ４０４とそれぞれインタフェースするために、両方とも１組のデータ、アドレス、制御信号ドライバ／レシーバ７４４に結合している。

【０１３０】

図２９は、ＣＰＵインタフェース７００とＨＣＢ４０２間のインタフェースの詳細ブロック図である。アドレス復号／要求生成ロジック７１４からのパケット読み出し要求信号が、ＴＸ　ＢＵＦ　７１２を含む送信バッファ７６２に結合されている送信パケット状態マシン７６０に供給される。アドレス復号／要求生成ロジック７１４からのパケット書き込み要求信号が、ＲＸ　ＢＵＦ　７１０を含む受信バッファ７７０に結合されている受信パケット状態マシン７６８に供給される。送信バッファ７６２および受信バッファ７７０は、バス７０５を介してＣＰＵインタフェース７００と、およびＨＣＢバス４３８を介してＨＣＢ　４０２とそれぞれインタフェースするために、両方とも１組のデータ、アドレス、制御信号ドライバ／レシーバ７６４に結合している。

【０１３１】

次に図３０を参照する。ＴＰＩ　２２０の簡略ブロック図であり、これの全体を示している。ＴＰＩ２２０は、ＨＳＢ２０６とＰＣＩバス２２２との間に介在してデータ転送を行い、ＴＬＡＮ　２２６とＥＰＳＭ　２１０との間でネットワーク・データの受け渡しを行う。ＴＰＩ２２０はＨＳＢ２０６上でスレーブとして動作し、ＥＰＳＭ２１０のポーリングに応答し、そしてＱＣデバイス２０２と同じようにＥＰＳＭ２１０とデータの受け渡しを行う。ＰＣＩバス２２２側では、ＴＰＩ２２０がＰＣＩバス２２２を介して４つのＴＬＡＮ２２６（ＰＯＲＴ２４、ＰＯＲＴ２５、ＰＯＲＴ２６、およびＰＯＲＴ２７）のそれぞれとネットワーク・データの受け渡しを行う。

【０１３２】

ＴＰＩ　２２０は、ＨＳＢコントローラ８０４、ＰＣＩバス・コントローラ８０２、およびメモリ８０６を含む。ＰＣＩバス・コントローラ８０２は、ＰＣＩバスの標準に従ってＰＣＩバス２２２とインタフェースし、ＴＰＩ２２０とＰＣＩバス２２２との間のデータ転送を簡便化する。ＰＣＩバス標準は、ＩｎｔｅｌＡｒｃｈｉｔｅｃｔｕｒｅ　Ｌａｂとその業界のパートナー各社によって定義されているものである。ＨＳＢコントローラ８０４は、ＨＳＢ２０６の定義済み動作に従ってＨＳＢ　２０６とインタフェースし、ＴＰＩ２２０とＥＰＳＭ２１０との間のデータ転送を簡便化する。メモリ８０６は１個所に集中あるいは分散配置することができ、複数のデータ・バッファ８０７および１つの制御リスト・メモリ８０８を含む。データ・バッファ８０７は、ＰＣＩバス２２２とＨＳＢ　２０６との間のデータ転送を簡便化するための一時的なメモリとして機能する。制御リスト・メモリ８０８はＰＣＩバス２２２上における各ＴＬＡＮ　２２６のバス・マスタ動作を簡便化する。

【０１３３】

次に図３１を参照する。ＴＰＩ　２２０の詳細ブロック図である。ＴＰＩ２２０は、ＰＣＩバス２２２とのインタフェースに用いられるバッファ、ドライバ、および関連の回路を

含んだＰＣＩバス・インタフェース・ロジック８１０を含む。本実施例のＰＣＩバス２２２は、データ幅が３２ビットで３３　ＭＨｚのクロック周波数で動作する。しかし、ＰＣＩバス２２２のデータ幅は特にこれでなくてもよく、また動作クロックも、例えば６６ＭＨｚといった任意の、あるいは使用可能ないずれの周波数でも構わないことはもとより理解されている。ＴＰＩ２２０はＰＣＩアービタ８１１を含み、これがＰＣＩバス２２２へのアクセスとこれの制御についてＴＬＡＮ　２２６、ＴＰＩ　２２０、およびＣＰＵ　２３０のそれぞれの間で調停を行う。特に、ＴＬＡＮ　２２６、ＴＰＩ　２２０、およびＣＰＵ２３０は、それぞれいくつかの要求信号ＲＥＱｍの１つをアサートしてＰＣＩバス２２２の制御を要求する。ＲＥＱｍ信号はＰＣＩアービタ８１１に受け取られる。ＰＣＩアービタ８１１は、応答してそれぞれの許可信号ＧＮＴｍをアサートすることによって要求しているデバイスの１つに制御を許可する。ＰＣＩアービタ８１１は必要に応じて他のアービトレーション・スキームを適用することもできるが、図解した実施形態においては、ＰＣＩアービタ８１１による調停はラウンドロビン法に基づいている。１つのＴＬＡＮ　２２６にＰＣＩバス２２２の制御を許可した後で、ＰＣＩアービタ８１１はＴＬＡＮ選択信号（ＴＳＥＬｍ）をアサートしてその特定のＴＬＡＮ　２２６を識別する。

【０１３４】

ＴＰＩ２２０は、ＴＰＩ２２０とＨＳＢ　２０６とのインタフェースに用いるバッファ、ドライバ、および関連の回路を含んだＨＳＢデータ転送インタフェース・ロジック８１９を含む。ＨＳＢデータ送信インタフェース・ロジック８１９は、ＨＳＢ　２０６上における同時リード／ライト・サイクルのためにリード・ラッチ８１９ａおよびライト・ラッチ８１９ｂを含む。ＨＳＢデータ転送インタフェース・ロジック８１９は、ＥＰＳＭ　２１０のポーリングに応答し、ＨＳＢ２０６上で実行されているサイクルを監視するために、ポート状態ロジック８２０を含む。特にポート状態ロジック８２０は、ＳＴＲＯＢＥ＊信号がＥＰＳＭ２１０によってアサートされればそれを検知し、応答してＰＫＴ＿ＡＶＩＡＬ［６］＊およびＢＵＦ＿ＡＶＡＩＬ［６］＊信号をＴＰＩ　２２０のデータ状態に基づいて多重化の方式でアサートする。ポート状態ロジック８２０は、ＲＥＡＤ＿ＯＵＴ＿ＰＫＴ［６］　＊およびＷＲＩＴＥ＿ＩＮ＿ＰＫＴ［６］＊信号をそれぞれアサートし、ＴＰＩ２２０に意図されたＨＳＢ　２０６上でのリードおよびライト・サイクルも検知する。ＴＰＩ２２０からＥＰＳＭ　２１０へのＨＳＢバス２０６を介したパケット・データの転送中、ポート状態ロジック８２０は転送されているデータがパケットの始め、またはパケットの終わりであれば、それぞれＳＯＰ＊またはＥＯＰ＊信号をＨＳＢ　２０６のバス・サイクルの期間アサートする。ＥＰＳＭ２１０からＴＰＩ　２２０へのＨＳＢバス２０６を介したパケット・データの転送中、ポート状態ロジック８２０はＳＯＰ＊またはＥＯＰ＊信号を読み取って、受信されているデータがパケットの始めであるか、あるいはパケットの終わりであるかを判断する。

【０１３５】

データ・バッファ８０７はいくつかの双方向ＦＩＦＯデータ・バッファ、８０７ａ　、８０７ｂ、８０７ｃ、および８０７ｄ（８０７ａ－ｄ）を含み、それぞれは３２ビット幅の送信バッファ（ＴＰＩ　ＴＸ　ＦＩＦＯ）および３２ビット幅の受信バッファ（ＴＰＩ　ＲＸ　ＦＩＦＯ）を含む。示した実施形態において、データ・バッファ８０７ａ、８０７ｂ、８０７ｃ、および８０７ｄは、それぞれポートＰＯＲＴ２４、ＰＯＲＴ２５、ＰＯＲＴ２６、およびＰＯＲＴ２７に対応する。各ＴＰＩ　ＲＸ　ＦＩＦＯは、ＰＣＩバス２２２を介してそれぞれのＴＬＡＮ　２２６からデータを受け取り、そのデータはＴＰＩ２２０によりＨＳＢ２０６を介してＥＰＳＭ２１０に送られる。各ＴＰＩ　ＴＸ　ＦＩＦＯは、ＨＳＢ２０６を介してＥＰＳＭ２１０からデータを受け取り、そのデータはＴＰＩ２２０によりＰＣＩバス２２２を介してそれぞれのＴＬＡＮ　２２６へ送られる。

【０１３６】

受信リスト復号ロジック８１２はＰＣＩバス・インタフェース・ロジック８１０に結合されており、少なくとも１つの受信制御リストを制御リスト・メモリ８０８の一部である受信制御リスト・メモリ（ＲＸ　ＣＮＴＬ　ＬＩＳＴ）８０８ａに格納する。受信リスト復

号ロジック８１２は、ＰＣＩバス２２２上のアドレスとしてアサートされたＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳに応答し、ＰＣＩバス２２２へのデータとしてＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａからの受信制御リストの書き込みを行う。示した実施形態においては、ＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａは一時に１つの受信制御リストを保持する。特に、それぞれのＴＬＡＮ　２２６はＰＣＩバス２２２の制御権を得てＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳをＰＣＩバス２２２上でアサートし、対応する受信制御リストをＲＸ　ＣＮＴＬ　ＬＩＳＴ　８０８ａから受け取る。受信制御リストは、ＴＬＡＮ　２２６が使用するＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳを含み、これは受信データを格納する場所を示すアドレスである。それぞれのポートからのデータ・パケットの受信に応答し、ＴＬＡＮ２２６は再びＰＣＩバス２２２の制御権を得て、受信データ・パケットからのデータを、予め取り出してある受信制御リスト内に格納されているアドレスを用いてＴＰＩ２２０へ転送する。本明細書で後に詳述するように、ＴＬＡＮ　２２６は調停を行ってＰＣＩバス２２２の制御を許可され、ＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳをＰＣＩバス２２２上でライト・サイクル中にアサートする。

【０１３７】

受信データ復号ロジック８１３、ＰＣＩ　ＲＸ　ＦＩＦＯ制御ロジック８１７、ＰＣＩアービタ８１１、およびＦＩＦＯ同期ロジック８１８が、ＰＣＩバス・インタフェース・ロジック８１０から対応するＴＰＩ　ＲＸ　ＦＩＦＯへの受信データの流れを制御する。ＰＣＩ　ＲＸ　ＦＩＦＯ制御ロジック８１７は、ＰＣＩバス・インタフェース・ロジック８１０からのデータを受け取る入力、およびそれぞれが対応するＴＰＩ　ＲＸ　ＦＩＦＯに結合されているいくつかの選択可能な出力を含む。ＰＣＩアービタ８１１はＴＳＥＬｍ信号をＦＩＦＯ同期ロジック８１８に供給し、これがＰＣＩ　ＲＸ　ＦＩＦＯ制御ロジック８１７への対応するＰＣＩバッファ選択信号（ＰＢＳＥＬｍ）をアサートし、ＰＣＩバス２２２へのアクセスが許可されている特定のＴＬＡＮ　２２６に基づいて適当なＴＰＩＲＸ　ＦＩＦＯを選択する。受信データ復号ロジック８１３は、ＰＣＩバス２２２上でライト・サイクルを実行中のＴＬＡＮ　２２６によってアサートされたＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳを受け取って復号化し、応答してＰＣＩ　ＲＸ　ＦＩＦＯ制御ロジック８１７への受信イネーブル信号（ＲＥＮ）をアサートして選択したＴＰＩ　ＲＸ　ＦＩＦＯにデータを渡す。

【０１３８】

ＰＣＩバス２２２とＨＳＢ　２０６との間における双方向データ・フローは、データ・バッファ８０７を介して実現されることに留意する。１つの実施形態において、ＰＣＩバス２２２とＨＳＢ　２０６は３３　ＭＨｚといった等しい速度で動作するが、代案の実施形態では異なったクロック周波数で動作することも考えられる。例えば別の実施形態において、ＨＳＢ　２０６が３３　ＭＨｚで動作し、一方ＰＣＩバス２２２で６６　ＭＨｚ動作する。クロックの速度が異なっても、ＴＰＩ２２０の機能が遂行されてデータ・フローを処理して同期が実現される。データ・バッファ８０７ａ－ｄのそれぞれのＴＰＩ　ＲＸ　ＦＩＦＯおよびＴＰＩ　ＴＸ　ＦＩＦＯは、データの書き込みと読み出しのためにポインタを両端に保持したサーキュラ・バッファとしての機能の遂行が望ましい。ＦＩＦＯ同期ロジック８１８は、一般に各ＦＩＦＯの両端のポインタの同期、保持、および更新のために動作し、適当なＴＰＩ　ＦＩＦＯとの正しいデータの読み書きを保証する。

【０１３９】

前述したように、各ＴＰＩ　ＲＸ　ＦＩＦＯはサーキュラ・バッファとして機能を遂行する。ＰＣＩ　ＲＸ　ＦＩＦＯ制御ロジック８１７はいくつかのＰＣＩ受信ポインタ（ＰＣＩ　ＲＸ　ＰＴＲ）を含み、選択されたＴＰＩ　ＲＸ　ＦＩＦＯ内の１ＤＷＯＲＤ（３２ビット）のデータを受け取る次のロケーションを指示あるいはアドレスするために、それぞれのＴＰＩ　ＲＸ　ＦＩＦＯに１つのポインタが充てられている。同様に、ＨＳＢ　ＲＸ　ＦＩＦＯ制御ロジック８２１が各ＴＰＩ　ＲＸ　ＦＩＦＯの他端にあり、いくつかのＰＣＩ受信「シンクロナイズド」ポインタ（ＰＣＩ　ＲＸ　ＳＰＴＲ）を含み、これらの

ポインタは、それぞれが対応する１つのＰＣＩ　ＲＸ　ＰＴＲのシンクロナイズされたコピーである。適当なＴＰＩ　ＲＸ　ＦＩＦＯを選択するためのＰＢＳＥＬm信号とともに、ＦＩＦＯ同期ロジック８１８も複数のＰＣＩカウント信号（ＰＣＮＴm）の対応する１つをアサートし、ＰＣＩ　ＲＸ　ＦＩＦＯ制御ロジック８１７内の当該のＰＣＩ　ＲＸ　ＰＴＲの同期的な更新すなわち増分を行う。ＦＩＦＯ同期ロジック８１８は、さらに複数のＨＳＢカウント信号（ＨＣＮＴm）の対応する１つをアサートし、ＨＳＢ　ＲＸ　ＦＩＦＯ制御ロジック８２１内の当該のＰＣＩＲＸ　ＳＰＴＲの同期的な更新すなわち増分を行う。このように、それぞれのＴＰＩ　ＲＸ　ＦＩＦＯの両端に１つずつ用意されたポインタによって、データを挿入すべき場所が指示される。

【０１４０】

ＰＣＩ　ＴＸ　ＦＩＦＯ制御ロジック８１６は、ＴＰＩ　ＴＸ　ＦＩＦＯのいずれかの中でデータを検出し、送信すべきデータを持っているＴＰＩ　ＴＸ　ＦＩＦＯに対応するＴＬＡＮ２２６対してコマンドを送るため、ＴＰＩ２２０にＰＣＩバス２２２の制御を要求させ、その制御を得させる。ＰＣＩ　ＴＸ　ＦＩＦＯ制御ロジック８１６は、１組のＴＰＩ制御レジスタ８４６から当該のＴＬＡＮ２２６のアドレスにアクセスする。ＴＰＩ２２０は当該のＴＬＡＮ２２６にコマンドを書き込み、ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳを用意し、ＴＬＡＮ２２６にそのＴＲＡＮＳＭＩＴ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳを使用するＴＰＩ２２０から送信制御リストを続いて要求させる。

【０１４１】

送信リスト・デコード・ロジック８１４は、ＰＣＩバス・インタフェース・ロジック８１０に結合されており、少なくとも１つの送信コントロール・リストをコントロール・リスト・メモリ８０８の一部である送信コントロール・リスト・メモリ（ＴＸ　ＣＮＴＬ　ＬＩＳＴ）８０８ｂに格納する。送信リスト・コントロール・ロジック８１４は、ＰＣＩバス２２２上のアドレスとしてアサートされた送信リスト・メモリ・ベース・アドレス（ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）に応答し、ＰＣＩバス２２２へのデータとしてＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂからの送信コントロール・リストの書き込みを行う。示した実施例においては、ＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂは、一時に１つの送信コントロール・リストを保持する。このようにして、それぞれのＴＬＡＮ　２２６はＰＣＩバス２２２の制御権を得て、ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳをＰＣＩバス２２２上でアサートし、対応する送信コントロール・リストをＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂから受け取る。送信コントロール・リストを取り出した後、ＴＬＡＮ２２６はＰＣＩバス２２２を要求し、そのバスの制御権を得ることによってその送信コントロール・リストを実行し、リード・サイクルを１回実行して、ＴＰＩ２２０の対応するＴＰＩ　ＴＸ　ＦＩＦＯからパケット・データ・メモリ・ベース・アドレス（ＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）を用いてデータを取り出す。

【０１４２】

送信データ・デコード・ロジック８１５、ＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６、ＰＣＩアービタ８１１、およびＦＩＦＯ同期ロジック８１８が、データ・バッファ８０７の各ＴＰＩ　ＴＸ　ＦＩＦＯからＰＣＩバス２２２へのデータの流れを制御する。ＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６は、ＰＣＩバス・インタフェース・ロジック８１０へデータを供給する出力、およびそれぞれがＴＰＩ　ＴＸ　ＦＩＦＯの対応する１つに結合されているいくつかの選択可能な入力を含む。ＴＬＡＮ２２６がデータを読み取るべくＰＣＩバス２２２上でリード・サイクルを実行するとき、ＰＣＩアービタ８１１はＴＳＥＬm信号をＦＩＦＯ同期ロジック８１８に供給し、これがＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６へのＰＢＳＥＬm信号をアサートし、ＰＣＩバス２２２の制御権を持っている特定のＴＬＡＮ２２６に基づいて、対応するＴＰＩ　ＴＸ　ＦＩＦＯを選択する。送信データ・デコード・ロジック８１５は、ＴＬＡＮ２２６によってアサートされたＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥ

ＳＳを受け取って復号化し、それに応答して、ＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６への送信イネーブル信号（ＴＥＮ）をアサートすることによって、選択されたＴＰＩ　ＴＸ　ＦＩＦＯへのデータの転送を可能とする。ＰＢＳＥＬm信号がＰＣＩ　ＲＸ　ＦＩＦＯコントロール・ロジック８１７とＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６の両方に供給されること、そしてＴＥＮおよびＲＥＮ信号によるＰＣＩ　ＲＸ　ＦＩＦＯコントロール・ロジック８１７とＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６との間の選択が、サイクルのタイプ、およびデータ・フローの方向に依存していることに留意する必要がある。

【０１４３】

示された本実施例において、各ＴＰＩ　ＴＸ　ＦＩＦＯはサーキュラ・バッファとして機能を遂行する。ＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６はいくつかのＰＣＩ送信ポインタ（ＰＣＩ　ＴＸ　ＰＴＲ）を含み、１つのデワード（ＤＷＯＲＤ）のデータを読み出すべき次のロケーションを指示あるいはアドレス指定するために、それぞれのＴＰＩ　ＴＸ　ＦＩＦＯに１つのポインタが充てられている。同様に、ＴＰＩ　ＴＸ　ＦＩＦＯの他端にある、本明細書で後に詳述するＨＳＢ　ＴＸ　ＦＩＦＯコントロール・ロジック８２２は、いくつかのＰＣＩ送信「同期（シンクロナイズド）」ポインタ（ＰＣＩ　ＴＸ　ＳＰＴＲ）を含み、これらのポインタは、それぞれが対応する１つのＰＣＩ　ＴＸ　ＰＴＲの同期されたコピーである。ＦＩＦＯ同期ロジック８１８は、ＰＣＩ　ＴＸＦＩＦＯコントロール・ロジック８１６から１つのＤＷＯＲＤのデータがＰＣＩバス２２２に供給される度に、対応する１つのＰＣＮＴm信号をアサートして当該のＰＣＩ　ＴＸ　ＰＴＲを増分し、対応する１つのＨＣＮＴm信号をアサートして当該のＰＣＩ　ＴＸ　ＳＰＴＲを増分する。このように、それぞれのＴＰＩ　ＴＸ　ＦＩＦＯの両端に１つずつ用意されたポインタによって、データを読み出すべき場所が指示される。

【０１４４】

ＨＳＢ　ＲＸ　ＦＩＦＯコントロール・ロジック８２１は、それぞれがＴＰＩＲＸ　ＦＩＦＯの対応する１つの出力に結合された幾つかの選択可能な入力を持っている。ＨＳＢ　ＲＸ　ＦＩＦＯコントロール・ロジック８２１は、ＨＳＢ２０６上でアサートされるべきデータをＨＳＢデータ転送インタフェース・ロジック８１９に供給するための１つの出力を持っている。ＨＳＢ　ＴＸ　ＦＩＦＯコントロール・ロジック８２２は、それぞれがＴＰＩ　ＴＸ　ＦＩＦＯの対応する１つの入力に結合された幾つかの選択可能な出力を持っている。ＨＳＢ　ＴＸ　ＦＩＦＯコントロール・ロジック８２２は、ＨＳＢデータ転送インタフェース・ロジック８１９からＨＳＢ２０６を介してデータを受け取るための１つの入力を持っている。

【０１４５】

ＨＳＢ　ＲＸ　ＦＩＦＯコントロール・ロジック８２１、ポート状態ロジック８２０、およびＦＩＦＯ同期ロジック８１８は、ＴＰＩ２２０からＥＰＳＭ２１０へのデータ転送中、データ・バッファ８０７ａ～８０７ｄのＴＰＩ　ＲＸ　ＦＩＦＯとＨＳＢ２０６との間におけるデータの流れを制御する。ポート状態ロジック８２０は、ＨＳＢ２０６上におけるリード・サイクルを示ＲＥＡＤ＿ＯＵＴ＿ＰＫＴ［６］＊信号がアサートとされたときにそれを検知し、選択されているポートの対応するＴＰＩ　ＲＸ　ＦＩＦＯを識別すべくＰＯＲＴ＿ＮＯ［１：０］信号をデコードする。特に、ＥＰＳＭ２１０は、ＰＯＲＴ＿ＮＯ［１：０］信号００、０１、１０、または１１をアサートして、ポートＰＯＲＴ２４、ＰＯＲＴ２５、ＰＯＲＴ２６、またはＰＯＲＴ２７にそれぞれ対応するデータ・バッファ８０７ａ、８０７ｂ、８０７ｃ、または８０７ｄの１つのＴＰＩ　ＲＸ　ＦＩＦＯを選択する。ポート状態ロジック８２０は、ＦＩＦＯ同期ロジック８１８へのポート選択信号（ＰＳＥＬm）をアサートして選択されたポートを表示し、　ＦＩＦＯ同期ロジック８１８が応答して対応するＨＳＢ選択信号（ＨＢＳＥＬm）をアサートし、対応するＴＰＩ　ＲＸ　ＦＩＦＯに結合されているＨＳＢ　ＲＸ　ＦＩＦＯ制御ロジック８２１の１つの出力を選択する。また、ポート状態ロジック８２０がＨＳＢイネーブル信号（ＨＲＥＮ）をアサートすることにより、ＨＳＢ　ＲＸ　ＦＩＦＯ制御ロジック８２１は、ＨＳＢ２０６上

でアサートされるべきデータをＨＳＢデータ転送インタフェース・ロジック８１９に供給することができる。

【０１４６】

ＨＳＢ　ＲＸ　ＦＩＦＯコントロール・ロジック８２１は、ＴＰＩ　ＲＸ　ＦＩＦＯ内における特定のデータのロケーションを示すためのＨＳＢ受信ポインタ（ＨＳＢ　ＲＸ　ＰＴＲ）を、それぞれのＴＰＩ　ＲＸ　ＦＩＦＯについて１つずつ含む。ＦＩＦＯ同期ロジック８１８は、ＨＣＮＴｍ信号の対応する１つをアサートして、ＴＰＩ　ＲＸ　ＦＩＦＯからＤＷＯＲＤが１つ読み出される度に、選択されているＴＰＩ　ＲＸ　ＦＩＦＯの対応するＨＳＢ　ＲＸ　ＰＲＴを更新すなわち減分する。また、ＰＣＩ　ＲＸ　ＦＩＦＯコントロール・ロジック８１７は、対応するＨＳＢ　受信「同期」ポインタ（ＨＳＢ　ＲＸ　ＳＰＴＲ）を含み、これはＦＩＦＯ同期ロジック８１８がＰＣＮＴｍ信号の対応する１つをアサートすることによって減分される。このように、ＨＳＢ　ＲＸ　ＦＩＦＯコントロール・ロジック８２１は、ＴＰＩ　ＲＸ　ＦＩＦＯのそれぞれについて２つのポインタを含み、ＰＣＩ　ＲＸ　ＳＰＴＲはデータを書き込むべき場所を指示し、ＨＳＢ　ＲＸ　ＰＴＲはデータを読み出すべき場所を指示する。ポート状態ロジック８２０もこれらのポインタにアクセスし、各ＴＰＩ　ＲＸ　ＦＩＦＯ内の有効なデータの量あるいは有効なデータ・バイト数を引き出す。このカウントは（ＴＢＵＳの値に対応している）対応するＲＢＳＩＺＥと比較され、ＨＳＢ２０６が、ＳＴＲＯＢＥ＊信号に応答して、ＰＫＴ＿ＡＶＡＩＬ［６］＊信号をアサートする方法を決定する。

【０１４７】

ＨＳＢ　ＴＸ　ＦＩＦＯコントロール・ロジック８２２、ポート状態ロジック８２０、およびＦＩＦＯ同期ロジックは、ＥＰＳＭ２１０からＴＰＩ２２０へのデータ転送中、ＴＰＩ　ＴＸ　ＦＩＦＯとＨＳＢ２０６との間におけるデータの流れを制御する。ポート状態ロジック８２０はＷＲＩＴＥ＿ＩＮ＿ＰＫＴ［６］＊信号がアサートとされたときにそれを検知し、ＥＰＳＭ２１０がＨＳＢ２０６上で実行しているライト・サイクルの間に、ＰＯＲＴ＿ＮＯ［１：０］信号からポート番号を検出する。ポート状態ロジック８２０はそれに応答して、ＰＳＥＬｍ信号およびＨＳＢ送信イネーブル信号（ＨＴＥＮ）をアサートし、当該するＴＰＩ　ＴＸ　ＦＩＦＯを示す。ＦＩＦＯ同期ロジック８１８はそれに応答して、ＨＢＳＥＬｍ信号をアサートし、当該ＴＰＩ　ＴＸ　ＦＩＦＯに対してＨＳＢ　ＴＸ　ＦＩＦＯコントロール・ロジック８２２の対応する入力を選択する。ＨＴＥＮ信号によってＨＳＢ　ＴＸ　ＦＩＦＯコントロール・ロジック８２２がイネーブルされ、ＨＳＢデータ転送インタフェース・ロジック８１９から選択されたＴＰＩ　ＴＸ　ＦＩＦＯにアサートすべきデータを受け取る。

【０１４８】

ＨＳＢ　ＴＸ　ＦＩＦＯコントロール・ロジック８２２は、それぞれのＴＰＩＴＸ　ＦＩＦＯについて１つのＨＳＢ送信ポインタ（ＨＳＢ　ＴＸ　ＰＴＲ）を含み、これによって、データを書き込むべきＴＰＩ　ＴＸ　ＦＩＦＯ内の特定のロケーションが指示される。ＦＩＦＯ同期ロジック８１８はＨＣＮＴｍ信号の対応する１つをアサートし、選択されたＴＰＩ　ＴＸ　ＦＩＦＯに１つのＤＷＯＲＤが書き込まれる度に、その選択されたＴＰＩ　ＴＸ　ＦＩＦＯの対応するＨＳＢ　ＴＸ　ＰＲＴを更新すなわち増分する。また、ＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６は、対応するＨＳＢ送信「同期」ポインタ（ＨＳＢＴＸ　ＳＰＴＲ）を含み、これは、ＦＩＦＯ同期ロジック８１８がＰＣＮＴｍ信号の対応する１つをアサートすることによって増分される。このように、ＨＳＢ　ＴＸ　ＦＩＦＯコントロール・ロジック８２２はＴＰＩ　ＴＸ　ＦＩＦＯのそれぞれについて２つのカウンタを含み、ＰＣＩ　ＴＸ　ＳＰＴＲはデータを読み出すべき場所を指示し、ＨＳＢ　ＴＸ　ＰＴＲはデータを書き込むべき場所を指示する。ポート状態ロジック８２０もこれらのポインタにアクセスし、各ＴＰＩ　ＴＸ　ＦＩＦＯ内の使用可能なスペース量あるいは空のデータ・バイト数を取り出す。このカウントは（ＴＢＵＳの値に対応している）対応するＸＢＳＩＺＥと比較され、ＨＳＢ　２０６がＳＴＲＯＢＥ＊信号に応答して、ＢＵＦ＿ＡＶＡＩＬ［６］＊信号をアサートする方法を決定する。

【０１４９】
ＴＰＩ２２０内には１組のＴＰＩ　ＰＣＩコンフィギュレーション・レジスタ８３５が用意されており、ＰＣＩバス２２２を介したアクセスのために、ＰＣＩバス・インタフェース・ロジック８１０に結合されている。また、ＴＰＩコントロール・レジスタ８４６が用意されており、ＴＰＩ２２０内の各種のデバイス、およびＰＣＩバス２２２を介したアクセスのために、ＰＣＩバス・インタフェース・ロジック８１０に結合されている。これらのレジスタ８４６および８３５の内容や構造は、後に詳述する。ＨＳＢデータ転送インタフェース・ロジック８１９は、ＰＡＣＫＥＴ　ＳＩＺＥタグ・レジスタ８１９ｃも含む。ＨＳＢデータ転送インタフェース・ロジック８１９は、ＥＰＳＭ２１０から送られる各パケット・データの最初のＤＷＯＲＤを捉え、パケット・サイズ（ＰＡＣＫＥＴ　ＳＩＺＥ）タグ・レジスタ８１９ｃに格納し、該レジスタ８１９ｃの内容を送信リストデコード・ロジック８１４のＴＸ　ＣＮＴＬ　ＬＩＳＴに書き込む。

【０１５０】
次に図３２を参照する。各ＴＬＡＮ２２６の構成と機能を示すブロック図である。ＴＬＡＮ２２６は、イーサネット（Ｅｔｈｅｒｎｅｔ）・ポート１１０、ＰＣＩバス・インタフェース８２４、およびイーサネット・ポート１１０とＰＣＩバス・インタフェース８２４との間に結合されたメモリ８２５を含む。

【０１５１】
イーサネット・ポート１１０は、対応するネットワーク１１２との間におけるパケット・データの送受信のために、１００Ｍｂのイーサネット・セグメント１１４の適合するコネクタを受容するための適宜のソケットを含む。イーサネット・ポート１１０は、受信したパケット・データをメモリ８２５内のデータ・バッファ８２６に供給する。イーサネット・ポート１１０はデータ・バッファ８２６からデータを取り出し、そのパケット・データをイーサネット・セグメント１１４に送信する。

【０１５２】
ＴＬＡＮ２２６は、その動作を制御するための１組のレジスタ８２８をメモリ８２５内に含む。レジスタ８２８は、外部のデバイスがＰＣＩバス２２２を介してコマンドを挿入できるようにするために、コマンド・レジスタ８２８ａを含む。レジスタ８２８は、外部のメモリからＰＣＩバス２２２を介してコマンド・リストをアクセスするためのアドレスを格納する、チャネル・パラメータ・レジスタ８２８ｂをさらに含む。コマンド・レジスタ８２８ａは、ＴＬＡＮ２２６に対し、コマンド・リストを取り出して実行するように指示するための（示していないが）ＧＯビットを含む。コマンド・レジスタ８２８ａは、ＴＬＡＮ２２６に対し、（ＲＸの場合）受信コマンド・リストを、そして（ＴＸの場合）送信コマンド・リストを取り出して実行するように指示するための（示していないが）ＲＸ／ＴＸビットも含む。メモリ８２５は現在のコントロール・リストを格納するためのリスト・バッファ８２７を含み、さらにリスト・バッファ８２７は、現在の受信・コントロール・リストを格納するための受信コントロール・リスト・バッファ８２７ａ、およびカレントの送信コントロール・リストを格納するための送信コントロール・リスト・バッファ８２７ｂを含む。

【０１５３】
ＰＣＩバス・インタフェース８２４は、ＰＣＩバス２２２に結合し、データ転送中にＰＣＩバス２２２のバス・マスタを動作させることによって、ＴＰＩ２２０とＴＬＡＮ２２６との間のデータ転送を制御するためのロジックを含む。ＴＰＩ２２０やＣＰＵ２３０のような外部のデバイスは、チャネル・パラメータ・レジスタ８２８ｂにアドレスを書き込み、コマンド・レジスタ８２８ａにコマンドを書き込む。ＴＬＡＮ２２６はそれに応答してＲＥＱｍ信号をアサートし、ＰＣＩバス２２２を仲裁に委ねる。ＧＮＴｍ信号を受け取ると、ＴＬＡＮ２２６は指示されたコマンド・リストを受け取ってリスト・バッファ８２７に格納するため、ＰＣＩバス２２２上で１サイクルを実行する。コマンドは、ＲＸ／ＴＸビットがＴＸにセットされていれば送信コマンドとみなされ、ＲＸ／ＴＸビットがＲＸにセットされていれば受信コマンドとみなされる。

【０１５４】
受信動作を開始するために、ＣＰＵ２３０は、受信リスト・メモリ・ベース・アドレス（ＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）をチャネル・パラメータ・レジスタ８２８ｂに書き込み、受信コマンドを各ＴＬＡＮ２２６のコマンド・レジスタ８２８ａに書き込む。ＴＬＡＮ２２６は、応答してＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳを用いて受信コントロール・リストを取り出すべく、ＰＣＩバス２２２を要求する。ＴＰＩ２２０は受信コントロール・リストをＴＬＡＮ２２６に供給し、そしてＴＬＡＮ２２６は、データの受信を待ってから受信コントロール・リストを実行する。受信コントロール・リストは順方向ポインタを含み、ＴＬＡＮ２２６はそれを用いて次の受信コントロール・リストを取り出し、コントロール・リストのチェーン（連鎖）を形成する。しかし、望ましい実施例では、ＴＰＩ２２０が各受信コントロール・リストの順方向ポインタを同一のＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳとともにロードする。ポート１１０からのデータがＴＰＩ２２０に受信さる場合、ＰＣＩバス・インタフェース８２４は仲裁に委ねて、ＰＣＩバス２２２の制御権を得てから、その受信コントロール・リスト・バッファ８２７ａ内の受信コントロール・リストを実行して、データをＰＣＩバス２２２を介してＴＰＩ２２０に転送する。データ・パケット全体の転送が完了したとき、ＴＬＡＮ２２６は、現在の受信コントロール・リストの順方向ポインタ内のＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳを使用し、新しく別の受信コントロール・リストを要求する。

【０１５５】
送信動作について説明する。ＴＰＩ２２０がそのＴＰＩ　ＴＸ　ＦＩＦＯのいずれかから送信すべきデータを検知し、仲裁に委ねてＰＣＩバス２２２の制御権を獲得する。それからＴＰＩ　ＲＸ　ＦＩＦＯは、送信リスト・メモリ・ベース・アドレス（ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）をそれぞれのＴＬＡＮ２２６のチャネル・パラメータ・レジスタ８２８ｂに、送信コマンドをコマンド・レジスタ８２８ａに書き込む。ＴＬＡＮ２２６は、ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳを用いて送信コントロール・リストを取り出すべく、ＰＣＩバス２２２を要求する。送信コントロール・リストが受け取られると、ＴＬＡＮ２２６はその送信コントロール・リストを送信コントロール・リスト・バッファ８２７ｂに格納し、そして、格納されている送信コントロール・リストを実行してパケット・データを受け取る。送信コントロール・リストも順方向ポインタを含み、通常はこれをＴＬＡＮ２２６が次のアドレスとして用いることによって次の送信コントロール・リストを受け取り、コントロール・リストのチェーンを形成する。ただし、示した実施例では、ＴＰＩ　２２０は各送信コントロール・リストの順方向ポインタをヌル値とともにロードする。従って、その送信コントロール・リスト・バッファ８２７ｂ内の送信コントロール・リストの実行後は、ＴＬＡＮ　２２６はＴＰＩ２２０が新しく別の送信コマンドを書き込むまで、待機することになる。

【０１５６】
次に図３３を参照する。該図はコントロール・リスト８３０を示している。これは受信と送信の両方のコントロール・リストの形式であり、さらにＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａおよびＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂの形式でもある。コントロール・リスト８３０は、ＦＯＲＷＡＲＤ＿ＰＯＩＮＴＥＲフィールド８３１、ＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａ、ＣＳＴＡＴフィールド８３２ｂ、ＣＯＵＮＴフィールド８３３、およびＤＡＴＡ＿ＰＯＩＮＴＥＲフィールド８３４を含む。各フィールドは３２ビットであるが、ＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａとＣＳＴＡＴフィールド８３２ｂは、１６ビットのフィールドである。

【０１５７】
ＦＯＲＷＡＲＤ＿ＰＯＩＮＴＥＲフィールドは、一般に複数のコントロール・リストをチェーン化するために使用される。受信動作については、ＦＯＲＷＡＲＤ＿ＰＯＩＮＴＥＲフィールド８３１がそれぞれのケースで同じＲＥＣＥＩＶＥＬＩＳＴ　ＭＥＭＯＲＹ　Ｂ

ＡＳＥ　ＡＤＤＲＥＳＳであるので、ＴＰＩ２２０が、ＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａから何度も繰り返して供給する受信コントロール・リストをＴＬＡＮ２２６が実行する。このように、各ＴＬＡＮ２２６は、そのカレントの受信コントロール・リストのＦＯＲＷＡＲＤ＿ＰＯＩＮＴＥＲフィールド８３１内のＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥＡＤＤＲＥＳＳを使用して、ネットワーク１１２から次のデータ・パケットが受信されたとき、次の受信コントロール・リストを要求する。従って、受信動作に関しては、ＴＬＡＮ２２６に対してＴＰＩ２２０が動作開始コマンドを出す必要がない。送信動作については、ＴＰＩ２２０が毎回同一のＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂからの送信コントロール・リストを実行する。しかし、ＴＰＩ２２０はＦＯＲＷＡＲＤ＿ＰＯＩＮＴＥＲフィールド８３１をヌル値（００００ｈ）にセットし、従ってＴＰＩ２２０によって開始されたときは、ＴＰＩ２２０およびそれぞれのＴＬＡＮ２２６は１つの送信動作を実行する。いずれかのＴＰＩＲＸ　ＦＩＦＯの中でデータが検知されて、ＴＰＩ２２０がＴＰＩ　ＲＸ　ＦＩＦＯのそれぞれのＴＬＡＮポート上で送信動作を行っていないとき、ＴＰＩ２２０は送信コマンドをそれぞれのＴＬＡＮ２２６に対して発生し、送信動作が開始される。それぞれのＴＬＡＮ２２６はＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂから送信コントロール・リストを取り出し、その送信コントロール・リストを実行し、そしてＦＯＲＷＡＲＤ＿ＰＯＩＮＴＥＲフィールド８３１のヌル値を検知したとき、デフォルトの状態に戻る。

【０１５８】

ＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａは、通常データ・パケットのサイズを表示する。受信動作については、ＴＰＩ２２０が最初にＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａ内のＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａをゼロにセットする。ＴＬＡＮ２２６がＴＰＩ２２０に対して１つのデータ・パケット全体の送信を完了した後、ＴＬＡＮ２２６は、ＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａのＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａおよびＣＳＴＡＴフィールド８３２ｂに対して最後のＤＷＯＲＤの書き込みを実行する。ＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａは実際のパケット・データのサイズでロードされ、ＣＳＴＡＴフィールド８３２ｂ内のフレーム完了ビットがセットされる。送信動作については、ＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂのＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａは、ＴＰＩ２２０によってＴＬＡＮ２２６に送信されるべきデータ・パケットのサイズでロードされる。ＨＳＢデータ転送インタフェース・ロジック８１９は、ＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂのＰＡＣＫＥＴ＿ＳＩＺＥレジスタ・タグ８１９ｃ内のパケット・サイズＤＷＯＲＤを送信リスト・デコード・ロジック８１４内のＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂに書き込む。そして、ＴＰＩ２２０が前述したように送信コマンドを対応するＴＬＡＮ２２６に書き込み、ＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂの内容が送信コントロール・リストとしてＴＬＡＮ２２６に対して要求されたときに供給される。

【０１５９】

ＣＳＴＡＴフィールド８３２ｂは、ＴＰＩ２２０とＴＬＡＮ２２６との間におけるコマンドおよび状態情報の受け渡しに使用される。ＴＰＩ２２０はＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａのＣＳＴＡＴフィールド８３２ｂを最初にゼロにセットする。ＴＬＡＮ２２６からそれぞれのＴＰＩ　ＲＸ　ＦＩＦＯへのパケット・データの転送が完了したとき、ＴＰＩ２２０はＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａ内のＣＳＴＡＴフィールド８３２ｂのフレーム完了ビット（ビット１４）をセットすることによって、パケット・データ転送の完了を表示する。ＴＰＩ２２０は、データ・パケットをＨＳＢ２０６を介してＥＰＳＭ２１０へ転送を開始できる状態にあることをポート状態ロジック８２０に知らせる。そしてポート状態ロジック８２０は、それぞれのＴＰＩ　ＲＸ　ＦＩＦＯ内にＥＰＳＭ２１０によるポーリングに応答して、ＥＰＳＭ２１０に対して送信可能なデータがあることを表示する。パケットの終わりは必ず転送しなければならないため、たとえパケットの終わりのデータ量がＲＢＳＩＺＥもしくはＴＢＵＳの値に適合しない場合でも、同様である。

【０１６０】

ＴＰＩ２２０は、ＥＰＳＭ　１０からのデータ・パケットの受信中におけるＡＬ＿ＦＣＳ

＿ＩＮ＊（またはＦＢＰＮ＊）信号の状態に基づいて、ＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂのＣＳＴＡＴフィールド８３２ｂ内のパス巡回冗長検査ＣＲＣ（Ｃｙｃｌｉｃ　Ｒｅｄｕｎｄａｎｃｙ　Ｃｈｅｃｋ）ビットをセットする。ＴＰＩ２２０は、データ・パケットがＣＲＣに使用されるデータを含んでいるかどうかを示すＣＲＣビットをセットする。ＣＲＣを含むイーサネットのデータ・パケットには、パケット・データに加えて誤り検査に用いられる４バイトのＣＲＣデータが入っている。

【０１６１】

ＤＡＴＡ＿ＰＯＩＮＴＥＲフィールド８３４は、データ転送動作中にＴＬＡＮ２２６によってアサートされるべきＰＣＩアドレスを指定する。このアドレスは、パケット・データ・メモリ・ベース・アドレス（ＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）であって、送信および受信動作の両方に同じものであることが望ましい。データ受信動作中、ＴＬＡＮ２２６がＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳをアサートし、受信データ復号ロジック８１３がＰＣＩバス２２２上のアドレスおよびライト・サイクルをデコードし、そして、選択されているＴＰＩ　ＲＸ　ＦＩＦＯ内へパケット・データが受容されるように、ＰＣＩ　ＲＸ　ＦＩＦＯコントロール・ロジック８１７をイネーブルする。データ送信動作中、ＴＬＡＮ２２６がＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳをアサートし、送信データ復号ロジック８１５がアドレスおよび読み出し動作をデコードし、そしてＴＰＩ　ＴＸ　ＦＩＦＯからのパケット・データの送信を促進するように、ＰＣＩ　ＴＸ　ＦＩＦＯ制御ロジック８１６をイネーブルする。

【０１６２】

ＣＯＵＮＴフィールド８３３は、存在するデータの量あるいはＤＡＴＡ＿ＰＯＩＮＴＥＲフィールド８３４の現在値における使用可能なバッファ・スペースを示す。データ受信動作中、受信リスト・デコード・ロジック８１２は、ＣＯＵＮＴフィールド８３３をＴＰＩコントロール・レジスタ８４６のＲＣＶ＿ＤＡＴＡ＿ＣＯＵＮＴレジスタ８４７ｂ（第８Ｆ図）内に書き込まれる値に設定する。ＲＣＶ＿ＤＡＴＡ＿ＣＯＵＮＴレジスタ８４７ｂの値でＴＰＩ２２０が受信すべき最大パケット・サイズが決まる。既定値は１，５１８バイトであって、これはＣＲＣの４バイトを含むイーサネット・データ・パケットの最大サイズである。データ送信動作中、ＴＰＩ２２０はＣＯＵＮＴフィールド８３３をＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａと同じ値に設定する。

【０１６３】

次に図３４を参照する。該図は、ＴＰＩ２２０に使用されるＴＰＩ　ＰＣＩコンフィギュレーション・レジスタ８３５の定義を示している。ＴＰＩ　ＰＣＩコンフィギュレーション・レジスタ８３５は、ＴＰＩ２２０専用の追加的なレジスタ、およびすべてのＰＣＩバスのアーキテクチャに共通のレジスタを含む。すべてのＰＣＩバスに共通のレジスタは、ＤＥＶＩＣＥ＿ＩＤレジスタ８３６ａ、ＶＥＮＤＯＲ＿ＩＤレジスタ８３６ｂ、状態（ＳＴＡＴＵＳ）レジスタ８３７ａ、コマンド（ＣＯＭＭＡＮＤ）レジスタ８３７ｂ、ＣＬＡＳＳ＿ＣＯＤＥレジスタ８３８ａ、ＲＥＶ＿ＩＤレジスタ８３８ｂ、ＢＩＳＴレジスタ８３９ａ、ＨＤＲ＿ＴＹＰＥレジスタ８３９ｂ、レイテンシすなわち待ち時間（ＬＡＴＥＮＣＹ）レジスタ８３９ｃ、ＣＡＣＨＥＬＳレジスタ８３９ｄ、ＭＡＸＬＡＴレジスタ８４５ａ、ＭＩＮＧＮＴレジスタ８４５ｂ、ＩＮＴＰＩＮレジスタ８４５ｃ、およびＩＮＴＬＩＮＥレジスタ８４５ｄである。

【０１６４】

ＴＰＩ２２０専用のレジスタは、ＴＰＩコントロールＩＯベース・アドレス（ＣＯＮＴＲＯＬ　ＩＯ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）レジスタ８４０、ＴＰＩコントロール・メモリ・ベース・アドレス（ＣＯＮＴＲＯＬ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）レジスタ８４１、送信リスト・メモリ・ベース・アドレス（ＴＲＡＮＳＭＩＴ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）レジスタ８４２、受信リスト・メモリ・ベース・アドレス（ＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）レジスタ８４３、およびパケット・データ・メモリ・ベース・アドレス（ＰＡＣＫＥＴ　Ｄ

ＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）レジスタ８４４である。

【０１６５】

初期化後、ＴＰＩコントロールＩＯベース・アドレス・レジスタ８４０にはＴＰＩコントロール・レジスタ８４６のためのＴＰＩ　ＣＯＮＴＲＯＬ　ＩＯ　ＢＡＳＥ　ＡＤＤＲＥＳＳが入っている。ＴＰＩコントロール・メモリ・ベース・アドレス・レジスタ８４１にはＴＰＩコントロール・レジスタ８４６のためのＴＰＩ　ＣＯＮＴＲＯＬ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳが入っている。このように、ＴＰＩコントロール・レジスタ８４６は、ＰＣＩバス２２２の入出力とメモリ・スペースの両方でアクセスが可能である。送信リスト・メモリ・ベース・アドレス・レジスタ８４２には、送信リストデコード・ロジック８１４によってデコードされるＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂのためのＴＲＡＮＳＭＩＴ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳが入っている。受信リスト・メモリ・ベース・アドレス・レジスタ８４３には、受信リストデコード・ロジック８１２によってデコードされるＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａのためのＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳが入っている。パケット・データ・メモリ・ベース・アドレス・レジスタ８４４には、ＴＰＩ２２０のデータ・バッファ８０７に対応するＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳが入っている。ＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳは、送信リスト・デコード・ロジック８１４と受信リスト・デコード・ロジック８１２の両方によってデコードされる。

【０１６６】

次に図３５を参照する。該図は、ＴＰＩ　２２０に使用されるＴＰＩコントロール・レジスタ８４６の定義の図解である。ＴＰＩコントロール・レジスタ８４６は、ＲＣＶ＿ＤＡＴＡ＿ＣＯＵＮＴレジスタ８４７ｂ、ＸＢＳＩＺＥ３レジスタ８４８ａ、ＸＢＳＩＺＥ２レジスタ８４８ｂ、ＸＢＳＩＺＥ１レジスタ８４８ｃ、ＸＢＳＩＺＥ０レジスタ８４８ｄ、ＲＢＳＩＺＥ３レジスタ８４９ａ、ＲＢＳＩＺＥ２レジスタ８４９ｂ、ＲＢＳＩＺＥ１レジスタ８４９ｃ、ＲＢＳＩＺＥ０レジスタ８４９ｄ、ＮＥＴ＿ＰＲＩ３レジスタ８５０ａ、ＮＥＴ＿ＰＲＩ２レジスタ８５０ｂ、ＮＥＴ＿ＰＲＩ１レジスタ８５０ｃ、ＮＥＴ＿ＰＲＩ０レジスタ８５０ｄ、ＴＬＡＮ０メモリ・ベース・アドレス（ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳ）レジスタ８５１、ＴＬＡＮ１メモリ・ベース・アドレス・レジスタ８５２、ＴＬＡＮ２メモリ・ベース・アドレス・レジスタ８５３、およびＴＬＡＮ３メモリ・ベース・アドレス・レジスタ８５４を含む。

【０１６７】

ＲＣＶ＿ＤＡＴＡ＿ＣＯＵＮＴレジスタ８４７ｂは、ＴＰＩ　２０が処理した受信データ・パケットの最大サイズを格納する。ＴＰＩ２２０は、この値を取り出してＲＸ　ＣＮＴＬ　ＬＩＳＴ　０８ａのＣＯＵＮＴフィールド８３３に入れる。ＸＢＳＩＺＥレジスタ８４８ａ～ｄの各々は、それぞれのポートについてＤＷＯＲＤ単位の送信バースト・サイズを保持している。すなわち、ＰＯＲＴ２４にはＸＢＳＩＺＥ０、ＰＯＲＴ２５にはＸＢＳＩＺＥ１、ＰＯＲＴ２６にはＸＢＳＩＺＥ２、そしてＰＯＲＴ２７にはＸＢＳＩＺＥ３である。ＸＢＳＩＺＥの送信バースト・サイズの値は、それぞれのポートに対してＥＰＳＭ２１０からデータを要求できるだけの十分なパケット・バッファ・スペースがそれぞれのＴＰＩＴＸ　ＦＩＦＯにあるかどうかを判定するとき、ＴＰＩ２２０のＨＳＢ　ＴＸＦＩＦＯコントロール・ロジック８２２およびポート状態ロジック８２０によって用いられる。ＲＢＳＩＺＥレジスタ８４９ａ～ｄの各々は、それぞれのポートについてＤＷＯＲＤ単位のＨＳＢ受信バースト・サイズを保持する。すなわち、ＰＯＲＴ２４にはＲＢＳＩＺＥ０、ＰＯＲＴ２５にはＲＢＳＩＺＥ１、ＰＯＲＴ２６にはＲＢＳＩＺＥ２、そしてＰＯＲＴ２７にはＲＢＳＩＺＥ３である。ＲＢＳＩＺＥの受信バースト・サイズの値は、それぞれのポートからＥＰＳＭ２１０に対する受信データ転送を要求できるだけの十分なパケット・データがそれぞれのＴＰＩ　ＲＸ　ＦＩＦＯにあるかどうかを判定するとき、ＨＳＢ　ＲＸ　ＦＩＦＯコントロール・ロジック８２１およびポート状態ロジック８２０によって用いられる。図解した実施例において、ＸＢＳＩＺＥおよびＲＢＳＩＺＥレジスタ８４

８、８４９の値はそれぞれが等しく、またＴＢＵＳの値とも等しい。しかし、ＸＢＳＩＺＥレジスタ８４８およびＲＢＳＩＺＥレジスタ８４９は、必要に応じて任意のバースト転送値でプログラミングされる。

【０１６８】

ＮＥＴ＿ＰＲＩレジスタ８５０は、それぞれのポートに関するそれぞれのネットワーク優先権の値を保持する。すなわち、ＰＯＲＴ２４にはＮＥＴ＿ＰＲＩ０、ＰＯＲＴ２５にはＮＥＴ＿ＰＲＩ１、ＰＯＲＴ２６にはＮＥＴ＿ＰＲＩ２、そしてＰＯＲＴ２７にはＮＥＴ＿ＰＲＩ３である。これらの値は、送信リスト・デコード・ロジック８１４がそれぞれのポートの送信優先権を設定するために使用される。ＴＬＡＮ０メモリ・ベース・アドレス・レジスタ８５１は、ＰＯＲＴ２４についてＴＬＡＮ０　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳというＰＣＩメモリ・アドレスを保持する。ＴＬＡＮ１メモリ・ベース・アドレス・レジスタ８５２は、ＰＯＲＴ２５についてＴＬＡＮ１　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳというＰＣＩメモリ・アドレスを保持する。ＴＬＡＮ２メモリ・ベース・アドレス・レジスタ８５３は、ＰＯＲＴ２６についてＴＬＡＮ２　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳというＰＣＩメモリ・アドレスを保持する。ＴＬＡＮ３メモリ・ベース・アドレス・レジスタ８５４は、ＰＯＲＴ２７についてＴＬＡＮ３　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳというＰＣＩメモリ・アドレスを保持する。これらのレジスタのそれぞれを、起動時にＣＰＵ　２３０が各ＴＬＡＮ　２２６のアドレスを認識してから初期化する。これらの値はＰＣＩ　ＴＸ　ＦＩＦＯコントロール・ロジック８１６に供給され、このロジックがＰＣＩバス２２２上にそれぞれの送信コマンドを出してパケット送信動作を開始するために該値を使用する。

【０１６９】

次に図３６を参照する。該図は、ネットワーク・スイッチ１０２の初期化、起動あるいはリセット時におけるＣＰＵ２３０のＰＣＩ初期化動作を図解したフローチャートである。最初のステップ８５５において、ＣＰＵ２３０はＰＣＩバス２２２のコンフィギュレーションを行い、それぞれのＴＬＡＮ２２６をＰＣＩメモリ・スペースにマッピングし、そして、このコンフィギュレーションをＰＣＩバス２２２を介してＴＰＩ　ＰＣＩコンフィギュレーション・レジスタ８３５に書き込む。ＰＣＩバス２２２のコンフィギュレーションを行う手順は既知であり、ここではさらに説明しない。

【０１７０】

特に、ＤＥＶＩＣＥ＿ＩＤレジスタ８３６ａは、標準のＰＣＩデバイスＩＤレジスタであり、その値は０ｘ５０００ｈに設定される。ＶＥＮＤＯＲ＿ＩＤレジスタ８３６ｂは標準のＰＣＩベンダＩＤレジスタであり、その値は０ｘ０Ｅ１１ｈに設定される。ＳＴＡＴＵＳレジスタ８３７ａは標準のＰＣＩデバイス状態レジスタである。ＣＯＭＭＡＮＤレジスタ８３７ｂは標準のＰＣＩデバイス・コマンド・レジスタである。ＣＬＡＳＳ＿ＣＯＤＥレジスタ８３８ａは標準のＰＣＩデバイス・クラス・コード・レジスタであり、その値は０ｘ０６０２００ｈに設定される。ＲＥＶ＿ＩＤレジスタ８３８ｂは標準のＰＣＩデバイス改定ＩＤレジスタであり、その値は０ｘ００ｈに設定される。ＢＩＳＴレジスタ８３９ａは標準のＰＣＩ　ＢＩＳＴ状態レジスタであり、その値は０ｘ００ｈに設定される。ＨＤＲ＿ＴＹＰＥレジスタ８３９ｂは標準のＰＣＩヘッダ・タイプ・レジスタであり、その値は０ｘ８０ｈに設定される。ＬＡＴＥＮＣＹ（待ち時間）レジスタ８３９ｃは標準のＰＣＩ待ち時間レジスタであり、ＣＰＵ２３０によって初期化される。ＣＡＣＨＥＬＳＺレジスタ８３９ｄは標準のＰＣＩキャッシュ・ライン・サイズ・レジスタであり、ＣＰＵ２３０によって初期化される。ＭＡＸＬＡＴレジスタ８４５ａは標準のＰＣＩ最長待ち時間レジスタであり、その値は０ｘ００ｈに設定される。ＭＩＮＧＮＴレジスタ８４５ｂは標準のＰＣＩデバイス・ミニマム・グラント・レジスタであり、その値は０ｘ００ｈに設定される。ＩＮＴＰＩＮレジスタ８４５ｃは標準のＰＣＩデバイス割り込みピン・レジスタであり、その値は０ｘ００ｈに設定される。ＩＮＴＬＩＮＥレジスタ８４５ｄは標準のＰＣＩデバイス割り込みライン・レジスタであり、ＣＰＵ　２３０によって設定される。

【０１７１】

ステップ８５５では、さらにＣＰＵ２３０が０ｘＦＦＦＦＦＦＦＦｈの値を次のそれぞれのレジスタに書き込む。すなわち、ＴＰＩ ＣＯＮＴＲＯＬ ＩＯ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４０； ＴＰＩ ＣＯＮＴＲＯＬ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４１； ＴＲＡＮＳＭＩＴ ＬＩＳＴ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４２； ＲＥＣＥＩＶＥ ＬＩＳＴ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４３； およびＰＡＣＫＥＴ ＤＡＴＡ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４４に書き込む。それぞれへの書き込み完了後、ＴＰＩ２２０が各レジスタ内の値を、指示された特定のレジスタに求められる量の入出力（Ｉ／Ｏ）またはメモリ・スペースを示す値に置き換える。ＣＰＵ２３０は、それに応答して各レジスタ内のそれぞれの新しい値を読み取り、各レジスタにベース（基準）・アドレスを書き返し、そのエンティティをＰＣＩ Ｉ／Ｏまたはメモリ・スペースにマッピングする。

【０１７２】

特に、必要なメモリ・スペースの量を決定してから、ＣＰＵ２３０はＣＯＮＴＲＯＬ ＩＯ ＢＡＳＥ ＡＤＤＲＥＳＳをＴＰＩ ＣＯＮＴＲＯＬ ＩＯ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４０に書き込んで、ＴＰＩコントロール・レジスタ８４６の入出力スペースへのアクセスを可能とし、ＣＰＵ２３０はＴＰＩＣＯＮＴＲＯＬ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳをＴＰＩ ＣＯＮＴＲＯＬ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４１に書き込んでＴＰＩコントロール・レジスタ８４６のメモリ・スペースへのアクセスを可能とし、ＣＰＵ２３０はＴＲＡＮＳＭＩＴ ＬＩＳＴ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳをＴＸ ＣＮＴＬ ＬＩＳＴ８０８ｂメモリ・ブロックのアドレスに対応するＴＲＡＮＳＭＩＴ ＬＩＳＴ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４２に書き込み、ＣＰＵ２３０は ＲＥＣＥＩＶＥ ＬＩＳＴＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳをＲＸ ＣＮＴＬ ＬＩＳＴ８０８ａのアドレスに対応するＲＥＣＥＩＶＥ ＬＩＳＴ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４３に書き込み、そしてＣＰＵ２３０はＰＡＣＫＥＴＤＡＴＡ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳをデータ・バッファ８０７のＰＣＩアドレスに対応するＰＡＣＫＥＴ ＤＡＴＡ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳレジスタ８４４に書き込む。

【０１７３】

次のステップ８５６ａにおいて、ＣＰＵ２３０はＰＣＩバス２２２上のそれぞれのＴＬＡＮ２２６に対して１つずつ問い合わせを行い、存在するＴＬＡＮの数、およびそれらのＴＬＡＮの対応するＰＣＩアドレスを認識する。続くステップ８５６ｂで、ＣＰＵ２３０は問い合わせたＴＬＡＮ２２６を既知で休止の状態に初期化する。そしてＣＰＵ２３０は、次のステップ８５７でＴＬＡＮ２２６がそれ以上存在するかどうかを調べ、もし存在すればステップ８５６ａに戻って、次のＴＬＡＮ２２６に対して問い合わせを行い、ＰＣＩバス２２２上のＴＬＡＮ２２６がすべて初期化されるまでこれを繰り返す。この時点では、ＴＬＡＮ０ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳ、ＴＬＡＮ１ ＭＥＭＯＲＹ ＢＡＳＥＡＤＤＲＥＳＳ、ＴＬＡＮ２ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳ、およびＴＬＡＮ３ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳの値は既知である。

【０１７４】

次のステップ８５８において、ＣＰＵ２３０は、図３５に関して前述したように、ＴＰＩコントロール・レジスタ８４６を適切な値に初期化する。これは、ＴＬＡＮ０ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳ、ＴＬＡＮ１ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳ、ＴＬＡＮ２ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳ、およびＴＬＡＮ３ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳの値を含む。続くステップ８５９で、ＣＰＵ２３０は、ＲＥＣＥＩＶＥ ＬＩＳＴ ＭＥＭＯＲＹ ＢＡＳＥ ＡＤＤＲＥＳＳをチャネル・パラメータ・レジスタ８２８ｂに書き込み、各ＴＬＡＮ２２６の受信動作の始動を開始する。受信動作の開始はステップ９６０で完了し、ＣＰＵ２３０が各ＴＬＡＮ２２６のコマンド・レジスタ８２８ａに対して書き込みを行う。このように初期化されて、それぞれのＴＬＡＮ２２６は受信コントロール・リストを要求するために、ＰＣＩバス２２２を要求して、直ちに受信動作を始める。

【０１７５】
次に図３７を参照する。該図は、各ＴＬＡＮ２２６についてネットワーク・スイッチ１０２が行う受信動作を図解するフローチャートである。動作は第１ステップ８６１ａで始まり、ＴＬＡＮ２２６は、ＰＣＩアービタ８１１にＰＣＩバス２２２を要求してこれを受け取る。ＴＬＡＮ２２６は第２ステップ８６１ｂでＲＥＣＥＩＶＥ　ＬＩＳＴ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳをＰＣＩバス２２２にアサートして受信コントロール・リストを要求し、ＴＰＩ２２０が第３のステップ８６１ｃで受信コントロール・リストをそのＴＬＡＮ２２６に供給する。受信コントロール・リストは、受信したデータ・パケットをどこで、もしくはどのように送信するかをＴＬＡＮ　２２６に知らせるためのＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳを含む。次の第４のステップ８６１１で、ＴＬＡＮ２２６はＰＣＩバス２２２の制御権を放棄する。

【０１７６】
ＴＬＡＮ２２６は、次のステップ８６２ａにおいて、最終的にネットワーク１１２からデータ・パケットを受信し、ステップ８６２ｂにおいて、ＰＣＩバス２２２の制御権を要求してこれを受け取る。ＴＬＡＮ２２６はステップ８６２ｃで、ＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳをＰＣＩバス２２２上のアドレスとして用い、１バーストのデータの書き込みを行い、一方ＴＰＩ２２０は、ステップ８６２ｄにおいて、そのデータを選択されたＴＰＩＲＸ　ＦＩＦＯに書き込む。書き込みバーストの完了と同時に、ＴＬＡＮ２２６は次のステップ８６２ｅに移って、ＰＣＩバス２２２の制御権を放棄する。さらに次のステップ８６５において、ＴＬＡＮ２２６は、最終ＤＷＯＲＤの書き込み動作で示されるべき、パケット・データの全体のＴＰＩ　ＲＸ　ＦＩＦＯに対する送出が完了したかどうかをチェックし、まだであれば、動作はステップ８６２ｂへ戻り、ＴＬＡＮ２２６はもう一度ＰＣＩバス２２２を要求するために別のバースト・データを送る。

【０１７７】
ＴＬＡＮ２２６は、データ・パケットの最終部分を送り終わった後、最後の反復動作を行って、ＴＰＩ　ＲＸ　ＦＩＦＯに対しパケットの終わりを知らせる。特にＴＬＡＮ２２６は、ＴＰＩ２２０のＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａ内のＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａおよびＣＳＴＡＴフィールド８３２ｂに対して、最後の１ＤＷＯＲＤの転送を実行する。このＤＷＯＲＤの転送によって、ＲＸ　ＣＮＴＬ　ＬＩＳＴ８０８ａが完了したばかりのデータ・パケットのパケットのサイズで更新され、ＣＳＴＡＴフィールド８３２ｂ内のフレーム完了ビットが更新される。ＴＰＩ２２０はこの書き込み動作をステップ８６５で検知して動作完了を表す内部フラグをセットし、ステップ８６６において、その適宜の状態をポート状態ロジック８２０に渡す。動作はステップ８６１ａへ戻って、別の受信コントロール・リストを要求する。

【０１７８】
次に図３８を参照する。該図は、ＴＰＩ２２０からＥＰＳＭ２１０へのＨＳＢ２０６を介した受信データ転送動作を図解するフローチャートである。動作は最初のステップ８７６で開始し、ＴＰＩ２２０のポート状態ロジック８２０が、ＴＰＩ　ＲＸ　ＦＩＦＯのいずれか１つに存在するＴＰＩコントロール・レジスタ８４６で用意されたそれぞれのＲＢＳＩＺＥと比べて等しいか大きい一定量のデータを検知するか、もしくは、ＴＬＡＮ２２６によって表示されているそのポートに関するパケットの終わりＥＯＰを検出する。

【０１７９】
次のステップ８７７では、ＴＰＩ２２０がＥＰＳＭ２１０のポーリングに応答し、各ＴＰＩ　ＲＸ　ＦＩＦＯ内に十分なデータが存在するか否かを表すＰＫＴ＿ＡＶＡＩＬ［６］＊信号を多重化の方式で適切にアサートする。このポーリングは、独立して発生し、クラリフィケーションの目的で含まれている。ＴＰＩ２２０のいずれかのＴＰＩ　ＲＸ　ＦＩＦＯ内に十分なデータが存在することをＰＫＴ＿ＡＶＡＩＬ［６］＊信号が表示した場合、ＥＰＳＭ２１０の使用可能な受信バッファ内に十分なバッファ記憶スペースがあれば、ＥＰＳＭ２１０はＨＳＢ

２０６上でリード・サイクルを開始する。
【０１８０】
ＴＰＩ２２０のポート状態ロジック８２０は、ＨＳＢ２０６上のリード・サイクルを検知し、当該のＴＰＩ　ＲＸ　ＦＩＦＯを選択して次のステップ８７９でデータを供給する。それからＴＰＩ２２０は、ステップ８８０において、ＥＰＳＭ２１０に対しＨＳＢ２０６を介してデータ・バーストを転送する。ステップ８８０でのデータ転送中、次のステップ８８１ａで、ポート状態ロジック８２０がＨＳＢ２０６を介した現在のデータ転送がパケットの始めであると判断すれば、データ転送中にＴＰＩ２２０がステップ８８１ｂにおいてＨＳＢ２０６上でＳＯＰ＊信号をアサートする。同様に、ステップ８８０でのデータ転送中、ステップ８８２ａにおいて、ポート状態ロジック８２０がＨＳＢ２０６を介した現在のデータ転送がパケットの終わりであると判断すれば、データ転送中にＴＰＩ２２０が、ステップ８８２ｂにおいて、ＨＳＢ２０６上でＥＯＰ＊信号をアサートする。ステップ８８２ａまたは８８２ｂから、動作はステップ８７６へ戻る。
【０１８１】
次に図３９を参照する。該図は、ＥＰＳＭ２１０からＴＰＩ２２０へパケット・データを送るためのＨＳＢ２０６を介した送信データ転送動作を図解するフローチャートである。動作はステップ８９０で開始し、ＴＰＩ２２０のポート状態ロジック８２０がＴＰＩ　ＴＸ　ＦＩＦＯののいずれか１つに、対応するＸＢＳＩＺＥと比較して等しいか大きいバッファ・スペースがあることを検知する。動作は次のステップ８９１へ進み、ポート状態ロジック８２０は、ＥＰＳＭ２１０のポーリングに応答してＢＵＦ＿ＡＶＡＩＬ［６］＊信号を多重化の方式で適切にアサートし、対応するＴＰＩ　ＴＸ　ＦＩＦＯ内に使用可能なバッファ・スペースがあることを表示する。前述したように、このポーリングは独立して発生し、クラリフィケーションの目的で含まれている。次のステップ８９２において、十分なスペースのあるＴＰＩ　ＴＸ　ＦＩＦＯに対してＥＰＳＭ２１０が転送するだけの十分なデータがあるとき、ＥＰＳＭ２１０は、ＨＳＢ２０６上でそのＴＰＩ　ＴＸ　ＦＩＦＯに対応するポートへのライト・サイクルを開始する。続くステップ８９３では、ＴＰＩ２２０のポート状態ロジック８２０がＨＳＢ２０６上のライト・サイクルを検知し、指示されたポートに適当なＴＰＩ　ＴＸ　ＦＩＦＯを選択する。次のステップ８９４において、ＥＰＳＭ２１０はＴＰＩ２２０に対してＨＳＢ２０６を介し１バーストのデータを転送し、ＴＰＩ２２０はそのデータをＴＰＩ２２０内の対応するＴＰＩ　ＴＸ　ＦＩＦＯに書き込む。
【０１８２】
ステップ８９５ａにおいて、ＴＰＩ２２０がステップ８９４のデータ・バースト中にＳＯＰ＊信号がアサートされたことを検知した場合、そのパケット・サイズを持っているデータの先頭のＤＷＯＲＤは、ステップ８９５ｂにおいてＰＡＣＫＥＴ　ＳＩＺＥタグ・レジスタ８１９ｃに入れられる。ステップ８９６ａにおいて、ＴＰＩ２２０が、ステップ８９４でのデータ・バースト中にＥＯＰ＊信号がアサートされたことを検知すれば、ＴＰＩ２２０はステップ８９６でパケットの終わりを表すＴＰＩ２２０内のフラグをセットする。ステップ８９６ａまたは８９６ｂから、動作はステップ８９０へ戻る。
【０１８３】
次に図４０を参照する。該図は、各ＴＬＡＮ２２６に関するネットワーク・スイッチ１０２の送信動作を図解するフローチャートである。最初のステップ８６７において、ＴＰＩ２２０はＴＰＩ　ＴＸ　ＦＩＦＯのいずれか１つの中にデータを検知し、それに応答してＰＣＩバス２２２を要求し、ＰＣＩアービタ８１１からこれの制御権を受け取る。次のステップ８６８で、ＴＰＩ２２０は、対応するＴＬＡＮ２２６のコマンド・レジスタ８２８ａに送信コマンドを書き込む。ＴＰＩ２２０は、その後ステップ８６９で、ＰＣＩバス２２２の制御権を放棄する。
【０１８４】
続くステップ８７０ａにおいて、送信コマンドを受け取ったＴＬＡＮ２２６は、ＰＣＩバス２２２の制御権を要求し、ＰＣＩアービタ８１１からこれの制御権を受け取り、ＴＰＩ

２２０に対して送信コントロール・リストを要求する。次のステップ８７０ｂで、ＴＰＩ２２０はＰＣＩバス２２２の制御権を持っているＴＬＡＮ２２６に送信コントロール・リストを供給し、ＴＬＡＮ２２６はその送信コントロール・リストをその送信コントロール・リスト・バッファ８２７ｂに入れる。続くステップ８７０ｃにおいて、ＴＬＡＮ２２６はＰＣＩバス２２２の制御権を放棄するが、直ちにステップ８７０ｄでＰＣＩバス２２２の制御権を再要求する。再びＰＣＩバス２２２の制御権を得ると、ＴＬＡＮ２２６はステップ８７１ａでＴＰＩ２２０に対して１バーストのデータを要求し、送信コントロール・リストの実行を開始する。特に、ＴＬＡＮ２２６は、ステップ８７１ａにおいて、ＰＣＩバス２２２上でＰＡＣＫＥＴ　ＤＡＴＡ　ＭＥＭＯＲＹ　ＢＡＳＥ　ＡＤＤＲＥＳＳをアサートする。続くステップ８７１ｂでは、ＴＰＩ２２０がそれに応答して、対応するＴＰＩ　ＴＸ　ＦＩＦＯを選択してイネーブルし、ＰＣＩバス２２２を介してＴＬＡＮ２２６にデータを供給する。それぞれのデータ・バースト後、ＴＬＡＮ２２６は、ステップ８７１ｃにおいてＰＣＩバス２２２の制御権を放棄する。ステップ８７２ａにおいて、データのパケット全体の転送が完了していないと判定すると、動作はステップ８７０ｃに戻り、ＴＬＡＮ２２６は再びＰＣＩバス２２２の制御権を要求し、最終的に該制御権を取り戻す。

【０１８５】

ステップ８７２ａにおいて、パケットの送信が完了していると判定すると、動作はステップ８７３ａに移り、ＴＬＡＮ２２６はＴＰＩ２２０に対してデータ転送完了の旨を書き込み、ＴＰＩ２２０はそれに応答して動作の完了を表示する。特に、ＴＬＡＮ２２６はＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂのＣＳＴＡＴフィールド８３２ｂに最後の１ＤＷＯＲＤの書き込みを行い、ＣＳＴＡＴフィールド８３２ｂ内のフレーム完了ビットをセットする。さらに、ＴＸ　ＣＮＴＬ　ＬＩＳＴ８０８ｂのＰＡＣＫＥＴ＿ＳＩＺＥフィールド８３２ａが、ＴＰＩ２２０によってＴＬＡＮ２２６に送信されるべきデータ・パケットのサイズでロードされる。ＴＬＡＮ２２６は、書き込み動作を完了すると、ステップ８７３ｂでＰＣＩバス２２２を放棄する。ステップ８７３ｂから、動作は次の送信動作に備えてステップ８６７に戻る。

【０１８６】

ＣＰＵ２３０による初期化後、ＴＰＩ２２０はＴＬＡＮ２２６と協調して動作するように構成されることによって、ＣＰＵ２３０がネットワーク・スイッチ１０２の他の重要なタスクや機能を遂行することができる点は、高い評価に値する。ＣＰＵ２３０は、ＰＣＩバス２２２上のデバイスのタイプや数を確認し、対応するアドレス値を割り当てて、ＰＣＩメモリおよび入出力スペースを初期化する。ＣＰＵ２３０は、ＴＬＡＮ２２６のアドレス値をＴＰＩ２２０に供給する。さらに、ＣＰＵ２３０はＴＰＩ２２０のアドレスの初期値をそれぞれのＴＬＡＮ２２６に供給し、コマンドを挿入して動作を起動する。ＴＬＡＮ２２６は、コントロール・リストを要求して該コントロール・リストを実行し、そのコントロール・リスト内のアドレスにあるメモリとの間で、データの読み書きを行うように構成される。ＴＰＩ２２０はまた、各コントロール・リストを更新し、それぞれを、要求している各ＴＬＡＮ２２６に供給するように構成される。さらに、ＴＰＩ２２０は、適宜のＴＬＡＮ２２６にコマンドを書き込んで送信動作を開始するよう構成され、また対応する送信コントロール・リストを後続の要求に応じて供給するように構成される。このようにして、ＣＰＵ２３０は初期化の実行後は、ネットワーク・スイッチ１０２の他の機能を自由に遂行することができる。

【０１８７】

図４１は、メモリ２１２の編成を図解するブロック図である。示した実施例では、メモリ２１２のサイズは４～１６メガバイト（Ｍｂｙｔｅ）であるが、このメモリ・サイズは可変であって、必要に応じた増減が可能である。図４１～図４７に示すメモリ・セクション・ブロックの幅、従って各メモリ・ラインの幅は、１ＤＷＯＲＤすなわち３２ビットである。メモリ２１２は、ハッシュ・メモリ・セクション９０２およびパケット・メモリ・セクション９０４という、２つの主なセクションに分けられる。ハッシュ・メモリ・セクシ

ョン９０２はネットワーク・デバイス識別セクションとして機能し、ネットワーク・スイッチ１０２に結合しているネットワーク１０６、１１２内の１つまたは複数のネットワーク・デバイスを識別する。ハッシュ・メモリ・セクション９０２のサイズは、必要なデバイス、関連のアドレスおよびエントリの数に基づいて、プログラミングすることができる。示した実施例において、ハッシュ・メモリ・セクション９０２は２５６キロバイト（Ｋｂｙｔｅ）のメモリで、最小８Ｋ（Ｋ＝2^{10}＝１，０２４）から最大１６Ｋまでのアドレスをサポートする。ハッシュ・メモリ・セクション９０２は、メモリ２１２内のどこに置かれてもよく、示した実施例ではメモリ２１２の先頭に位置している。パケット・メモリ・セクション９０４のサイズは、メモリ２１２の残りの領域、すなわちハッシュ・メモリ・セクション９０２が使用していない部分である。

【０１８８】

図４２は、メモリ２１２のハッシュ・メモリ・セクション９０２の編成を示すブロック図である。ハッシュ・メモリ・セクション９０２は長さが２５６キロバイトとして示されているが、ハッシュ・メモリ・セクションのサイズは固定、あるいは必要に応じてプログラミング可能であることは理解されよう。ハッシュ・メモリ・セクション９０２は、一次的なハッシュ・エントリのための１番目の１２８キロバイトの一次ハッシュ・エントリ・セクション９０６、およびチェーン（連鎖）・ハッシュ・エントリ用の２番目の１２８キロバイトのチェーン・ハッシュ・エントリ・セクション９０８という、２つの１２８キロバイトのセクションに分かれている。セクション９０６、９０８の各々は、それぞれの長さが１６バイトの８Ｋのエントリを含む。

【０１８９】

図４３は、一次ハッシュ・エントリ・セクション９０６とチェーン・ハッシュ・エントリ・セクション９０８の両方を含む、ハッシュ・メモリ・セクション９０２内の各エントリを表すハッシュ・テーブル・エントリ９１０の編成の図解である。各エントリ９１０は、ネットワーク・スイッチ１０２に結合しているネットワーク１０６、１１２のネットワーク・デバイスの１つに対応する。各一次エントリは１つのハッシュ・アドレスに存在し、ハッシュ・アドレスはそのデバイスのＭＡＣアドレスを「ハッシュ」して決定される。特に、ネットワーク・デバイスには、物理アドレスあるいはＭＡＣアドレスとも呼ばれる４８ビットのハードウェア・アドレスが割り当てられ、このアドレスは製造過程において、あるいはネットワークの設置中に、ジャンパまたはスイッチを設定して各ネットワーク・デバイスに割り当てられる一意の数値である。このＭＡＣアドレスの一部は、米国電気電子技術者協会ＩＥＥＥ（Ｉｎｓｔｉｔｕｔｅ ｏｆ Ｅｌｅｃｔｒｉｃａｌ ａｎｄ Ｅｌｅｃｔｒｏｎｉｃｓ Ｅｎｇｉｎｅｅｒｓ）によって製造業者に割り当てられたもので、その製造業者のすべての製品に共通しており、ハードウェア・アドレスの他の一部は、ハードウェアの製造業者が割り当てた一意の値である。ハッシュ・テーブル・エントリ９１０最初の６バイト、すなわちバイト５～０には、その項目に関連するデバイスのＭＡＣアドレスが入っている。従ってネットワーク・スイッチ１０２は、そのＭＡＣソース・アドレスを含むデータ・パケットを送信する各ネットワーク・デバイスに、１つのハッシュ・テーブル・エントリを付加する。

【０１９０】

ネットワーク１０６、１１２内の各ネットワーク・デバイスから送信されるそれぞれのデータ・パケットは、一般に送信元と受信先のＭＡＣアドレスを含み、これらは両方とも、いくつかのアルゴリズムの１つに従ってハッシュされるものである。示した実施例においては、各ＭＡＣアドレスの２つの部分を論理的に結合あるいは比較して対応するハッシュ・アドレスを算出する。各部分は１３ビットから１６ビットであり、排他的論理和（ＸＯＲ）のロジックを使用してビット単位の方式で結合され、１３から１６ビットのハッシュ・アドレスを形成する。例えば、最初の１６ビットのＭＡＣアドレスＭＡ［１５：０］と、次の１６ビットのＭＡＣアドレスＭＡ［３１：１６］とのビット単位の方式による論理和が、ハッシュ・アドレスＨＡ［１５：０］となる。ある実施例では、ハッシュされた結果の最初の１３、１４、１５、または１６ビットが、ハッシュ・アドレスＨＡとして使用

される。あるいは、ＭＡＣアドレスの最初の１３ビットのＭＡ［１２：０］を次の１３ビットのＭＡ［２５：１３］とハッシュして、１３ビットのハッシュ・アドレスＨＡ［１２：０］を得る。もしくは、ＭＡＣアドレスの最初の１４ビットのＭＡ［１３：０］を次の１４ビットのＭＡ［２７：１４］とハッシュして、１４ビットのハッシュ・アドレスＨＡ［１３：０］とするなど、以下同様に行われる。ハッシュ処理には多種多様なアルゴリズムが知られており、当業者には既知のように、アドレス・ビットの特定の組み合わせの結合に用いられること、および本発明は何ら特定のハッシュ法に限定されるものではないことは、理解されであろう。

【０１９１】

ハッシュ・アドレスは、一次ハッシュ・エントリ・セクション９０６内のそれぞれのハッシュ・エントリの位置を特定するための実アドレス、またはオフセット・アドレスとして使用される。ＭＡＣアドレスは一意であるが、ハッシュ・アドレスの場合は、異なった２つのＭＡＣアドレスが同じハッシュ・アドレスにハッシュする限りにおいて、一意である必要はない。チェーン・ハッシュ・エントリ・セクション９０８は、本明細書で後に詳述するように、異なったデバイスの重複したハッシュ・アドレスを格納すべく用意されている。ハッシュ・アドレスでアクセスできる一次ハッシュ・エントリ・セクション９０６と、一次ハッシュ・エントリ・セクション９０６の先頭エントリ内にあるリンク・アドレスでアクセス可能なチェーン・ハッシュ・エントリ・セクション９０８による編成により、少なくとも１つのブランチ動作が節約できる。ポインタのリストを用いてテーブル・エントリにアクセスするのではなく、メモリ２１２内の最初のエントリは１回のブランチ動作で検索され、次のエントリは２回目のブランチ動作で、というように以下同様である。このように、メモリ２１２の以上のような編成によってアクセス１回につき少なくとも１つのブランチ動作が節約できるため、ハッシュ・エントリに対するアクセスの効率が向上する。

【０１９２】

ハッシュテーブル・エントリ９１０の次のバイト（６）には、デバイスが接続されている関連のポート番号を識別する２進ポート番号（ＰｏｒｔＮｕｍ）が入っており、ここでＰＯＲＴ０のポート番号はゼロ、ＰＯＲＴ１のポート番号は１、（ＣＰＵ２３０の）ＰＯＲＴ２８のポート番号は２８、というようになっている。次のバイト（７）は、制御およびエイジ情報バイト（ＣＯＮＴＲＯＬ／ＡＧＥ）であり、エントリが有効であるかどうかを識別するバリッド・ビット（ＶＡＬＩＤＥＮＴＲＹ）を含み、これがロジック１であればそのエントリは有効、ロジック０であればその項目は有効でない、つまり空ビットであることを示す。ＣＯＮＴＲＯＬ／ＡＧＥバイトは、このデバイスに関する最後のソース・アクセスからの経過時間を表す２進のエイジ数（ＡＧＥ）を含む。最後のソース・アクセスから予め決められた不使用の時間量が経過すれば、デバイスは老化してＣＰＵ２３０によってハッシュ・エントリから削除される。経過時間はいくつかの方法の１つを用いて測られ、その単位は秒かそれ以下、分、時、その他である。デバイスを老化とみなす不使用時間は、プログラミングが可能である。他の実施例において、ＡＧＥ数は特定のデバイスが「旧」であるかどうかを表すために用いられる１つのビットであり、一定の経過時間あるいはそのような要因で設定される。

【０１９３】

次の４バイト（Ｂ：８）は、もし適用されていれば、ポートのグループを表す２９ビットの仮想ＬＡＮ（ＶＬＡＮ）のビットマップ値を定義する。ＶＬＡＮ値の各ビットはポートのそれぞれ１つに対応し、デバイスかポートがそのポートとグループ化されれば、セットされる。従ってＶＬＡＮ値は、その他のポートのうち、どのポートがデバイスとグループ化されたかを識別する。これにより、ネットワーク１０６、１１２を任意の組み合わせでグルーピプ化して、ネットワーク・スイッチ１０２に結合された複数の異なったＴＬＡＮを形成することができる。例えば、最初の５ポートＰＯＲＴ０～ＰＯＲＴ４が一緒になってグルーピングされれば、それぞれのＶＬＡＮ値は００００００１Ｆｈとなる。ここで、ｈは１６進数を示す。ＰＯＲＴ２に結合されている１つのデバイスから送られたＢＣパケ

ットは、ＰＯＲＴ０、ＰＯＲＴ１、およびＰＯＲＴ３にリピートされ、ネットワーク・スイッチ１０２のその他のすべてのポートにはリピートされない。ＶＬＡＮ値が全部１か１ＦＦＦＦＦＦＦｈであれば、そのデバイスにグループ化が適用されていないことを表している。１つのデバイスを複数のグループに関連させられることに留意する必要がある。他の実施例においては、各デバイスが属するいくつかのＶＬＡＮグループがあれば、それらの２つ以上を識別するために１つのＶＬＡＮフィールドを含むことができる。

【０１９４】

各ハッシュ・テーブル・エントリ９１０の最後の４バイト（Ｆ：Ｃ）は、チェーン・ハッシュ・エントリ・セクション９０８内で、もしあれば、同じハッシュ・アドレスを持った次のエントリを指示するリンク・アドレス（ＬＩＮＫ　Ａ［３１：０］すなわちＬＩＮＫ　ＡＤＤＲＥＳＳ）である。次のエントリは、チェーン・ハッシュ・エントリ・セクション９０８内で次の使用可能なロケーションに格納されている。このように、２つの異なったデバイスの２つのＭＡＣアドレスが同一のハッシュ・アドレスにハッシュすれば、最初の、すなわち「一次」エントリが一次ハッシュ・エントリ・セクション９０６に格納され、２番目のエントリがチェーン・ハッシュ・エントリ・セクション９０８内に格納され、一次エントリのＬＩＮＫ　ＡＤＤＲＥＳＳが２番目のエントリを指示する。別のＭＡＣアドレスが最初の２つと同じハッシュ・アドレスをハッシュすれば、各追加エントリはチェーン・ハッシュ・エントリ・セクション９０８に格納され、ＬＩＮＫＡＤＤＲＥＳＳによって連続した順序で、一緒に連鎖される。従って、最初が２番目を指示し、２番目が３番目を指示し、以下同様となる。それぞれのエントリは、ハッシュ・テーブル・エントリ９１０のフォーマットに従う。ＬＩＮＫ　ＡＤＤＲＥＳＳの形式は、適宜自由に定義することができる。ＬＩＮＫ　ＡＤＤＲＥＳＳは一般に、メモリ２１２内のハッシュ・メモリ・セクション９０２を指示するベース・アドレス・ポーション、およびハッシュ・メモリ・セクション９０２内の実際のエントリへのオフセット・ポーションを含む。下位のアドレス・ビットは、バイト整合のために必要に応じてゼロに設定する。各チェーン内の最後のエントリは、ＬＩＮＫ　ＡＤＤＲＥＳＳの一部をゼロにセットして識別する。例えば、ＬＩＮＫ　ＡＤＤＲＥＳＳのビット［３１：２８］をゼロにセットして、最後のエントリを表す。

【０１９５】

図４４は、メモリ２１２のパケット・メモリ・セクション９０４の編成を示すブロック図である。示した実施例において、メモリ２１２のパケット・メモリ・セクション９０４は複数の隣接した等しいサイズのセクタ９１２として編成され、各セクタ９１２は、セクタ・プレフィクス９１４と呼ばれるセクタ情報セクション、および１つまたは複数のパケット・データ・ブロックを含むパケット・セクション９１６を含む。各セクタ９１２は、設計を簡略化しオーバーヘッドを下げるため、メモリ２１２の機能を遂行するメモリ・デバイスのページ・サイズに対応して、そのサイズを２Ｋバイトにすることが望ましい。示した実施例において、ＦＰＭ　ＤＲＡＭ　ＳＩＭＭは４Ｋバイトのページ・バウンダリ（境界）で編成され、同期ＤＲＡＭ　ＳＩＭＭは２Ｋｂｙｔｅのページ・バウンダリで編成されている。従って、２Ｋバイトのセクタ・サイズで、サポートされるタイプのメモリ・デバイスに十分である。セクタ９１２は初期において空であるが、ＬＩＮＫ　ＡＤＤＲＥＳＳと一緒にチェーン化されて、空メモリ・セクタのＦＲＥＥＰＯＯＬ　ＣＨＡＩＮ（フリープール・チェーン）を形成する。

【０１９６】

ポート１０４、１１０のそれぞれから新しい情報のパケットが受け取られると、１つまたは複数のセクタ９１２がＦＲＥＥＰＯＯＬ　ＣＨＡＩＮから切り離され、１ポートにつき１つのＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮ内で一緒にリンクされる。また、各パケットは、同じまたは別のＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮ内で他のパケットとリンクされ、１ポートにつき１つのＴＲＡＮＭＩＴ　ＳＥＣＴＯＲ　ＣＨＡＩＮ形成する。このようにして、１つのポートのＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮ内のパケットは、さらに別のポートのＴＲＡＮＭＩＴ　ＳＥＣＴＯＲ　ＣＨＡＩＮにも入れられ

る。セクタ９１２のパケット・セクション９１６内のデータがすべて受信先のポートへ送信されると、そのセクタは、そのＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮから解放され、再びＦＲＥＥＰＯＯＬ　ＣＨＡＩＮに戻ってリンクされる。ＲＥＣＥＩＶＥ　ＳＥＣＴＯＲおよびＦＲＥＥＰＯＯＬチェーンの機能は、本明細書で後に詳述する方式で、１つのセクタから次のセクタへのリンク・アドレスあるいはポインタを用いて遂行される。

【０１９７】

図４５は、パケット・メモリ・セクション９０４の各セクタ９１２の各セクタ・プレフィクス９１４の編成の図解である。セクタ・プレフィクス９１４は、対応するセクタ９１２の情報を含み、さらに次のセクタ９１２があれば、それへのリンクとして機能する。プレフィクスの情報部分は、セクタ９１２内のどこに入っていてもよい点に留意されたい。最初のバイト（０）は、そのときのセクタ９１２内のパケットまたはパケット片の数を表す２進のセクタ・パケット・カウント（ＳｅｃＰｋｔＣｎｔ）を定義する。セクタ・パケット・カウントは、そのセクタにパケット・データが格納されると増分され、受信先のポートによる送信のためにデータが読み出されると減分される。セクタ・パケット・カウントＳｅｃＰｋｔＣｎｔがゼロに減分されたとき、そのセクタがＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮの最後にあるものでなければ、該セクタは解放されてＦＲＥＥＰＯＯＬ　ＣＨＡＩＮに戻る。次のバイト（１）は、受信したパケットの送信元ポートを示すセクタ・ソース値（ＳｅｃＳｏｕｒｃｅ）である。この値は、そのセクタが解放されてＦＲＥＥＰＯＯＬ　ＣＨＡＩＮに戻るとき、当該受信ポート・セクタ・カクント（ＲｘＳｅｃＣｎｔ）を識別して減分するために必要である。次の２つのバイト（３：２）は、リザーブすなわち未使用となっている。

【０１９８】

それぞれのセクタ・プレフィクス９１４内の次の４バイトは、対応するＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮまたはＦＲＥＥＰＯＯＬ　ＣＨＡＩＮ内の次のセクタへのネクスト・リンク・アドレス（ＮｅｘｔＳｅｃＬｉｎｋ）である。同一のリンク・アドレスが両方の目的に使用されているが、異なったリンク・アドレスを用いてもよい。示した実施例において、ＮｅｘｔＳｅｃＬｉｎｋアドレスは３２ビットで、ベース（基準）およびオフセットの部分から成る。下位の“ｎ”個のビットは、ＮｅｘｔＳｅｃＬｉｎｋのセクタ・サイズに応じた整合のために、ゼロにセットしてもよい。整数“ｎ”は、４Ｋバイトのセクタでは１２、２Ｋバイトのセクタでは１１、１１Ｋバイトのセクタでは１０、そして５１２Ｋバイトのセクタでは９である。示した実施例においては、ｎは２Ｋバイトのセクタに１１、などとなっている。このようにして、ポート１０４、１１０から１つまたは複数のパケットが受け取られると、そのポートによって受信されたその１つまたは複数のパケットを格納すべく、セクタ９１２のＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮが１つ割り当てられる。複数のセクタ９１２は、そのチェーン内の各セクタ９１２のセクタ・プレフィクス９１４内のＮｅｘｔＳｅｃＬｉｎｋアドレスを用いて、チェーン化の方式で一緒にリンクされる。パケット・データは、各ＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮ内のそれぞれのセクタ９１２のパケット・セクション９１６内に順に格納される。１つのパケットのパケット・データは、ＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮ内のセクタ・バウンダリを越えてもよいという点に留意する必要がある。セクタ・プレフィクス９１４の最後の８バイト（１５：８）は、リザーブすなわち未使用となっている。

【０１９９】

図４６は、パケット・セクション９１６内の各パケット・データ・ブロックを表す例示的なパケット・データ・ブロック９１７の図である。パケット・データ・ブロック９１７は、パケット・ブロック・ヘッダ９１８およびパケット・データ・セクション９２０という２つの部分に分かれている。パケット・ブロック・ヘッダ９１８は、ＭＣＢ４０４によって各パケットの前に付加されてパケット・データ・ブロック９１７を形成するのが望ましい。パケット・ブロック・ヘッダ９１８の最初の２バイト（１：０）は、パケットの長さをバイト数で定義する１５ビットの２進パケット長（ＰｋｔＬｅｎｇｔｈ）値、およびＣＴモードのパケットがポートの停動（ｓｔａｌｌ）のためメモリ２１２へ転送されたとき

にセットされる１ビットの中間（ミッド）パケットＣＴ値（ＭｉｄＰｋｔＣＴ）を形成する。ＭＣＢ４０４は、ＴＬＡＮ２２６のポートＰＯＲＴ２４とＰＯＲＴ２７、およびＣＰＵ２３０のポートＰＯＲＴ２８へ送信するとき、ＰｋｔＬｅｎｇｔｈを含むこの最初のＤＷＯＲＤをパケットに付加する。パケット・ブロック・ヘッダ９１８の次のバイト（２）は、パケットのソース・ポートすなわち送信元ポート（ＳｏｕｒｃｅＰｏｒｔ）番号を識別する。これは、ソース・アドレスに関するポート番号を識別するための８ビットで２進のポートＩＤ番号である。送信元ポートは、そのパケットが格納されている特定のＲＥＣＥＩＶＥ　ＳＥＣＴＯＲＣＨＡＩＮによっても識別される。次のバイト（４）は、宛先ポートすなわち受信先ポート（ＤｅｓｔＰｏｒｔ）番号を識別する。これは、ＳｏｕｒｃｅＰｏｒｔ値の場合と同様に、受信先のポート番号を識別するための８ビットで２進のポートＩＤ番号である。受信先ポートは、そのパケットが属する特定のＴＲＡＮＭＩＴ　ＰＡＣＫＥＴ　ＣＨＡＩＮによっても識別される。

【０２００】

パケット・ブロック・ヘッダ９１８の４バイト（１１：８）は、ＴＲＡＮＭＩＴ　ＰＡＣＫＥＴ　ＣＨＡＩＮ内の次のデータ、またはパケット・データ・ブロック９１７への３２ビットのネクスト・リンク・アドレス（ＮｅｘｔＴｘＬｉｎｋ）を定義する。送信パケット・カウント（ＴｘＰｋｔＣｎｔ）がゼロまで減分されたとき、ＴＲＡＮＭＩＴ　ＰＡＣＫＥＴ　ＣＨＡＩＮの終わりが表示される。ＮｅｘｔＴｘＬｉｎｋアドレスの下位ビットＡ０は、次のパケットがブロードキャストであるか否かを示すＢＣパケット・ビット（ＮｅｘｔＰｋｔＢＣ）として使用される。ＮｅｘｔＰｋｔＢＣ＝１であれば、次のパケットは後述するブロードキャストの形式であり、もしＮｅｘｔＰｋｔＢＣ＝０であれば、次のパケットは非ブロードキャストである。ＮｅｘｔＴｘＬｉｎｋアドレスの次の下位ビットＡ１は、次のパケットがＳｎＦであるか否かを同様に表示するＳｎＦパケット・ビット（ＮｅｘｔＰｋｔＳｎＦ）として使用される。ＮｅｘｔＴｘＬｉｎｋアドレスの下位半バイト（４ビット）は、その半バイトの実際の値にかかわらず、バイト整合の目的にゼロと想定してもよいことに留意されたい。従って、例えばＮｅｘｔＴｘＬｉｎｋアドレスが読み取られるとき、ビットＡ［３：０］が実際はＮｅｘｔＰｋｔＢＣ＝１のような値であっても、これを無視してゼロと想定することができる。これにより、これらのビットは代替用途に使用することができる。示した実施例においては、下位ビットＡ［３：０］がゼロと想定されるように、データ構造が１６バイト整合となっている。

【０２０１】

示した実施例においては、パケット・データ・セクション９２０がパケット・ブロック・ヘッダ９１８の直後に置かれ、パケット・ヘッダ内でデータフィールドの長さが定義される。しかし、示した実施例における各セクタの特定の序列や各値の特定の位置などは多少恣意的であって例示の域を出ないものであり、従って本発明の範囲を越えない限りにおいて、編成は必要に応じて任意である。

【０２０２】

先に述べたように、パケットは、ポートＰＯＲＴ０～ＰＯＲＴ２８の各々から検索され、セクタ９１２の対応する受信セクタ・チェーン（ＲＥＣＥＩＶＥ　ＳＥＣＴＯＲ　ＣＨＡＩＮ）に格納される。受信セクタ・チェーンは、ポート当たり１つ対応して設けられている。図４８示されるように、第１の受信セクタ・チェーン９３０がＰＯＲＴ０に対して示され、ここで第１のセクタ９３１のセクタ・プレフィックス９１４におけるＮｅｘｔＳｅｃＬｉｎｋを用いて、セクタ９３１が別のセクタ９３２にリンクされる。必要に応じて、セクタ・プレフィックス９１４におけるリンク・アドレスを用いて、更に他のセクタがリンクされる。また、第２の受信セクタ・チェーン９４０がＰＯＲＴ１に対して示され、このポートで、第１のセクタ９４１のセクタ・プレフィックス９１４におけるＮｅｘｔＳｅｃＬｉｎｋを用いて、セクタ９４１が別のセクタ９４２にリンクされる。あるポートで受取られた各パケットごとに、パケット・ブロック・ヘッダ９１８が、対応する受信セクタ・チェーンのその時のセクタ（現在セクタ）９１２のパケット・セクション９１６において前に受取られたパケット・データ・ブロック９１７の直後に置かれ、パケット・ブロッ

ク・ヘッダ９１８に、そのパケット・データ・セクション９２０が後続する。現在セクタ９１２のパケット・セクション９１６がパケット・データ・ブロック９１７を格納中に一杯になると、別のセクタ９１２がフリープール・チェーン（ＦＲＥＥＰＯＯＬ　ＣＨＡＩＮ）から割付けられ、当該ポートに対する受信セクタ・チェーンリンクされる。このように、ポートから受取ったパケット・データ・ブロック９１７は、当該ポートに関して対応する受信セクタ・チェーン内に連続的に配置される。また、セクタ９１２のパケット・セクションは、パケット全体および（または）パケットの部分を含むことができる。

【０２０３】

したがって、図４８に示されるように、ポートＰＯＲＴ０で受取られたパケット・データ・ブロック９３４、９３５および９３６が、セクタ９３１、９３２内に配置される。パケット・データ・ブロック９３５がセクタ９３１、９３２に跨がることに注目されたい。同様に、ポートＰＯＲＴ１で受取られたパケット・データ・ブロック９４４および９４５が、図示のように、セクタ９４１、９４２内に置かれ、パケット・データ・ブロック９４５がセクタ９４１、９４２に跨がっている。

【０２０４】

各パケットはまた、各宛先ポートに対するパケットの送信パケット・チェーン（ＴＲＡＮＳＭＩＴ　ＰＡＣＫＥＴ　ＣＨＡＩＮ）と関連させられ、該ポートでは、これらのパケットが、ＮｅｘｔＳｅｃＬｉｎｋアドレスを用いて、一緒にリンクされる。各送信パケット・チェーンにおけるパケットは一般に、ネットワーク・スイッチ１０２により受取られる時間に基いて順序付けられ、その結果、関連する宛先ポートへ送られる時、この順序が維持される。例えば、図４８に示されるように、パケット・データ・ブロック９３４、９４４がポートＰＯＲＴ１０から送られべきであり、そしてパケット・データ・ブロック９３４がパケット・データ・ブロック９４４の直前に送られるべきならば、パケット・データ・ブロック９３４のパケット・ブロック・ヘッダ９１８のＮｅｘｔＴｘＬｉｎｋアドレスが、パケット・データ・ブロック９４４を指示する。パケット・データ・ブロック９４４のパケット・ブロック・ヘッダ９１８のＮｅｘｔＴｘＬｉｎｋアドレスは、次に送られるべきパケット・データ・ブロックを指示する、、、の如くである。伝送の実際の順序は、１つのパケットが送信パケット・チェーンへリンクされる時に決定される。ＣＴモード・パケットは、このパケットが受取られる時の初めにリンクされ、ＳｎＦモード・パケットは、パケット全体が格納された後にリンクされる。中間パケット暫定ＣＴモード・パケットは、適切な順序付けを保証するため、対応する送信パケット・チェーンの前にリンクされる。

【０２０５】

図４７は、正規（通常）のパケット・ブロック・ヘッダ９１８を置換する、ＢＣ（ブロードキャスト）パケットに対して用いられる１２８バイトのパケット・ヘッダ９２２を示すブロック図である。ＢＣパケットにおいては、ＮｅｘｔＰｋｔＢＣ値が前のパケットにセットされて現在パケットがＢＣパケットであることを示す。各送信パケット・チェーンが、伝送されるＢＣパケットを含む全てのポートに対して維持されるべきである。従って、ＢＣパケット・ヘッダ９２２は、０～２８が番号が付された各ポート（ポート１０４、１１０及びＣＰＵ２３０を含む）ごとに、４バイトのリンク・アドレス（Ｐｏｒｔ＃ＮｅｘｔＴｘＬｉｎｋ）を含み、各ＮｅｘｔＴｘＬｉｎｋアドレスが、リストにおける場所（ポート番号Ｐｏｒｔ＃）により識別される対応ポートと関連する送信パケット・チェーンにおける次のパケットを指示する。このように、ＮｅｘｔＴｘＬｉｎｋアドレスは、バイト（１１：８）で始まり、バイト（１２３：１２０）で終る。第１のＮｅｘｔＴｘＬｉｎｋアドレス・エントリ（１１：８）は、第１のポートＰＯＲＴ０に対するメモリ２１２における次のパケットと対応し、第２のエントリ（バイト１５：１２）は、第２のポートＰＯＲＴ１に対するメモリ２１２における次のパケットに対するＮｅｘｔＴｘＬｉｎｋアドレスである。このように、ＣＰＵ２３０に関する次のパケットに対するＮｅｘｔＴｘＬｉｎｋである最後のエントリ（バイト１２３：１２０）まで続いている。各ＢＣリンク・アドレスもまた、各送信パケット・チェーンにおける次のパケットがＢＣパケットか否かを

示す次のＢＣパケット（ＮｅｘｔＰｋｔＢＣ）ビットと、各送信パケット・チェーンにおける次のパケットがＳｎＦパケットか否かを示す次のＳｎＦパケット（ＮｅｘｔＰｋｔＳｎＦ）ビットとを含んでいる。

【０２０６】

ＢＣパケット・ヘッダ９２２の最初の４バイト（３：０）は、正規のパケット・ブロック・ヘッダ９１８の最後の４バイトに類似し、ＭｉｄＰｋｔｃｔ値がＢＣパケットに対してゼロであることを除いて、ＰｋｔＬｅｎｇｔｈ、ＭｉｄＰｋｔＣＴ、ＳｏｕｒｃｅＰｏｒｔ（ソース・ポート）およびＤｅｓｔＰｏｒｔ（宛先ポート）の値を含んでいる。ＢＣパケット・ヘッダ９２２の次の４バイト（７：４）は、バイト２８：０の各々がＢＣパケット・データを受取るポートに対応するブロードキャスト・ポート・ビットマップ（ＢＣ＿Ｐｏｒｔｓ）である。各ビットは、パケットが対応するポートへ送られる時にクリアされる。全てのＢＣ＿ポート・ビットがクリアされた時、先に述べたＳｅｃＰｋｔＣｎｔカウントもまた減分される。

【０２０７】

図４９には、各々が同じＢＣパケット１０１０を包含する幾つかの送信パケット・リンクを示すブロック図が例示される。この例では、ポート１、５、１１および１２が、ＶＬＡＮ関数などを用いてグループ化され、その結果、ポート１２の如き１つのソース・ポート（例えば、ポート１２）で受取られるＢＣパケット１０１０のデータが当該グループにおける残りのポート（ポート１、５および１１）に複写される。４つの送信パケット・チェーン１００２、１００４、１００６および１００８が、それぞれポート１、５、１１および１２に対して示される。送信パケット・チェーン１００２、１００４および１００６は、幾つかの一般的な非ブロードキャスト・パケット１０００をＢＣパケット１０１０とリンクする。ポート１２がソース・ポートであるから、ＢＣパケット１０１０はポート１２に送られず、従ってこのポートは送信パケット・チェーン１００８には含まれない。ＢＣパケット１０１０はＢＣパケット・ヘッダ１０１２を含み、このヘッダは、ポート１の送信パケット・チェーン１００２における次のパケット１０００を指示するリンク・アドレス１０１６と、ポート５の送信パケット・チェーン１００４における次のパケット１０００を指示するリンク・アドレス１０１８と、ポート１１の送信パケット・チェーン１００６における次のパケット１０００を指示するリンク・アドレス１０２０とを含む、各ポートに１つずつリンク・アドレスのリストを含んでいる。このように、送信パケット・チェーン１００２、１００４および１００６の各々が保持される。各送信パケット・チェーンが１つ以上のＢＣパケットを含み、これが必要に応じて、非連続的あるいは連続的に現れることも判る。

【０２０８】

図５０は、１組のＭＣＢパケット制御レジスタ１１０２を示すブロック図であり、これらのレジスタはＳＲＡＭ６５０内に備えられて、ネットワーク・スイッチ１０２のＣＰＵ２３０を含む２９個のポート１０４、１１０の各々に対して同様に備えられている。ＣＰＵ２３０は、スパニング・ツリー処理のためのブリッジ・プロトコル・データ・ユニット（ＢＰＤＵ）の送受などのある目的のため、「ポート（ＰＯＲＴ２８）」として扱われる。各ＭＣＢパケット制御レジスタ１１０２は、受信セクション１１０４と送信セクション１１０６とを含んでいる。受信セクション１１０４では、２８ビットの受信パケット・ヘッダのベース・ポインタ（ＲｘＢａｓｅＰｔｒ）が、当該ポートに対する受信セクタ・チェーンの初めである対応ポートに対応するその時の受信パケット・ヘッダのベース（基底）に対するポインタである。メモリ２１２について先に述べたように、ＳＲＡＭ６５０に対するデータ構造は、１６バイトが割り当てられ、全てのポインタの最下位ビットＡ［３：０］がゼロと仮定される。２８ビットのそのときの受信ポインタ（ＲｘＣｕｒＰｔｒ）は、当該ポートの受信セクタ・チェーンに関するその時のデータ記憶場所に対するポインタである。ＲｘＣｕｒＰｔｒ値の下位４ビットは、受信ＢＣパケット表示ビット（ＲｘＢＣ）と、「パケット開始（ＳＯＰ）」フラグとして用いられる受信伝送進行中（ＲｘＩＰ）ビットと、その時のパケットがセクター境界と交差するかどうかを示す多重セクタ・パケ

ット（ＭｕｌｔｉＳｅｃＰｋｔ）ビット１と、送信リンクがパケットの終りで更新されることを示すＳｎＦビット０とを含む、制御ビットである。受信セクション１１０４は更に、ＭｉｄパケットＣＴビット（ＭｉｄＣＴ）と、ＲｘＣｕｒＰｔｒまでのバイトで受取られるその時のパケットの長さに等しい１６ビットの受信パケット長（ＲｘＰｋｔＬｎ）値と、対応するポートによりその時使用中であるセクタの数を示す１６ビットの受信ポート・セクタ・カウント（ＲｘＳｅｃＣｎｔ）と、各ポートまたは受信セクタ・チェーンに対して許容されるＣＰＵプログラムされたセクタ最大数を識別する１６ビットの受取りセクタ閾値（ＲｘＳｅｃＴｈｒｅｓｈｏｌｄ）値とを含んでいる。ＲｘＳｅｃＴｈｒｅｓｈｏｌｄ値は、該ＲｘＳｅｃＴｈｒｅｓｈｏｌｄをＲｘＳｅｃＣｎｔと比較することにより、バックプレシャが当該ポートに対して加えられるべきかどうかを決定するために用いられる。バックプレシャがディスエーブル（不動作）状態にされると、ＲｘＳｅｃＴｈｒｅｓｈｏｌｄ値を用いて、対応するポートで受取られる更なるパケットをドロップする（捨てる）。

【０２０９】

受信セクション１１０４は更に、対応するポートに対する送信パケット・チェーンにおける最後のパケットのベースを示す２８ビットのポインタである送信キュー・ポインタ（ＥｎｄＯｆＴｘＱＰｔｒ）の終りを含んでいる。最後に、送信キューＢＣ（ＥＯＱ＿ＢＣ）の終りが、対応するポートに対する送信パケット・チェーンにおける最後のパケットに対するブロードキャスト・フォーマットを示すようにセットされる。

【０２１０】

送信セクション１１０６は、対応するポートに送信パケット・チェーンに関する情報を提供する。送信ベース・ポインタ（ＴｘＢａｓｅＰｔｒ）は、その時の伝送パケット・ヘッダのベースに対する２８ビットのポインタであり、別の２８ビットの送信の現在ポインタ（ＴｘＣｕｒＰｔｒ）が、対応するポートに対するその時のデータ検索場所を指示する。送信ブロードキャスト（ＴｘＢＣ）ビットが、パケット・ヘッダがブロードキャスト・フォーマットであることを示すようにセットされる。送信進行中（ＴｘＩＰ）ビットが論理値１にセットされると、それにより、送信がその時ポートに対して進行中であり、ＳＯＰを示す。８ビットの送信ソース・ポート（ＴｘＳｒｃＰｏｒｔ）番号は、ＳＯＰにおけるパケット・ヘッダから読出されるその時の送信パケットのソース・ポート番号ある。１６ビット送信パケット長（ＴｘＰｋｔＬｎ）値は、その時の送信パケットに対して送られるべき残りのバイトと等しい。あるパケットが伝送されるとき、パケットのパケット・ブロック・ヘッダ９１８におけるＰｋｔＬｅｎｇｔｈ値が送信セクション１１０６におけるＴｘＰｋｔＬｎ値へ複写され、次いでＴｘＰｋｔＬｎ値は、パケットが伝送される時、ＴＸコントローラ６０６によって減分される。ＴｘＰｋｔＬｎ減分されてゼロになると、ＥＰＳＭ２１０が、パケットの終りを示す対応するＥＯＰ＊信号を生成する。１６ビットの最大パケット数（ＴｘＰｋｔＴｈｒｅｓｈｏｌｄ）値は、各ポートに対してキューさせられるＣＰＵプログラムされたパケットの最大数に等しい。ＣＰＵ２３０を宛て先とするパケットがＴｘＰｋｔＴｈｒｅｓｈｏｌｄまたはＲｘＰｋｔＴｈｒｅｓｈｏｌｄの制限を受けないことが判る。最後に、１６ビットの送信パケット・カウント（ＴｘＰｋｔＣｎｔ）は、対応するポートに対してその時にキューされるパケットの数に等しい。

【０２１１】

図５１は、ＳＲＡＭ６５０に置かれたフリープール・パケット制御レジスタ１１０８を示すブロック図であり、これらのレジスタは、レジスタのフリープール・チェーン（ＦＲＥＥＰＯＯＬ　ＣＨＡＩＮ）と関連されている。各フリープール・レジスタ１１０８は、フリープール・チェーンにおける次の自由なフリー・セクタに対するポインタ（ＮｅｘｔＦｒｅｅＳｅｃＰｔｒ）と、フリープール・チェーンにおける最後のセクタに対するポインタ（ＬａｓｔＳｅｃＣｎｔ）と、その時利用可能なフリー・セクタの数に等しいフリー・セクタ・カウント（ＦｒｅｅＳｅｃＣｎｔ）と、メモリ・オーバーフロー・フラグ（ＭＯＦ）がバックプレシャまたはフィルタリング（パケットの抜き取り）の目的のためにセットされる前に許容される、ＣＰＵプログラムされたセクタの最小数に等しいフリー・セク

タ閾値（ＦｒｅｅＳｅｃＴｈｒｅｓｈｏｌｄ）数と、その時にメモリ２１２にあるＢＣパケット数に等しいＢＣパケット・カウント（ＢＣ　ＰｋｔＣｎｔ）と、メモリ２１２に許容されるＢＣパケットのＣＰＵプログラムされた最大数に等しいＢＣパケット閾値（ＢＣ　ＰｋｔＴｈｒｅｓｈｏｌｄ）カウントとを含んでいる。

【０２１２】

図５２は、メモリ２１２へのデータ・パケットの受取りのため、およびＣＴ動作モードにおけるデータ・パケットの送信のための、ネットワーク・スイッチ１０２の動作をフロー図で示している。データは通常、リアルタイムであるいは全体的にパケットの形態におけるネットワーク・スイッチ１０２のポートＰＯＲＴ０～ＰＯＲＴ２７により送受信され、セグメント１０８、１４に跨がって送られている間は、細分割されることはない。しかし、ネットワーク・スイッチ１０２内のＦＩＦＯは一般に、全パケットを格納するのに充分なほどには大きくない。このため、パケット・データは、ネットワーク・スイッチ１０２内で、パケットの一部あるいはパケットの細分割の形態で、１つのＦＩＦＯから別のＦＩＦＯへ送られる。

【０２１３】

第１のステップ１２００において、ＥＰＳＭ２１０は、信号ＰＫＴ＿ＡＶＡＩＬｍ＊の表示により、ポート１０４、１１０の一方により受取られる新たなパケットを検出する。次のステップ１２０２において、パケットの初めの部分即ちヘッダがソース・ポートから検索されて、ハッシュ・リクエスト・ロジック５３２へ読込まれる。ヘッダは、宛先ＭＡＣアドレスおよびソースＭＡＣアドレスを含んでいる。ハッシュ・リクエスト・ロジック５３２は、宛先アドレスおよびソース・アドレスとソース・ポート番号を、ＨＡＳＨ＿ＤＡ＿ＳＡ［１５：０］信号中に与え、ＭＣＢ４０４へＨＡＳＨ＿ＲＥＱ＊信号をアサートする。ＭＣＢ４０４は、それに応答して、パケットに対する適切な動作を決定するためのハッシング手順を呼出し、ソース・アドレスおよび宛先アドレスがハッシュされて、このアドレスのいずれかがそれ以前にメモリ２１２に格納されたかどうかを判定する。ＭＣＢ４０４は、ＨＣＢ４０２に対して充分な情報が得られる場合に、信号ＨＡＳＨ＿ＤＯＮＥ＊をアサートして、パケットに対してとるべき適切な動作を判定する。図５２に示されるフロー図は、宛先アドレスおよびソース・アドレスに関する２つの主要部分を含んでおり、これについては後述する。図示した実施例では、宛先アドレスが最初にハッシュされ、ソース・アドレスがその後に続くが、これらの手順は同時に実行しても良いし、所望の順番で実行してもよい。

【０２１４】

宛先アドレスの場合は、処理はステップ１２０４へ進み、ハッシング手順が呼出されて宛先アドレスをハッシュする。信号ＨＡＳＨ＿ＤＯＮＥ＊に応答して動作がステップ１２０４からステップ１２０８へ進んで、ユニキャストとＢＣパケットの双方に対するスレッショルド・コンディション（閾値条件）を調べる。ステップ１２０８において、関連するスレッショルド・コンディションを新たなパケットが違反するかどうかが判定される。特に、ＦｒｅｅＳｅｃＣｎｔ数がＦｒｅｅＳｅｃＴｈｒｅｓｈｏｌｄ数と等しいかあるいはこれより小さければ、パケットをメモリ２１２に格納するのに充分な余地がないことである。また、ＲｘＳｅｃＣｎｔがソース・ポートに対するＲｘＳｅｃＴｈｒｅｓｈｏｌｄより大きいかあるいはこれに等しければ、ネットワーク・スイッチ１０２が、パケットをドロップする（捨てる）ことを決定する。ＢＣパケットの場合は、ＢＣ＿ＰｋｔＴｈｒｅｓｈｏｌｄ数が、ＢＣパケットの実際数であるＢＣ＿ＰｋｔＣｎｔ数に比較されて、ＢＣパケットの最大数が既に受取られたかどうか判定する。ユニキャスト・パケットの場合は、ＴｘＳｅｃＴｈｒｅｓｈｏｌｄ数が、宛先ポートに対するＴｘＳｅｃＣｎｔに比較される。

【０２１５】

ステップ１２０８からステップ１２０５へ進み、ここでＨＣＢ４０２が、ＨＡＳＨ　ＳＴＡＴＵＳ［１：０］信号から、またスレッショルド・コンディションのどれかの比較から、パケットをドロップすべきかどうか判定する。このパケットは、先に述べたような様々な他の理由、例えば、ソース・ポートと宛先ポートが等しいなどの理由からドロップされ

る。パケットがドロップされるべきであれば、動作はステップ１２０５からステップ１２０７へ進み、ここでパケットがドロップされるかあるいはバックプレシャ（ＢＣ）が加えられる。条件ＦｒｅｅＳｅｃＴｈｒｅｓｈｏｌｄまたはＲｘＳｅｃＴｈｒｅｓｈｏｌｄが違反され、かつバックプレシャがイネーブル状態にされソース・ポートがハーフ２重モードで動作しているならば、バックプレシャが提供される。さもなければ、パケットがドロップされる。バックプレシャにおいては、ＥＰＳＭ２１０がＨＳＢ２０６においてバックプレシャ・サイクルを実行して、ソース・ポートに送出側装置に対するジャミング・シーケンスをアサートさせる。ＡＢＯＲＴ＿ＯＵＴ＊信号により示されるように、バックプレシャ指令がソース・ポートにより受入れられなければ、この指令がジャミング・シーケンスのアサートに遅すぎて提供されたことであり、パケットがドロップされる。また、ＢＣ＿ＰｋｔＴｈｒｅｓｈｏｌｄ条件が抵触される唯一つのスレッショルド・コンディションであっても、パケットがドロップされる。ネットワーク・スイッチ１０２がドロップされるパケットの残部を受取り続けるが、パケットは格納されず、あるいは別のポートへ送られない。動作は、ステップ１２０７からステップ１２１４へ進み、ここでＭＣＢコンフィギュレーション・レジスタ４４８における適切な統計レジスタが、ステップ１２０７で行われる動作に基いて更新される。当該統計レジスタは、オーバーフロー条件によりパケットがドロップされたかバックプレシャされたかを示す。例えば、ポート当たりの「ドロップされたパケット－バッファなし」カウントがソース・ポートに対して増分されて、パケットがオーバーフロー条件により捨てられるか、あるいは、パケットがバックプレシャされるならば、「バックプレシャされたパケット」カウントが増分されることを示す。

【０２１６】

パケットがドロップされなければ、動作はステップ１２０５からステップ１２０６へ進み、ここで宛先アドレス（ＤＡ）がハッシュ・メモリ・セクション９０２で見出されたかどうか、またパケットがブロードキャストされるべきかどうか判定される。宛先アドレスが認識されず従って宛先ポートが未知であるか、あるいはパケット内のグループ・ビットがセットされるならば、パケットがブロードキャストされる。宛先アドレスが見出されなければ、あるいは、パケットがステップ１２０６で判定されるようなＢＣパケットであるならば、パケットがブロードキャストされて動作がステップ１２１０へ進み、ＥＰＳＭ２１０のＭＣＢ４０４が、必要に応じて、新たなパケットに対するメモリ２１２内に別のセクタを割付ける。現在のすなわちその時のセクタがパケットに対して充分な余地を有するならば、新たなセクタは不要である。次いで、動作はステップ１２１６へ進み、パケットの残りがバースト単位でＥＰＳＭ２１０を介してバッファ記憶され、そしてメモリ２１２へ送られる。ポート設定の如何を問わず、ＢＣパケットがＳｎＦモードで処理され、ここで全パケットが伝送される前にメモリ２１２に格納される。ステップ１２１６から、動作がステップ１２１７へ進んで、パケット・エラーによるパケットの受取り中に信号ＡＢＯＲＴ＿ＯＵＴ＊がアサートされたことを判定する。ポートＰＯＲＴ１～ＰＯＲＴ２７により、ＦＩＦＯオーバーラン、ラン・パケット、オーバーサイズ・パケット、パケットが不正ＦＣＳを持つこと（フレーム検査シーケンス）の検出、あるいはＰＬＬエラーが検知されたかのような、幾つかのエラー条件が調べられる。パケット・エラーがステップ１２１７において検出されるならば、動作はステップ１２１９へ進み、ここでパケットがメモリ２１２から除去される。

【０２１７】

パケット・エラーがステップ１２１７で検出されなければ、動作はステップ１２１８へ進み、ここでＢＣパケットのパケット・ヘッダ９２２におけるブロードキャスト・ポート・ビット・マップＢＣ＿Ｐｏｒｔｓが、ＢＣパケットが送られるべきアクティブなポートで更新される。次のポート、即ち、ソース・ポートか、ソース・ポートがＣＰＵ２３０であるならば前送（ＦＯＲＷＡＲＤＩＮＧ）状態ではない任意のポートか、あるいは、ソース・ポートがＣＰＵ２３０であるならばディスエーブル状態の任意のポート、および対応するＴｘＰｋｔＴｈｒｅｓｈｏｌｄより大きいかこれと等しいＴｘＰｋｔＣｎｔ数を持つポートを除いて、ＢＣパケットがポート１０４、１１０の全てへ送られる。ＶＬＡＮがイネ

ーブル状態にあるならば、ハッシュ・テーブル・エントリ９１０におけるＶＬＡＮビット・マップ値もまた調べられ、これが更に、ポートをＶＬＡＮグループにおけるアクティブ状態の関連ポートに限定する。また、パケットが未知の宛先アドレスによりブロードキャストされるミスＢＣパケットが、ＭｉｓｓＢＣＢｉｔＭａｐレジスタに従って前送される。パケットがいずれのポートにも送られないように、得られたＢＣ＿Ｐｏｒｔのビットマップが全てゼロであるならば、この判定がステップ１２０５で行われて当該パケットがステップ１２０７でドロップされ、あるいはパケットはステップ１２１９でメモリ２１２から除去されることが判る。

【０２１８】

動作はステップ１２１８からステップ１２２０へ進み、得られたＢＣ＿Ｐｏｒｔのビット・マップにおける各ポートに対する送信パケット・チェーンへ、パケットが付加される。特に、パケット・ヘッダ９２２におけるＢＣ＿Ｐｏｒｔビット・マップで示される各ポートに対するＮｅｘｔＴｘＬｉｎｋリンク・アドレスの各々が更新されて、ＢＣパケットを適切なポートの送信パケット・チェーンに挿入する。他の全てのレジスタ、あるいはネットワーク・スイッチ１０２におけるカウント値および統計数値も、例えばＢＣ＿ＰｋｔＣｎｔ数のように同様に然るべく更新される。

【０２１９】

再びステップ１２０６に戻り、宛先アドレスが見出されたがパケットがＢＣパケットでなければ、動作はステップ１２２２へ進み、ここでハッシュ・キャッシュ・テーブル６０３が更新される。次いで、動作は次のステップ１２２４へ進み、ここでソース・ポートあるいは宛先ポートのいずれかがＳｎＦモードに対してセットされるかどうか質問される。両方のポートがＣＴモードにセットされ、等しいポート速度および宛先ポートに対するＴＢＵＳ設定がソース・ポートに対するＴＢＵＳ設定に等しいなどの、他のＣＴ条件が満たされるならば、動作はステップ１２２５へ進み、ここで宛先ポートへの経路が使用中であるかどうかが質問される。ステップ１２２４で決定されるようにＳｎＦモードが指示されるか、あるいはＣＴモードに指示されるが暫定ＣＴモードが開始されるように宛先ポートがステップ１２２５で決定されるように使用中であるならば、動作はステップ１２２６へ進み、ここでＥＰＳＭ２１０のＭＣＢ４０４が、必要に応じて、新たなパケットに対してメモリ２１２内のスペースを割付ける。ステップ１２２６から、動作はステップ１２２８へ進み、ここでパケットの残りの部分がＥＰＳＭ２１０へ検索され、メモリ２１２へ送られる。ステップ１２１７に類似するステップ１２２９で示されるように、パケットの受取り中にパケット・エラーが生じるならば、動作はステップ１２１９へ進んでメモリ２１２からこのパケットを除去する。さもなければ、動作は次のステップ１２３０へ進み、ここでパケットは宛先ポートの送信パケット・チェーンに付加され、適切なリンク・アドレス、カウントおよびチェーンが更新される。

【０２２０】

再びステップ１２２５において、宛先ポートの経路が使用中でなければ、動作はステップ１２３１へ進み、ここでソース・ポートおよび宛先ポートが、その時のパケットに対する正規のＣＴ動作に関して指定される。正規のＣＴモードでは、各々の残りのパケット部分がメモリ２１２へは送られず、その代わり、ＣＴ　ＢＵＦ（ＣＴバッファ）５２８を介して宛先ポートへバッファ記憶される。パケットのヘッダは、ＥＰＳＭ２１０のＲＸ　ＦＩＦＯから直接宛先ポートへ送られる。次のステップ１２３２は、ＣＴバッファ５２８へのデータ・パケット部分の受信及び宛先ポートへのパケット部分の転送とを示している。ＣＴ動作の間、次のステップ１２３３は、宛先ポートまたは経路が使用中かあるいは利用できないかどうかを質問する。ステップ１２３３でのこの質問は、データが主アービタ５１２によりＣＴバッファ５２８に受取られる前に行われる。宛先ポートが更なるデータに対してまだ利用可能である間、動作はステップ１２３４へループして、全パケットが宛先ポートへ送られたかどうかを判定し、送られなかったならば、再びステップ１２３２へ戻り更なるデータを伝送する。全パケットがステップ１２３４で判定されたようにＣＴモードで転送された時、このパケットに対する動作が完了する。

【０２２１】
ステップ１２３３で宛先ポートが使用中または利用できないと判定すると、動作はステップ１２３５へ進んで、メモリ２１２にパケットの残りの部分を受取って、中間パケットの暫定ＣＴモードを開始する。中間パケット暫定ＣＴモードでは、パケットの残りの部分がメモリ２１２にバッファ記憶される。パケットが伝送の途中にあったため、メモリ２１２へ送られる残りのパケット・データはこのポートに対する送信パケット・チェーンの初めに置かれて、次のステップ１２３６で示される適切なパケット順序付けを保証する。正規のＣＴ動作モードにおけるように、中間パケット暫定ＣＴモードの間にメモリ２１２へ供給された各データ部分は、受取り後すぐに宛先ポートへの転送のために利用可能である。
【０２２２】
再びステップ１２０２において、動作はソース・アドレスをハッシュするためステップ１２４０へ進む。次いで、ステップ１２４２へ進み、ソース・アドレスがハッシュ・メモリ・セクション９０２で見出されたかどうか、かつパケット内のグループ（ＧＲＯＵＰ）ビットがセットされたかどうかが判定される。ソース・アドレスが見出され、ＧＲＯＵＰビットがセットされなかったならば、動作はステップ１２４４へ進み、ハッシュ・メモリ・セクション９０２のＡＧＥフィールドがＡＧＥ情報で更新される。例えば、ＡＧＥ値はゼロにセットされる。ソースＭＡＣアドレスおよびソース・ポート番号が前のエントリとは対応しないことが判る。このことは、例えば、ネットワークまたはデータ装置が１つのポートから他のポートへ移される場合に生じる。この情報は、ステップ１２４４において比較され、更新される。
【０２２３】
ステップ１２４２において、ソース・アドレスが見出されず、あるいはＧＲＯＵＰビットがセットされたならば、ステップ１２４６へ進み、ＣＰＵ２３０に対して割込みが生成され、ＣＰＵが以降のステップを実行する。次のステップ１２４８において、ＣＰＵ２３０は、メモリ２１２のハッシュ・メモリ・セクション９０２にハッシュ・テーブル・エントリを割り当て、あるいは新たなソース・ポート・アドレスに関するハッシュ・キャッシュ・テーブル６０３のハッシュ・テーブル・エントリ、あるいはリースト・レーセントリ・ユーズド（最低使用頻度：ＬＲＵ）セクションを割付ける。次いで、ステップ１２５０へ進み、割付けられたハッシュ・エントリにおけるソースＭＡＣアドレス、ソース・ポート番号およびＡＧＥ情報等が更新される。
【０２２４】
図５３は、メモリ２１２から１つ以上の宛先ポートへデータを送信するためのネットワーク・スイッチ１０２の一般的動作を示すフロー図である。この送信手順は一般に、以下に述べるように、ＳｎＦモードおよび中間パケット暫定ＣＴ動作モードに適用し、ＢＣパケットに適用する。第１のステップ１２６０は一般に、先に述べた手順に従って、パケット・データがメモリ２１２において待ち行列に入れられる（キューされる）ことを表わす。次のステップ１２６２へ進み、ＭＣＢ４０４がＨＣＢ４０２に対してパケット・データが得られることを示す。中間パケット・データ暫定ＣＴモードにおいては、この表示は、宛先ポートへ送るためにデータが直ちに得られるので、データの最初のデワード（ＤＷＯＲＤ）がメモリ２１２に格納するためＭＣＢ４０４へ送られると、直ちに与えられる。しかし、ＳｎＦモードの場合は、この表示は、全パケットが送信に先立ち格納されるので、データ・パケットに対するデータの最後のデワードがＭＣＢ４０４へ送られた後にのみ与えられる。パケット・データが送信のために利用可能であると、動作はステップ１２６４へ進み、伝送されるパケット・データを受取るために利用可能なバッファ・スペースを宛先ポートが有するかどうかが判定される。ステップ１２６４は一般に、先に述べたように、対応する信号ＢＵＦ＿ＡＶＡＩＬｍ＊に応答するポート１０４、１１０のそれぞれをポーリングするため、ＥＰＳＭ２１０により行われるポーリング手順を表わす。宛先ポートがパケット・データの受取りに利用可能なバッファ・スペースを持つことを示すまで、動作はステップ１２６４に止まる。
【０２２５】

ステップ１２６４において、送信先ポートすなわち宛先ポートがバッファ・スペースを持つと判定すると、動作はステップ１２６６へ進み、ＨＣＢ４０２が宛先ポートに対するデータの転送を要求する。そして、ステップ１２６８において、データのバーストがメモリ２１２から宛先ポートへ送られる。次のステップ１２７０へ進み、メモリ２１２におけるデータの全てが宛先ポートへ送られたかどうかが判定される。送られなかったならば、ステップ１２６４へ戻って、宛先ポートがデータの別の転送のため利用可能なより多くのバッファ・スペースを有することになるまで、待機する。最終的には、ＳｎＦモードおよび暫定ＣＴモードの場合における全データ・パケット、あるいは中間パケット・データ暫定ＣＴモードの場合における残りのパケット・データが、転送され、これはステップ１２７０で判定される。

【０２２６】

動作はステップ１２７２へ進み、パケットがＢＣパケットであるかないか判定される。パケットがＢＣパケットであるならば、動作はステップ１２７４へ進んで、全パケットが全てのアクティブ・ポートへ転送されたかどうか判定する。転送されなかったならば、その時のパケットに対しては、動作が完了する。この手順は、パケットが全てのアクティブ・ポートへ転送されるまで、各ポートに対して再び実行される。各ＢＣパケットに対する各宛先ポートに関して、ステップ１２６４～ステップ１２７０が行われることを、ステップ１２７２と１２７４が表わしていることが判る。このように、全ＢＣデータ・パケットは、送信のため全てのアクティブな宛先ポートへ送られるまで、メモリ２１２に保持される。当該パケットがＢＣパケットでなければ、あるいはステップ１２７４で示されるようにＢＣパケットに対する全てのアクティブ・ポートへ全パケットが送られた後は、動作はステップ１２７６へ進み、ＢＣパケットを保持するメモリ２１２におけるバッファ・スペースが解放される。特に、パケット・データを保持するセクターは、メモリ２１２内のフリー・メモリ・セクタのフリープール・チェーン（ＦＲＥＥＰＯＯＬ　ＣＨＡＩＮ）へ戻される。

【０２２７】

図５４には、ＥＰＳＭ２１０のハッシュ・ルックアップ動作を示すフロー図が示される。図５４のフロー図におけるステップは、ＭＣＢ４０４によって行われる。初期ステップ１３０２は、信号ＨＡＳＨ＿ＲＥＱ＊のアサートにより示されるハッシュ・リクエストを検出する。ＨＣＢ４０２は、新たなパケットとしてのパケットのヘッダを識別し、ソース・アドレスおよび宛先アドレス、およびソース・ポート番号を決定し、ＭＣＢ４０４のハッシュ・コントローラ６０２に対する信号ＨＡＳＨ＿ＤＡ＿ＳＡ［１５：０］を表明する。次に、ＭＣＢ４０４は、ソース・アドレスおよび宛先ＭＡＣアドレス、およびソース・ポート番号を検索して、ハッシング手順を実施し、これにより、パケットに対する適切な動作を決定する。

【０２２８】

ＭＣＢ４０４は一般に、各パケットに関して、ソース・ポート番号、ソース・アドレスおよび宛先ＭＡＣアドレスに基づく、４つの動作の１つを行う。特に、ハッシュ・コントローラ６０２は、信号ＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］を決定し、これにより、パケットを宛先ポートへ送るようにＦＯＲＷＡＲＤ＿ＰＫＴをセットし、パケットをドロップして無視するようにＤＲＯＰ＿ＰＫＴをセットし、宛先ＭＡＣアドレスが新規でありかつ未知であってパケットが他の全てのポートにブロードキャストすなわち送信される場合にＭＩＳＳ＿ＢＣをセットし、又は、パケットがサブセットの関連ポートに複写されて送信されるべき場合にＧＲＯＵＰ＿ＢＣをセットする。ステップ１３０２からステップ１３０４へ進んで、以下の式（１）により、パケットを捨てるかどうかを決定する。

【０２２９】

【数１】

DropPkt:=(SrcState=DIS)or(!FilterHit&SrcState!=FWD)　　（１）

但し、ＳｒｃＳｔａｔｅは、ソース・ポートのスパニング・ツリー状態を識別するものであり、ＨｉｌｔｅｒＨｉｔは、ソースＭＡＣアドレスが予め定めた範囲内にある場合にア

サートされるビットであり、間投詞「！」記号は論理的否定を示し、記号「！＝」は関数「に不等」を示し、記号「：＝」は関数「に等」を示す。各ポートは、ＨＳＢコンフィギュレーション・レジスタ４４８に提供され、学習（ＬＲＮ）、前送（ＦＷＤ）、ブロックド（ＢＬＫ）、リスニング（ＬＳＴ）およびディスエーブル状態（ＤＩＳ）を含む、ＩＥＥＥ仕様８０２．１のスパニング・ツリー関数により決定されるような、５つの状態の１つを持つ。図示した実施例においては、ＢＬＫ状態とＬＳＴ状態は同じものとして処理される。このため、ソース・ポートがディスエーブル状態にされるか、あるいはソースＭＡＣアドレスが予め定めたフィルタ範囲内になく、かつソース・ポートの状態が前送されなければ、パケットはドロップすなわち捨てられる。

【０２３０】

ＤｒｏｐＰｋｔがステップ１３０４で死んであると判定されると、ステップ１３０５へ進み、パケットを無視するかさもなければ捨てるようＨＣＢ４０２に命令するように、信号ＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］が００ｂ＝ＤＲＯＰ＿ＰＫＴに等しくセットされる。ＤｒｏｐＰｋｔが偽であるならば、ステップ１３０６へ進み、ＦｉｌｔｅｒＨｉｔビットが調べられて、ソースＭＡＣアドレスが予め定めた範囲内に含まれるかどうかを判定する。この予め定めた範囲は、ＣＰＵ２３０をソースとし、あるいは宛先とするパケットを識別し、ＣＰＵ２３０へ送られるブリッジ・プロトコル・ユニット（ＢＰＤＵ）を含む。ＦｉｌｔｅｒＨｉｔビットがステップ１３０６で死んであると判定されると、ステップ１３０８へ進んで、宛先ポート（ＤｓｔＰｒｔ）を識別する。パケットがＣＰＵ２３０からであれば（ＳｒｃＰｒｔ＝ＣＰＵ）、宛先ポートは、前の動作においてＣＰＵ２３０によりセットされる値ＦｌｔｒＰｒｔに等しくセットされる（ＤｓｔＰｒｔ：＝ＦｌｔｒＰｒｔ）。さもなければ、パケットはＣＰＵ２３０へ送られる（ＤｓｔＰｒｔ：＝ＰＯＲＴ２８）。次いで、ステップ１３０８からステップ１３１０へ進んで、以下の式（２）に基づいて、パケットを前送する（ＦｗｄＰｋｔ）かどうかを判定する。

【０２３１】

【数２】

FwdPkt:

=(DstPrt!=SrcPrt)&((DstState=FWD)or(SrcPrt=CPU&DstState!=DIS))　　（２）

但し、式（２）において、ＤｓｔＳｔａｔｅは、宛先ポート（ＤｓｔＰｒｔ）のスパニング・ツリー状態であり、「＆」は論理的ＡＮＤ演算を示す。このように、宛先ポートおよびソース・ポートが同じでなく、かつ宛先ポートの状態が前送であるならば、あるいはソース・ポートがＣＰＵ２３０であり宛先ポートの状態がディスエーブル状態でなければ、パケットは宛先ポートへ送られる。ハッシュ・ルックアップがなくとも、宛先ポートは、ＣＰＵ２３０であるか、あるいはＣＰＵ２３０によりＦｌｔｒＰｒｔにより判定されるので、既知である。ＦｌｔｒＰｒｔが偽であれば、ステップ１３０５へ進んでパケットを捨てる。さもなければ、ＦｌｔｒＰｒｔが真の場合、ステップ１３１２へ進み、ＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］信号が１１ｂ＝ＦＯＲＷＡＲＤ＿ＰＫＴに等しくセットされ、パケットが宛先ポートへ送られるべきことを示す。また、ＨＡＳＨ＿ＤＳＴＰＲＴ［４：０］信号はＤｓｔＰｒｔ宛先ポート番号と関連させられる。

【０２３２】

ステップ１３０６において、ソース・アドレスが予め定めた範囲内になく、従ってフィルタされたＭＡＣアドレス外であるならば、動作はステップ１３１４へ進んで、パケットがＢＣパケットであるか否かを示す、受取ったパケット内のＧＲＯＵＰビットを調べる。ＧＲＯＵＰが偽（ＧＲＯＵＰビット＝論理値０）であれば、ステップ１３１６へ進んで、宛先ＭＡＣアドレス（ＤＡ）のハッシュ・ルックアップを行う。ＭＡＣアドレスは、２つの異なるセット（組）のビットをアドレスから取り、そしてこの２つのセットを一緒にビット単位で論理的に組合わせて比較することにより、ハッシュされる。これにより、先に述べたように、対応する１３～１６ビットのハッシュ・アドレスを形成する。ＭＡＣアドレスの任意のビットをハッシング手順の目的のために選定することができる。図５５のフロー図に関連して以下に述べる別個のルーチンまたは関数により、実際のルップアップ手順

が行われる。

【０２３３】

ステップ１３１６におけるルックアップ手順が、ＨＩＴと呼ばれるビットを含む１つ以上の値を必要に応じて返送し、これが宛先アドレスに対するＤＡ＿Ｈｉｔとして、あるいはソース・アドレスに対するＳＡ＿Ｈｉｔとして返送される。ＨＩＴビットは、ハッシュされたアドレスがハッシュ・メモリ・セクション９０２に見出されたかどうかを判定する。ステップ１３１６からステップ１３１８へ進み、ここでＤＡ＿Ｈｉｔ値が調べられてアドレスが見出されたか否かを判定する。このアドレスは、宛先ＭＡＣアドレスと対応する装置がパケットを前に送信した場合に、メモリ２１２中に見出される。ＤＡ＿Ｈｉｔが真ならば、動作はステップ１３１０へ進んで、先に述べたようにパケットを前送するかどうかを判定する。ハッシュ・アドレスが見出されずＤＡ＿Ｈｉｔが偽であるならば、ステップ１３２０へ進み、ここでＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］信号が１０ｂ＝ＭＩＳＳ＿ＢＣにセットされて、新たなＭＡＣアドレスを示す。宛先装置と関連するポート番号が未知であるので、パケットは他の全てのアクティブ・ポート（ＶＬＡＮにより定性化されるポート、および他の論理的ポート）へ、ブロードキャストされて、パケットが適切な宛先装置へ送られることを保証する。最終的には、宛先装置は、ソース・アドレスと同じＭＡＣアドレスを含む新たなパケットに答する。この時、ネットワーク・スイッチ１０２は、ＭＡＣアドレスをポートとポート番号とに関連付けて、これに対応してハッシュ・メモリ・セクション９０２を更新する。ステップ１３１４において、ＧＲＯＵＰビットが真（即ち、論理値１）であるならば、動作はステップ１３２２へ進み、ここでＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］信号が０１ｂ＝ＧＲＯＵＰ＿ＢＣにセットされ、パケットが他の全てのポート、あるいはＶＬＡＮ関数により指定されるポートのグループへブロードキャストされることを示す。

【０２３４】

ステップ１３０５、１３１２、１３２０あるいは１３２２のいずれかから、ステップ１３２４へ進んで、ＳｒｃＬｏｏｋＵｐ値を調べることにより、ソースＭＡＣアドレスについてハッシュ・メモリ・セクション９０２を検索するかどうかを判定する。ＳｒｃＬｏｏｋＵｐ値は、以下の式（３）に従って決定される。

【０２３５】

【数３】

SrcLookUp:=(SrcState=(LRNorFWD))&SrcPrt!=CPU　　（３）

式（３）は、ソース・ポートが学習モードあるいは前送モードにあり、かつ該ソース・ポートがＣＰＵ２３０でない場合は、ＭＡＣソース・アドレスが探索されることを示している。ＳｒｃＬｏｏｋＵｐがアサートされて真であるとステップ１３２４で判定されると、動作はステップ１３２６へ進み、２つの値ＶＬＡＮおよびＳｅｃｕｒｅＰｏｒｔが調べられる。ＶＬＡＮモードのどれかがイネーブル状態にされるならばＶＬＡＮビットは真であるが、それ以外は偽である。ソース・ポートが確実であればＳｅｃｕｒｅＰｏｒｔが真である、すなわちアサートされ、ここでは新たなアドレスはハッシュ・メモリ・セクション９０２へは付加されず、未知のソース・アドレスからのパケットが捨てられる。ＶＬＡＮが真でなくポートが確実でなければ、動作はステップ１３２８へ進み、ＨＡＳＨ＿ＤＯＮＥ＊信号がアサートされて、一時的にアサート状態を保つ。この時、信号ＨＡＳＨ＿ＳＴＡＴＵＳおよびＨＡＳＨ＿ＤＳＴＰＲＴが、ＨＣＢ４０２により捕捉される。

【０２３６】

ステップ１３２６において、ＶＬＡＮが真であるか、あるいはＳｅｃｕｒｅＰｏｒｔが真であると判定された場合、あるいはステップ１３２８が行われた後は、ソース・アドレス・ルックアップの後まで、ＨＡＳＨ＿ＤＯＮＥ＊信号のアサートが遅延される。次いでステップ１３３０へ進み、宛先ＭＡＣアドレスに関して先に述べたと類似の方法で、ハッシュ・ルックアップがソースＭＡＣアドレス（ＳＡ）に関して行われる。ステップ１３３０において、対応する装置に関するハッシュ・アドレスが見出されるならば、値ＳＡ＿Ｈｉｔが真に戻される。ステップ１３３０からステップ１３３２へ進み、ここで値Ｓｒｃ＿Ｈ

ｉｔが調べられる。Ｓｒｃ＿Ｈｉｔは、以下の式（４）によりＳＡ＿Ｈｉｔに関連付けられる。

【０２３７】

【数４】

Src_Hit:=SA_Hit&(HshPrt=SrcPort)　　（４）

式（４）において、ソース・ヒットが生じ（ＳＡ＿Ｈｉｔが真）、かつハッシュ・メモリ・セクション９０２におけるエントリで見出されたポート番号がパケットが受取られた実際のソース・ポート番号と等しければ、Ｓｒｃ＿Ｈｉｔは真である。格納されたソース・ポート番号が実際のソース・ポート番号と等しくなければ、以下に述べるように、装置は別のポートへ移されたことであり、ハッシュ・メモリ・セクション９０２はＣＰＵ２３０により更新される。Ｓｒｃ＿Ｈｉｔが真であれば、動作はステップ１３３４へ進み、ＶＬＡＮが偽ならば、ＨＡＳＨ＿ＤＯＮＥ＊信号がアサートされる。次いで、動作はステップ１３３６へ進み、装置のＡＧＥ番号がゼロであるか判定される。ＡＧＥがゼロに等しくなければ、ＡＧＥ番号はステップ１３３８においてゼロに等しくセットされる。ステップ１３３６でＡＧＥ番号がゼロであると判定された場合、あるいはステップ１３３８においてゼロにセットされた後、ステップ１３４０へ進み、ＶＬＡＮビットが再び調べられる。ＶＬＡＮが真であれば、ステップ１３４２へ進み、ここでハッシュＶＬＡＮルーチンすなわち手順が調べられて、関連するポートをハッシュ・テーブル・エントリ９１０における対応するＶＬＡＮビット・マップ値から決定されたものとして、識別する。ステップ１３４０でＶＬＡＮが真でないと判定すると、動作はステップ１３４４へ進み、既にアサートされていない場合は、ＨＡＳＨ＿ＤＯＮＥ＊信号がある期間だけアサートすなわちパルスが発生され、次に否定される。ステップ１３４４の終了により、この手順の動作が完了する。ＨＡＳＨ＿ＤＯＮＥ＊信号の否定信号により、ＨＣＢ４０２のハッシュ・ルックアップを終了する。

【０２３８】

ステップ１３３２において、Ｓｒｃ＿Ｈｉｔが偽ならば、ステップ１３５０へ進み、ＬｅａｒｎＤｉｓＰｒｔ値を調べることにより、ソース・ポートがディスエーブル状態にされたことを学習しているかどうか判定される。もし学習していなければ、ステップ１３５２へ進み、パケットの新たな情報が適切なレジスタへロードされ、ＣＰＵ２３０が割込みされる。ＣＰＵ２３０は、これに応答して、ハッシュ・メモリ・セクション９０２を新たなハッシュ・テーブル・エントリ９１０で更新する。ソース・ポートがステップ１３５０でディスエーブル状態にされたことを学習していると判定した場合、あるいはハッシュ・メモリ・セクション９０２がステップ１３５２で更新された後は、ステップ１３５４へ進んで、ＳｅｃｕｒｅＰｏｒｔビットを調べる。ＳｅｃｕｒｅＰｏｒｔが真ならば、動作はステップ１３５６へ進み、ここでＨＡＳＨ＿ＳＴＡＴＵＳ［１：０］信号が００ｂ＝ＤＲＯＰ＿ＰＫＴへ変更される。この場合、アドレスが新しく、かつ新アドレスが保全ポートでは許容されないので、新たなパケットがドロップされる。また、必要に応じて、セキュリティ（保全）違反割込みがＣＰＵ２３０に対してアサートされて、セキュリティ違反に応答して適切な処置を行う。ステップ１３５６からステップ１３４４へ進む。再びステップ１３５４において、ＳｅｃｕｒｅＰｏｒｔビットが非保全ポートを示す偽であるならば、ステップ１３４０へ進む。ステップ１３２４において、ＳｒｃＬｏｏｋＵｐが偽であれば、直接ステップ１３４４へ進む。

【０２３９】

図５５には、ハッシュ・メモリ・セクション９０２におけるハッシュ・テーブル・エントリ９１０の全てを探索するためのハッシュ・ルックアップ手順を示すフロー図が示されている。最初のステップ１４０２において、アドレス値Ａがステップ１３１６または１３３０から送られる受取られたハッシュ・アドレスに等しくセットされる。動作はステップ１４０４へ進み、ここで受取られたハッシュ・アドレスと関連する主ハッシュ・エントリ・セクション９０６内のハッシュ・テーブル・エントリ９１０が読出される。動作はステップ１４０６へ進み、ＶＡＬＩＤＥＮＴＲＹ（エントリ有効）ビットが読出され、新たなパ

ケットのＭＡＣアドレスが格納されたＭＡＣアドレスと比較される。エントリが有効であり正確な整合がＭＡＣアドレス間に生じるならば、動作はステップ１４０８へ進み、ＨＩＴビットが真にセットされてハッシュ・ビットを示し、動作は呼出し手順即ちルーチンへ戻る。エントリは有効でないか、あるいはアドレスの整合が起きなかったならば、ステップ１４１０へ進み、ここでＶＡＬＩＤＥＮＴＲＹビットと、エントリのＥＯＣ（チェーン終り）値が調べられる。エントリが有効でないか、あるいはＥＯＣに到達しなければ、動作はＨＩＴビットを偽として戻る。さもなければ、ハッシュ・アドレスが、ステップ１４１２においてハッシュ・エントリ内のリンク・アドレス（バイトＦ：Ｃ）に等しくセットされ、ステップ１４０４へ戻って、チェーン化されたハッシュ・エントリ・セクション９０８内の次のチェーン化エントリを試みる。ＭＡＣアドレス整合による有効なエントリが見出されるまでか、あるいはＥＯＣ値に遭遇するまで、動作はステップ１４０４、１４０６、１４１０および１４１２間をループする。

【０２４０】

以下のテーブル（１）は、本発明により実現された特定の実施形態におけるＣＰＵ２３０の入出力（Ｉ／Ｏ）スペース・レジスタを示している。テーブル（１）は、単に例示的に示したものであり、また、該例においては、レジスタが特殊な実施例中か又はそれ以外で実現されるか、若しくは同様なレジスタが異なる呼称で呼ばれている。

【０２４１】

【表１】

テーブル1： CPU230 I／Oスペース・レジスタ

オフセット(h)	マスク	シャドウ化	アクセス (R/W)	Reg_name/Bit_name	説明
0	PCB 406		CPU: R PCB: W MCB: ---- HCB: ----	割り込みソース1 ビット0: MCB_INT 1: MEM_RDY 2: ABORT_PKT 3: STAT_RDY 4-31: リザーブ	(1)
4	PCB 406		CPU: R/W PCB: R MCB: ---- HCB: ----	割り込みマスク1 ビット0: MCB_INT 1: MEM_RDY 2: ABORT_PKT 3: STAT_RDY 4: HASH_MISS 5-31: リザーブ	(2)
8	PCB 406		CPU: R/W PCB: R/W MCB: ---- HCB: ----	パケット情報 RdPkt ビット0: SOP 1: EOP 2-15: リザーブ 16-23: 長さ（EOPに対して） 24-31: リザーブ	(3)
C	PCB 406		CPU: R/W PCB: R/W MCB: ---- HCB: ----	パケット情報 WrPkt ビット0: SOP 1: EOP 2-5: BE（SOPに対して） 6-15: リザーブされる 16-23: 長さ 24-31: リザーブ	(4)

【０２４２】
【表２】

10	PCB 406		CPU: R PCB: R/W MCB: ---- HCB: ----	SIMM 存在検出 ビット 0-3: simm1_pd[0..3] 4-7: simm2_pd[0..3] 8-11: simm3_pd[0..3] 12-15: simm4_pd[0..3] 16-31: リザーブ	(5)
14	PCB 406		CPU: R/W PCB: W MCB: ---- HCB: ----	ポーリング・ソース (1 & 2) ビット 0: MCB_INT 1: MEM_RDY 2: PKT_AVAIL 3: BUF_AVAIL 4: ABORT_PKT 5: STAT_RDY 6: HASH_MISS 7-31: リザーブ	(6)
18	PCB 406		CPU: R PCB: W MCB: ---- HCB: ----	割り込みソース 2 ビット 0: PKT_AVAIL 1: BUF_AVAIL 2-31: リザーブ	(7)
1c	PCB 406		CPU: R/W PCB: R MCB: ---- HCB: ----	割り込みマスク 2 ビット 0: PKT_AVAIL 1: BUF_AVAIL 2-31: リザーブ	(8)

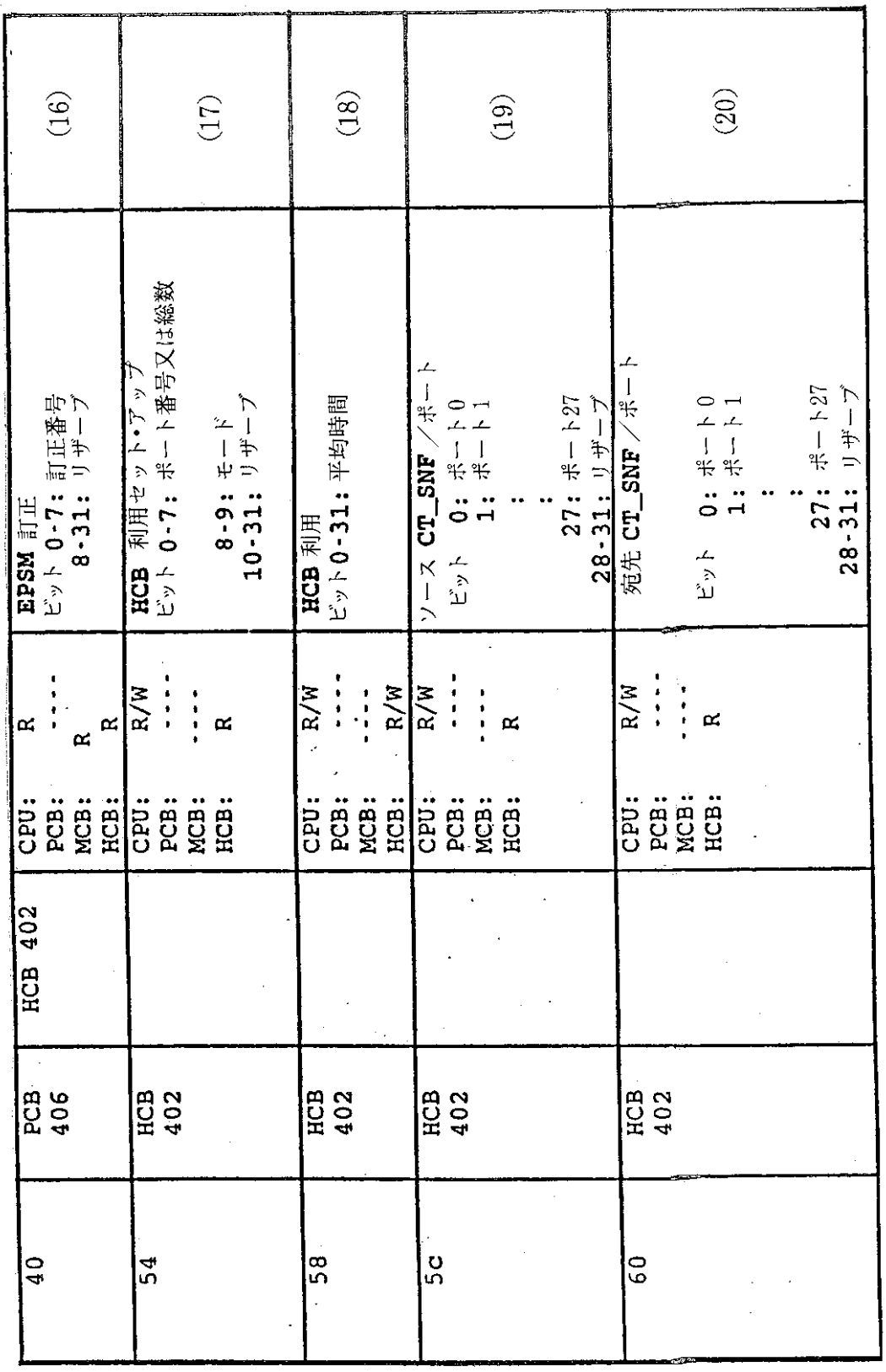
【０２４３】
【表３】

20	PCB 406		CPU: R/W PCB: R/W MCB: ---- HCB: ---	QCスタティスティック情報 ビット 0-1: ポート番号 2-4: QC番号 5-9: レジスタ番号 10-14: レジスタの数 15-19: レジスタの最大数 20-31: リザーブ	(9)
24	PCB 406		CPU: R PCB: R/W MCB: ---- HCB: ----	総パケット情報 ビット0-15: パケット長 16-23: ソース・ポート 24-31: 宛先ポート	(10)
28	PCB 406		CPU: WO PCB: R/W MCB: ---- HCB: ----	フラッシュFIFO	(11)
30	PCB 406	MCB 404 HCB 402	CPU: R/W PCB: R MCB: R HCB: R	EPSM ステップアップ ビット 0: TPI インストール 1: EXP インストール 2: マスタ・スイッチ・イネーブル 3-4: QcXferSize[1:0] 5-6: TPIXferSize[1:0] 7: AI_FCS 8: DramWrDis 9: SramWrDis 10-12: Epsm Addr Dcd 13: ClkSel 14-21: CPU ポート番号 22-31: リザーブ	(12)

【０２４４】
【表４】

34	PCB 406	HCB 402	CPU: R/W PCB: ---- MCB: R HCB: R	ポート・スピード ビット 0: ポート0スピード 1: ポート1スピード : : 27: ポート27スピード 28-31: リザーブ	(13)
38	PCB 406	MCB 404 HCB 402	CPU: R PCB: ---- MCB: R HCB: R	ポート・タイプ ビット 0: ポート0タイプ 1: ポート1タイプ : : 27: ポート27タイプ 28-31: リザーブ	(14)
3c	PCB 406	MCB 404	CPU: R/W PCB: R MCB: R HCB: ----	MEM リクエスト ビット 0-23: Mem アドレス 24: メモリ選択 25: 転送サイズ 26-29: バイト・イネーブル 30: RW 31: ロック済ページ・ヒット	(15)

【０２４５】
【表５】



40	PCB 406	HCB 402	CPU: R ---- PCB: ---- MCB: R HCB: R	**EPSM** 訂正 ビット **0-7:** 訂正番号 **8-31:** リザーブ	(16)
54	HCB 402		CPU: R/W ---- PCB: ---- MCB: ---- HCB: R	**HCB** 利用セット・アップ ビット **0-7:** ポート番号又は総数 **8-9:** モード **10-31:** リザーブ	(17)
58	HCB 402		CPU: R/W ---- PCB: ---- MCB: ---- HCB: R/W	**HCB** 利用 ビット **0-31:** 平均時間	(18)
5c	HCB 402		CPU: R/W ---- PCB: ---- MCB: ---- HCB: R	ソース **CT_SNF** ／ポート ビット **0:** ポート0 **1:** ポート1 ： ： **27:** ポート27 **28-31:** リザーブ	(19)
60	HCB 402		CPU: R/W ---- PCB: ---- MCB: ---- HCB: R	宛先 **CT_SNF** ／ポート ビット **0:** ポート0 **1:** ポート1 ： ： **27:** ポート27 **28-31:** リザーブ	(20)

【０２４６】
【表６】

64	HCB 402 (High 2 bits of each xfersz)		CPU: R/W PCB: ---- MCB: ---- HCB: R	XferSize／ポート ビット 0-3: ポート 0 xfersize 4-7: ポート 1 xfersize 8-11: ポート 2 xfersize 12-15: ポート 3 xfersize 16-19: ポート 4 xfersize 20-23: ポート 5 xfersize 24-27: ポート 6 xfersize 28-31: ポート 7 xfersize	(21)
68	HCB 402 (High 2 bits of each xfersz)		CPU: R/W PCB: ---- MCB: ---- HCB: R	**Xfersize／ポート** ビット 0-3: ポート 8 xfersize 4-7: ポート 9 xfersize 8-11: ポート10 xfersize 12-15: ポート11 xfersize 16-19: ポート 12 xfersize 20-23: ポート 13 xfersize 24-27: ポート 14 xfersize 28-31: ポート 15 xfersize	(22)

【０２４７】
【表７】

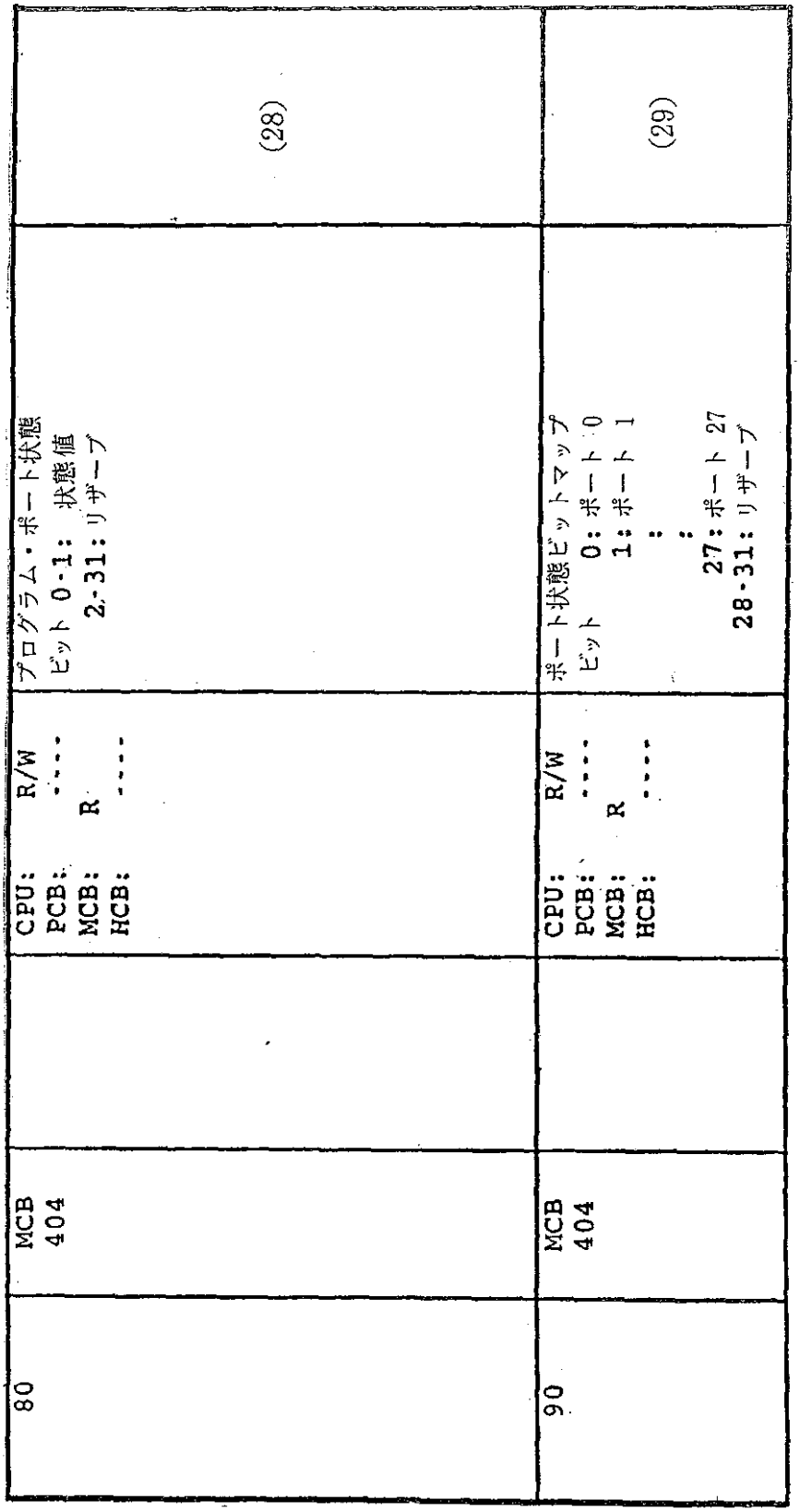
リザーブ

6c	HCB 402 (High 2 bits of each xfersz)		CPU: R/W PCB: ---- MCB: ---- HCB: R	**XferSize**／ポート ビット 0-3:ポート16 xfersize 4-7:ポート17 xfersize 8-11:ポート18 xfersize 12-15:ポート19 xfersize 16-19:ポート20 xfersize 20-23:ポート21 xfersize 24-27:ポート22 xfersize 28-31:ポート23 xfersize	(23)
70	HCB 402 (High 2 bits of each xfersz)		CPU: R/W PCB: ---- MCB: ---- HCB: R	**XferSize**／ポート ビット 0-3:ポート24 xfersize 4-7:ポート25 xfersize 8-11:ポート26 xfersize 12-15:ポート27 xfersize 16-19:ポート28 xfersize 20-31:リザーブ	(24)

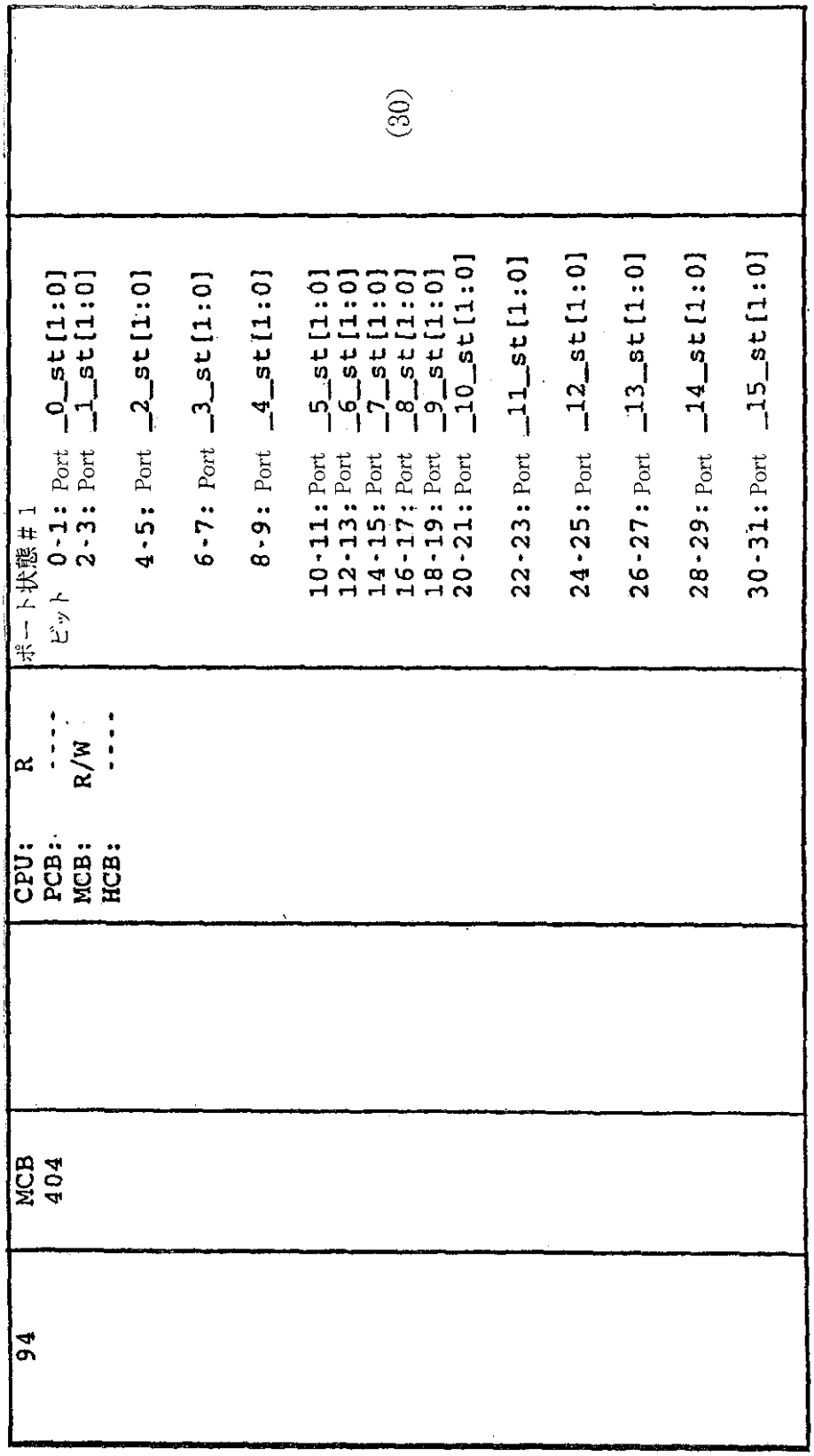
【０２４８】
【表８】

74	HCB 402		CPU: R/W PCB: ---- MCB: ---- HCB: R	**Arb_Mode** ビット 0-1: モード値 2-31: リザーブ	(25)
78	HCB 402		CPU: R/W PCB: ---- MCB: ---- HCB: R	**HCB Misc** コントロール ビット 0: イネーブル CT FIFO 1: イネーブル **Rd Extra** WS 2: イネーブル **CC Rd/Wr** Qc 3: イネーブル **CC Rd/Wr** Qe 4: 初期的イネーブル AD 5-31: リザーブ	(26)
7c	HCB 402		CPU: R/W PCB: ---- MCB: ---- HCB: R	ポート・シャットダウン ビット 0-27: ビット マップ	(27)

【０２４９】
【表９】


(28)
(29)
プログラム・ポート状態
ビット 0-1: 状態値
2-31: リザーブ
CPU: R/W ----
PCB: ----
MCB: R ----
HCB: ----
ポート状態ビットマップ
ビット 0: ポート 0
1: ポート 1
:
27: ポート 27
28-31: リザーブ
CPU: R/W ----
PCB: ----
MCB: R ----
HCB: ----
MCB 404
MCB 404
80
90

【０２５０】
【表１０】


94
MCB
404
CPU: R
PCB: ----
MCB: R/W
HCB: ----
ポート状態#1
ビット 0-1: Port _0_st[1:0]
2-3: Port _1_st[1:0]
4-5: Port _2_st[1:0]
6-7: Port _3_st[1:0]
8-9: Port _4_st[1:0]
10-11: Port _5_st[1:0]
12-13: Port _6_st[1:0]
14-15: Port _7_st[1:0]
16-17: Port _8_st[1:0]
18-19: Port _9_st[1:0]
20-21: Port _10_st[1:0]
22-23: Port _11_st[1:0]
24-25: Port _12_st[1:0]
26-27: Port _13_st[1:0]
28-29: Port _14_st[1:0]
30-31: Port _15_st[1:0]
(30)

【０２５１】
【表１１】

98	MCB 404		CPU: R ---- PCB: ---- MCB: R/W HCB: ----	ポート状態＃2 ビット 0-1: Port _16_st[1:0] 2-3: Port _17_st[1:0] 4-5: Port _18_st[1:0] 6-7: Port _19_st[1:0] 8-9: Port _20_st[1:0] 10-11: Port _21_st[1:0] 12-13: Port _22_st[1:0] 14-15: Port _23_st[1:0] 16-17: Port _24_st[1:0] 18-19: Port _25_st[1:0] 20-21: Port _26_st[1:0] 22-23: Port _27_st[1:0] 24-31: リザーブ	(31)
9c	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	宛先ミスブロードキャスト ビット 0-28: DestMissBCビットマップ 29-31: リザーブ	(32)

【０２５２】
【表１２】

a8	MCB 404		CPU: R/W PCB: ---- MCB: R/W HCB: ----	メモリ・バス・モニタ・コントロール ビット 0-14: モニタ・モード 15: モニタ選択 16-23: モニタ・ポート選択 24-27: フィルタ時間スケール 28: モニタ・クリア 29: カウント／フィルタ・モード 30: Backpress イネーブル 31: アラーム	(33)
ac	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	メモリ・バス・モニタ・スレッショルド ビット 0-7: アラーム・セット・スレッショルド 8-15: アラーム・クリア・スレッショルド 16-19: リザーブ 20-31: ピーク BW	(34)
b0	MCB 404		CPU: R PCB: ---- MCB: R/W HCB: ----	メモリ・バス利用 ビット 0-31: パーセント利用	(35)
b8	MCB 404		CPU: R PCB: ---- MCB: R/W HCB: ----	メモリによってドロップしたパケット ビット 0-31: パケットの数	(36)

【０２５３】
【表１３】

bc	MCB 404		CPU: R PCB: ---- MCB: R/W HCB: ----	BCによってドロップしたパケット ビット0-31:パケットの数	(37)
c0	MCB 404		CPU: R/W PCB: ---- MCB: R ----' HCB:	ハッシュ・テーブルの定義 ビット0-14:アドレス [16:2] 15-23:アドレス [25:17] 24-25:テーブル・サイズ 26:ロック・ハッシュ・サイクル 27: Vlan グループ BC 28: Vlan ミス BC 29: Vlanユニキャスト 30-31:リザーブ	(38)
c4	MCB 404		CPU: R ---- PCB: ---- MCB: R/W HCB: ----	Rx セクタ・カウント ビット 0-28:ビットマップ 29-31:リザーブ	(39)
c8	MCB 404		CPU: R ---- PCB: ---- MCB: R/W HCB: ----	Tx パケット・カウント ビット 0-28:ビットマップ 29-31:リザーブ	(40)
cc	MCB 404		CPU: R ---- PCB: ---- MCB: R/W HCB: ----	ハッシュ・アドレス・ロー ビット0-31:バイト 0-3	(41)

【０２５４】
【表１４】

d0	MCB 404		CPU: R PCB: ---- MCB: R/W HCB: ----	ハッシュ・アドレス・ハイ ビット 0-15: バイト 4-5 16-23:ソース・ポート 24:ポート・ミス 25-31:リザーブ	(42)
d4	MCB 404		CPU: R PCB: ---- MCB: R/W HCB: ----	受信メモリ・セクタによりドロップしたパケット ビット 0-31:パケットの数	(43)
d8	MCB 404		CPU: R PCB: ---- MCB: R/W HCB: ----	送信メモリ・セクタによりドロップしたパケット ビット 0-31:パケットの数	(44)
dc	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	受信オーバーフローによりドロップしたパケット ビット 0-28:ポート・ビットマップ 29-31:リザーブ	(45)
e0	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	送信オーバーフローによりドロップしたパケット ビット 0-28:ポート・ビットマップ 29-31:リザーブ	(46)
e4	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	学習ディスエーブル・ポート ビット 0-27:学習ディスエーブル・ポート・ビットマップ 28-31:リザーブ	(47)
e8	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	確実ポート ビット 0-27:確実ポート・ビットマップ 28-31:リザーブ	(48)

【０２５５】
【表１５】

ec	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	セキュリティ・バイオレーション状態 ビット 0-31:カウント	(49)
f0	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	セキュリティ・バイオレーション ビット 0-27:ポート・ビットマップ 28-31:リザーブ	(50)
f4	MCB 404	Rasenx	CPU: R/W PCB: ---- MCB: R/W HCB: ----	メモリ・コントロール ビット 0-1:メモリ・タイプ 2:メモリ・スピード 3: EDO テスト・モード 4: Dbl リンク・モード 5: DisRcPgHits 6: DisTxPGHits 7-31:リザーブ	(51)
f8	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	RAS 選択 ビット 0-31: Rasenx [1:0]	(52)
fc	MCB 404		CPU: R/W PCB: R MCB: ---- HCB: ----	リフレッシュ・カウンタ ビット 0-9:カウント 10-31:リザーブ	(53)

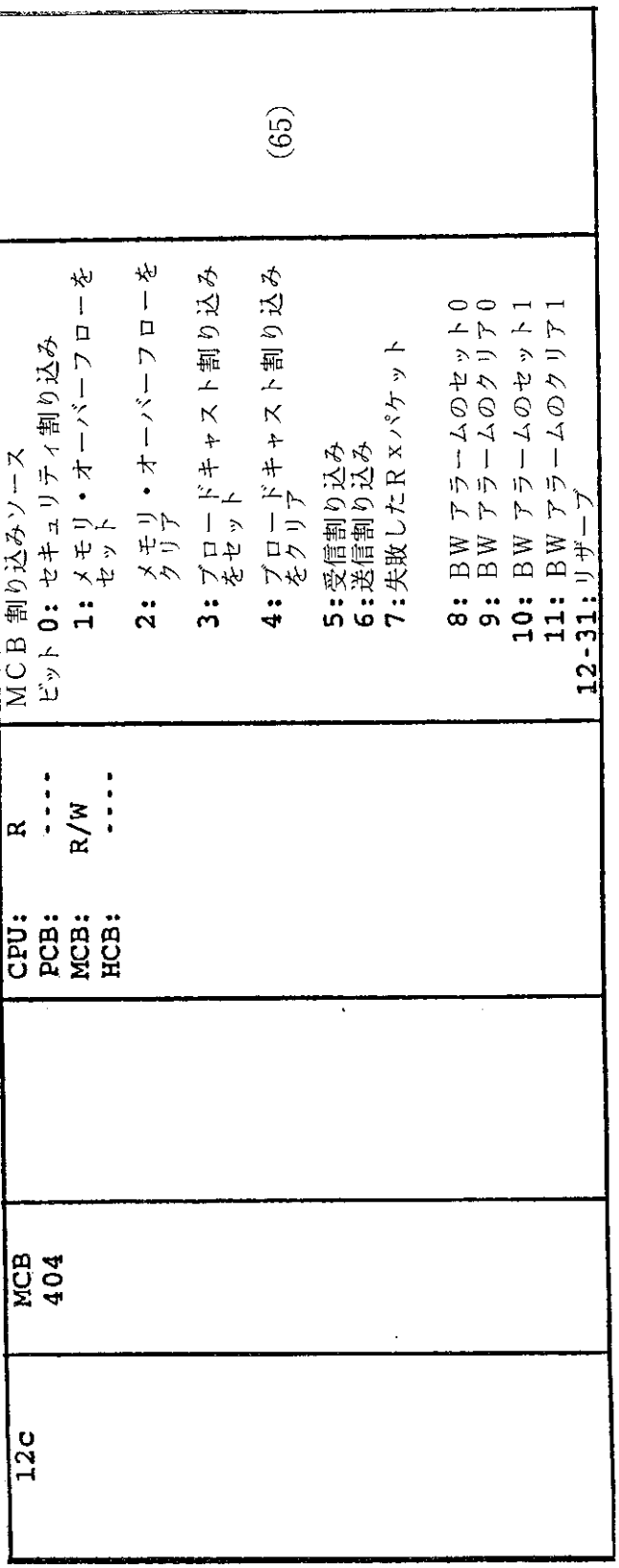
【０２５６】
【表１６】

100	MCB 404 (bit 4-7)	CPU: R/W PCB: ---- MCB: R HCB: ----	フィルタ・コントロール ビット 0-3:アドレス・イネーブル [3:0] 4-7: マスク・イネーブル [3:0] 8-31: リザーブ	(54)	
104	MCB 404	CPU: R/W PCB: ---- MCB: R HCB: ----	マスク・アドレス・フィルタ・ロー ビット0-31:バイト 0－3	(55)	
108	MCB 404	CPU: R/W PCB: ---- MCB: R HCB: ----	マスク・アドレス・フィルタ・ハイ ビット0-15:バイト 4－5 16-31:リザーブ	(56)	
10c	MCB 404	CPU: R/W PCB: ---- MCB: R HCB: ----	アドレス・フィルタ0・ロー ビット0-31:バイト 0－3	(57)	
110	MCB 404	CPU: R/W PCB: ---- MCB: R HCB: ----	アドレス・フィルタ1・ハイ ビット0-15:バイト 4－5 16-23:宛先ポート 24-31:フィルタ・マスク0	(58)	
114	MCB 404	CPU: R/W PCB: ---- MCB: R HCB: ----	アドレス・フィルタ1・ロー ビット0-31:バイト 0－3	(59)	
118	MCB 404	CPU: R/W PCB: ---- MCB: R HCB: ----	アドレス・フィルタ1・ハイ ビット0-15:バイト 4－5 16-23:宛先ポート 24-31:フィルタ・マスク1	(60)	

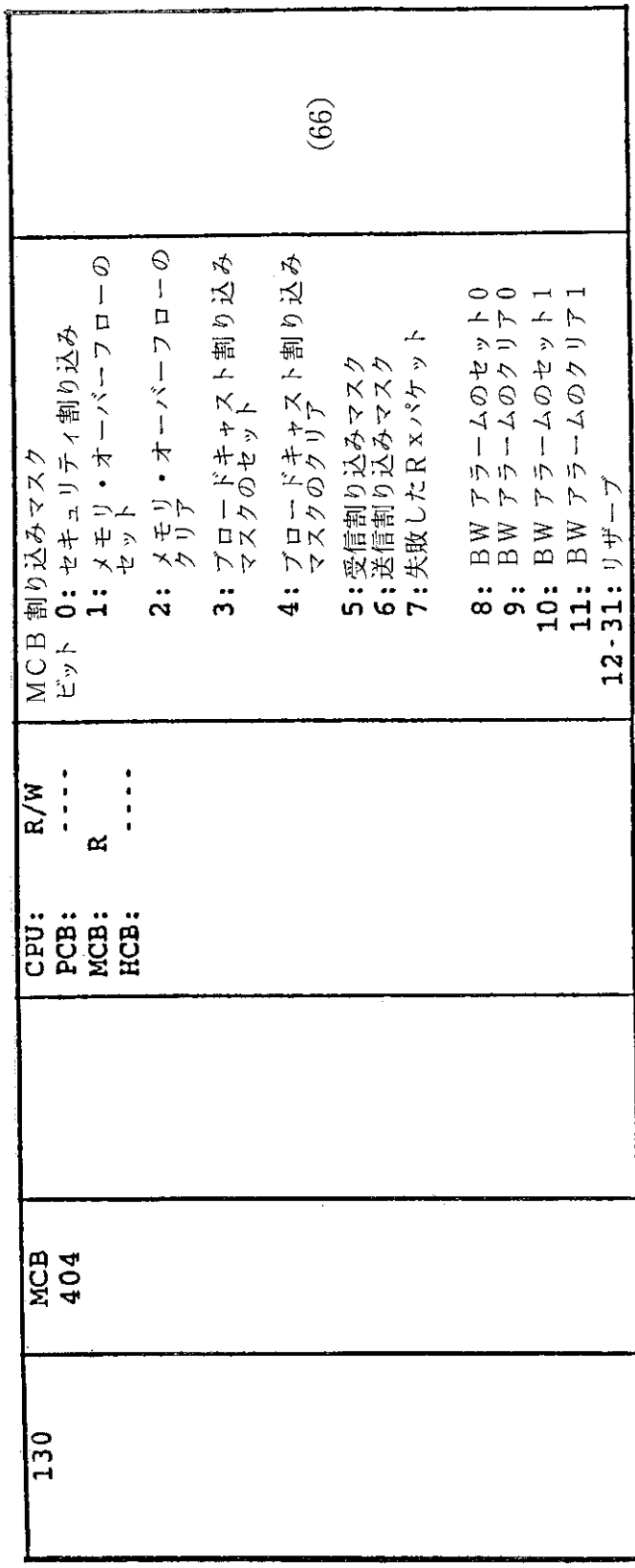
【０２５７】
【表１７】

11c	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	アドレス・フィルタ2・ロー ビット0-31:バイト 0－3	(61)
120	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	アドレス・フィルタ2・ハイ ビット0-15:バイト 4－5 16-23:宛先ポート 24-31:フィルタ・マスク2	(62)
124	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	アドレス・フィルタ3・ロー ビット0-31:バイト 0－3	(63)
128	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	アドレス・フィルタ・ハイ ビット0-15:バイト 4－5 16-23:宛先ポート 24-31:フィルタ・マスク3	(64)

【０２５８】
【表１８】


12c
MCB
404
CPU: R
PCB: ----
MCB: R/W
HCB: ----
MCB 割り込みソース
ビット 0: セキュリティ割り込み
1: メモリ・オーバーフローをセット
2: メモリ・オーバーフローをクリア
3: ブロードキャスト割り込みをセット
4: ブロードキャスト割り込みをクリア
5: 受信割り込み
6: 送信割り込み
7: 失敗したRxパケット
8: BW アラームのセット0
9: BW アラームのクリア0
10: BW アラームのセット1
11: BW アラームのクリア1
12-31: リザーブ
(65)

【０２５９】
【表１９】


130
MCB
404
CPU:
PCB:
MCB:
HCB:
R/W

R

MCB 割り込みマスク
ビット 0: セキュリティ割り込み
1: メモリ・オーバーフローのセット
2: メモリ・オーバーフローのクリア
3: ブロードキャスト割り込みマスクのセット
4: ブロードキャスト割り込みマスクのクリア
5: 受信割り込みマスク
6: 送信割り込みマスク
7: 失敗したRxパケット
8: BW アラームのセット0
9: BW アラームのクリア0
10: BW アラームのセット1
11: BW アラームのクリア1
12-31: リザーブ
(66)

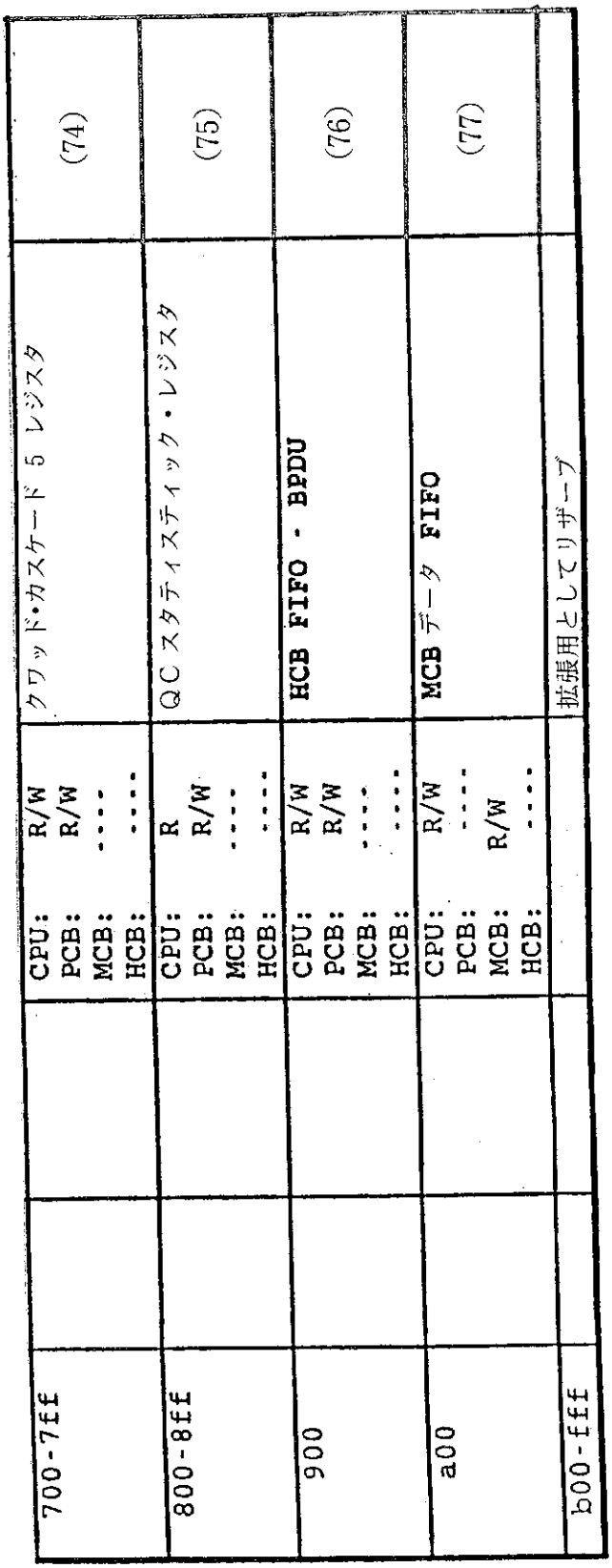
【０２６０】
【表２０】

134	MCB 404		CPU: R/W ---- PCB: ---- MCB: R/W HCB: ----	MCBポーリング・ソース ビット 0:セキュリティ割り込み 1:メモリ・オーバーフローのセット 2:メモリ・オーバーフローのクリア 3:ブロードキャスト・ポーリング・ソースのセット 4:ブロードキャスト・ポーリング・ソースのクリア 5:受信ポーリング・ソース 6:送信ポーリング・ソース 7:失敗したRxパケット 8:BWアラームのセット0 9:BWアラームのクリア0 10:BWアラームのセット1 11:BWアラームのクリア1 12-31:リザーブ	(67)
138	MCB 404		CPU: R/W ---- PCB: ---- MCB: R HCB: ----	バックプレッシャ・イネーブル ビット0-23:リザーブ 24-27:ポート・ビットマップ 28-31:リザーブ	
13c	MCB 404		CPU: R/W ---- PCB: ---- MCB: R HCB: ----	結合ポートのセット0 ビット0-27:ポート・ビットマップ 28-31:リザーブ	
140	MCB 404		CPU: R/W ---- PCB: ---- MCB: R HCB: ----	結合ポートのセット1 ビット0-27:ポート・ビットマップ 28-31:リザーブ	

【０２６１】
【表２１】

144	MCB 404		CPU: R/W PCB: ---- MCB: R HCB: ----	デフォルトVlanビットマップ ビット0-28:ビットマップ	
148	MCB 404		CPU: R/W PCB: ---- MCB: ---- HCB: R	契約ポート ビット 0-7: ポート 8-15: Ｒｘモニタ・ポート番号 16-23: Ｔｘモニタ・ポート番号 24-31: リザーブ	(68)
200-2ff			CPU: R/W PCB: R/W MCB: ---- HCB: ----	クワッド・カスケード０レジスタ	(69)
300-3ff			CPU: R/W PCB: R/W MCB: ---- HCB: ----	クワッド・カスケード１レジスタ	(70)
400-4ff			CPU: R/W PCB: R/W MCB: ---- HCB: ----	クワッド・カスケード２レジスタ	(71)
500-5ff			CPU: R/W PCB: R/W MCB: ---- HCB: ----	クワッド・カスケード３レジスタ	(72)
600-6ff			CPU: R/W PCB: R/W MCB: ---- HCB: ----	クワッド・カスケード４レジスタ	(73)

【０２６２】
【表２２】



700-7ff			CPU: R/W PCB: R/W MCB: ---- HCB: ----	クワッド・カスケード５レジスタ	(74)	
800-8ff			CPU: R PCB: R/W MCB: ---- HCB: ----	ＱＣスタティスティック・レジスタ	(75)	
900			CPU: R/W PCB: R/W MCB: ---- HCB: ----	HCB FIFO - BPDU	(76)	
a00			CPU: R/W PCB: ---- MCB: R/W HCB: ----	MCB データ FIFO	(77)	
b00-fff				拡張用としてリザーブ		

【０２６３】
テーブル１の（１）～（７７）の説明
（１）ＣＰＵ２３０に対する任意の割り込み（１又は複数）のソース。これらの割り込みはＣＰＵ２３０が割り込みをアクノレッジ（確認）するときに該ＣＰＵ２３０によってクリアされる
（２）ＣＰＵ２３０に対するマスクされるべき割り込み
（３）このレジスタはＣＰＵ２３０によって書き込まれる

（４）このレジスタはＥＰＳＭ２１０によって書き込まれる
（５）このレジスタはシフト・レジスタ・インターフェースを通じてＳＩＭＭ上の情報を含む
（６）ＣＰＵ２３０に対するマスクされている任意の割り込み（１又は複数）のソース
（７）ＣＰＵ２３０に対する任意の割り込み（１又は複数）のソース。これらの割り込みはＣＰＵ２３０が割り込みをアクノレッジ（確認）するときに該ＣＰＵ２３０によってクリアされる
（８）ＣＰＵ２３０に対するマスクされるべき割り込み
（９）このレジスタに書き込むＣＰＵ２３０は、適当なポートの統計リード（読み出し）を発行するようにＱＣインターフェースに知らせる
（10）このレジスタはＥＰＳＭ２１０によって書き込まれる
（11）このレジスタは、書き込まりれたときに、ＦＩＦＯ内容をフラッシュ（消去）し、ＥＯＰを受信するまでフラッシュを続ける
（12）このレジスタは一般的セットアップ・パラメータを保持する
（13）これはポート速度ビットマップ・レジスタである。ポートに対するビットがリセットされたとき、これは１０ｍｈｚポートであり、ビットがセットされたとき、これは１００ｍｈｚポートである。即ち、０＝１０ｍｈｚ、１＝１００ｍｈｚである。パワーアップ・デフォールトは正しい値を含むべきである
（14）これはポート・タイプ・ビットマップ・レジスタである。ポートに対するビットがリセットされたとき、これはＱＣポートであり、ビットがセットされたとき、これはＴＬＡＮポートである。即ち、０＝ＱＣ、１＝ＴＬＡＮである。パワーアップ・デフォールトは正しい値を含むべきである
（15）これは、ＣＰＵ２３０からのメモリ転送に対するアドレス及びコントロールを含むレジスタである
（16）このリード・オンリ・レジスタはＥＰＳＭ２１０に対する改訂番号（revision number）を供給する
（17）このレジスタは、ＨＣＢ４０２使用率（ユーティリゼーション）について観察されるべきポート及びモード・ビットを選択する。その可能なモードは、ＴＸ、ＲＸ及びその両方である
（18）ＨＣＢ４０２ユーティリゼーションは、選択されたポートがバス上にある平均時間である
（19）このレジスタは、どのソース・ポートがＣＴを行えるか及びどのものがＳｎＦを行うことのみできるかを示すための、ポートに対するビットマップである
（20）このレジスタは、どの宛て先ポートがＣＴを行えるか及びどのものがＳｎＦを行うことのみできるかを示すための、ポートに対するビットマップである
（21）このレジスタは指定されたポートに対するエクスファーサイズ（ｘｆｅｒｓｉｚｅ）を含む
（22）このレジスタは指定されたポートに対するエクスファーサイズ（ｘｆｅｒｓｉｚｅ）を含む
（23）このレジスタは指定されたポートに対するエクスファーサイズ（ｘｆｅｒｓｉｚｅ）を含む
（24）このレジスタは指定されたポートに対するエクスファーサイズ（ｘｆｅｒｓｉｚｅ）を含む
（25）このレジスタはアービトレーション（仲裁）・モード値を含む。使用可能なアービトレーション・モードはＦＣＦＳ、ウエイト（重み付け）、又はラウンド・ロビン（round_robin）である
（26）ＨＣＢ４０２のサブセクションに対する種々のコントロール
（27）ディスエーブルされるべきポートのビットマップ
（28）このレジスタは、ポート状態ビットマップ・レジスタにおいて示されたポートがどの状態に変更すべきかを教える

状態値
条件
００ｂ
ディスエーブルされる
０１ｂ
ブロックされる／
聞く
１０ｂ
学習する
１１ｂ
送る
（29）このレジスタは、どのポートがその状態を変化させるかを示す。このレジスタはプログラム・ポート状態レジスタと組になって、ポート状態レジスタを満たす
（30）各ポートに対しての２ビットが、以下のように、アービタにポートがどの状態にあるかを教える
状態値
条件
００ｂ
ディスエーブルされる
０１ｂ
ブロックされる／
聞く
１０ｂ
学習する
１１ｂ
送る
（31）各ポートに対しての２ビットが、以下のように、アービタにポートがどの状態にあるかを教える
状態値
条件
００ｂ
ディスエーブルされる
０１ｂ
ブロックされる／
聞く
１０ｂ
学習する
１１ｂ
送る
（32）宛て先ミス・ブロードキャスト（放送）・ビットマップ
（33）メモリ・バス２１４モニタ・コントロールは、メモリ・バス２１４上で行われる監視（モニタリング）を（それが行われる場合に）セットアップするために用いられる
（34）メモリ・バス２１４モニタ・スレッショルドは、アラームをセットするため及びアラームをクリアするために用いられる
（35）メモリ・バス２１４ユーティリゼーション・レジスタ
（36）メモリ・スレッショルド・カウンタによるメモリ・スペースの欠如によってドロップされる（落とされる）パケットの数。このレジスタはリード（読み出し）されるときにクリアされる
（37）ブロードキャスト・メモリ・スペースの欠如によってドロップされるブロードキャスト・パケットの数

（38）ハッシュ・テーブルのベースに対するアドレス。ハッシュ・テーブルのサイズはレジスタの定義で説明したものである
（39）受信セクタ・スレッショルド・オーバーフローのセット又はクリアの何れかによってＣＰＵ２３０に割り込みを行ったポートのビットマップ
（40）送信パケット・スレッショルド・オーバーフローのセット又はクリアの何れかによってＣＰＵ２３０に割り込みを行ったポートのビットマップ
（41）ハッシュ・テーブル内を見たときにミスしたアドレス
（42）残りのハッシュ・アドレス及びソース・ポート
（43）受信メモリ・セクタ・オーバーフローによってドロップされたパケットの数。このレジスタは読み出されたときにクリアされる
（44）送信メモリ・セクタ・オーバーフローによってドロップされたパケットの数。このレジスタは読み出されたときにクリアされる
（45）このレジスタは、受信オーバーフローによってパケットをドロップしたポートのビットマップである
（46）このレジスタは、送信オーバーフローによってパケットをドロップしたポートのビットマップである
（47）学習（ラーニング、learning）ディスエーブル・ポート・ビットマップ
（48）セキュア（機密保護、secure）・ポート・ビットマップ
（49）このレジスタはポート・セキュリティによってドロップされたパケットの合計を含む
（50）このレジスタは、セキュリティによってパケットをドロップしたポートのビットマップである
（51）このレジスタはメモリ・タイプ、速度その他を含む
（52）ＲＡＳがメモリの４Ｍブロックをイネーブルにする
（53）リフレッシュ・カウンタがメモリ・コントローラに対してリフレッシュ信号を生成する
（54）このレジスタはアドレス・フィルタリング及びアドレスのマスキングをイネーブルにする
（55）このレジスタはアドレス・フィルタリングに対するマスク・ビットを含む
（56）このレジスタはアドレス・フィルタリングに対するマスク・ビットを含む
（57）このレジスタはアドレス・フィルタ０のバイト０－３を含む
（58）このレジスタはアドレス・フィルタ０のバイト４－５を含む
（59）このレジスタはアドレス・フィルタ１のバイト０－３を含む
（60）このレジスタはアドレス・フィルタ１のバイト４－５を含む
（61）このレジスタはアドレス・フィルタ２のバイト０－３を含む
（62）このレジスタはアドレス・フィルタ２のバイト４－５を含む
（63）このレジスタはアドレス・フィルタ３のバイト０－３を含む
（64）このレジスタはアドレス・フィルタ３のバイト４－５を含む
（65）このレジスタはＭＣＢ４０４において開始された任意の割り込みのソースを含む
（66）このレジスタはＭＣＢ４０４において開始された任意の割り込みに対するマスキングを含む
（67）このレジスタは、マスクされたＭＣＢ４０４において開始された任意の割り込みのソースを含む
（68）このレジスタはプロミスキュアス（無差別、promiscuous）・モードで観察されているポートの値を保持する。また、ＲＸトラフィック及びＴＸトラフィックが現れるポートを含む
（69）これはカッド・カスケード・レジスタに対するオフセットである。これはＱＣ０に対するものである
（70）これはカッド・カスケード・レジスタに対するオフセットである。これはＱＣ１に対するものである

（71）これはカッド・カスケード・レジスタに対するオフセットである。これはＱＣ２に対するものである
（72）これはカッド・カスケード・レジスタに対するオフセットである。これはＱＣ３に対するものである
（73）これはカッド・カスケード・レジスタに対するオフセットである。これはＱＣ４に対するものである
（74）これはカッド・カスケード・レジスタに対するオフセットである。これはＱＣ５に対するものである
（75）これは、カッド・カスケードから読み出されたばかりの統計バッファに対するアドレス・スペースである
（76）これは、パケット・データをＨＣＢ４０２へ送信するため／パケット・データをＨＣＢ４０２から受信するための、ＦＩＦＯのアドレスである
（77）これは、データをＭＣＢ４０４へ送信するため／データをＭＣＢ４０４から受信するための、ＦＩＦＯのアドレスである
テーブル（１）のレジスタを明瞭にするため、下記のレジスタ定義を提供する。
【０２６４】
【表２３】
割込み情報
ＥＰＳＭ２１０からＣＰＵ２３０に対する３つの割込みピン；ＣＰＵＩＮＴＨＡＳＨＬ、ＣＰＵＩＮＴＰＫＴＬおよびＣＰＵＩＮＴＬがある。ＣＰＵＩＮＴＨＡＳＨＬは、ハッシュ・ミスが生じた時にのみ代入され、（オフセット 'hocにおける）ハッシュ・アドレス・ロー・レジスタを読出すことによりクリアされる。ＣＰＵＩＮＴＰＫＴＬは、パケット・インターフェースＦＩＦＯで利用可能なパケットがある時か、あるいはパケット・インターフェースＦＩＦＯが更に多くのパケット・データを送るためクリアされるバッファ・スペースを有するならば、代入される。ＣＰＵＩＮＴＬは、４つのあり得るソースに対して代入され；これらソースの１つがＭＣＢ４０４における８つのあり得るソースを指す。割込みソースは、これらソースがマスクされなければ、ＣＰＵ２３０を割込みさせる。ＣＰＵ２３０が割込みされることなく割込みソースの情報を利用可能にさせるため、ポーリング機構が利用可能である。割込みソースのマスキングが割込みをＣＰＵ２３０からブロックさせるが、情報はいぜんとしてポーリング・ソース・レジスタで利用可能である。例えば、要求される統計数字が利用可能である時にＳＴＡＴ＿ＲＤＹマスク・ビットがセットされるならば、割込みは生じないが、ＣＰＵ２３０はいぜんとして統計数字がポーリング・レジスタの読出しにより読出す用意があると判定することができる。注：割込みソース・レジスタはこれを読出すことによりクリアされるが、ポーリング・ソース・レジスタはこれをクリアするため書込まれなければならない。
割込みソース１レジスタ：（オフセット＝'ｈ００）ＣＰＵ２３０へ送られるＣＰＵＩＮＴＬ割込みのソース。このレジスタは、ＥＰＳＭ２１０により更新され、その時割込みがＣＰＵ２３０へ送出される。ＣＰＵ２３０がこのレジスタに達すると内容がクリアされる。１ビットにおける１の値は、割込みが生じたことを表示する。デフォルト＝３２'ｈ０００００＿００００。
ビット０（Ｗ／Ｒ）　ＭＣＢ＿ＩＮＴは、割込みがＭＣＢ４０４に生じたことおよび割込みを更に理解するためＭＣＢ割込みソース・レジスタが読出される必要があることをＣＰＵ２３０に通知する割込みである。デフォルトは０。
ビット１（Ｗ／Ｒ）　ＭＥＭ＿ＲＤＹは、要求されたメモリ・データがバッファ・スペースで利用可能であることをＣＰＵ２３０に通知する割込みである。デフォルトは０。
ビット２（Ｗ／Ｒ）　ＡＢＯＲＴ＿ＰＫＴは、ＡＢＯＲＴ＿ＩＮ＊信号がＰＣＢ４０６へ表明されたことをＣＰＵ２３０に通知する割込みである。デフォルトは０。
ビット３（Ｗ／Ｒ）　ＳＴＡＴ＿ＲＤＹは、要求された統計数字情報がＰＣＢ　４０６のバッファ・スペースにおいて用意があることをＣＰＵ２３０に通知する割込みである。デ

フォルトは０。 ビット４～３１（ＲＯ） 予約（ＲＥＳＥＲＶＥＤ） 常に０として読出す。 割込みソース・レジスタに対するｐｃｂｒｅｇｓインターフェース
ＭｃｂＩｎｔ（ｉｎ） ＭＣＢからの入力、ビット０を判別
ＭｅｍＲｄｙ（ｉｎ） メモリＦＩＦＯからの入力、ビット１を判別
ＡｂｏｒｔＰｋｔＩｎｔ（ｉｎ） ＨＣＢ４０２インターフェースからの入力、ビット４を判別
ＳｔａｔＲｄｙＩｎｔ（ｉｎ） ＱＣインターフェースからの入力、ビット５を判別
ＣｐｕＩｎｔ＿（ｏｕｔ） 割込みが生じたことを示すＣＰＵ２３０への信号。
割込みマスク１レジスタ （オフセット＝'ｈ０４）ＣＰＵ２３０によりマスクされる割込み。任意のビットにおける１の値が、割込みがマスクされることを示す。デフォルト＝３２’ｈ ００００＿００１ｆ
ビット０（Ｗ／Ｒ） ＣＰＵ２３０に対するＭｃｂＩｎｔ割込みをマスク。デフォルトは１
ビット１（Ｗ／Ｒ） ＣＰＵ２３０に対するＭｅｍＲｄｙ割込みをマスク。デフォルトは１
ビット２（Ｗ／Ｒ） ＣＰＵ２３０に対するＡｂｏｒｔＰｋｔＩｎｔ割込みをマスク。デフォルトは１
ビット３（Ｗ／Ｒ） ＣＰＵ２３０に対するＳｔａｔＲｄｙＩｎｔ割込みをマスク。デフォルトは１
ビット４（Ｗ／Ｒ） ＣＰＵ２３０に対するＨａｓｈＭｉｓｓ割込みをマスクデフォルトは１
ビット５～３１（ＲＯ） 予約。常に０として読出し。
割込みソース２レジスタ （オフセット＝'ｈ１８）ＣＰＵ２３０へ送られるＣＰＵＩＮＴＰＫＴＬ割込みのソース。このレジスタはＥＰＳＭ２１０により更新され、次に割込みがＣＰＵ２３０へ送られる。ＣＰＵ２３０がこのレジスタを読出す時、内容がクリアされる。１ビットにおける１の値は、割込みが生じたことを示す。デフォルト＝３２’ｈ００００＿００００
ビット０（Ｗ／Ｒ） ＰＫＴ＿ＡＶＡＩＬは、パケット・データがＣＰＵ２３０に対して利用可能であることをＣＰＵ２３０へ通知する割込み。デフォルトは０
ビット１（Ｗ／Ｒ） ＢＵＦ＿ＡＶＡＩＬは、パケット・データを送出するためバッファ・スペースがＣＰＵ２３０に対して利用可能であることをＣＰＵ２３０に通知する割込み。デフォルトは０
ビット２～３１（ＲＯ） 予約。常に０として読出し
割込みソース・レジスタに対するｐｃｂｒｅｇｓインターフェース
ＰｋｔＡｖａｉｌＩｎｔ（ｉｎ） ＴＸ ＦＩＦＯからの入力、ビット２を判別
ＢｕｆＡｖａｉｌＩｎｔ（ｉｎ） ＲＸ ＦＩＦＯからの入力、ビット３を判別
ＣｐｕＩｎｔ＿Ｐｋｔ＿（ｏｕｔ） パケット割込みが生じたことを示すＣＰＵ２３０に対する信号
インターフェース・マスク２レジスタ （オフセット＝'ｈｌｃ）ＣＰＵ２３０によりマスクされる割込み。任意のビットにおける１の値は、割込みがマスクされることを示す。デフォルト＝３２'ｈ０００００＿０００３。
ビット０（Ｗ／Ｒ） ＣＰＵ２３０に対するＰｋｔＡｖａｉｌＩｎｔ割込みをマスク。デフォルトは１
ビット１（Ｗ／Ｒ） ＣＰＵ２３０に対するＢｕｆＡｖａｉｌＩｎｔ割込みをマスク。デフォルトは１
ビット２～３１（ＲＯ） 予約。常に０として読出し
ポーリング・ソース１＆２レジスタ （オフセット＝'ｈ１４）このレジスタはマスクされた割込み情報を含み、所望のビットをクリアするため１を書込むＣＰＵ２３０によりクリアされる。このため、ＣＰＵ２３０が割込みされる代わりにポーリングすることを可能

にする。ＣＰＵは代わりにポーリングを欲する任意の割込みソースをマスクしなければならない
ビット０（Ｗ／Ｒ）　ＭＣＢ＿ＩＮＴは、割込みがＭＣＢ４０４に生じたことおよび割込みを更に理解するためＭＣＢ割込みソース・レジスタが読出される必要がある。デフォルトは０
ビット１（Ｗ／Ｒ）　ＭＥＭ＿ＲＤＹは要求されたメモリ・データがバッファ・スペースで利用可能であることをＣＰＵ２３０に通知する割込み。デフォルトは０
ビット２（Ｗ／Ｒ）　ＰＫＴ＿ＡＶＡＩＬは、パケット・データがＣＰＵ２３０に対して利用可能であることをＣＰＵ２３０に通知する割込み。デフォルトは０
ビット３（Ｗ／Ｒ）　ＢＵＦ＿ＡＶＡＩＬは、バッファ・スペースがＣＰＵ２３０がパケット・データを送るために利用可能であることをＣＰＵ２３０に通知する割込み。デフォルトは０
ビット４（Ｗ／Ｒ）　ＡＢＯＲＴ＿ＰＫＴは、ＡＢＯＲＴ＿ＩＮ信号がＰＣＢ４０６へアサートされたことをＣＰＵ２３０に通知する割込み。デフォルトは０
ビット５（Ｗ／Ｒ）　ＳＴＡＴ＿ＲＤＹは、要求された統計情報がＰＣＢ４０６のバッファ・スペースにおいて用意があることＣＰＵ２３０に通知する割込み。デフォルトは０
ビット６（Ｗ／Ｒ）　ＨＡＳＨ＿ＭＩＳＳは、ハッシュ・ミスが生じたことをＣＰＵ２３０に通知する割込み。
ビット７～３１（ＲＯ）　予約。常に０として読出し
ポーリング・ソース・レジスタに対するｐｃｂｒｅｇｓインターフェース
ＭｃｂＩｎｔ（ｉｎ）　ＭＣＢからの入力。ビット０を判別
ＭｅｍＲｄｙ（ｉｎ）　メモリＦＩＦＯからの入力。ビット２を判別
ＰｋｔＡｖａｉｌＩｎｔ（ｉｎ）　ＴＸ　ＦＩＦＯからの入力
ビット２を判別
ＢｕｆＡｖａｉｌＩｎｔ（ｉｎ）　ＲＸ　ＦＩＦＯからの入力
ビット３を判別
ＡｂｏｒｔＰｋｔＩｎｔ（ｉｎ）　ＨＣＢ４０２インターフェースからの入力
ビット４を判別
ＳｔａｔＲｄｙＩｎｔ（ｉｎ）　ＱＣインターフェースからの入力
ビット６を判別
ｍ＿ＨａｓｈＩｎｔ（ｉｎ）　ＭＣＢ４０４からの入力　ビット６を判別。
【０２６５】
【表２４】
パケット・データのコンフィギュレーション
パケット転送のため使用される３つのレジスタがあり、１つは受取られたパケットに対し、２つは伝送パケットに対する。受信パケットは、ＨＳＢ２０６からのＲｅａｄＯｕｔＰｋｔ信号と関連させられる。送信パケットは、ＨＳＢ２０６からのＷｒｉｔｅＩｎＰｋｔ信号と関連させられる。注：受信と送信の用語は、ＨＳＢ２０６から参照される。ＣＰＵ２３０は、パケット・データ・バッファをアクセスする前に適切なレジスタをアクセスしなければならない。
パケット情報ＲｄＰｋｔレジスタ　（オフセット＝'ｈ０８）ＣＰＵ２３０により送られるデータのパケットに対する必要な情報。ＨＳＢ２０６から参照される受信パケットデフォルト＝３２'ｈ０００００＿００００
ビット０（Ｗ／Ｒ）　ＳＯＰ　ＣＰＵ２３０からのパケットの初め
１＝ＳＯＰ
ビット１（Ｗ／Ｒ）　ＥＯＰ　ＣＰＵ２３０からのパケットの終り
１＝ＥＯＰ
ビット２～１５（ＲＯ）　予約。常，に０として読出し
ビット１６～２３（Ｗ／Ｒ）　ＥＯＰがアサートされる時、ＦＩＦＯにおけるデータ長さ（バイト数）

ビット２４～３１（ＲＯ）　予約。常に０として読出し
パケット情報ＲｄＰｋｔレジスタに対するｐｃｂｒｅｇｓインターフェース
ｒ＿Ｓｏｐ（ｏｕｔ）　ＨＳＢ２０６インターフェースに与えられたパケット・インジケータの開始
ｒ＿Ｅｏｐ（ｏｕｔ）　ＨＳＢ２０６インターフェースに与えられたパケット・インジケータの終り
ｒ＿ｌｅｎｇｔｈ（ｏｕｔ）　ＥＯＰが表示される時バッファにおけるデータのバイト長
パケット情報ＷｒＰｋｔレジスタ　（オフセット＝'ｈ０ｃ）　ＨＳＢ２０６により送られるデータのパケットに対する必要情報。ＨＳＢ２０６から照会される伝送パケットデフォルト＝３２'ｈ０００００＿００００
ビット０（Ｗ／Ｒ）　ＳＯＰ。ＨＳＢ２０６からのパケットの開始
ｌ＝ＳＯＰ
ビット１（Ｗ／Ｒ）　ＥＯＰ。ＨＳＢ２０６からのパケットの終り
Ｉ＝ＥＯＰ
ビット２～５（Ｗ／Ｒ）　ＳＯＰまたはＥＯＰと関連するＤＷＯＲＤに対するバイト使用可能。通常は、全てのバイトが使用可能化される。ｌ＝使用可能化
ビット６～１５（ＲＯ）　予約。常に０として読出し
ビット１６～２３（Ｗ／Ｒ）ＦＩＦＯにおけるデータ長さ（バイト数）
ビット２４～３１（ＲＯ）　予約。常に０として読出し
パケット情報ＷｒＰｋｔレジスタに対するｐｃｂｒｅｇｓインターフェース
ｈ＿ＳｏｐＩｎ＿（ｉｎ）　ＨＳＢ２０６インターフェースからのＳＯＰインジケータ
ｈ＿ＥｏｐＩｎ＿（ｉｎ）　ＨＳＢ２０６インターフェースからのＥＯＰインジケータ
ｈ＿ＢｙｔｅＶａｌＩｎ＿（ｉｎ）　ＨＳＢ２０６インターフェースからのバイト使用可能
合計パケットＩｎｆｏ（オフセット＝'ｈ２４）これは、ＭＣＢ４０４がパケットをＣＰＵ２３０へ送る前にそのパケットに付加する情報。この値は、ＣＰＵ向けパケットに対するＳＯＰがある時にセットされる
デフォルト＝３２'ｈ００００＿００００
ビット０～１５　　パケット長
ビット１６～２３（ＲＯ）　ソース・ポート
ビット２４～３１（ＲＯ）　宛先ポート
【０２６６】
【表２５】
メモリ存在の検出
ＳＩＭＭ／ＤＩＭＭ存在検出レジスタ　（オフセット＝'ｈ１０）システムにおけるＳＩＭＭについての情報を保持。この情報は、オンボードのシフト・レジスタからリセットされた僅かに後でロードされる
ビット０～３（ＲＯ）　ｓｉｍｍ１＿ｐｄ［０．．３］
ビット４～７（ＲＯ）　ｓｉｍｍ２＿ｐｄ［０．．３］
ビット８～１１（ＲＯ）　ｓｉｍｍ３＿ｐｄ［０．．３］
ビット１２～１５（ＲＯ）　ｓｉｍｍ４＿ｐｄ［０．．３］
ビット１６～３１（ＲＯ）　予約。常に０として読出し
存在検出レジスタに対するｐｃｂｒｅｇｓインターフェース
ｉ＿ＰＤＳｅｒＩｎ（ｉｎ）　存在検出シフト・レジスタからのシリアル入力
【０２６７】
【表２６】
クワッドカスケード統計セットアップ
ＱＣ統計Ｉｎｆｏレジスタ　（オフセット＝'ｈ２０）クワッドカスケード統計レジスタの読出し動作のためのセットアップ情報。ＣＰＵは、このレジスタに統計読出しを開始することを書込む。デフォルト＝３２'ｈ００００＿８０００

ビット０～１（Ｗ／Ｒ） ポート番号。これは、その統計数字が読出されるポート番号。読出すべきポートは、この番号と指定クワッドカスケードにより決定される
ビット２～４（Ｗ／Ｒ） ＱＣ番号。アクセスするクワッドカスケードを指示予約された組合わせ：３’ｂ１１０および３’ｂ１１１
ビット５～９（Ｗ／Ｒ） レジスタ番号。これは、指定されたポートに対して読出されるべき第１のレジスタの番号
ビット１０～１４（Ｗ／Ｒ） レジスタ数。これは、読出すべきレジスタ数注：ソフトウエアは、この数を、読出すため利用可能なレジスタ範囲内のレジスタ番号と共に保持するため要求される
ビット１５～１９（Ｗ／Ｒ） 最大レジスタ数。これは、クワッドカスケードで利用可能な統計的レジスタの最大数。デフォルト＝６'ｈ１７
ビット２０～３１（Ｗ／Ｒ） 予約。常に０として読出し
クワッドカスケード統計セットアップ・レジスタに対するｐｃｂｒｅｇｓインターフェース
ｒ＿ＱｃＳｔａｔＰｏｒｔＮｏ（ｏｕｔ） 読出された統計に対するポート番号。これは、０と３間の値。この値は、ＱＣ数と共に用いられて、スイッチにおけるどのポートが観察されつつあるかを決定する
ｒ＿ＱｃＳｔａｔＱｃＮｏ（ｏｕｔ） Ｑｃ数。ポート番号と共に用いられる。
ｒ＿ＳｔａｔＲｅｇＮｏ（ｏｕｔ） 始動レジスタ番号。これは、読出されるべき最初の統計レジスタの番号
ｒ＿ＮｏＳｔａｔＲｅｇｓ（ｏｕｔ） 読出すべき統計レジスタ数
ｒ＿Ｍａｘｒｅｇｓ（ｏｕｔ） 存在する統計レジスタの最大数。これは、保持される統計数字が変更されるならば将来の使用のために特に利用可能。
【０２６８】
【表２７】
ＥＰＳＭ２１０のセットアップ
ＥＰＳＭセットアップ・レジスタ （オフセット＝'ｈ３０）ＥＰＳＭ２１０に対する汎用セットアップ・パラメータ。デフォルト＝３２'ｈ０００７＿１０００または３２'ｈ０００７＿３０００、ｃｋｌｌｓｅｌ入力に依存
ビット０（Ｗ／Ｒ） ＴＰＩインストール。ｌ＝ＴＰＩ２２０インストールデフォルト＝０。このビットは、マスタ・スイッチ使用可能（ビット２）が否定される時にのみ、書込まれる
ビット１（Ｗ／Ｒ） ＥＸＰインストール。ｌ＝拡張インストール。デフォルト＝０。このビットは、マスタ・スイッチ使用可能（ビット２）が否定される時にのみ、書込まれる
ビット２（Ｗ／Ｒ） マスタ・スイッチ使用可能。ｌ＝パケット・トラフィック使用可能。デフォルト＝０
ビット３～４（Ｗ／Ｒ）ＱｃＸｆｅｒＳｉｚｅ［１：０］これらビットは、マスタ・スイッチ使用可能（ビット２）が否定される時にのみ、書込まれる
００＝ＨＳＢ２０６における１６バイト転送サイズ
０１＝ＨＳＢ２０６における３２バイト転送サイズ
１０＝ＨＳＢ２０６における６４バイト転送サイズ
１１＝無効組合わせ
ビット５～６（Ｗ／Ｒ）ＴＰＩＸｆｅｒＳｉｚｅ［１：０］これらのビットはマスタ・スイッチ使用可能（ビット２）が否定される時にのみ、書込まれる
００＝ＨＳＢ２０６における１６バイト転送サイズ
０１＝ＨＳＢ２０６における６４バイト転送サイズ
１０＝ＨＳＢ２０６における１２８バイト転送サイズ
１１＝ＨＳＢ２０６における２５６バイト転送サイズ
ビット７（Ｗ／Ｒ） ＡＩＦＣＳ。このビットは、クワッドカスケードがＦＣＳビットを自動挿入することを可能にするため使用される。これは、ＣＰＵ２３０からのパケット

に対してのみ用いられる
ビット８（Ｗ／Ｒ）　ＤｒａｍＷｒＤｉｓ。これは、セットされた時、ＣＰＵ２３０からＤＲＡＭへの書込みを使用不能状態にする。デフォルト＝０
ビット９（Ｗ／Ｒ）　ＳｒａｍＷｒＤｉｓ。これは、セットされた時、ＣＰＵ２３０から内部ＳＲＡＭへの書込みを使用不能状態にする
ビット１０～１２（Ｗ／Ｒ）ＥＰＳＭ２１０アドレス・デコード。これらビットは、ＥＰＳＭ２１０のレジスタ・スペースとメモリ・インターフェースをデコードするため用いられる
ビット１３（ＲＯ）　ｃｌｋｌｓｅｌ
１＝ＣＬＫ２周波数は１Ｘ　ＣＬＫ１周波数である
０＝ＣＬＫ２周波数は２Ｘ　ＣＬＫ１周波数である
ビット１４～２１（ＲＯ）　ＣＰＵポート番号。ＣＰＵ２３０のポート番号を指示。デフォルト＝８'ｈ１ｃ
ビット２２～～３１（ＲＯ）　予約。常に０として読出し
ＥＰＳＭセットアップ・レジスタに対するｐｃｂｒｅｇｓインターフェース
ｃｌｋｌｓｅｌ（ｉｎ）　ｃｌｋ１１およびｃｌｋ２が同じレートであるかどうかを判別するためピンからの入力
ｒ＿ＤｒａｍＷｒＤｉｓ（ｏｕｔ）　ＣＰＵ２３０インターフェースに、ＤＲＡＭへの書込みが使用不能化されることを知らせる
ｒ＿ＳｒａｍＷｒＤｉｓ（ｏｕｔ）　ＣＰＵ２３０インターフェースに、内部ＳＲＡＭへの書込みが使用不能化されることを知らせる
ｒ＿ＥＰＳＭＡｄｒＤｃｄ（ｏｕｔ）　この３ビット数は、ＣＰＵ２３０バスにおけるアドレス・ビット３１：２９と比較される。
ＥＰＳＭセットアップ・レジスタに対するｈｃｂｒｅｇｓインターフェース
ｒ＿ＭｓｔｒＳｗＥｎ（ｏｕｔ）　スイッチがパケット通信量に対して使用可能化されることをアービタなどに対して通知
ｒ＿ＴｐｉＩｎｓｔ（ｏｕｔ）
ｒ＿ＥｘｐＩｎｓｔ（ｏｕｔ）
ｒ＿ＮｏｎＵＬＢＣＭｏｄｅ［１：０］（ｏｕｔ）
ｒ＿ＵＬＢＣＭｏｄｅ［１：０］（ｏｕｔ）
ｒ＿ＡＩＦＣＳ（ｏｕｔ）
ＥＰＳＭセットアップ・レジスタに対するｍｃｂｒｅｇｓインターフェース
ｒ＿ＤｒａｍＷｒＤｉｓ（ｏｕｔ）　ＤＲＡＭ書込みのＣＰＵ要求を使用不能
ｒ＿ＳｒａｍＷｒＤｉｓ（ｏｕｔ）　内部ＳＲＡＭ書込みのＣＰＵ要求を使用不能
ＥＰＳＭ改訂レジスタ　（オフセット＝'ｈ４０）ＥＰＳＭ２１０の改訂番号
ビット０～７（ＲＯ）　ＥＰＳＭ２１０の改訂番号
ビット８～31（ＲＯ）　リザーブ（予約）。常に０として読出し
ＥＰＳＭ改訂レジスタに対するｐｃｂｒｅｇｓインターフェース
なし。
【０２６９】
【表２８】
ポート・セットアップ
ポート速度レジスタ　（オフセット＝'ｈ３４）各ポートの速度を含むビット・マップ。
１＝１００ＭＨｚ；０＝１０ＭＨｚ。デフォルト＝３２'ｈ０ｆ００＿０００
ビット０（Ｗ／Ｒ）　ポート０の速度
ビット１（Ｗ／Ｒ）　ポート１の速度
：
：
ビット２７（Ｗ／Ｒ）　ポート２７の速度
ビット２８～３１（ＲＯ）　留保済み。常に０として読出し

ポート速度レジスタに対するｈｃｂｒｅｇｓインターフェース
ｒ＿ＰｏｒｔＳｐｄ［２７：０］（ｏｕｔ）　ＨＣＢ４０２ブロックに対するポート速度ビット・マップ
ポート・タイプ・レジスタ　（オフセット＝’ｈ３８）各ポートのタイプを含むビット・マップ。１＝ＴＬＡＮ；０＝カッドカスケード。デフォルト＝３２'ｈ０ｆ００＿００００
ビット０（Ｗ／Ｒ）　ポート０タイプ
ビット１（Ｗ／Ｒ）　ポート１タイプ
：
：
ビット２７（Ｗ／Ｒ）　ポート２７タイプ
ビット２８～３１（Ｗ／Ｒ）予約。常に０として読出し
ポート・タイプ・レジスタに対するｍｃｂｒｅｇｓ＆ｈｃｂｒｅｇｓインターフェース
ｒ＿ＰｏｒｔＴｙｐｅ［２７：０］（ｏｕｔ）　ＨＣＢ４０２およびＭＣＢ４０４に対するポート・タイプ・ビット・マップ
ＣＰＵメモリ要求
ＣＰＵ２３０によるメモリ要求は、２つの方法で行うことができる。下記のレジスタが両方法で用いられる；ＣＰＵ２３０は、初期レジスタ／ＦＩＦＯメモリ要求法を用いる時に、レジスタを直接アクセスするのみ。
メモリ要求レジスタ　（オフセット＝'ｈ３ｃ）ＣＰＵは、このレジスタにメモリ読出しまたは書込みを要求するよう書込む。この要求された機構は、外部ＤＲＡＭまたは内部ＳＲＡＭのいずれかのアクセスのため用いられる。
ビット０～２３（Ｗ／Ｒ）　転送の開始アドレス［２５：２］。ＳＲＡＭアクセスのためには、ビット２３－８が留保されるビット７：０が２５６の２４ビット・ワードをアドレス指定する
ビット２４（Ｗ／Ｒ）　　０＝外部ＤＲＡＭアクセス（即ち、パケット＆ハッシュ・メモリ）
１＝内部ＳＲＡＭアクセス（即ち、パケット制御レジスタ）
ビット２５（Ｗ／Ｒ）　　転送長
０＝１転送（４バイト）
１＝４転送（１６バイト）
注：開始アドレス＆転送長さは、転送が２Ｋページ境界に跨がるようにセットされるべきではない。これを保証する１つの方法は、全てのデータ構造（ハッシュ・エントリの如き）が１６バイト整合されることを確認することである
ビット２６～２９（Ｗ／Ｒ）　バイト使用可能［３：０］。（１＝表明）部分ワード書込みに有効。また、ＣＡＳを含まない読出しを行うようセットされたＥＤＯテスト・モードと共に使用される。１より大きい転送長の書込みのため、ＢｙｔｅＥｎａｂｌｅｓは１１１１でなければならない。これらは、ＥＤＯテスト・モードがセットされなければ、読出しは問題外（ｄｏｎ’ｔ　ｃａｒｅ）である。
ビット３０（Ｗ／Ｒ）　書込み／読出し。０＝読出し、１＝書込み
ビット３１（Ｗ／Ｒ）　ロックされたページ・ヒット。別のＣＰＵ要求が同じメモリ・ページ内に続くことを示す。ＤＲＡＭメモリ・アービタはメモリ・システムが別の要求をすることを許容せず、ＲＡＳはその時のサイクル後に表明されたままとなる。ＥＤＯテスト・モードのみにおいて用いられる。リフレッシュを含む他の要求側は、セットされる間はメモリ・アクセスを行わない。ＳＲＡＭがロックされる間パケット・メモリ通信量入力が停止するので、ＳＲＡＭアクセスにおいては決して使用すべきでない（ハードウエアのデバギングを除く）。
メモリ要求レジスタに対するｍｃｂｒｅｇｓインターフェース
ＣｐｕＡｄｒ［２５：２］（ｏｕｔ）　開始アドレスｍｅｍｃｔｌおよびｍｃｂｓｒａｍモジュールをパス

ＣｐｕＢＥ［３：０］（ｏｕｔ） ｍｅｍｃｔｌおよびｍｃｂｓｒａｍモジュールへＢｙｔｅＥｎａｂｌｅｓをパス
ＣｐｕＬｎ［１：０］（ｏｕｔ） ｍｅｍｃｔｌおよびｍｃｂｓｒａｍへ転送長さをパス（００ １のＩｎ＝１ならば、００；Ｉｎ＝４ならば、１１）
ＣｐｕＭｅｍＳｅｌ（ｏｕｔ） 外部ＤＲＡＭ（０）および内部ＳＲＡＭ（１）データ間のｍｕｘ制御
ＣｐｕＷｒ（ｏｕｔ） 書込み／読出しビット＝１ならば、ｍｅｍｃｔｌおよびｍｃｂｓｒａｍモジュールへアサート
ＣｐｕＰｇＨｉｔ（ｏｕｔ） ロック・ページ・ヒット・ビット＝１ならば、ｍｅｍｃｔｌおよびｍｃｂｓｒａｍモジュールへアサート
ＣｐｕＲｅｑ（ｏｕｔ） メモリ要求レジスタが書込まれメモリ選択＝０である時、ｍｅｍｃｔｌモジュールへアサートＣｐｕＡｃｋがアサートされるまで、アサートされたままでなければならない
ＣｐｕＡｃｋ（ｉｎ） ＣｐｕＲｅｑが受入れられる時、ｍｅｍｃｔｌモジュールからｍｃｂｒｅｇｓへアサートされる
ＣｐｕＩｎｔｅｒｎａｌＲｅｑ（ｏｕｔ） メモリ要求レジスタが書込まれ、メモリ選択＝１である時、ｍｃｂｓｒａｍモジュールへアサート。ＣｐｕＩｎｔｅｒｎａｌＡｃｋがアサートされるまで、アサートされたままでなければならない
ＣｐｕＩｎｔｅｒｎａｌＡｃｋ（ｉｎ） ＣｐｕＩｎｔｅｒｎａｌＲｅｑが受信される時、ｍｃｂｓｒａｍモジュールからｍｃｂｒｅｇｓへアサートされる
注：以下のシーケンスをＥＤＯメモリに対するテストに用いるべきである：
１．ＥＤＯテスト・モード・ビットをメモリ制御レジスタにセット
２．ＤＷＯＲＤを００００ｈでテスト中のバンクに書込む
３．ロックド・ページ・ヒット・セットおよびバイト使用可能＝１１１１ｂを持つ同じＤＷＯＲＤを読出す。その後、ＦＰＭ ＤＲＡＭがＭＤをフロートさせる間ＥＤＯ ＤＲＡＭがＭＤをローに保持し、約１００ｎｓ後にＭＤ［０］におけるプルアップ・レジスタがこの線をハイに引上げる
４．ロックド・ページ・ヒット・ビットがクリヤされ、バイト使用可能＝００００ｂによりＤＷＯＲＤを再び読出す。これは、ＣＡＳがアサートされない読出しである。ＭＤ［０］は、ＥＤＯ ＤＲＡＭに対してロー、ＦＰＭに対してハイとなる
５．インストールされたメモリの各バンクごとにステップ１～４を繰返す。全てのバンクがＥＤＯ ＤＲＡＭを含むだけで、メモリ・タイプがＥＤＯ ＤＲＡＭへセットされる
６．ＥＤＯテスト・モード・ビットをクリヤして、メモリ・タイプをセットする。ＥＤＯテスト・モードをセットしたままにしてはならない。
【０２７０】
【表２９】
混雑ポート
混雑ポート・レジスタ （オフセット＝'ｈ１４８）ポートが混雑モードで観察される制御がレジスタに含まれる。デフォルト＝３２'ｈ０００００＿０００００。このレジスタは、マスタ・スイッチ使用可能（ＥＰＳＭセットアップ・レジスタ）が否定される時にのみ書込まれる
ビット０～７（Ｗ／Ｒ） 混雑モードで観察されるポート番号
ビット８～15（Ｗ／Ｒ） 受取られつつあるデータが現れるポート
ビット１６～２３（Ｗ／Ｒ） 観察されるポートへ送られるデータが現れるポート
ビット２４～３１（Ｗ／Ｒ） 予約。常に０として読出し。
【０２７１】
【表３０】
高速バス・モニタ
ＨＳＢ使用セットアップ・レジスタ （オフセット＝'ｈ５４）どのポートがＨＳＢ２０６の使用のためのモニターとなるか制御デフォルト＝３２'ｈ００００＿００００

ビット０～７（Ｗ／Ｒ）　ポート番号または合計
ビット８～９（Ｗ／Ｒ）　モード
ビット１０～３１（ＲＯ）　　予約。常に０として読出し
ＨＳＢ使用レジスタ　　（オフセット＝'ｈ５８）ＨＳＢ２０６の使用は、選択されたポートがＨＳＢ２０６にある平均時間である。デフォルト＝３２'ｈ０００００＿０００００
ビット０～３１（ＲＯ）　選択された平均時間ポートはＨＳＢ２０６にある。
【０２７２】
【表３１】
クワッドスルー／ストアＮ前送情報
ソースＣＴ＿ＳＮＦレジスタ　（オフセット＝'ｈ５ｃ）ソース・ポートのＣＴ／ＳｎＦ状態を含むビット・マップ。０＝ＣＴ；１＝ＳＮＦデフォルト＝３２'ｈ００００＿００００
ビット０（Ｗ／Ｒ）　ポート０ソースＣＴ＿ＳＮＦ
ビット１（Ｗ／Ｒ）　ポート１ソースＣＴ＿ＳＮＦ
：
：
ビット２７（Ｗ／Ｒ）　ポート２７ソースＣＴ＿ＳＮＦ
ビット２８～３１（Ｗ／Ｒ）　予約。常に０として読出し
ソースＣＴ＿ＳＮＦレジスタに対するｈｃｂｒｅｇｓインターフェース
ＴｂｌＳｒｃＰｒｔ（ｉｎ）　その時のパケット・ソース・ポート。８ビット入力
ｒ＿ＲｘＰｏｒｔＣｔＳｎｆ（ｏｕｔ）　ＴｂｌＳｒｃＰｒｔに対するＣＴ＿ＳＮＦ状態　１ビット出力
宛先ＣＴ＿ＳＮＦレジスタ　（オフセット＝'ｈ６０）宛先ポートのＣＴ／ＳｎＦ状態を含むビット・マップ。０＝ＣＴ；１＝ＳＮＦ。デフォルト＝３２'ｈ００００＿００００
ビット０（Ｗ／Ｒ）　ポート０宛先ＣＴ＿ＳＮＦ
ビット１（Ｗ／Ｒ）　ポート１宛先ＣＴ＿ＳＮＦ
：
：
ビット２７（Ｗ／Ｒ）　ポート２７宛先ＣＴ＿ＳＮＦ
ビット２８～３１（ＲＯ）予約。常に０として読出し
ソースＣＴ＿ＳＮＦレジスタに対するｈｃｂｒｅｇｓインターフェース
ＴｂｌＤｓｔＰｒｔ（ｉｎ）　その時のパケット宛先ポート。８ビット入力
ｒ＿ＴｘＰｏｒｔＣｔＳｎｆ（ｏｕｔ）ＴｂｌＤｓｔＰｒｔに対するＣＴ＿ＳＮＦ状態。１ビット出力。
【０２７３】
【表３２】
調停情報
調停モード・レジスタ　（オフセット＝'ｈ７４）調停モード値を含む。デフォルト＝３２'ｈ００００＿００００。このレジスタは、マスタ・スイッチ使用可能（ＥＰＳＭセットアップ・レジスタ）が否定される時にのみ書込まれる
ビット０～１（Ｗ／Ｒ）　調停（アービタレション）モード
２'ｂ００：先入れ先サーブ調停モード
２'ｂ０１：重み付け優先調停モード
２'ｂ１０：ラウンド・ロビン調停モード
２'ｂ１１：これも先入れ先サーブ・モード
ビット２～３１（ＲＯ）　　予約。常に０として読出し
調停モード・レジスタに対するｈｃｂｒｅｇｓインターフェース
ｒ＿ＡｒｂＭｏｄｅ（ｏｕｔ）　ＨＣＢ４０２における調停モジュールで必要である先に示した２ビット値
調停重み付けレジスタ＃１　（オフセット＝'ｈ６４）重み付け優先調停モードに対する

ポート０～７の重み
ビット０～３（Ｗ／Ｒ）　重み付け優先モードに対するポート０調停重み
ビット４～７（Ｗ／Ｒ）　重み付け優先モードに対するポート１調停重み
ビット８～１１（Ｗ／Ｒ）　重み付け優先モードに対するポート２調停重み
ビット１２～１５（Ｗ／Ｒ）　重み付け優先モードに対するポート３調停重み
ビット１６～１９（Ｗ／Ｒ）　重み付け優先モードに対するポート４調停重み
ビット２０～２３（Ｗ／Ｒ）　重み付け優先モードに対するポート５調停重み
ビット２４～２７（Ｗ／Ｒ）　重み付け優先モードに対するポート６調停重み
ビット２８～３１（Ｗ／Ｒ）　重み付け優先モードに対するポート７調停重み
調停重みレジスタ＃１に対するｈｃｂｒｅｇｓインターフェース
ｒ＿ＡｒｂＷｔ０（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート０に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート１に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート２に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ３（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート３に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ４（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート４に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ５（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート５に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ６（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート６に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ７（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート７に対する重み付けのためＨＣＢ４０２により使用される
調停重みレジスタ＃２　　（オフセット＝'ｈ６８）重み付けされた優先調停モードのためのポート８～１５に対する重み
ビット０～３（Ｗ／Ｒ）　重み付け優先モードのためのポート８調停重み
ビット４～７（Ｗ／Ｒ）　重み付け優先モードのためのポート９調停重み
ビット８～１１（Ｗ／Ｒ）　重み付け優先モードのためのポート１０調停重み
ビット１２～１５（Ｗ／Ｒ）重み付け優先モードのためのポート１１調停重み
ビット１６～１９（Ｗ／Ｒ）重み付け優先モードのためのポート１２調停重み
ビット２０～２３（Ｗ／Ｒ）重み付け優先モードのためのポート１３調停重み
ビット２４～２７（Ｗ／Ｒ）重み付け優先モードのためのポート１４調停重み
ビット２８～３１（Ｗ／Ｒ）重み付け優先モードのためのポート１５調停重み
調停重みレジスタ＃２に対するｈｃｂｒｅｇｓインターフェース
ｒ＿ＡｒｂＷｔ８（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート８に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ９（ｏｕｔ）　これら４ビットが、重み付け調停モードにおけるポート９に対する重みつけのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１０（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１０に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１１（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１１に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１２（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１２に対する重みつけのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１３（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１３に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１４（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１

４に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１５（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１５に対する重み付けのためＨＣＢ４０２により使用される
調停重みレジスタ＃３　　（オフセット＝'ｈ６ｃ）重み付け優先モードに対するポート１６～２３の重み
ビット０～３（Ｗ／Ｒ）　重み付け優先モードに対するポート１６調停重み
ビット４～７（Ｗ／Ｒ）　重み付け優先モードに対するポート１７調停重み
ビット８～１１（Ｗ／Ｒ）　重み付け優先モードに対するポート１８調停重み
ビット１２～１５（Ｗ／Ｒ）重み付け優先モードに対するポート１９調停重み
ビット１６～１９（Ｗ／Ｒ）重み付け優先モードに対するポート２０調停重み
ビット２０～２３（Ｗ／Ｒ）重み付け優先モードに対するポート２１調停重み
ビット２４～２７（Ｗ／Ｒ）重み付け優先モードに対するポート２２調停重み
ビット２８～３１（Ｗ／Ｒ）重み付け優先モードに対するポート２３調停重み
調停重みレジスタ＃３に対するｈｃｂｒｅｇｓインターフェース
ｒ＿ＡｒｂＷｔ１６（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１６に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１７（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート17に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１８（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１８に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ１９（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート１９に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２０（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２０に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２１（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２１に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２２（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２２に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２３（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２３に対する重み付けのためＨＣＢ４０２により使用される
調停重みレジスタ＃４　　（オフセット＝'ｈ７０）重み付け優先モードに対するポート１６～２３の重み
ビット０～３（Ｗ／Ｒ）　重み付け優先モードに対するポート２４調停重み
ビット４～７（Ｗ／Ｒ）　重み付け優先モードに対するポート２５調停重み
ビット８～１１（Ｗ／Ｒ）　重み付け優先モードに対するポート２６調停重み
ビット１２～１５（Ｗ／Ｒ）重み付け優先モードに対するポート２７調停重み
ビット１６～１９（Ｗ／Ｒ）重み付け優先モードに対するポート２８調停重み
ビット２０～３１（ＲＯ）　予約。常に０として読出し
調停重みレジスタ＃４に対するｈｃｂｒｅｇｓインターフェース
ｒ＿ＡｒｂＷｔ２４（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２４に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２５（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２５に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２６（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２６に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２７（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２７に対する重み付けのためＨＣＢ４０２により使用される
ｒ＿ＡｒｂＷｔ２８（ｏｕｔ）これら４ビットが、重み付け調停モードにおけるポート２８に対する重み付けのためＨＣＢ４０２により使用される。
【０２７４】

【表３３】
ＨＣＢ４０２混雑制御
ＨＣＢ混雑制御　（オフセット＝'ｈ７８）ＨＣＢ４０２に対する混雑制御
デフォルト＝３２'ｈ０００００＿００００
ビット０（Ｗ／Ｒ）　ＣＴ　ＦＩＦＯ使用可能。１＝ＣＴ　ＦＩＦＯ使用可能
ビット１（Ｗ／Ｒ）　読出し使用可能特別待機状態。１＝待機状態使用可能
ビット２（Ｗ／Ｒ）　クワッドカスケードに対する同時読出しおよび書込みの使用可能
ビット３（Ｗ／Ｒ）　ＱＥ１１０に対する同時読出しおよび書込みの使用可能
ビット４（Ｗ／Ｒ）　早期アドレス使用可能
ビット５～３１（ＲＯ）　予約。常に０として読出し。
【０２７５】
【表３４】
ポート遮断
ポート遮断　（オフセット＝'ｈ７ｃ）ポートが遮断されるビット・マップ
デフォルト＝３２'ｈ００００＿００００
ビット０～２７（Ｗ／Ｒ）　ポート０ないし２７に対するビット・マップ
１＝ポートが遮断。＃
ビット２８～３１（ＲＯ）　予約。常に０として読出し。
【０２７６】
【表３５】
ポート状態セットアップ
１つのポートの状態をセットアップまたは変更するために、２つのレジスタが書込まれなければならない。書込む第１のレジスタは、変更されるポートのビット・マップを含むポート状態ビット・マップ・レジスタである。書込む第２のレジスタは、状態値を含み、２つのポート状態レジスタのプログラミングを開始するプログラム・ポート状態レジスタである。ＣＰＵのポート状態は、常に前送中であり、決して変更できない。
ポート状態ビット・マップ・レジスタ　（オフセット＝'ｈ９０）その状態が変化するポートのビット・マップ。１＝当該ポート状態をプログラム・ポート状態レジスタにおける値へ変更
デフォルト＝３２'ｈ００００＿００００
ビット０（Ｗ／Ｒ）　ポート０。このビットの設定がポート０の状態の変更を使用可能にする
ビット１（Ｗ／Ｒ）　ポート１。このビットの設定がポート１の状態の変更を使用可能にする
：
：
ビット２７（Ｗ／Ｒ）　ポート２７。このビットの設定がポート２７の状態の変更を使用可能にする
ビット２９～３１（ＲＯ）　予約。常に０として読出し
プログラム・ポート状態レジスタ　（オフセット＝'ｈ８０）ポート状態。ＣＰＵが、ポート状態レジスタのプログラミングを開始する当該レジスタに書込む。当該ポート状態ビット・マップ・レジスタは、「最初に書込まれねばならない」
デフォルト＝３２'ｈ００００＿００００
ビット０～１（Ｗ／Ｒ）　状態値。この値は、オフセット３０におけるビット・マップに示されるポートに置かれる

状態値	条件
００ｂ	使用不能
０１ｂ	ブロック／リスニング
１０ｂ	学習
１１ｂ	前送

ビット２～３１（ＲＯ）　予約。常に０として読出し
ポート状態＃１レジスタ　（オフセット＝'ｈ９４）ポート０ないし１５の状態
プログラム・ポート状態レジスタおよびポート状態ビット・マップ・レジスタによりプログラムされる
デフォルト＝３２'ｈ０００００＿００００
状態値　　条件
００ｂ　　使用不能
０１ｂ　　ブロック／リスニング
１０ｂ　　学習
１１ｂ　　前送
ビット０～１（ＲＯ）　Ｐｏｒｔ＿０＿ｓｔ［１：０］
ビット２～３（ＲＯ）　Ｐｏｒｔ＿１＿ｓｔ［１：０］
ビット４～５（ＲＯ）　Ｐｏｒｔ＿２＿ｓｔ［１：０］
ビット６～７（ＲＯ）　Ｐｏｒｔ＿３＿ｓｔ［１：０］
ビット８～９（ＲＯ）　Ｐｏｒｔ＿４＿ｓｔ［１：０］
ビット１０～１１（ＲＯ）　Ｐｏｒｔ＿５＿ｓｔ［１：０］
ビット１２～１３（ＲＯ）　Ｐｏｒｔ＿６＿ｓｔ［１：０］
ビット１４～１５（ＲＯ）　Ｐｏｒｔ＿７＿ｓｔ［１：０］
ビット１６～１７（ＲＯ）　Ｐｏｒｔ＿８＿ｓｔ［１：０］
ビット１８～１９（ＲＯ）　Ｐｏｒｔ＿９＿ｓｔ［１：０］
ビット２０～２１（ＲＯ）　Ｐｏｒｔ＿１０＿ｓｔ［１：０］
ビット２２～２３（ＲＯ）　Ｐｏｒｔ＿１１＿ｓｔ［１：０］
ビット２４～２５（ＲＯ）　Ｐｏｒｔ＿１２＿ｓｔ［１：０］
ビット２６～２７（ＲＯ）　Ｐｏｒｔ＿１３＿ｓｔ［１：０］
ビット２８～２９（ＲＯ）　Ｐｏｒｔ＿１４＿ｓｔ［１：０］
ビット３０～３１（ＲＯ）　Ｐｏｒｔ＿１５＿ｓｔ［１：０］
ポート状態＃２レジスタ　（オフセット＝'ｈ９８）ポート１６ないし２８の状態。プログラム・ポート状態レジスタおよびポート状態ビット・マップ・レジスタによりプログラムされる。デフォルト＝３２'ｈ０３００＿００００
状態値　　条件
００ｂ　　使用不能
０１ｂ　　ブロック／リスニング
１０ｂ　　学習
１１ｂ　　前送
ビット０～１（ＲＯ）　Ｐｏｒｔ＿１６＿ｓｔ［１：０］
ビット２～３（ＲＯ）　Ｐｏｒｔ＿１７＿ｓｔ［１：０］
ビット４～５（ＲＯ）　Ｐｏｒｔ＿１８＿ｓｔ［１：０］
ビット６～７（ＲＯ）　Ｐｏｒｔ＿１９＿ｓｔ［１：０］
ビット８～９（ＲＯ）　Ｐｏｒｔ＿２０＿ｓｔ［１：０］
ビット１０～１１（ＲＯ）　Ｐｏｒｔ＿２１＿ｓｔ［１：０］
ビット１２～１３（ＲＯ）　Ｐｏｒｔ＿２２＿ｓｔ［１：０］
ビット１４～１５（ＲＯ）　Ｐｏｒｔ＿２３＿ｓｔ［１：０］
ビット１６～１７（ＲＯ）　Ｐｏｒｔ＿２４＿ｓｔ［１：０］
ビット１８～１９（ＲＯ）　Ｐｏｒｔ＿２５＿ｓｔ［１：０］
ビット２０～２１（ＲＯ）　Ｐｏｒｔ＿２６＿ｓｔ［１：０］
ビット２２～２３（ＲＯ）　Ｐｏｒｔ＿２７＿ｓｔ［１：０］
ビット２４～２５（ＲＯ）　Ｐｏｒｔ＿２８＿ｓｔ［１：０］　ＣＰＵポートは常に前送（１１）
ビット２６～３１（ＲＯ）　予約。常に０として読出し
ポート状態セットアップ・レジスタに対するｍｃｂｒｅｇｓインターフェース

ＳｏｕｒｃｅＰｏｒｔ［７：０］（ｉｎ） ｍｃｂｈａｓｈモジュールからのソース・ポート番号
ｍ＿ＨａｓｈＤｓｔｐｒｔ［７：０］（ｉｎ） ｍｃｂｈａｓｈモジュールからの宛先ポート
ＳｒｃＰｒｔＳｔａｔｅ［１：０］（ｏｕｔ） ソース・ポート・レジスタおよびポート状態レジスタに基くｍｃｂｈａｓｈに対する組合わせ出力
ＤｓｔＰｒｔＳｔａｔｅ［１：０］（ｏｕｔ） ｍ＿ＨａｓｈＤｓｔＰｒｔおよびポート状態レジスタに基くｍｃｂｈａｓｈに対する組合わせ出力。
【０２７７】
【表３６】
パケット・メモリ定義
メモリ・セクター情報レジスタ （オフセット＝'ｈａ０）パケット・メモリは、固定数のセクターからなる。このレジスタは、セクター・サイズを定義する。２Ｋバイトの最小セクター・サイズは、最大パケット（１５１８バイト＋オーバーヘッド）が１つ以上のセクター境界の抵触を行い得ないことを保証する。２Ｋバイトの現在唯一つのセクター・サイズがサポートされる。このレジスタは、マスタ・スイッチ使用可能（ＥＰＳＭセットアップ・レジスタ）が否定される時にのみ書込まれる
ビット０～１（Ｗ／Ｒ） セクター・サイズ。２Ｋバイトの現在唯一つのセクターがサポートされる
００＝２Ｋバイト（デフォルト）
０１＝４Ｋバイト
１０＝８Ｋバイト
１１＝１６Ｋバイト
ビット２～３１（ＲＯ） 予約。常に０として読出し。
【０２７８】
【表３７】
メモリ・バス帯域幅モニタ
メモリ・バス・モニタ制御レジスタ （オフセット＝'ｈａ８）当該レジスタにより制御される２つの独立バス・モニタがある。モニタ選択ビット（２４）は、どのモニタがアクセス中であるかを選択するため用いられる。このビットはまた、メモリ・バス・モニタ閾値レジスタとメモリ使用レジスタとに対するアクセスを制御する。モニタ・ビットは、当該レジスタの高いバイトのみを書込むことにより独立的にセットすることができる
ビット０～９（Ｗ／Ｒ） モニタ・モード［９：０］。監視されるべきバスの活動状態のタイプを定義
デフォルトは１０'ｈ３ＦＦ（全てを監視）
ＣｙｃｌｅＴｙｐｅ（１つ以上のビットをセット）ビット０ パケット（パケット関連トラフィックを監視するようセット）
ビット１ ハッシュ（ハッシュ索引トラフィックを監視するようセット）
ビット２ ＣＰＵ（メモリに対するＣＰＵアクセスを監視するようセット）
ビット３ リフレッシュ（リフレッシュ・サイクルを監視するようセット）
パケット・タイプ（パケット・ビット（０）がセットされるならば、一方または両方のビットをセットしなければならない）
ビット４ ユニキャスト（既知の個々のアドレス・ード・パケットを監視するようセット）
ビット５ ブロードキャスト（グループ・ビット・セットまたはハッシュ・ミスを持つパケットを監視するようセット）
パケットＴｘ／Ｒｃ（パケット・ビット（０）がセットされるならば、一方または両方のビットをセットしなければならない）
ビット６ 送信（送信関連トラフィックを監視するようセット）
ビット７ 受信（受信関連トラフィックを監視するようセット）

パケット・データ／オーバーヘッド（パケット・ビット（０）がセットされるならば、一方または両方のビットをセットしなければならない）
ビット８　データ（パケット転送のデータ部分をモニタするようセット）
ビット９　オーバーヘッド（パケット転送の非データ関連部分、即ち、バス調停、パケット・メモリ保守、未使用サイクルをモニタするようセット）
ビット１０～１５（ＲＯ）　予約。常に０を読出す
ビット１６～１９（ＲＯ）　フィルタ・タイム・スケール。ＬＰフィルタリングのための略々時定数をセット
０ｈ＝７５ミリ秒　　４ｈ＝３００ミリ秒　８ｈ＝予約　Ｃｈ＝予約
１ｈ＝６００ミリ秒　５ｈ＝２．５秒　　９ｈ＝予約　Ｄｈ＝予約
２ｈ＝５ミリ秒　　６ｈ＝２０秒　　Ａｈ＝予約　Ｅｈ＝予約
３ｈ＝４０ミリ秒　　７ｈ＝２．５分　　Ｂｈ＝予約　Ｆｈ＝予約
デフォルト＝０ｈ。フィルタ・モードにおいてのみ適用
ビット２０（Ｗ／Ｒ）　カウント／フィルタ・モード。（デフォルト＝０、フィルタ・モード）
０＝モニターが、フィルタ・タイムスケールにより定義されるように低域通過フィルタとして動作
バス使用レジスタの読出しは、フィルタ・モニタにおける値に影響を及ぼさない
１＝モニタ・カウント・バス・サイクル、しかし、フィルタ動作は行わない。カウント・モードにある時、読出し時はバス使用レジスタはクリアされる
ビット２１（Ｗ／Ｒ）　タイマ・モード。カウント・モードにある時のみ適用（デフォルト＝０）
０＝モニター・モード・ビットにより定義されるサイクルのみをカウント
１＝クロック・サイクルごとにカウンタを増分
ビット２２（Ｗ／Ｒ）　バックプレシャ可能化。１＝全てのポートにおけるバックプレシャを使用可能するため当該モニタからのアラーム使用。デフォルト＝０、不使用可能
ビット２３（Ｗ／Ｒ）　ブロードキャスト制御使用可能。１＝任意のポートから受信されたブロードキャスト・パケットを捨てるためモニタからのアラームを使用
デフォルト＝０、不使用可能
ビット２４（Ｗ／Ｒ）　モニタ選択。０＝モニタ０（デフォルト）
１＝モニタ１
ビット２５（ＲＯ）　　予約。常に０を読出し
メモリ・バス・モニター閾値／ＢＷレジスタ（オフセット＝'ｈａｃ）モニタ選択ビットは、当該レジスタのアクセスに先立ちセットされねばならない
ビット０～７（Ｗ／Ｒ）　アラーム設定閾値。バス使用がこの値に達するかあるいはこれを越えるならば、アラーム・フラグがセットされ、ＣＰＵ割込みが生成される。使用可能にされるならば、バックプレシャまたはブロードキャスト制御が適用される（デフォルト＝８'ｈ００）
ビット８～15（Ｗ／Ｒ）　アラーム・クリア閾値。バス使用が、この値まで低減するかあるいはこれより低減する時、アラーム・フラグがクリアされ、ＣＰＵ割込みが生成されるバックプレシャおよびブロードキャストの制御が解放される
（デフォルト＝８'ｈ００）
ビット１６～２３（ＲＯ）ピークＢＷ。最後の読出し以後、最大帯域幅が検出される。読出し時に、クリアされる
ビット２４～３１（ＲＯ）その時のＢＷ。バス帯域幅使用フィルタのその時の値。００ｈ値は、０％の使用を表わし、ＦＦｈの値は１００％の使用を表わす
メモリ・バス使用レジスタ（オフセット＝'ｈｂ０）当該レジスタのアクセスに先立ち、モニタ選択ビットがセットされねばならない
ビット０～３１（ＲＯ）　バス使用［３１：０］。メモリ・バス使用カウンタカウント・

モードでは、カウンタが最後に始動された後は、この値は使用中のバス・サイクル数のカウントである。読出し時にクリアされる。両方の「バス利用レジスタ」が読出された時、両方のフィルタのカウンタが同時に始動する
フィルタ・モードでは、上位８ビットがカウントＢＷとして閾値／ＢＷに複写されるので、このレジスタを読出す必要がない。ＢＷに対して８ビットより多くを使用することが望ましければ、最大域幅値が常に３２'ｈＦＦ００＿００００となり、かつ選択されるタイム・スロットに応じて最小値が
３２'ｈＦＦ００＿００００と３２'ｈ００ＦＦ＿ＦＦＦＦの間になることに注意すべきである
フィルタ・モードで読出された時は、クリアされない。

【０２７９】

【表３８】

メモリ帯域幅モニターに対するｍｃｂｒｅｇｓインターフェース

SelectedBandWidth[31:0](in)　選択されたモニタに対するメモリ・バス利用レジスタ［３１：０］。また、ビット２４～３１が閾値／ＢＷレジスタにおけるその時のＢＷである
SelectedMaxBW[7:0](in)　閾値／ＢＷレジスタ・ビット１６～２３におけるピークＢＷ
Alarm0(in)　モニター０に対するアラーム・フラグ。このフラグがセットされクリアされる時、ｍｃｂｒｅｇｓが割込みＢＷＡＬＡＲＭＳＥＴ０とＢＷＡＬＡＲＭＣＬＲ０を生成する
Alarm1(in)　モニタ１に対するアラーム・フラグ。このフラグがセットされクリアされる時、ｍｃｂｒｅｇｓが割込みＢＷＡＬＡＲＭＳＥＴ１とＢＷＡＬＡＲＭＣＬＲ１を生成する
r_MonMode0[9:0](out)　モニタ０に対するモニタ・モード
r_MonMode1[9:0](out)　モニタ１に対するモニタ・モード
r_BwScale0[2:0](out)　モニタ０に対するフィルタ・タイムスケール
r_BwScale1[2:0](out)　モニタ１に対するフィルタ・タイムスケール
r_CountOnly0(out)　モニタ０に対するカウント／フィルタ・モード
r_CountOnly1(out)　モニタ１に対するカウント／フィルタ・モード
r_TimerMode0(out)　モニタ０に対するタイマ・モード
r_TimerMode1(out)　モニタ１に対するタイマ・モード
r_BackPresOnAlarm0(o)　モニタ０に対するバックプレシャ使用可能
r_BackPresOnAlarm1(o)　モニタ１に対するバックプレシャ使用可能
r_BackBcPktsOnAlarm0(o)　モニタ０に対するブロードキャスト制御使用可能ビット
r_DropBcPktsOnAlarm1(o)　モニタ１に対するブロードキャスト制御使用可能ビット
r_FilterSelect(out)　モニタ選択ビット
r_AlarmSet0[7:0](out)　モニタ０に対するアラーム・セット閾値
r_AlarmSet1[7:0](out)　モニタ１に対するアラーム・セット閾値
r_AlarmClr[7:0](out)　モニタ１に対するアラーム・クリア閾値
ClrBwCtr0(out)　モニタ０に対する利用レジスタが読出される時１クロックに対してアサート
ClrBwCtr1(out)　モニタ１に対する利用レジスタが読出される時１クロックに対してアサート
ClrMaxBW0(out)　モニタ０に対する閾値／ＢＷレジスタが読出される時１クロックに対してアサート
ClrMaxBW0(out)　モニタ０に対する閾値／ＢＷレジスタが読出される時１クロックに対してアサート。

【０２８０】

【表３９】

ドロップ・パケット統計

メモリのオーバーフロー、同報オーバーフロー、受信セクター・オーバーフローおよび送信セクター・オーバーフローによりドロップされたパケットがカウントされる。受信セクター・オーバーフローおよび送信セクター・オーバーフローに対するこれらのカウントおよびビット・マップが保持される。これら条件もまた、ＣＰＵ２３０に対する割込みを生じる。割込み情報が、ＭＣＢ割込みソース・レジスタに保持される
ドロップ・パケット・メモリ・オーバーフロー・レジスタ　（オフセット＝′ｈｂ８）このレジスタは、２つの条件により生じるメモリ・オーバーフローによりドロップされたパケット数を含む。これら条件は、パケットが記憶されている時のハッシュ索引および実際のメモリ・オーバーフロー時に閾値を越えさせられ、これが打切りパケットを生じる
ビット０～３１（Ｗ／Ｒ）　メモリ・オーバーフローによりドロップされたパケット数
ドロップ・パケット・ブロードキャスト・オーバーフロー・レジスタ（オフセット＝ｈｂｃ）
このレジスタは、ブロードキャスト閾値オーバーフローによりドロップされたパケット数を含む
ビット０～３１（Ｗ／Ｒ）　ブロードキャスト閾値オーバーフローによりドロップされたパケット数
ドロップ・パケット受信セクタ・オーバーフロー・レジスタ　（オフセット＝′ｈｄ４）
このレジスタは、受信セクタ・オーバーフローにより外されたパケット数を保持する
ビット０～３１（Ｗ／Ｒ）　受信セクタ・オーバーフローにより外されたパケット数
ドロップ・パケット送信セクタ・オーバーフロー・レジスタ　（オフセット＝′ｈｄ８）
このレジスタは、送信セクタ・オーバーフローにより外されたパケット数を保持する
ビット０～３１（Ｗ／Ｒ）　送信セクタ・オーバーフローにより外されたパケット数
ドロップ・パケット受信セクタ・ビット・マップ・レジスタ　（オフセット＝′ｈｄｃ）
このレジスタは、受信セクタ・オーバーフローによりパケットをドロップしたポートのビット・マップを保持する
ビット０～２８（Ｗ／Ｒ）　受信セクタ使用のオーバーフローを通知するポートのビット・マップ
ドロップ・パケット送信セクタ・ビット・マップ・レジスタ　（オフセット＝′ｈｅ０）
このレジスタは、送信セクター・オーバーフローによりパケットをドロップしたポートのビット・マップを保持する
ビット０～２８（Ｗ／Ｒ）　送信セクタ使用のオーバーフローを通知するポートのビット・マップ
ドロップ・パケット統計に対するｍｃｂｒｅｇｓインターフェース
ｘ＿ＲｘＰｋｔＡｂｏｒｔｅｄ＿　メモリ・オーバーフローによりパケットが打切られた時を通知するＸＣＢからストローブ
ＤｒｏｐＰｋｔＳｔｂ＿ＭｅｍＯＦ　メモリをオーバーフローするのでパケットが外された時を通知するストローブ
ＤｒｏｐＰｋｔＳｔｂ＿ＢＣＯＦ　ブロードキャスト閾値がオーバーフローするのでパケットが打切られた時を通知するストローブ
ＤｒｏｐＰｋｔＳｔｂ＿ＲｘＯＦ　受信セクター閾値がオーバーフローするのでパケットがドロップされた時を通知するストローブ
ＤｒｏｐＰｋｔＳｔｂ＿ＴｘＯＦ　送信セクタ閾値がオーバーフローするのでパケットがドロップされた時を通知するストローブ。
【０２８１】
【表４０】
ハッシュ・テーブル定義
ハッシュ・テーブル定義レジスタ　（オフセット＝′ｈｃ０）主要ハッシュ・エントリ・テーブルの基底アドレスおよびサイズを定義。ハッシュ・テーブルの多重コピーがメモリに保持されるならば、ＥＰＳＭ２１０スイッチを間に持つようにこのレジスタが使用されるビット０～１４（ＲＯ）　主要ハッシュならば、テーブル基底アドレス［１６：２］。

常に０
ビット１５～２３（ＲＯ）主要ハッシュ・テーブル基底アドレス［２５：１７］常に０
ビット２４～２５（Ｗ／Ｒ）主要ハッシュ・テーブル・サイズ［１：０］。（デフォルトは００）
００＝キー・サイズ１３ビット。テーブル・サイズ１
２８Ｋバイト（８Ｋ　１６バイト・エントリ）
０１＝キー・サイズ１４ビット。テーブル・サイズ２
５６Ｋバイト（１６Ｋ　１６バイト・エントリ）
（基底アドレス・ビット１７が無視され、内部で０に強制される）
１０＝キー・サイズ１５ビット。テーブル・サイズ５
１２Ｋバイト（３２Ｋ　１６バイト・エントリ）
（基底アドレス・ビット１８：１７が無視され、内部で０に強制される）
１１＝キー・サイズ１６ビット。テーブル・サイズ１
メガバイト（６４Ｋ　１６バイト・エントリ）
（基底アドレス・ビット１９：１７が無視され、内部で０に強制される）
ビット２６（Ｗ／Ｒ）　ハッシュ・サイクル・ロック。このビットのセッティングが、ハッシュ索引中のメモリ・サイクルをロックさせる。これは、メモリに対するパケット読出しおよび書込み転送を遅らせることを代償に、ハッシュ索引時間を最短化する。デフォルトは０
ビット３１：２７（ＲＯ）　予約。常に０として読出し
ハッシュ・テーブル定義レジスタに対するｍｃｂｒｅｇｓインターフェース
r_HashBaseAdr[25:17](out)　基底アドレスをｍｅｍｈａｓｈモジュールへ通過
r_HashKeySize[1:0](out)　キー・サイズをｍｅｍｈａｓｈモジュールへ通過
r_LockHashCycs(out)　ロック・ハッシュ・サイクル・ビットがセットされるならば、ｍｃｂｈａｓｈモジュールへアサート
HashLookUpIP(in)　ハッシュ索引が進行中でありハッシュ・テーブル定義レジスタに対する書込みが無視されるまで延期されるべきことを表示するためｍｃｂｈａｓｈモジュールによりアサート。ｍｃｂｈａｓｈが、ＨａｓｈＬｏｏｋＵｐＩＰが否定されると任意の立上がりクロック・エッジでレジスタを更新。
【０２８２】
【表４１】
ソース・ポート学習
ハッシュ・ソース・ミス・レジスタ・ロー　（オフセット＝'ｈｃｃ）ハッシュ・テーブルに付加される新しいソース・アドレスのバイト３：０。これらレジスタがロードされ、ハッシュＳＡが未知であるか、あるいはポートが変化し、かつソース・ポートが学習不使用にされる時に、割込みが発される。レジスタは、ハッシュ・ソース・ミス・レジスタ・ハイのレジスタが読出されるまでロックされる（ローのレジスタが最初に読出されねばならない）。レジスタがロックされる間遭遇された未知のＳＡまたはポート変化は否定される。
ビット０～７（ＲＯ）　ハッシュ・テーブルに追加されるべきＭＡＣアドレスのバイト０（高次のアドレス・バイト。グループ・ビット＝ビット０）
ビット８～１５（ＲＯ）ハッシュ・テーブルに追加されるべきＭＡＣアドレスのバイト１
ビット１６～２３（ＲＯ）ハッシュ・テーブルに追加されるべきＭＡＣアドレスのバイト２
ビット２４～３１（ＲＯ）ハッシュ・テーブルに追加されるべきＭＡＣアドレスのバイト３
ハッシュ・ソース・ミス・レジスタ・ハイ（オフセット＝'ｈｄ０）新たなソース・アドレスとソース・ポートＩＤのバイト５：４
ビット０～７（ＲＯ）　ハッシュ・テーブルに追加されるべきＭＡＣアドレスのバイト４

ビット８～１５（ＲＯ）ハッシュ・テーブルに追加されるべきＭＡＣアドレスの
バイト５
ビット１６～２３（ＲＯ）ハッシュ・テーブルへ追加されるべきソース・ポート
ＩＤ［７：０］
ビット２４～３１（ＲＯ）予約。常に０として読出し
学習不能ポート・レジスタ　（オフセット＝'ｈｅ４）ビット・マップされた学習不能ポート・レジスタ。ＣＰＵには適用しない
ビット０（Ｗ／Ｒ）　ポート０学習不能。１＝使用不能。デフォルト＝０
ビット１（Ｗ／Ｒ）　ポート１学習不能。１＝使用不能。デフォルト＝０
．．．
ビット２８（Ｗ／Ｒ）ポート２８学習不能。１＝使用不能。デフォルト＝０
ビット２９～３１（Ｗ／Ｒ）予約。常に０として読出し
ソース・ポート学習に対するｍｃｂｒｅｇｓインターフェース
SelectedAdr[47:0](in)　ｍｅｍｈａｓｈモジュールからのソース・アドレス
SourcePort[47:0](in)　ｍｅｍｈａｓｈモジュールからのソース・ポート
SrcMissStb(in)　ハッシュＳＡミスが生じた時ｍｅｍｈａｓｈモジュールによりアサートされ、ソース・ミス・レジスタおよびソース・ポートが妥当する。ハッシュ・ソース・ミス・レジスタに対してゲートとして使用されるならば、ｍｅｍｈａｓｈは保持時間を保証する
SrcMissLock(out)　学習不能がポートに対してセットされたかどうかアサートする。これは、ソース・ポート入力と学習不能レジスタに基くｍｅｍｈａｓｈに対する組合せ出力であり、連続的に評価される。ｍｅｍｈａｓｈはサンプルする時を知る。
【０２８３】
【表４２】
ポート・セキュリティ
ソース・ポート・レジスタ　（オフセット＝'ｈｅ８）ビット・マップされたソース・ポート・レジスタ。（セキュリティが使用可能にされたポートに対して学習不能ビットをセットすることも望ましい）
ビット０（Ｗ／Ｒ）　ポート０セキュリティ可能。１＝使用可能
デフォルト＝０
ビット１（Ｗ／Ｒ）　ポート１セキュリティ可能。１＝使用可能
デフォルト＝０
．．．
ビット２８（Ｗ／Ｒ）ポート２８セキュリティ使用可能。１＝使用可能
デフォルト＝０
ビット２９～３１（ＲＯ）予約。常に０として読出し
セキュリティ違反レジスタ　（オフセット＝'ｈｆ０）ポートによりビット・マップされたセキュリティ違反。読出し時にクリア。０に初期設定。最初のビットがセットされる時割込みが発され、読出される時クリアされる
ビット０（ＲＯ）　セキュリティ違反ポート０。１＝違反の発生
ビット１（ＲＯ）　セキュリティ違反ポート１。１＝違反の発生
．．．
ビット２８（ＲＯ）セキュリティ違反ポート２８。１＝違反の発生
ビット２９～３１（ＲＯ）予約。常に０として読出し
セキュリティ違反統計レジスタ　（オフセット＝'ｈｅｃ）全てのポートにおける全セキュリティ違反のカウント。読出し時にクリア。０に初期設定
ビット０～３１（ＲＯ）セキュリティ違反カウント［３１：０］
ポートセキュリティに対するｍｃｂｒｅｇｓインターフェース
SourcePort[7:0](in)　ｍｅｍｈａｓｈモジュールからのソース・ポート番号
SecurePort(out)　セキュリティ・モードがポートに対してセットされるかどうかア

サート。これは、ＳｅｃｕｒｅＰｏｒｔ入力およびソース・ポート・レジスタに基くｍｅｍｈａｓｈに対する組合わせ出力であり、連続的に評価される。ｍｅｍｈａｓｈはサンプルする時を知る
SecViolationStb(in)　セキュリティ違反が表示されたソース・ポートに生じたことを示すストローブ。ソース・ポートにより示されるセキュリティ違反レジスタに対してゲートとして用いられるならば、ｍｅｍｈａｓｈが保持時間を保証する。
【０２８４】
【表４３】
メモリ・コンフィギュレーション
メモリ制御レジスタ　（オフセット＝'ｈｆ４）種々のメモリ制御機能。マスタ・スイッチ使用可能（ＥＰＳＭセットアップ・レジスタ）が否定される時に、このレジスタのみが書込まれる
ビット０～１（Ｗ／Ｒ）　メモリ・タイプ
００＝高速ページ・モードＤＲＡＭ（デフォルト）
０１＝ＥＤＯ　ＤＲＡＭ
１０＝シンクロナスＤＲＡＭ
１１＝留保
ビット２（Ｗ／Ｒ）　メモリ速度（０＝６０ｎｓ、１＝５０ｎｓ）
デフォルトは０ビット３（Ｗ／Ｒ）
ＥＤＯテスト・モード（１＝可能化）。デフォルトは０
ビット４（Ｗ／Ｒ）　ダブル・リンク・モード。デフォルトは０
ビット５（Ｗ／Ｒ）　使用不能受信ページ・ヒット。デフォルトは０
ビット６（Ｗ／Ｒ）　使用不能送信ページ・ヒット。デフォルトは０
ビット７～３１（ＲＯ）予約。常に０として読出し
メモリ制御レジスタに対するｍｃｂｒｅｇｓインターフェース
r_MemEDO(out)　メモリ・タイプが０１ならば、ｍｅｍｃｔｌモジュールに対してｍｃｂｒｅｇｓによりアサート
r_MemSync(out) メモリ・タイプが１０ならば、ｍｅｍｃｔｌモジュールに対してｍｃｂｒｅｇｓによりアサート
r_Mem50ns(out) メモリ速度ビットが１ならば、ｍｅｍｃｔｌモジュールに対してｍｃｂｒｅｇｓによりアサート
r_TestForEDO(out)ＥＤＯテスト・モードが１ならば、ｍｅｍｃｔｌモジュールに対してｍｃｂｒｅｇｓによりアサート
ビット０～１（Ｗ／Ｒ）　００００００００ｈ－０３ＦＦＦＦＦＦｈに対するＲＡＳ選択（４Ｍ）
ビット２～３（Ｗ／Ｒ）　０４０００００ｈ－０７ＦＦＦＦＦＦｈに対するＲＡＳ選択（８Ｍ）
ビット４～５（Ｗ／Ｒ）　０８０００００ｈ－０ＢＦＦＦＦＦＦｈに対するＲＡＳ選択（１２Ｍ）
ビット６～８（Ｗ／Ｒ）　０Ｃ０００００ｈ－０ＦＦＦＦＦＦＦｈに対するＲＡＳ選択（１６Ｍ）
．．．
ビット３０～３１（Ｗ／Ｒ）３Ｃ０００００ｈ－３ＦＦＦＦＦＦＦｈに対するＲＡＳ選択（６４Ｍ）
ＲＡＳ選択は次のようにコード化される。即ち、００＝ＲＡＳ０、０１＝ＲＡＳ１、０＝ＲＡＳ２、１１＝ＲＡＳ３。デフォルトは常に０
メモリＲＡＳ選択レジスタに対するｍｃｂｒｅｇｓインターフェース
r_RasSelReg[31:0](out)　データをｍｃｂｒｅｇｓからｍｅｍｃｔｌモジュールへ送る
メモリ・リフレッシュ・カウント・レジスタ　（オフセット＝'ｈｆｃ）リフレッシュ要求間のＣＬＫサイクル数を定義

ビット０～９（Ｗ／Ｒ）　リフレッシュ・カウント［９：０］。リフレッシュ・カウント×ＣＬＫ周期は１５．６２５ミリ秒より小さいかこれと等しくなければならない。デフォルトは２０．８ｈ。（３０ｎｓＣＬＫに対しては、１５．６０ミリ秒）
ビット１０～３１（ＲＯ）予約。常に０として読出し
メモリ・リフレッシュ・カウント・レジスタに対するｍｃｂｒｅｇｓインターフェース
RefReq(out)　ｍｅｍｃｔｌモジュールに対するリフレッシュ要求ストローブ。ｍｅｍｃｔｌが正のエッジにおける要求を検出するので、ストローブは任意の長さでよい。確認は返さない。
【０２８５】
【表４４】
ＭＡＣアドレス・フィルタリング
ＣＰＵ２３０に出入りさせるようパケットを指向するため、宛先アドレスに基くフィルタリングが行われる。その時２つのみに対する要求が存在するが、４つのフィルタが設けられる。その時要求がなくとも、アドレス比較に「ドント・ケア」を含むマスキングが得られる。１つがＣＰＵ２３０の個々のアドレスを持ち他が（スパニング・ツリーに対する）ＢＰＤＵ多重キャスト・アドレスを持つ２つのフィルタをセットアップすべきである。ＣＰＵ２３０でないポートからの受信パケットがフィルタ・アドレスにヒットするならば、（ＢＣまたはＭＣであっても）パケットはＣＰＵ２３０のみへ送られる。ＣＰＵ２３０から起生したパケットがフィルタ・アドレス（ＢＰＤＵアドレス）にヒットするならば、このパケットはフィルタ・アドレス・レジスタに指定された宛先ポートへ送られる。パケットがフィルタ・アドレスにヒットするならば、ハッシュ・テーブルの宛先索引が迂回される
フィルタ制御レジスタ　（オフセット＝'ｈ１００）ＭＡＣ宛先アドレス・フィルタリングを制御
ビット０～３（Ｗ／Ｒ）　フィルタ使用可能アドレス［３：０］。１＝対応するアドレス・フィルタ・レジスタ［３：０］に対する個々の宛先アドレス・フィルタリング使用可能。デフォルトは０
ビット４～７（Ｗ／Ｒ）　アドレス・マスク使用可能［３：０］。１＝アドレス・フィルタ・レジスタ［３：０］がアドレス・フィルタ・マスク・レジスタを持つならば、マスキング使用可能。デフォルトは０
フィルタ・マスク・レジスタ・ロー　（オフセット＝'ｈ１０４）デフォルト＝０
ビット０～７（Ｗ／Ｒ）　ＭＡＣアドレス・マスクのバイト０（１＝マスク・アドレス・ビット）
ビット８～１５（Ｗ／Ｒ）ＭＡＣアドレス・マスクのバイト１（１＝マスク・アドレス・ビット）
ビット１６～２３（Ｗ／Ｒ）ＭＡＣアドレス・マスクのバイト２（１＝マスク・アドレス・ビット）
ビット２４～３１（Ｗ／Ｒ）ＭＡＣアドレス・マスクのバイト３（１＝マスク・アドレス・ビット）
フィルタ・マスク・レジスタ・ハイ　（オフセット＝'ｈ１０８）デフォルト＝０
ビット０～７（Ｗ／Ｒ）　ＭＡＣアドレス・マスクのバイト４（１＝マスク・アドレス・ビット）
ビット８～１５（Ｗ／Ｒ）ＭＡＣアドレス・マスクのバイト５（１＝マスク・アドレス・ビット）
ビット１６～３１（ＲＯ）予約。常に０として読出し
フィルタ・アドレス・レジスタ０ロー　（オフセット＝'ｈ１０ｃ）
ビット０～７（Ｗ／Ｒ）　前送されるＭＡＣアドレスのバイト０
ビット８～１５（Ｗ／Ｒ）前送されるＭＡＣアドレスのバイト１
ビット１６～２３（Ｗ／Ｒ）前送されるＭＡＣアドレスのバイト２
ビット２４～３１（Ｗ／Ｒ）前送されるＭＡＣアドレスのバイト３

フィルタ・アドレス・レジスタ０ハイ　（オフセット＝'ｈ１１０）
ビット０～７（Ｗ／Ｒ）　送られるＭＡＣアドレスのバイト４
ビット８～１５（Ｗ／Ｒ）　送られるＭＡＣアドレスのバイト５
ビット１６～２３（Ｗ／Ｒ）宛先ポート。ソース・ポートがＣＰＵ２３０ならば、ＭＡＣアドレスがフィールド・アドレスに一致するならば、このフィールドがパケットをどのポートへ送るべきかを指定する。ソース・ポートがＣＰＵ２３０でなければ、このフィールドが無視されて、フィールドＭＡＣアドレスに対するヒットがＣＰＵ２３０へ送られる
ビット２４～３１（Ｗ／Ｒ）予約。常に０として読出し
フィルタ・アドレス・レジスタ１ロー　（オフセット＝'ｈ１１４）前参照
フィルタ・アドレス・レジスタ１ハイ　（オフセット＝'ｈ１１８）前参照
フィルタ・アドレス・レジスタ２ロー　（オフセット＝'ｈ１１ｃ）前参照
フィルタ・アドレス・レジスタ２ハイ　（オフセット＝'ｈ１２０）前参照
フィルタ・アドレス・レジスタ３ロー　（オフセット＝'ｈ１２４）前参照
フィルタ・アドレス・レジスタ３ハイ　（オフセット＝'ｈ１２８）前参照
アドレス・フィルタリングに対するｍｃｂｒｅｇｓインターフェース
SelectedAdr[47: 0](in)　ｍｅｍｈａｓｈモジュールからの宛先アドレス
filterHit(out)　　フィルタ・アドレス・ヒットが生じるならば、アサートこれは、SelectedAdrおよびフィルタ・レジスタに基くｍｅｍｈａｓｈに対する組合せ出力であり、連続的に評価される。ｍｅｍｈａｓｈはサンプルする時を知る
FilterPort[7:0](out)　ソース・ポートがＣＰＵ２３０であれば、ＦｉｌｔｅｒＰｏｒｔは、フィルタ・ヒットを生成するフィルタ・レジスタからの宛て先ポート・フィールドに等しい。ソース・ポートがＣＰＵ２３０でなければ、ＦｉｌｔｅｒＰｏｒｔは（ＥＰＳＭセットアップ・レジスタからの）ＣｐｕＰｏｒｔに等しい
SourcePort[7:0](in)　ｍｅｍｈａｓｈモジュールからのソース・ポート番号
SrcPrtIsCpu　　ＳｏｕｒｃｅＰｏｒｔ入力がＥＰＳＭセットアップ・レジスタにおけるＣｐｕＰｏｒｔ番号と整合するならば、アサートされる。
【０２８６】
【表４５】
ＭＣＢ割込み情報
ＭＣＢ４０４には８つの割込みソースがある。割込みソースは、ソースがマスクされなければ、ＣＰＵ２３０を割込みさせる。ＣＰＵ２３０が割込みされずに割込みソースの情報を得ることを可能にするため、ポーリング機構が使用可能である。割込みソースのマスキングは、割込みをＣＰＵ２３０からブロックさせるが、情報はポーリング・ソース・レジスタから依然として得られる。
ＭＣＢ割込みソース・レジスタ　（オフセット＝'ｈ１２ｃ）ＣＰＵ２３０へ送られる割込みのソース。このレジスタは、ＥＰＳＭ２１０により更新され、割込みがＣＰＵ２３０へ送られる。ＣＰＵ２３０がこのレジスタを読出す時、内容はクリアされる。ビットにおける１の値は、割込みが生じたことを示す。デフォルト＝３２'ｈ００００＿００００
ビット０（Ｗ／Ｒ）　セキュリティ割込み。セキュリティ違反が生じると、この割込みが生じる
ビット１（Ｗ／Ｒ）　メモリ・オーバーフロー・セット。メモリがパケットで一杯になりオーバーフロー閾値が送られると、この割込みが生じる
ビット２（Ｗ／Ｒ）　メモリ・オーバーフロー・クリア。メモリが空になりオーバーフロー閾値が送られると、この割込みが生じる
ビット３（Ｗ／Ｒ）　セットのブロードキャスト。ブロードキャスト・パケットがメモリを一杯にし、ブロードキャスト閾値が送られると、この割込みが生じる
ビット４（Ｗ／Ｒ）　クリアのブロードキャスト。ブロードキャスト・パケットがメモリから空になり、ブロードキャスト閾値が送られると、この割込みが生じる
ビット５（Ｗ／Ｒ）　ＯＦの受取り。ポートがパケットを受取るためのその割付けスペースを越えると、この割込みが生じる

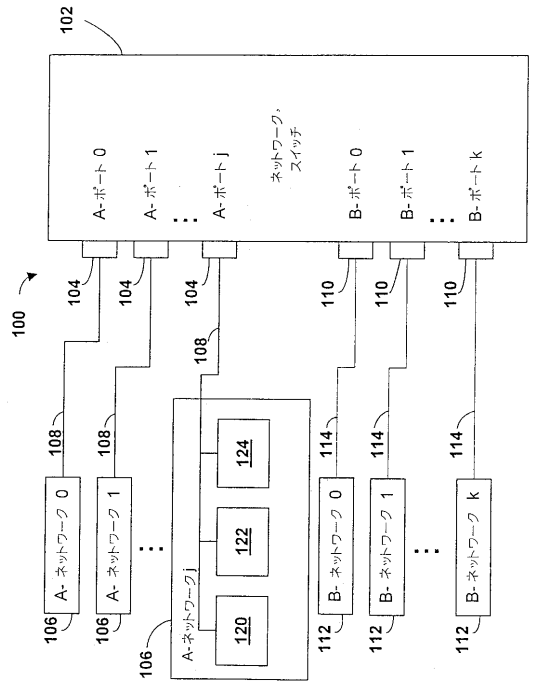
ビット６（Ｗ／Ｒ）　ＯＦの送出。パケットを送信しているポートがその割付けスペースを越えると、この割込みが生じる
ビット７（Ｗ／Ｒ）　Ｒｘパケット打切り。パケットが記憶され始め、メモリが超過すると判定されると、パケットが打切られ、この割込みが生じる
ビット８～３１（ＲＯ）予約。常に０として読出し
割込みソース・レジスタに対するｍｃｂｒｅｇｓインターフェース
割込みマスク・レジスタ　（オフセット＝'ｈ１３０）ＣＰＵ２３０によりマスクされる割込み。任意のビットにおける１の値は、割込みがマスクされることを示す。デフォルト＝３２'ｈ０００００＿０００００
ビット０（Ｗ／Ｒ）　セキュリティ割込みに対するマスク
ビット１（Ｗ／Ｒ）　メモリ・オーバーフロー・セット割込みに対するマスク
ビット２（Ｗ／Ｒ）　メモリ・オーバーフロー・クリア割込みに対するマスク
ビット３（Ｗ／Ｒ）　同報ＯＦセット割込みに対するマスク
ビット４（Ｗ／Ｒ）　同報ＯＦクリア割込みに対するマスク
ビット５（Ｗ／Ｒ）　受信ＯＦ割込みに対するマスク
ビット６（Ｗ／Ｒ）　送信ＯＦ割込みに対するマスク
ビット７（Ｗ／Ｒ）　Ｒｘパケット打切り割込みに対するマスク
ビット８～３１（Ｗ／Ｒ）予約。常に０として読出し
ポーリング・ソース・レジスタ　（オフセット＝'ｈ１３４）このレジスタは、マスクされた割込み情報を含み、所望のビットをクリアするため、ＣＰＵ２３０が１を書込むことによりクリアされる。これにより、ＣＰＵ２３０が割込みの代わりにポーリングすることを許容する。ＣＰＵは、代わりにポーリングを欲する任意の割込みソースをマスクしなければならない。
ビット０（Ｗ／Ｒ）　セキュリティ割込み。セキュリティ違反が生じるならば、この割込みが生じる
ビット１（Ｗ／Ｒ）　メモリ・オーバーフロー・セット。メモリがパケットで一杯となりオーバーフロー閾値が送られると、この割込みが生じる
ビット２（Ｗ／Ｒ）　メモリ・オーバーフロー・クリア。メモリが空になりオーバーフロー閾値が送られると、この割込みが生じる。
ビット３（Ｗ／Ｒ）　ブロードキャストＯＦセット。ブロードキャスト・パケットがメモリを一杯にしてブロードキャスト閾値が送られと、この割込みが生じる
ビット４（Ｗ／Ｒ）　ブロードキャストＯＦクリア。ブロードキャスト・パケットがメモリから空になりブロードキャスト閾値が送られると、この割込みが生じる
ビット５（Ｗ／Ｒ）　受取りＯＦ。ポートがパケットを受取るその割付けスペースを越えると、この割込みが生じる
ビット６（Ｗ／Ｒ）　送信ＯＦ。パケットを送出しているポートがその割付けスペースを越えると、この割込みが生じる
ビット７（Ｗ／Ｒ）　Ｒｘパケット打切り。パケットが記憶され始め、メモリが越えられると判定されると、パケットが打切られてこの割込みが生じる
ビット８～３１（Ｗ／Ｒ）予約。常に０として読出し
ポーリング・ソース・レジスタに対するｍｃｂｒｅｇｓインターフェース。
【０２８７】
【表４６】
バックプレシャ
バックプレシャ使用可能　（オフセット＝'ｈ１３８）バックプレシャを使用可能にするビット・マップ
ビット０～２３（ＲＯ）　予約。常に０として読出し
ビット２４～２７（Ｗ／Ｒ）ビット・マップ
ビット２８～３１（ＲＯ）　予約。常に０として読出し。
【０２８８】

【表４７】
ポート・ボンディング
２組の結合されたポートがある。従って、どのポートが一緒に結合されるかを通知する２つのレジスタがある。
（註）各レジスタにおける僅かに２ビットがセットされるべきであり、即ち、２つのポートが一緒に結合されるべきである。
結合ポート・セット０　（オフセット＝'ｈ１３ｃ）このビット・マップはどのポートが当該セットにおいて一緒に結合されるかを通知する
ビット０～２７（Ｗ／Ｒ）　セット０に対するビット・マップ
ビット２８～３１（ＲＯ）　予約。常に０として読出し
結合ポート・セット１　（オフセット＝'ｈ１４０）このビット・マップが、どのポートが当該セットにおいて一緒に固定されるかを通知する
ビット０～２７（Ｗ／Ｒ）　セット１に対するビット・マップ
ビット２８～３１（ＲＯ）　予約。常に０として読出し
ＶＬＡＮ
デフォルトＶＬＡＮレジスタ　（オフセット＝'ｈ１４４）
【０２８９】
ネットワーク・スイッチに対する多重ポート・ポーリング・システムが複数のネットワーク・ポートに対する受送信状態を決定するための有効なシステムを提供することが判る。ポーリング・ロジックが、１つ照会信号をアサートして複数の送受信状態信号を受取り、これにより多重ポートの状態を一時に受取る。ポーリング・ロジックが、全てのポートの状態を連続的に追跡するように送受信リストを然るべく更新する。このことが、ソース・ポートからのデータを検索する時と伝送のためポートへデータを提供する時とを決定するためリストを検査する調停および制御ロジックを容易にする。
【０２９０】
本出願の好適な実施例について説明してきたが、本発明の変形及び変更が本発明の技術的思想を変更することなく可能であることは、当業者に明らかであろう。
【図面の簡単な説明】
【図１】本発明によるネットワーク・スイッチを含むネットワーク・システムを示す簡略図である。
【図２】図１のネットワーク・スイッチの更に詳細なブロック図である。
【図３】ネットワーク・スイッチのポートを構成する図２のクワッド・カスケード装置形態を示すブロック図である。
【図４】図３に示した特定のクワッド・カスケード装置の信号を示す図である。
【図５】図３のクワッド・カスケード装置のプロセッサ読出しタイミングを示すタイミング図である。
【図６】図３のクワッド・カスケード装置のプロセッサ書込みタイミングを示すタイミング図である。
【図７】図３のクワッド・カスケード装置のプロセッサ・バースト読出しアクセスを示すタイミング図である。
【図８】図３の各ポートのバッファ状態照会を示す模範的タイミング図である。
【図９】図２のＨＳＢにおける同時読出し書込みサイクルを示すタイミング図である。
【図１０】図２のＨＳＢにおける同時読出し書込みサイクルを実行する手順を示すフロー図である。
【図１１】図２のスイッチ・マネージャを示すブロック図である。
【図１２】図４のバス・コントローラ・ブロックの更に詳細なブロック図である。
【図１３】図５Ａのバス・コントローラ・ブロックのメモリ内のバッファを示す図である。
【図１４】図１２のバス・コントローラ・ブロック内の受信ポーリング状態マシンの動作を示す状態図である。

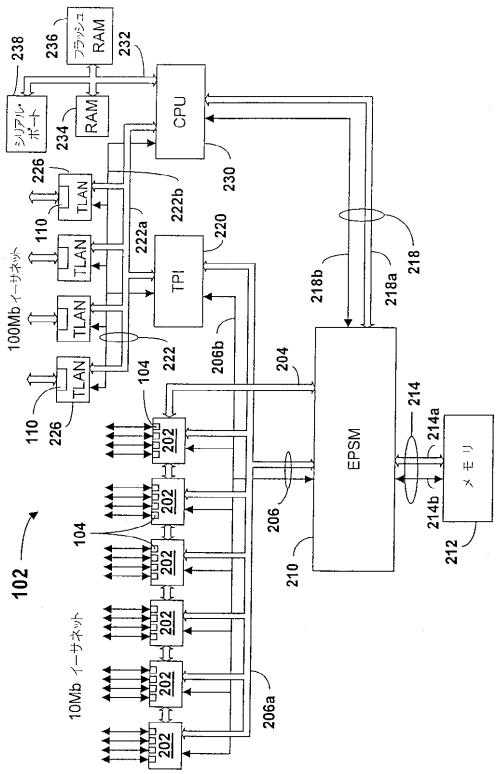
【図１５】図１２のバス・コントローラ・ブロック内の受信ポーリング状態マシンの動作を示す状態図である。
【図１６】図１２のバス・コントローラ・ブロック内の受信ポーリング状態マシンの動作を示す状態図である。
【図１７】図１２のバス・コントローラ・ブロック内の受信ポーリング状態マシンの動作を示す状態図である。
【図１８】図１２のバス・コントローラ・ブロック内の受信ポーリング状態マシンの動作を示す状態図である。
【図１９】図１２のバス・コントローラ・ブロック内の送信ポーリング状態マシンの動作を示す状態図である。
【図２０】図１２のバス・コントローラ・ブロック内の送信ポーリング状態マシンの動作を示す状態図である。
【図２１】図１２のバス・コントローラ・ブロック内の送信ポーリング状態マシンの動作を示す状態図である。
【図２２】図１２のバス・コントローラ・ブロック内の送信ポーリング状態マシンの動作を示す状態図である。
【図２３】図１２のバス・コントローラ・ブロック内の送信ポーリング状態マシンの動作を示す状態図である。
【図２４】図１１のメモリ・コントローラ・ブロックの更に詳細なブロック図である。
【図２５】図１１のプロセッサ・コントローラ・ブロックの更に詳細なブロック図である。
【図２６】図１１のプロセッサ・コントローラ・ブロックの更に詳細なブロック図である。
【図２７】図１１のプロセッサ・コントローラ・ブロックの更に詳細なブロック図である。
【図２８】図１１のプロセッサ・コントローラ・ブロックの更に詳細なブロック図である。
【図２９】図１１のプロセッサ・コントローラ・ブロックの更に詳細なブロック図である。
【図３０】図２のＴｈｕｎｄｅｒＬＡＮポート・インターフェース（ＴＰＩ）を示す簡略ブロック図である。
【図３１】ＴＰＩの更に詳細なブロック図である。
【図３２】図２のＴｈｕｎｄｅｒＬＡＮ（ＴＬＡＮ）の各々の構成と機能とを示すブロック図である。
【図３３】任意のＴＬＡＮにより実行される制御リストの全体フォーマットを示す図である。
【図３４】図２のＰＣＩバスと関連するＴＰＩにより使用されるＴＰＩ周辺要素相互接続（ＰＣＩ）構成レジスタの定義を示す図である。
【図３５】ＴＰＩにより使用されるＴＰＩ制御レジスタの定義を示す図である。
【図３６】図２のＣＰＵのＰＣＩ初期設定動作を示すフロー図である。
【図３７】ＴＬＡＮの各々に対する受取り動作を示すフロー図である。
【図３８】図２の高速バス（ＨＳＢ）に跨がる受取りデータ転送動作を示すフロー図である。
【図３９】ＨＳＢに跨がる伝送データ転送動作を示すフロー図である。
【図４０】ＴＬＡＮの各々に対する伝送動作を示すフロー図である。
【図４１】図２のメモリの構成を示すブロック図である。
【図４２】図２のメモリの構成を示すブロック図である。
【図４３】図２のメモリの構成を示すブロック図である。
【図４４】図２のメモリの構成を示すブロック図である。
【図４５】図２のメモリの構成を示すブロック図である。

【図４６】図２のメモリの構成を示すブロック図である。
【図４７】図２のメモリの構成を示すブロック図である。
【図４８】図２のメモリの構成を示すブロック図である。
【図４９】同報パケットを組込んだ幾つかの伝送パケット・リンクを示すブロック図である。
【図５０】図６のスタティック・メモリの構成を示すブロック図である。
【図５１】図６のスタティック・メモリの構成を示すブロック図である。
【図５２】メモリに対するデータ・パケットの受取りと動作のカットスルーモードにおけるデータ・パケットの送出とのための図２のネットワーク・スイッチの全体動作を示すフロー図である。
【図５３】メモリからデータ・パケットを伝送するための図２のネットワーク・スイッチの全体動作を示すフロー図である。
【図５４】図２のスイッチ・マネージャのハッシュ索引動作を示すフロー図である。
【図５５】図２のメモリにおけるハッシュ・テーブル・エントリを探索するためのハッシュ索引手順を示すフロー図である。

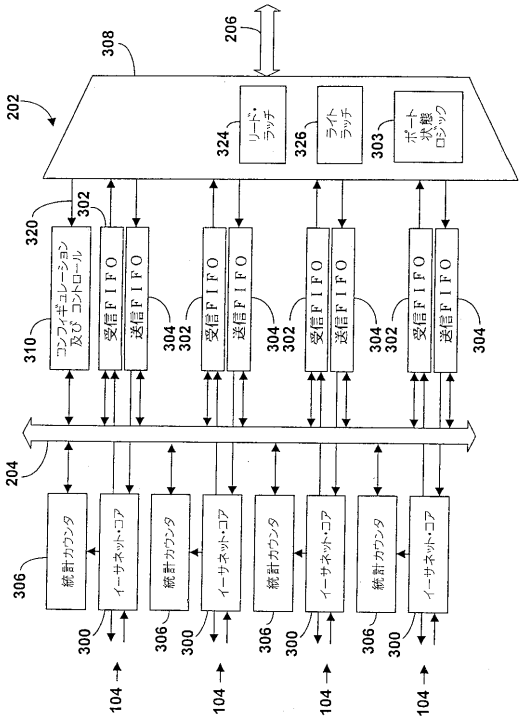
【図１】



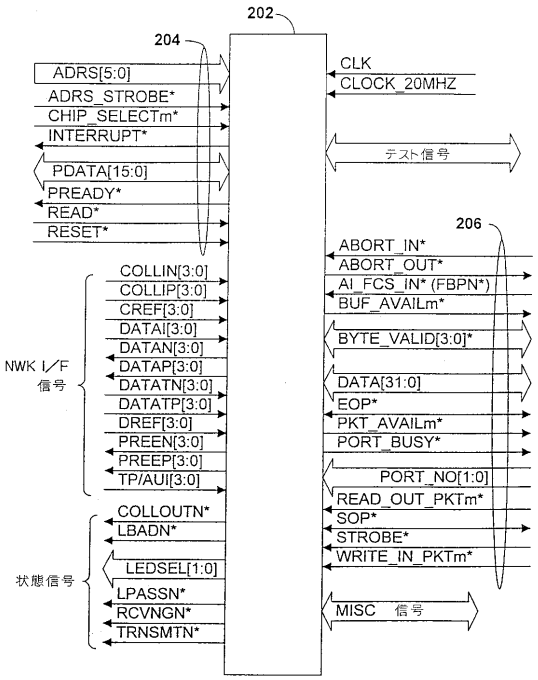
【図２】



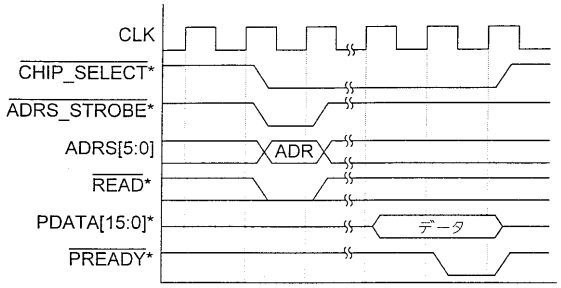
【図３】


206
308
202
324
リード・ラッチ
326
ライト・ラッチ
303
ポート状態ロジック
320
302
コンフィギュレーション及びコントロール
受信ＦＩＦＯ
送信ＦＩＦＯ
310
304
204
306
統計カウンタ
イーサネットコア
300
104

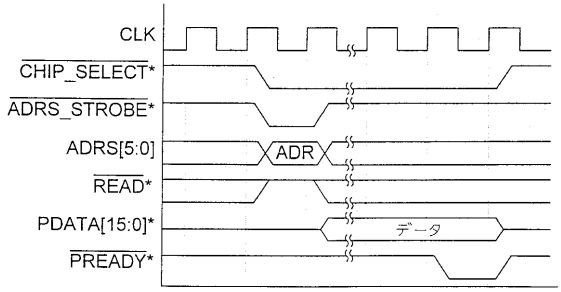
【図４】


202
204
ADRS[5:0]
ADRS_STROBE*
CHIP_SELECTm*
INTERRUPT*
PDATA[15:0]
PREADY*
READ*
RESET*
CLK
CLOCK_20MHZ
テスト信号
206
ABORT_IN*
ABORT_OUT*
AI_FCS_IN* (FBPN*)
BUF_AVAILm*
BYTE_VALID[3:0]*
DATA[31:0]
EOP*
PKT_AVAILm*
PORT_BUSY*
PORT_NO[1:0]
READ_OUT_PKTm*
SOP*
STROBE*
WRITE_IN_PKTm*
MISC 信号
NWK I/F 信号
COLLIN[3:0]
COLLIP[3:0]
CREF[3:0]
DATAI[3:0]
DATAN[3:0]
DATAP[3:0]
DATATN[3:0]
DATATP[3:0]
DREF[3:0]
PREEN[3:0]
PREEP[3:0]
TP/AUI[3:0]
状態信号
COLLOUTN*
LBADN*
LEDSEL[1:0]
LPASSN*
RCVNGN*
TRNSMTN*

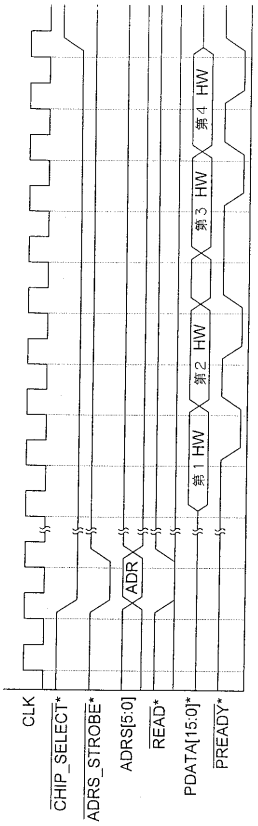
【図５】


CLK
CHIP_SELECT*
ADRS_STROBE*
ADRS[5:0]
ADR
READ*
PDATA[15:0]*
データ
PREADY*

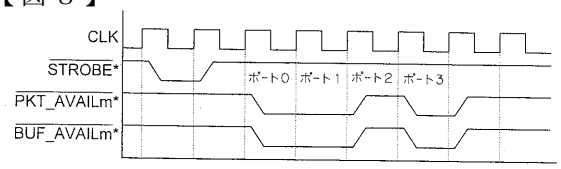
【図６】


CLK
CHIP_SELECT*
ADRS_STROBE*
ADRS[5:0]
ADR
READ*
PDATA[15:0]*
データ
PREADY*

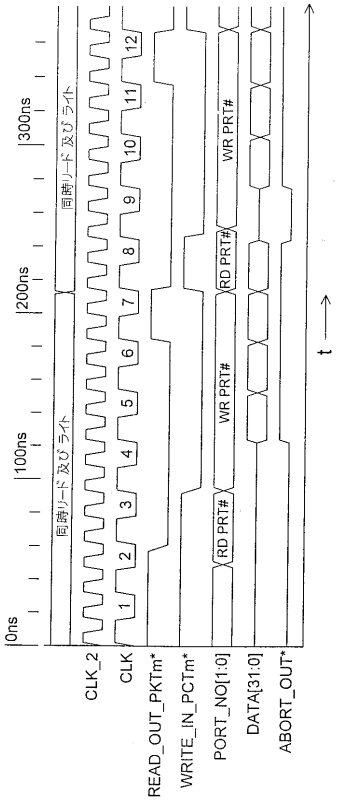
【図７】


第1 HW
第2 HW
第3 HW
第4 HW
ADR
CLK
CHIP_SELECT*
ADRS_STROBE*
ADRS[5:0]
READ*
PDATA[15:0]*
PREADY*

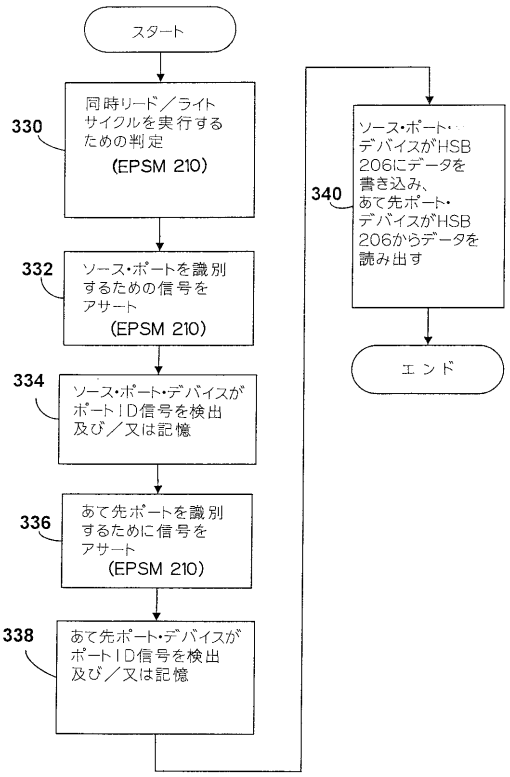
【図８】


CLK
STROBE*
PKT_AVAILm*
BUF_AVAILm*
ポート0
ポート1
ポート2
ポート3

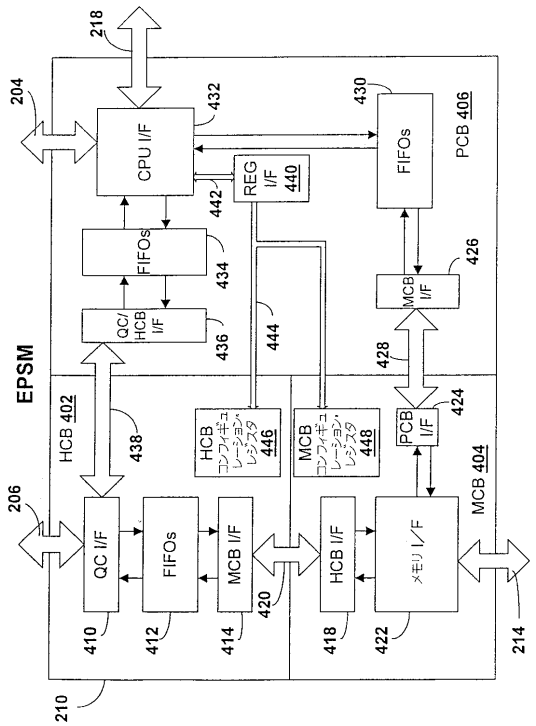
【図９】


0ns
100ns
200ns
300ns
同時リード及びライト
同時リード及びライト
1
2
3
4
5
6
7
8
9
10
11
12
RD PRT#
WR PRT#
RD PRT#
WR PRT#
t
CLK_2
CLK
READ_OUT_PKTm*
WRITE_IN_PCTm*
PORT_NO[1:0]
DATA[31:0]
ABORT_OUT*

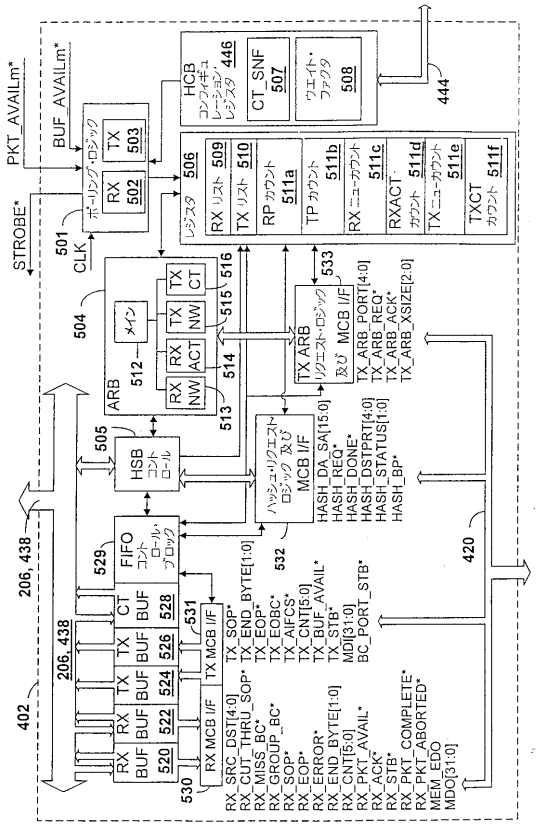
【図１０】


スタート
同時リード／ライトサイクルを実行するための判定
(EPSM 210)
330
ソース・ポートを識別するための信号をアサート
(EPSM 210)
332
ソース・ポート・デバイスがポートID信号を検出及び／又は記憶
334
あて先ポートを識別するために信号をアサート
(EPSM 210)
336
あて先ポート・デバイスがポートID信号を検出及び／又は記憶
338
ソース・ポート・デバイスがHSB 206にデータを書き込み、あて先ポート・デバイスがHSB 206からデータを読み出す
340
エンド

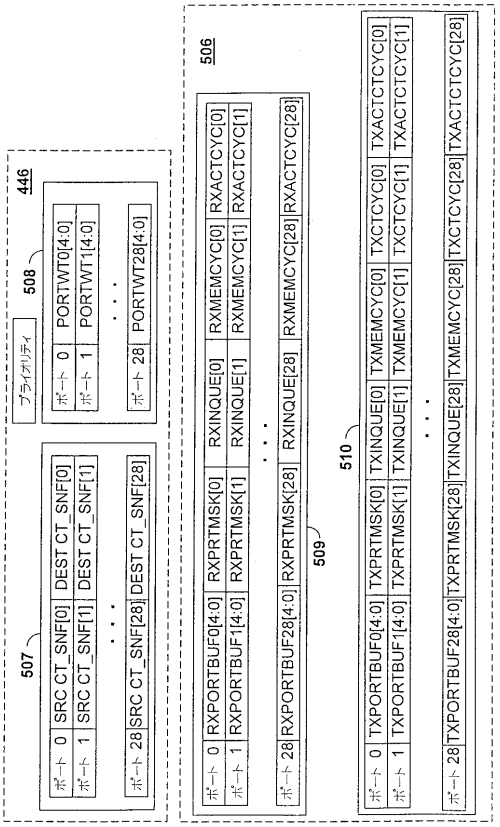
【図１１】


218
204
CPU I/F
432
430
FIFOs
PCB 406
REG I/F 440
442
FIFOs
434
426
MCB I/F
QC/HCB I/F
436
444
428
EPSM
HCB 402
438
HCB コンフィギュレーション・レジスタ 446
MCB コンフィギュレーション・レジスタ 448
PCB I/F
424
206
QC I/F
FIFOs
MCB I/F
HCB I/F
メモリ I/F
MCB 404
410
412
414
420
418
422
214
210

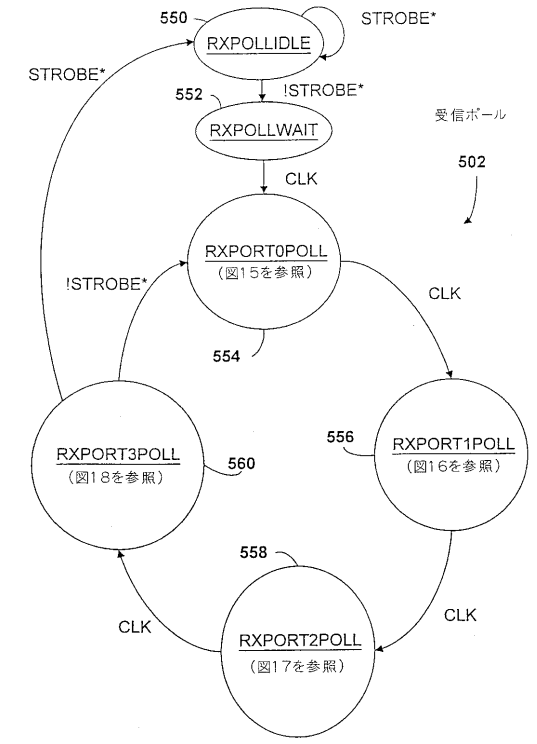
【図１２】



【図１３】



【図１４】



【図１５】

554

```
RXPORT0POLL
IF ((RXNEWCNT+1 != RPCOUNT)
|| (RXACTCNT+1 != RPCOUNT)), THEN
{
1) IF (!PKT_AVAIL*[0] && !RXPRTMSK[0]), THEN
(IF (WTPRIORITY), THEN RXPORTBUF0 = PORTWT[0],
ELSE  RXPORTBUF0 = RPCOUNT;
RXPRTMSK[0]  = 1; RXINCCNTBY[0] = 1);

2) IF (!PKT_AVAIL[1]* && !RXPRTMSK[4]), THEN
(IF (WTPRIORITY), then RXPORTBUF4 = PORTWT[4],
ELSE RXPORTBUF4 = RPCOUNT + RXINCCNTBY[0];
RXPRTMSK[4] = 1; RXINCCNTBY[1] = 1);
. . .
8) IF (!PKT_AVAIL[7]* && !RXPRTMSK[28]), THEN
(IF (WTPRIORITY), THEN RXPORTBUF28 = PORTWT[28],
ELSE RXPORTBUF28 = RPCOUNT +
BITSUM(RXINCCNTBY[6:0]);
RXPRTMSK[28] = 1; RXINCCNTBY[7] = 1);

9) RPCOUNT = RPCOUNT +
BITSUM(RXINCCNTBY[7:0])
}
```

【図１６】

556

```
RXPORT1POLL
IF ((RXNEWCNT+1 != RPCOUNT)
|| (RXACTCNT+1 != RPCOUNT)), THEN
{
1) IF (!PKT_AVAIL[0]* && !RXPRTMSK[1]), THEN
(IF (WTPRIORITY),
THEN RXPORTBUF1 = PORTWT[1],
ELSE RXPORTBUF1 = RPCOUNT;
RXPRTMSK[1] = 1; RXINCCNTBY[0] = 1);
. . .
7) IF (!PKT_AVAIL[6]* && !RXPRTMSK[25]), THEN
(IF (WTPRIORITY),
THEN RXPORTBUF25 = PORTWT[25],
ELSE RXPORTBUF25 =
RPCOUNT + BITSUM(RXINCCNTBY[5:0]);
RXPRTMSK[25] = 1; RXINCCNTBY[6] = 1);

8) (SAME EQUATION 8 AS IN STATE 554);
9) RPCOUNT = RPCOUNT +
BITSUM(RXINCCNTBY[6:0])
}
```

【図１７】

558

```
RXPORT2POLL
IF ((RXNEWCNT+1 != RPCOUNT)
|| (RXACTCNT+1 != RPCOUNT)), THEN
{
1) IF (!PKT_AVAIL[0]* && !RXPRTMSK[2]), THEN
(IF (WTPRIORITY),
THEN RXPORTBUF2 = PORTWT[2],
ELSE RXPORTBUF2 = RPCOUNT;
RXPRTMSK[2] = 1; RXINCCNTBY[0] = 1);
. . .
7) IF (!PKT_AVAIL[6]* && !RXPRTMSK[26]), THEN
(IF (WTPRIORITY),
THEN RXPORTBUF26 = PORTWT[26],
ELSE RXPORTBUF26 = RPCOUNT +
BITSUM(RXINCCNTBY[5:0]);
RXPRTMSK[26] = 1; RXINCCNTBY[6] = 1);

8) (SAME EQUATION 8 AS IN STATE 554);
9) RPCOUNT = RPCOUNT +
BITSUM(RXINCCNTBY[6:0])
}
```

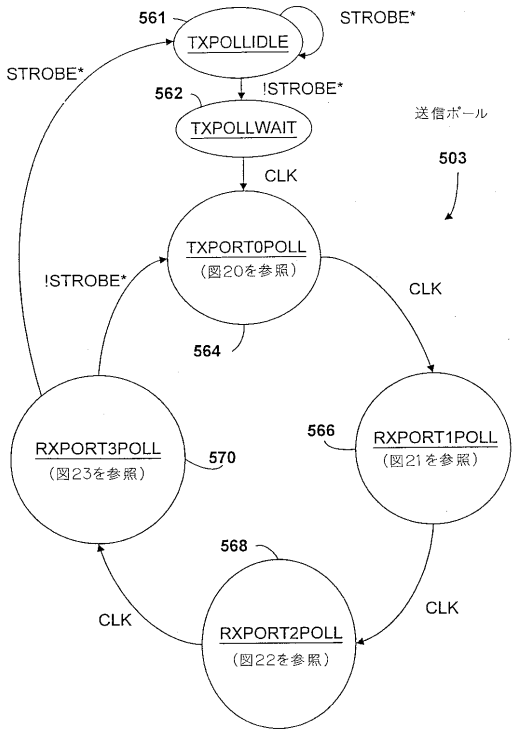
【図１８】

560

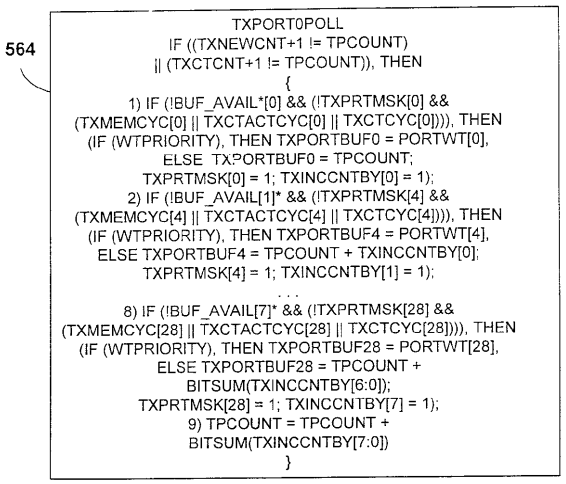
```
RXPORT3POLL
IF ((RXNEWCNT+1 != RPCOUNT)
|| (RXACTCNT+1 != RPCOUNT)), THEN
{
1) IF (!PKT_AVAIL[0]* && !RXPRTMSK[3]), THEN
(IF (WTPRIORITY),
THEN RXPORTBUF3 = PORTWT[3],
ELSE RXPORTBUF3 = RPCOUNT;
RXPRTMSK[3] = 1; RXINCCNTBY[0] = 1);
. . .
7) IF (!PKT_AVAIL[6]* && !RXPRTMSK[27]), THEN
(IF (WTPRIORITY),
THEN RXPORTBUF27 = PORTWT[27],
ELSE RXPORTBUF27 = RPCOUNT +
BITSUM(RXINCCNTBY[5:0]);
RXPRTMSK[27] = 1; RXINCCNTBY[6] = 1);

8) (SAME EQUATION 8 AS IN STATE 554);
9) RPCOUNT = RPCOUNT +
BITSUM(RXINCCNTBY[6:0])
}
```

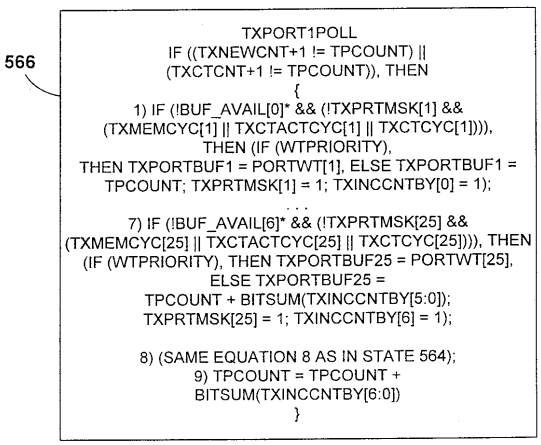
【図１９】



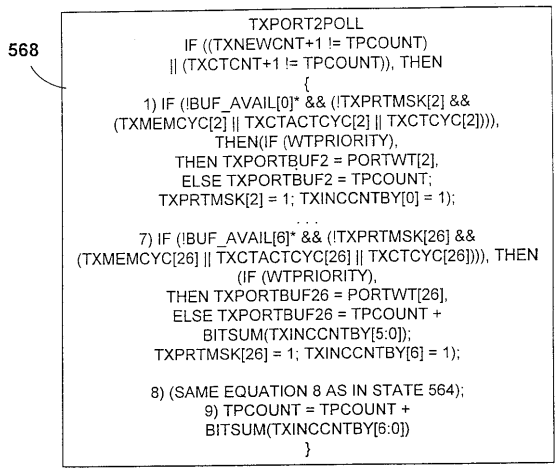
【図２０】



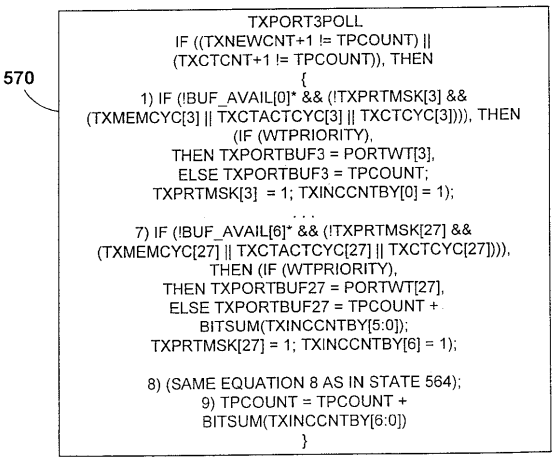
【図２１】



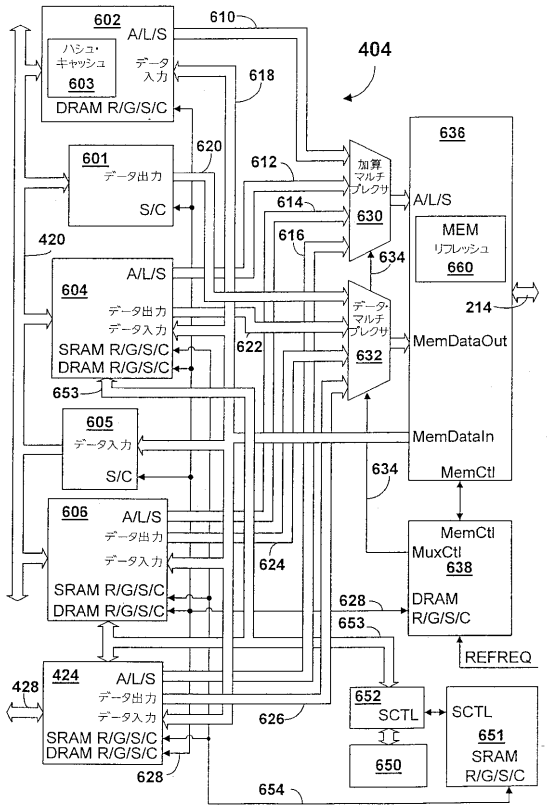
【図２２】



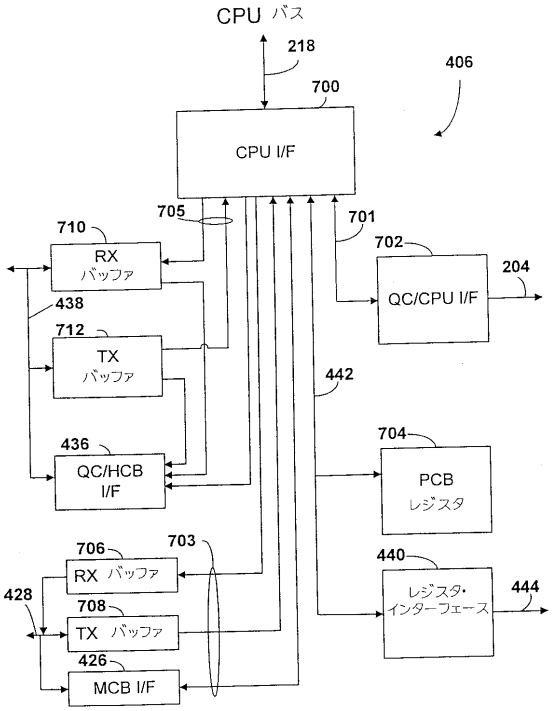
【図２３】



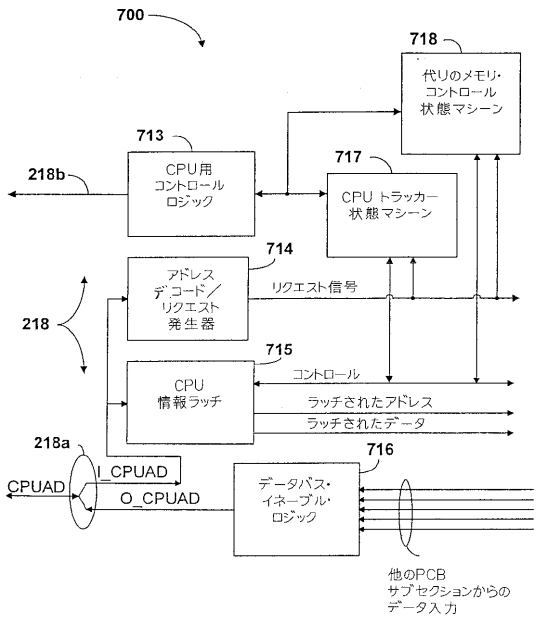
【図２４】



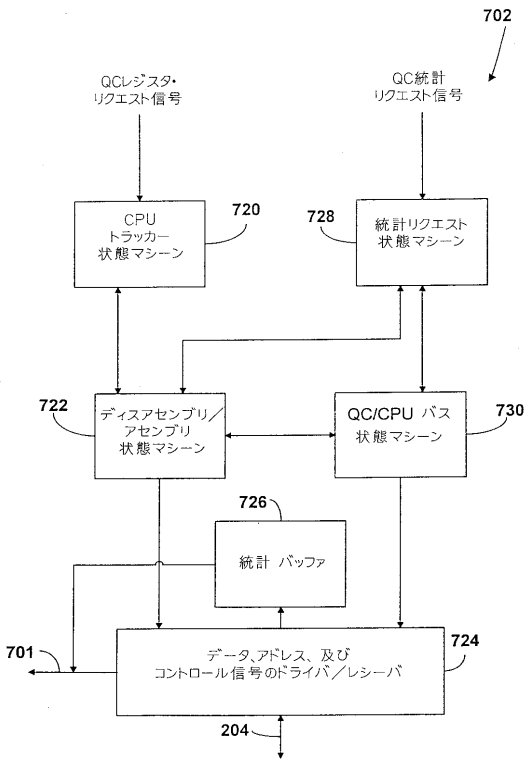
【図２５】



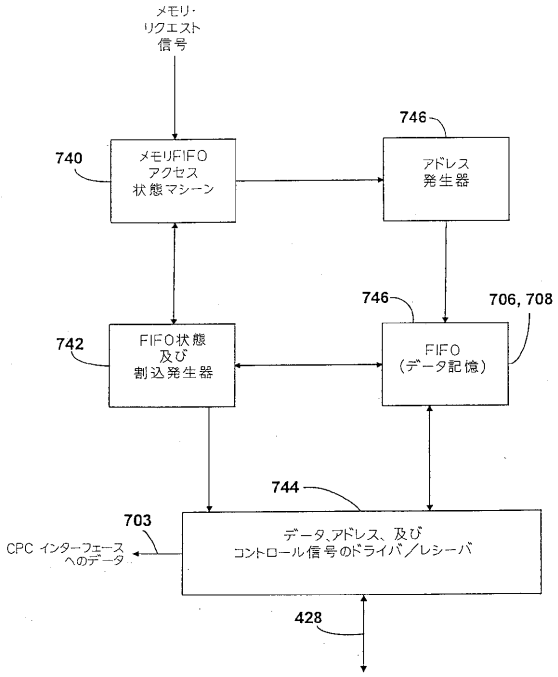
【図２６】



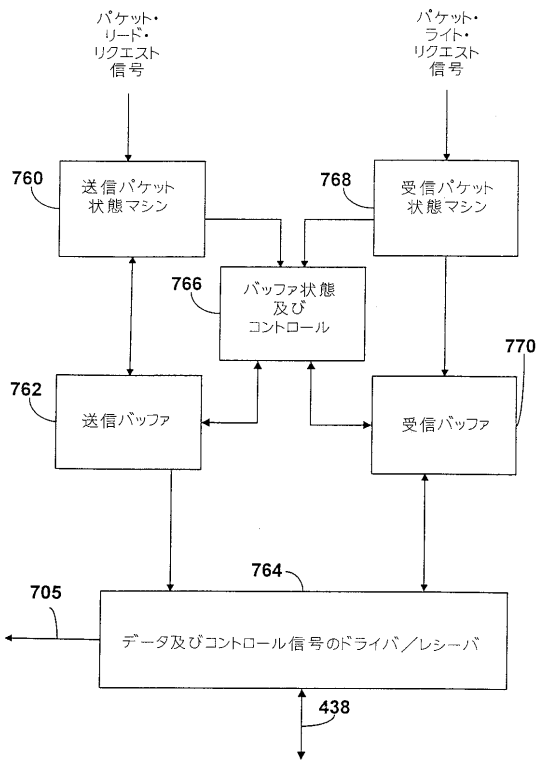
【図２７】



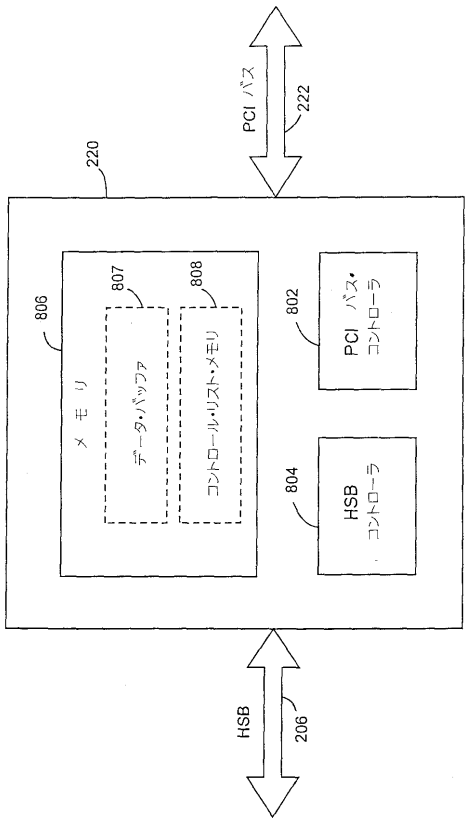
【図２８】



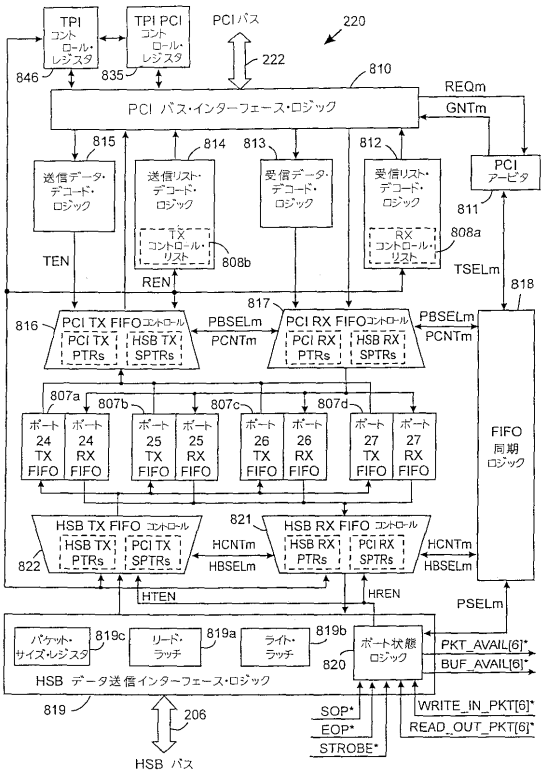
【図２９】



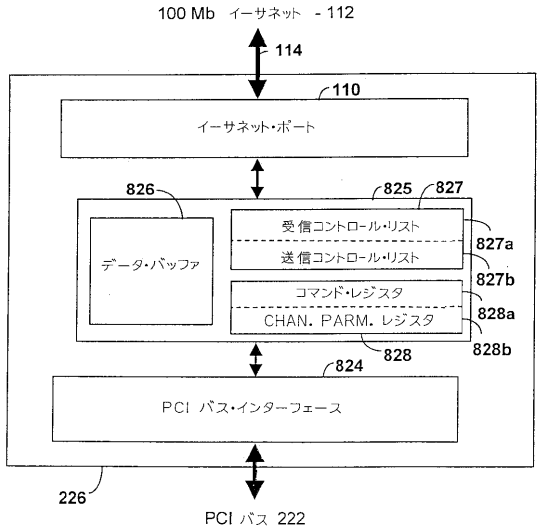
【図３０】



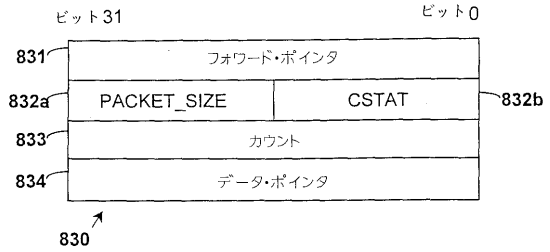
【図３１】



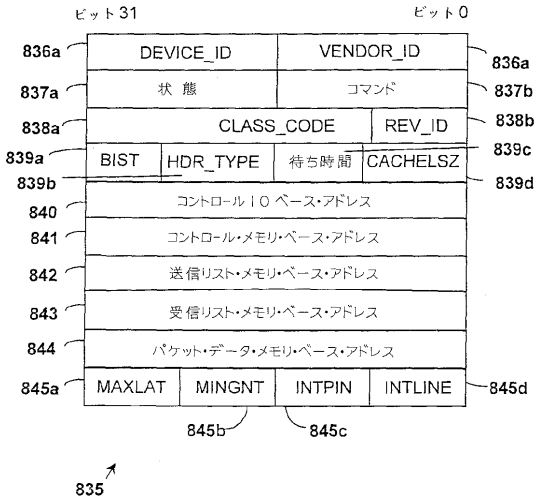
【図３２】



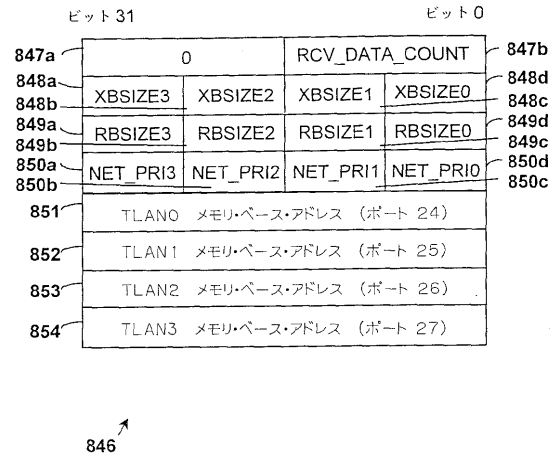
【図３３】



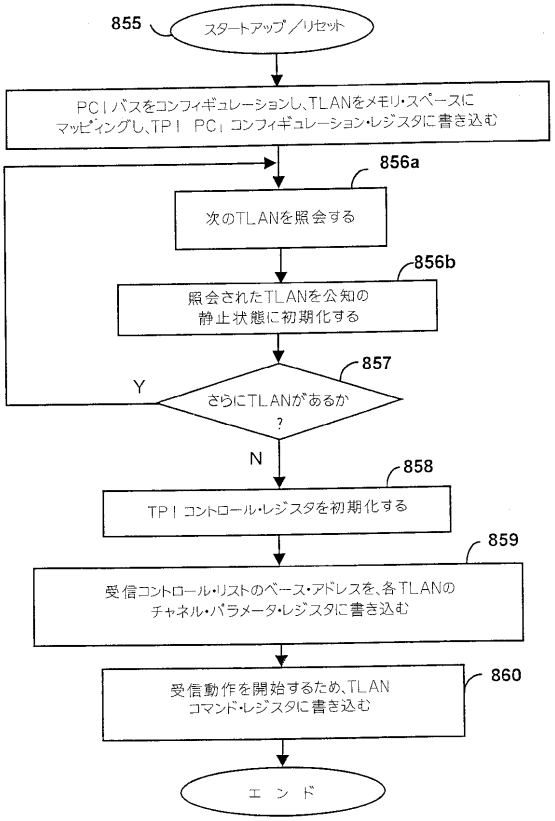
【図３４】



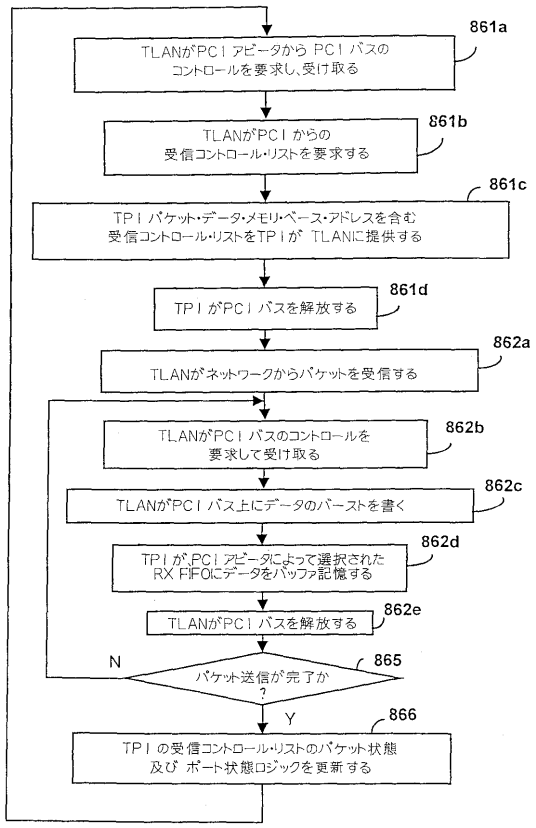
【図３５】



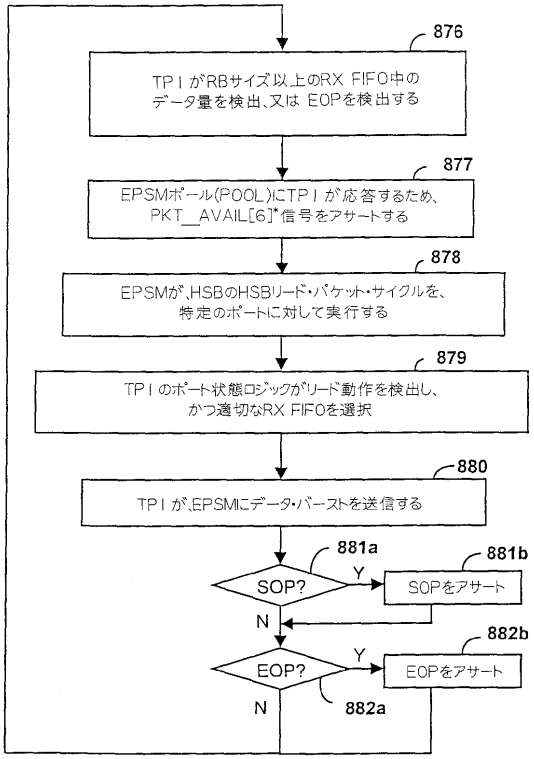
【図３６】



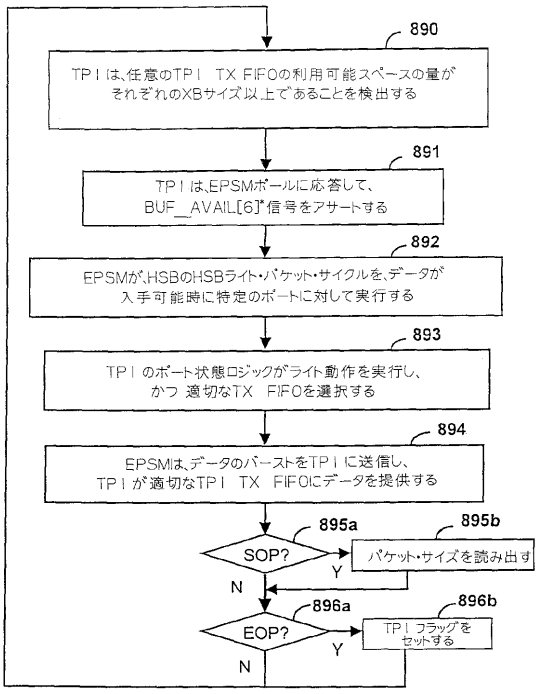
【図３７】



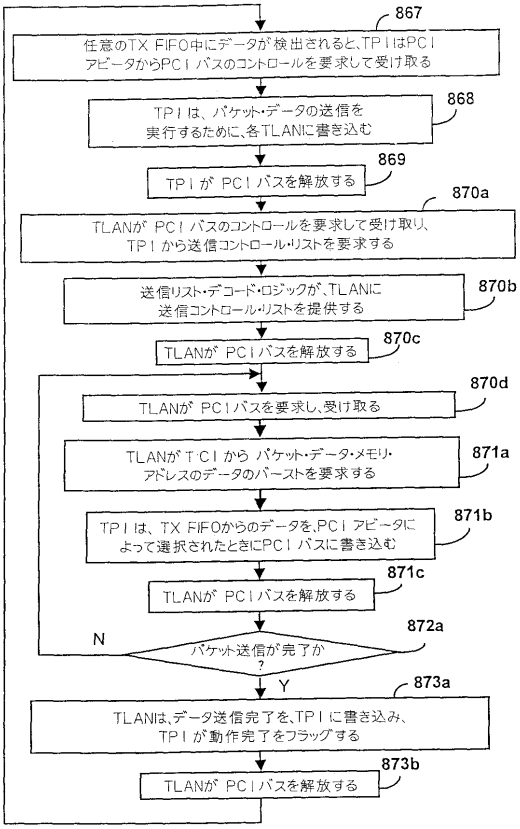
【図３８】



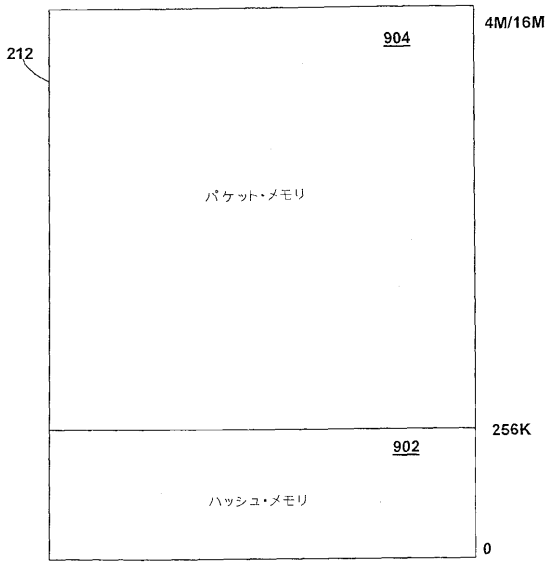
【図３９】



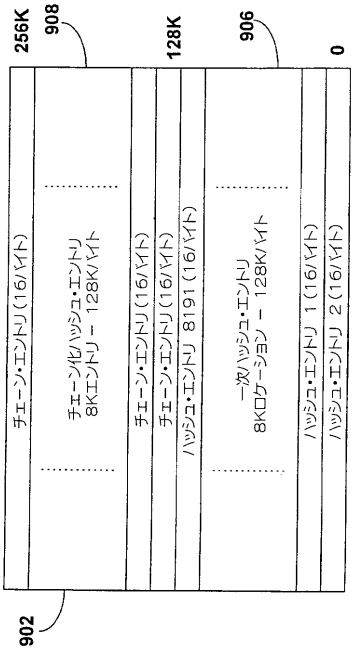
【図４０】



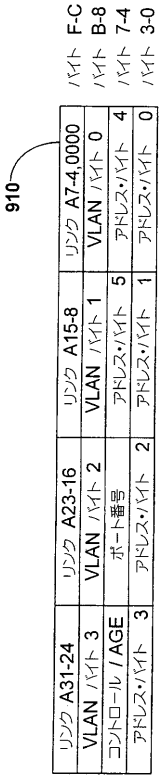
【図４１】



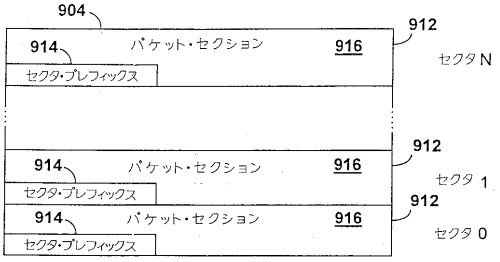
【図４２】



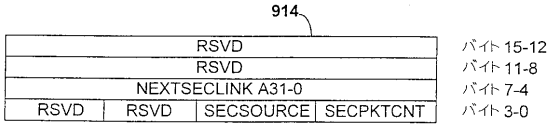
【図４３】



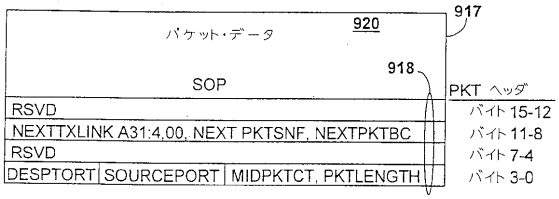
【図４４】



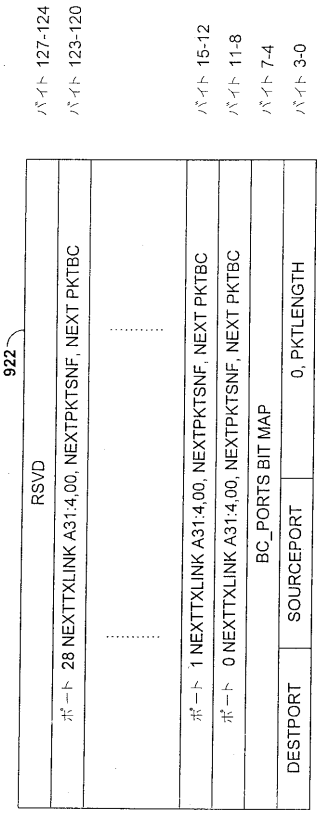
【図４５】



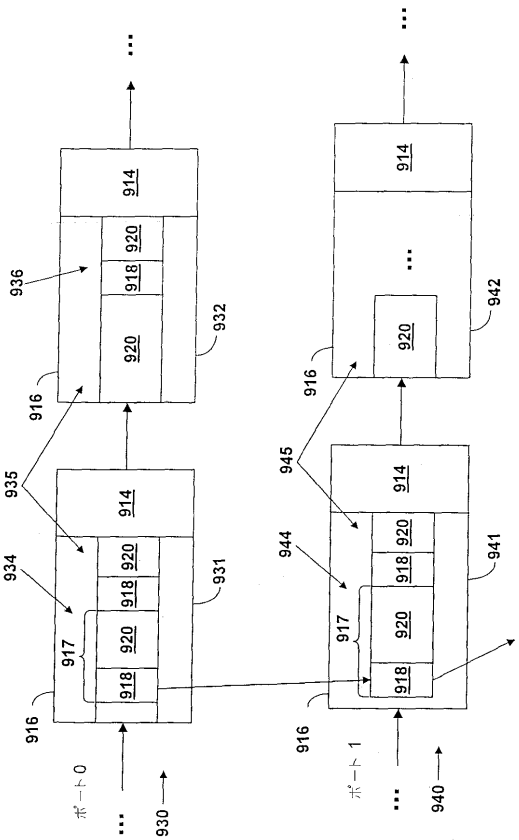
【図４６】



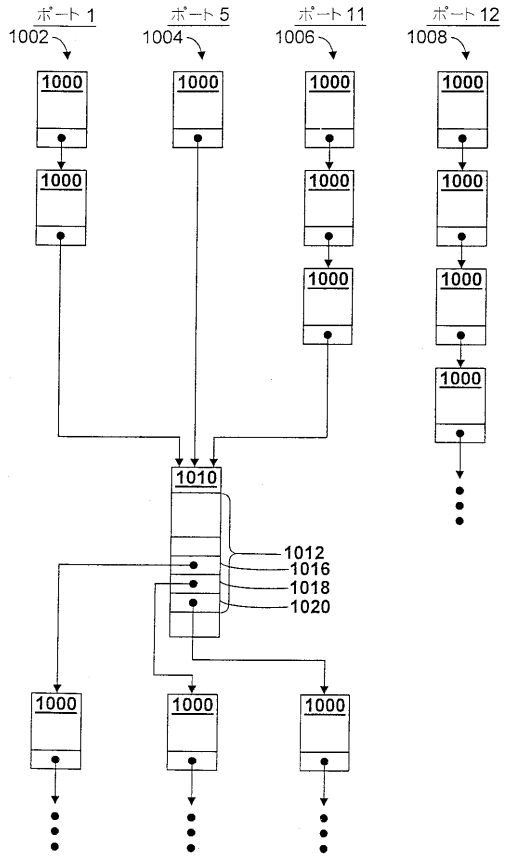
【図４７】



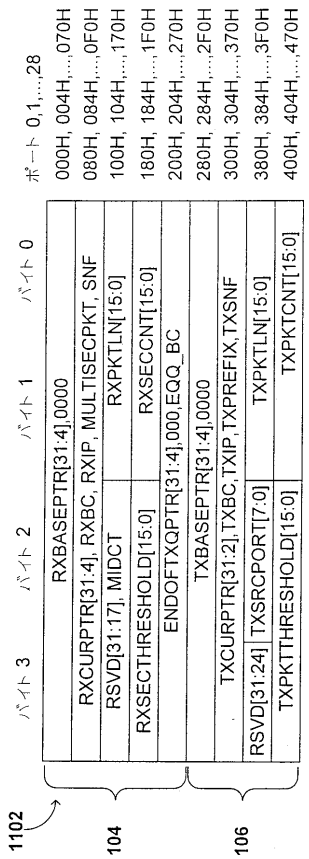
【図４８】



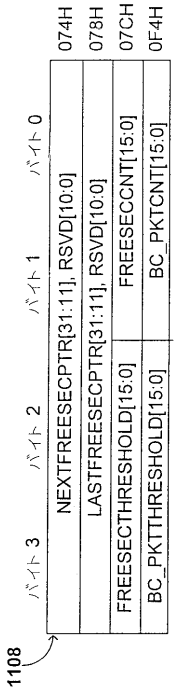
【図４９】



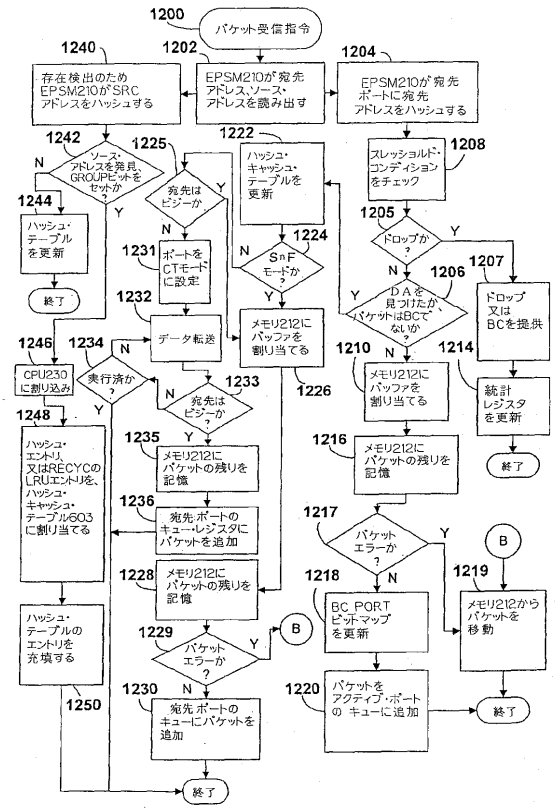
【図５０】



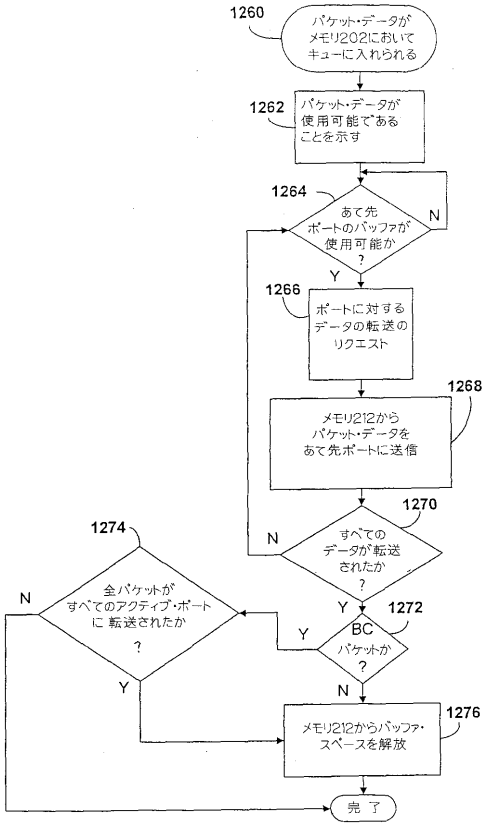
【図５１】



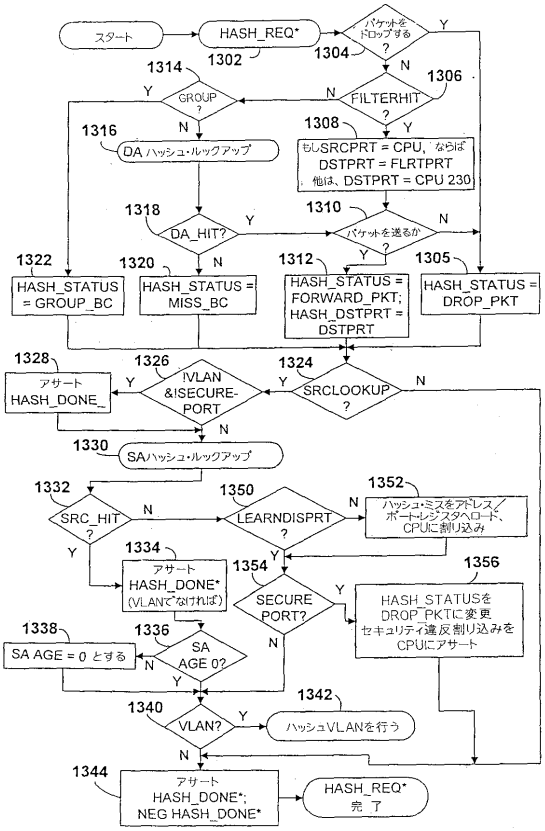
【図５２】



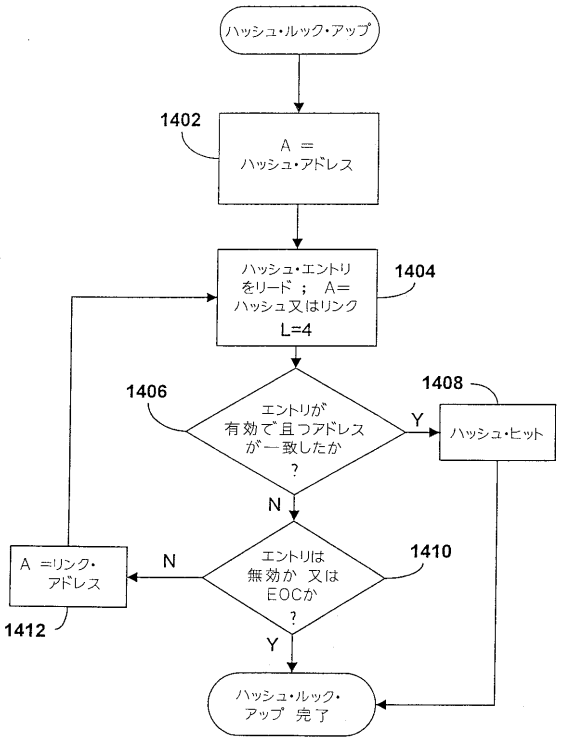
【図５３】



【図５４】



【図５５】


ハッシュ・ルック・アップ
1402
A ＝
ハッシュ・アドレス
ハッシュ・エントリ
をリード ； A＝
ハッシュ又はリンク
L=4
1404
1406
エントリが
有効で且つアドレス
が一致したか
？
Y
1408
ハッシュ・ヒット
N
エントリは
無効か 又は
EOCか
？
1410
N
A ＝リンク・
アドレス
1412
Y
ハッシュ・ルック・
アップ 完了

フロントページの続き

(74)代理人 100071124
弁理士 今井 庄亮
(74)代理人 100076691
弁理士 増井 忠弐
(74)代理人 100075236
弁理士 栗田 忠彦
(74)代理人 100075270
弁理士 小林 泰
(74)代理人 100096068
弁理士 大塚 住江
(74)代理人 100107696
弁理士 西山 文俊
(72)発明者 ウィリアム・ジェイ・ウォーカー
アメリカ合衆国テキサス州７７０７０，ヒューストン，ミルズ・リバー １３１５４
(72)発明者 ゲイリー・ビー・コズアー
アメリカ合衆国テキサス州７７３８８，スプリング，フォーレスト・エルムズ・ドライブ １８４０６
(72)発明者 マイケル・エル・ウィットコウスキー
アメリカ合衆国テキサス州７７３７５，トムボール，エイヴンプレイス・ロード １６２２３
(72)発明者 パトリシア・イー・ハレスキー
アメリカ合衆国テキサス州７７０７０，ヒューストン，ケイン・クリーク・コート １６１０６
(72)発明者 デール・ジェイ・メイヤー
アメリカ合衆国テキサス州７７０７０，ヒューストン，ムーアクリーク １１８１９

審査官 矢頭 尚之

(56)参考文献 特表平９－５００７７４（ＪＰ，Ａ）
特表平８－５１０８７４（ＪＰ，Ａ）
特表平１０－５１１２３６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
H04L 12/44
H04L 12/66