

(19) 日本国特許庁(JP)

(12) 公表特許公報(A)

(11) 特許出願公表番号

特表2004-507164

(P2004-507164A)

(43) 公表日 平成16年3月4日(2004.3.4)

(51) Int.Cl.⁷
H04L 12/56F I
H04L 12/56F
テーマコード (参考)
5K030

審査請求 未請求 予備審査請求 未請求 (全 154 頁)

(21) 出願番号 特願2002-520494 (P2002-520494)
 (86) (22) 出願日 平成13年8月9日 (2001.8.9)
 (85) 翻訳文提出日 平成14年4月11日 (2002.4.11)
 (86) 国際出願番号 PCT/US2001/025083
 (87) 国際公開番号 W02002/015496
 (87) 国際公開日 平成14年2月21日 (2002.2.21)
 (31) 優先権主張番号 60/225,034
 (32) 優先日 平成12年8月11日 (2000.8.11)
 (33) 優先権主張国 米国 (US)
 (31) 優先権主張番号 09/670,007
 (32) 優先日 平成12年9月25日 (2000.9.25)
 (33) 優先権主張国 米国 (US)

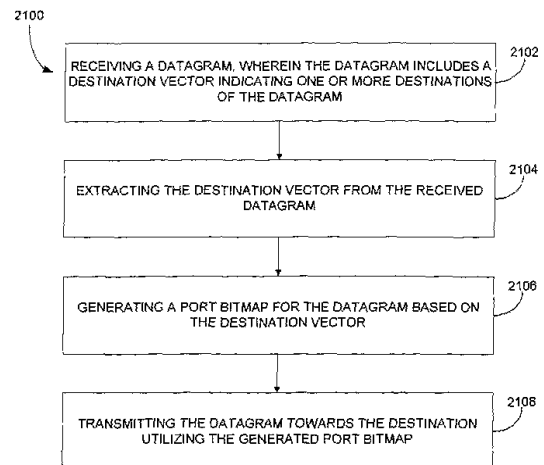
(71) 出願人 502129634
 パイオン・カンパニー・リミテッド
 大韓民国・135-280・ソウル・カン
 ナム・ク・デチードン・996-1・ミレ
 アセット タワー・フォース フロア
 (74) 代理人 100064621
 弁理士 山川 政樹
 (72) 発明者
 チャン, ユースン
 大韓民国・305-701・タエジョン・
 ユソン・ク・クスン・ドン・373-1・
 チップス ビルディング・サード フロア
 ・ケイエアイエスティ

最終頁に続く

(54) 【発明の名称】 新しいスイッチ・ファブリック・チップセット・システムおよび方法

(57) 【要約】

スイッチ・ファブリック (2000) を介してデータグラム (2100) を送信するためのシステム、方法および製造品を提供する。受信されたデータグラム (2102) から、宛先ベクトル (2104) が抽出される。宛先ベクトル (2104) は、データグラム (2102) の1つまたは複数の宛先 (すなわち、ユニキャストまたはマルチキャスト) を指示する。ポート・ビットマップ (2106) がデータグラム (2102) のために、宛先ベクトル (2104) に基づいて生成される。データグラム (2108) は続いて宛先へ向けて、ポート・ビットマップ (2106) を利用して送信される。



【特許請求の範囲】

【請求項 1】

スイッチ・ファブリックを介してデータグラムを送信するための方法であって、
(a) データグラムを受信することを含み、前記データグラムは、前記データグラムの 1 つまたは複数の宛先を指示する宛先ベクトルを含み、さらに、
(b) 前記宛先ベクトルを前記受信されたデータグラムから抽出すること、
(c) 前記データグラムのためのポート・ビットマップを、前記宛先ベクトルに基づいて生成すること、および
(d) 前記データグラムを前記宛先へ向けて、前記生成されたポート・ビットマップを利用して送信することを含む方法。

10

【請求項 8】

スイッチ・ファブリックを介してデータグラムを送信するためのシステムであって、
(a) データグラムを受信するための論理を含み、前記データグラムは、前記データグラムの 1 つまたは複数の宛先を指示する宛先ベクトルを含み、さらに、
(b) 前記宛先ベクトルを前記受信されたデータグラムから抽出するための論理と、
(c) 前記データグラムのためのポート・ビットマップを、前記宛先ベクトルに基づいて生成するための論理と、
(d) 前記データグラムを前記宛先へ向けて、前記生成されたポート・ビットマップを利用して送信するための論理とを含むシステム。

20

【請求項 15】

スイッチ・ファブリックを介してデータグラムを送信するためのコンピュータ・プログラム製品であって、
(a) データグラムを受信するためのコンピュータ・コードを含み、前記データグラムは、前記データグラムの 1 つまたは複数の宛先を指示する宛先ベクトルを含み、さらに、
(b) 前記宛先ベクトルを前記受信されたデータグラムから抽出するためのコンピュータ・コードと、
(c) 前記データグラムのためのポート・ビットマップを、前記宛先ベクトルに基づいて生成するためのコンピュータ・コードと、
(d) 前記データグラムを前記宛先へ向けて、前記生成されたポート・ビットマップを利用して送信するためのコンピュータ・コードとを含むコンピュータ・プログラム製品。

30

【発明の詳細な説明】

【0001】

(関連出願の相互参照)

本願は、2000年8月11日出願の「GIGABIT ETHERNET IP SWITCH CHIPSET」という名称の米国仮特許出願第60/225034号の利点を主張するものであり、これは全体としてあらゆる目的のために参照により本明細書に組み込まれる。

【0002】

(発明の分野)

本発明は一般にデータ・ルーティング・システムに関し、より詳細には、ネットワーク・スイッチを介したパケットのルーティングに関する。

40

【0003】

(発明の背景)

イーサネットは、幅広く導入されているローカル・エリア・ネットワーク技術である。米国電気電子学会(IEEE)802.3の規格において指定されたイーサネットは当初、Xeroxによって開発され、次いでさらにXerox、DECおよびIntelによって開発された。イーサネットLANは通常、同軸ケーブルまたは特殊なグレードのツイスト・ペア線を使用する。一般に導入されたイーサネット・システムは10BASE-Tと呼ばれ、最大10Mbpsの伝送速度を提供する。デバイスがケーブルに接続され、キャリア検知多重アクセス衝突検出(CSMA/CD)プロトコルを使用して、アクセスのた

50

めに競争する。ファスト・イーサネットまたは100BASE-Tは最大100メガビット毎秒の伝送速度を提供し、LANバックボーン・システムのために使用することができ、10BASE-Tカードを有するワークステーションをサポートする。ギガビット・イーサネットは、さらにより高レベルのバックボーン・サポートを1000メガビット毎秒（1ギガビットまたは10億ビット毎秒）で提供する。

【0004】

ファスト・イーサネットはローカル・エリア・ネットワーク伝送規格であり、100メガビット毎秒のデータ率を提供する（「100BASE-T」と呼ばれる）。既存の10メガビット毎秒（10BASE-T）のイーサネット・カードを有するワークステーションを、ファスト・イーサネット・ネットワークに接続することができる。（100メガビット毎秒は共有データ率であり、各ワークステーションへの入力は10Mbpsカードによって制約される）。ギガビット・イーサネットはローカル・エリア・ネットワーク伝送規格であり、10億ビット毎秒（1ギガビット）のデータ率を提供する。ギガビット・イーサネットはIEEE802.3規格において定義されている。ギガビット・イーサネットは、エンタープライズ・バックボーンとして使用することができる。ギガビット・イーサネットを、光ファイバにおいて搬送することができる（銅媒体上では非常に短い距離が可能）。10および100Mbpsカードを有する既存のイーサネットLANは、ギガビット・イーサネット・バックボーンに送られることが可能である。

10

【0005】

データを、パケット交換網を介して、デジタル信号を使用して送信することができる。パケット交換網では、ユーザが同じパスを同時に共有することができ、データ・ユニットが移動するルートは、状態が変化すると変わる可能性がある。パケット交換では、メッセージがパケットに分割され、これらはある数のバイトのユニットである。送信側および宛先のネットワーク・アドレスがパケットに追加される。各ネットワーク・ポイントがパケットを見て、これを次にどこへ送信するべきかを調べる。同じメッセージにおけるパケットは異なるルートを移動する可能性があり、それらが送信された順序と同じ順序で到着しない可能性がある。宛先では、メッセージにおけるパケットが収集され、元のメッセージに再び組み立てられる。

20

【0006】

階層化は、プログラミングを別々のステップに編成することであり、これらは順次実行され、特定のインターフェイスによって、各ステップの結果を次のプログラムまたは層へ、ある量の情報の送信または受信などの全体の機能が完了するまで渡すために、定義される。通信プログラムはしばしば階層化される。通信プログラムのための参照モデル、開放型システム間相互接続（OSI）は、階層化されたプロトコルのセットであり、これにおいては2つのプログラムが通信交換の両側にあり、等しいセットの層を使用する。OSIは7つの層を含み、それぞれが、プログラム対プログラムの通信をコンピュータの間で行うために実行されなければならない異なる機能を反映する。伝送制御プロトコルおよびインターネット・プロトコル（TCP/IP）は、2層セットのプログラムの一例であり、インターネット通信のためのトランスポートおよびネットワーク・アドレス機能を提供するものである。

30

40

【0007】

スイッチはインターネットワーキング・デバイスであり、データのユニットをその次の宛先へ送信するためのパスまたは回路を選択する。スイッチはルータの機能も含み、ルート、および特に、何の隣接ネットワーク・ポイントにデータが送信されるべきであるかを決定することもできる。階層化の開放型システム間相互接続（OSI）通信モデルに関して、スイッチをデータ・リンク層—レイヤ2に関連付けることができる。スイッチは、ネットワーク層—レイヤ3のルーティング機能を実行可能にすることもできる。レイヤ3スイッチは時としてIPスイッチとも呼ばれる。

【0008】

OSIのデータ・リンク層—レイヤ2は、データをネットワークにおいて物理リンクに渡

50

って移動させることに関係する。ネットワークでは、スイッチを、データ・メッセージをレイヤ2レベルでリダイレクトするデバイスにすることができ、これは媒体アクセス制御(MAC)アドレスを使用して、どこにメッセージをダイレクトするべきであるかを決定する。データリンク層は2つのサブレイヤを含み、これらはIEEE-802のLAN規格において記載されており、すなわち媒体アクセス制御(MAC)および論理リンク制御(LLC)である。データ・リンク層は、最初の接続がセット・アップされていることを確かめ、出力データをデータ・フレームに分割し、受信側からの、データがうまく到着したという応答を処理する。これはまた、入力データがうまく受信されたことも保証し、これは、フレームにおける特殊な場所でのビット・パターンを解析することによって行う。

【0009】

10

OSIのネットワーク層ーレイヤ3は、ネットワークにおける隣接ノードのアドレスを知ること、ルートおよびサービス品質を選択すること、およびローカル・ホスト・ドメインのための入力メッセージを認識し、かつトランスポート層へ転送することに関係する。スイッチをレイヤ3デバイスにすることができ、これはレイヤ3の機能を実行することができる。IPアドレスはレイヤ3アドレスである。

【0010】

交換は、その本質により、一時データ・フローを含む。ギガビット速度で、これらの一時的なものは非常に高速に、かつ非常にランダムな方法において発生する。過去においては、十分に高速な速度で、スイッチ・データ・フローのどのコンポーネントがダイレクトされるべきであるかを決定して、スイッチを通過するデータ・フローを最大限にし、かつス

20

【0011】

(発明の概要)

スイッチ・ファブリックを介してデータグラムを送信するためのシステム、方法および製造品を提供する。受信されたデータグラムから、宛先ベクトルが抽出される。宛先ベクトルは、データグラムの1つまたは複数の宛先(すなわち、ユニキャストまたはマルチキャスト)を指示する。ポート・ビットマップがデータグラムのために、宛先ベクトルに基づいて生成される。データグラムは続いて宛先へ向けて、ポート・ビットマップを利用して送信される。

【0012】

30

本発明の一態様では、宛先ベクトルが32ビットのデータを含むことができる。本発明の別の態様では、ポート・ビットマップが8ビットのデータを含むことができる。本発明の一実施形態では、ポート・ビットマップを生成することがさらに、データグラムのすべての宛先を宛先ベクトルから得ること、および次いで、スイッチング・テーブルを利用して、スイッチ・ファブリック・コンポーネントのどのスイッチ・ポートを介してデータグラムを宛先に到達するようにルーティングすることができるかを決定することを、含むことができる。次いで、ポート・ビットマップを生成して、それを介してデータグラムをその所期の宛先へ到達するようにルーティングすることができる、決定されたスイッチ・ポートのすべてをその中に指示することができる。

【0013】

40

本発明の別の実施形態では、データグラムを送信することが、ポート・ビットマップに基づいてマスクを作成すること、およびマスクをデータグラムの送信において使用してポート・ビットマップを修正することも含むことができる。このような実施形態では、マスクを使用してポート・ビットマップを修正することが、データグラムが送信された先の宛先に関連付けられたデータのポート・ビットマップをクリアすることも含むことができる。本発明のさらなる実施形態では、それを介してデータグラムを宛先へ到達するようにルーティングすることができるスイッチ・ポートの可用性も、決定することができる。次いで、データグラムを、使用可能であると決定されたスイッチ・ポートを介して送信することができる。

【0014】

50

以上および他の特徴、態様、および利点は、以下の詳細な説明、添付の請求項、および添付の図面から、より良く理解される。

【0015】

(詳細な説明)

イーサネット・スイッチ・ファブリック・チップセット・システムは、スイッチ・ファブリック・トポロジ、および、32ギガビットのイーサネット・データ・ストリームのワイヤスピード、ノンブロッキング、レイヤ2およびレイヤ3のスイッチングを支持することができるマルチプロセッサ・ポート・コントローラを使用して開示されている。これは、ネットワーク・プロセッサとして当技術分野で知られているものに拡張することができる。スイッチング技術の基本的な理解のために、読者は、Rich Seifert (John Wiley & Sons, Inc. 2000) による The Switch Book: the complete guide to LAN switching technology を指示される。これは、あらゆる目的のために、参考文献によって完全に本明細書に組み込まれている。

【0016】

本明細書に記述したシステムにより、どのスイッチ・ファブリック構成要素の過渡的なデータ・フローを向けることができるかに関する決定は、他のデータ・フロー・トラフィックによるブロッキングを防止しながら、スイッチ・ファブリック・システムを通るデータ・フローを最大にするのに十分な速さで実施することが可能である。これは、過渡現象が生じる前に可能な限り多くの決定を実施し、かつ、考慮の決定を、スイッチ・ファブリックを通る経路全体ではなく、単に経路の次のスイッチ・ファブリック構成要素とすることによって、達成される。この解決法の一部として、スイッチ・ファブリックを通して状況と制御の情報を送り返すことに備え、かつ、制御経路における初期接続手順の必要性を軽減または排除することを支援するために、データ経路の他に、分離制御経路を提供することが可能である。この解決法の一部として、制御経路を使用して更新することができる各スイッチ・ファブリック構成要素の内側に、スイッチング・テーブルを提供することが可能である。

【0017】

(システム構成)

図1は、本発明の実施形態による、マルチポート・スイッチ・アーキテクチャ100の概略的なブロック図である。一般に、このマルチポート・スイッチ・アーキテクチャ100は、スイッチ・ファブリック104を介して互いに相互接続された複数のポート・コントローラ(例示した実施形態では32のポート・コントローラ)102、ポート・コントローラ102に接続されたネットワーク制御プロセッサ106、およびスイッチ・ファブリック104を備える。図1に示したように、要素間の接続(例えば108、110、112)は、双方向である。

【0018】

さらに図1を参照すると、このマルチポート・スイッチ・アーキテクチャ100の特定の実施形態では、スイッチ・ファブリックは、クロス・ネットワーク構成/トポロジとして知られているものにおいて双方向コンジット(例えばコンジット110)を介して共に接続されている複数のスイッチ・ファブリック構成要素(例示した実施形態では12のスイッチ・ファブリック構成要素)104a、104b、104c、104d、104e、104f、104g、104h、104i、104j、104k、104lを備えることが可能である。図1に示した特定の構成は、スイッチ・ファブリック構成要素(104a、104b、104c、104dおよび104i、104j、104k、104l)の外側ステージの対を接続するスイッチ・ファブリック構成要素(104e、104f、104g、104h)の中間ステージを有する3つのステージまたは層のクロス・ネットワーク・トポロジとして知られている。

【0019】

スイッチ・ファブリック104のクロス・ネットワーク・トポロジにより、各外側ステー

ジのスイッチ・ファブリック構成要素104a、104b、104c、104dおよび104i、104j、104k、104lは、独立したコンジットを介して、中間ステージのスイッチ・ファブリック構成要素104e、104f、104g、104hの各々に直接結合される。例えば、図1では、スイッチ・ファブリック構成要素104aは、それを4つの中間ステージのスイッチ・ファブリック構成要素104e、104f、104g、104hに結合する4つのコンジットを有する。逆に、スイッチ・ファブリック104のクロス・ネットワーク・トポロジにより、中間ステージのスイッチ・ファブリック構成要素104e、104f、104g、104hの各々は、独立したコンジットを介して、外側ステージのスイッチ・ファブリック構成要素104a、104b、104c、104d、104i、104j、104k、104lの各々に直接結合される。したがって、図1に示した実施形態では、中間ステージのスイッチ・ファブリック構成要素104e、104f、104g、104hは、外側ステージのスイッチ・ファブリック構成要素104a、104b、104c、104d、104i、104j、104k、104lの各々への直接的で独立した接続を有する。

【0020】

また、スイッチ・ファブリックを横切る単一の packets のために、単一の外側ステージのスイッチ／ファブリック構成要素を備える入力ステージ、中間ステージのスイッチ・ファブリック構成要素を備える中間ステージ、および、スイッチ・ファブリック構成要素間の双方向の接続性のために残りの外側ステージのスイッチ・ファブリック構成要素を備える外側ステージを有するような、図1に示した特定の実施形態を考慮することが可能であることに留意されたい。

【0021】

各外層のスイッチ・ファブリック構成要素104a、104b、104c、104d、104i、104j、104k、104lは、直接それに結合／接続された複数のポート・コントローラを有する。図1に示した実施形態では、外層のスイッチ・ファブリック構成要素の各々は、1つのコンジット（例えばコンジット108）が各ポート・コントローラに対応する、4つのコンジットによってそれに結合された4つのポート・コントローラを有する。例えば、スイッチ・ファブリック104aは、4つの別々の双方向コンジットによってそれに結合されたポート・コントローラ102a、102b、102c、102dを有する。この実施形態の一態様では、ポート・コントローラ102は、すべて、ギガビットのポート・コントローラを備えることが可能である。

【0022】

スイッチ・ファブリックのクロス・ネットワーク・トポロジの特徴は、ノンブロッキング・トポロジであるということである。各ポート・コントローラから、中間ステージのスイッチ・ファブリック構成要素104e、104f、104g、104hのそれぞれを通る同じスイッチ・ファブリック構成要素に直接結合されていない他のすべてのポート・コントローラまで、経路が存在する。例えば、スイッチ・ファブリック構成要素104aに接続されたポート・コントローラ102a、102b、102c、102dは、中間ステージのスイッチ・ファブリック構成要素104e、104f、104g、104hの各々を介して、他の外側ステージのスイッチ・ファブリック構成要素104b、104c、104d、104i、104j、104k、104lのすべてに結合されたポート・コントローラに接続されている。

【0023】

図1に示した構成100の好ましい実施形態では、全体で32ギガビットのイーサネット・ポート・コントローラのためのスイッチ・ファブリック構成要素の各外側ステージに提供された、16ギガビットのイーサネット・ポート・コントローラが存在することが可能である。この好ましい実施形態では、16ギガビットのイーサネット・ポート・コントローラの各セットからの最大スループットの要求は、 16×2 （2倍）ギガビット・イーサネット・データ・ストリーム = $32 \text{ Gbps} + [(32 \text{ Gbps} + \text{オーバーヘッド}) + (32 \text{ Gbps} + \text{オーバーヘッド})]$ の両方向における最大スループットの要求に対するオ

ーバーヘッド＝64 Gbps＋オーバーヘッドとなることが可能である。そのような好ましい実施形態では、32ギガビットのイーサネット・ポート構成におけるスイッチ・ファブリック構成要素の各々は、16 Gbpsにオーバーヘッドを加えたスループット能力を有する変更された8ポートのクロスバー・スイッチ・ファブリックとすることが可能である。スイッチ・ファブリックの中間ステージは、これらのデバイスの4つを備えるので、中間ステージのスループットは、全体で64 Gbpsにオーバーヘッドを加えたものとなることが可能である。

【0024】

図2は、本発明の実施形態による、他のマルチポート・スイッチ・アーキテクチャ200の概略的なブロック図である。この実施形態では、マルチポート・スイッチ・アーキテクチャ200は、単一のスイッチ・ファブリック構成要素204を備えるスイッチ・ファブリックを介して共に相互接続された4つのポート・コントローラ202a、202b、202c、202dを備える。そのような実施形態の一態様では、ポート・コントローラ202a、202b、202c、202dは、図1に示したポート・コントローラのアーキテクチャと同様のアーキテクチャを有するファスト・イーサネット・ポート・コントローラを備えることが可能である。

10

【0025】

図3は、本発明の実施形態による、ギガビット・アップリンク300の概略的なブロック図である。アップリンク300は、双方向に一緒に結合された第1ポート・コントローラ302aと第2ポート・コントローラ302bを備える。そのような実施形態の一態様では、ポート・コントローラの一方は、図2に示したマルチポート・スイッチ・アーキテクチャ200で使用したタイプのようなファスト・イーサネット・ポート・コントローラ302aを備えることが可能であり、他のポート・コントローラ302bは、図1に示したマルチ・ポート・スイッチ・アーキテクチャ100で使用したタイプのようなギガビット・ポート・コントローラを備えることが可能である。

20

【0026】

(ポート・コントローラ)

各ポート・コントローラは、パイプライン処理と呼ばれる技術を実施する。パイプライン処理は、複数の命令が実行の際に部分的に重なる実施技術である。パイプラインは、一連のステージとして定義することができ、各ステージで何らかの作業が行われる。一般にポート・コントローラは、物理インターフェイス(PHY)からイーサネット・パケットを受け取り、それらを処理し、スイッチ・ファブリックに送るための、第1のパイプラインを有する。ポート・コントローラはまた、類似するが複雑さのより低い第2のパイプラインも有し、これはスイッチ・ファブリックからイーサネット・パケットを受け取り、それらを物理インターフェイス(PHY)に送るためのものである。

30

【0027】

一般に、ポート・コントローラの好ましいバージョンが少なくとも2つあるものとなることができる。第1の好ましいバージョンは、単一(二重)ギガビット・イーサネット・ラインに対してワイヤ速度のパケット処理およびその他のポート・コントローラ機能を提供することができる。第2の好ましいバージョンは、8本のファスト・イーサネット(またはイーサネット)・ラインに対して同じ機能を提供することができる。これらのデバイスは、スイッチ・ファブリックによって構成して、32本のギガビット・イーサネット・ラインまで、または256本のファスト・イーサネット・ラインまでのワイヤ速度レイヤ2/3切り換えを実現することができる。

40

【0028】

図4は、本発明の一実施形態によるポート・コントローラ400のアーキテクチャの概略ブロック図である。ポート・コントローラ400は、メディア・アクセス制御(MAC)ユニット402、入口ダイレクト・メモリ・アクセス(入口DMAまたはIDMA)ユニット404、ヘッダ待ち行列406、パケット・プロセッサ・ユニット408、ダイナミック・ランダム・アクセス・メモリ(DRAM)コントローラ410、出力向けスケジュー

50

ーラ 4 1 2、インターフェイス・トランスミッタ 4 1 4、フリー・セル・マネージャ 4 1 6、同期スタティック・ランダム・アクセス・メモリ (S S R A M) コントローラ 4 1 8、インターフェイス・レシーバ 4 2 0、出口コントローラ 4 2 2、出口バッファ 4 2 4、およびネットワーク・インターフェイス 4 2 6 を備える。D R A M コントローラ 4 1 0 は、外部パケット・バッファ／メモリと呼ばれる第 1 メモリに結合されるが、このメモリは、同期 D R A M (S D R A M) など何らかの種類のダイナミック・ランダム・アクセス・メモリであることが好ましい。D R A M コントローラ 4 1 0 は、第 1 メモリ／外部パケット・バッファを制御し、外部パケット・バッファに対してデータの読取りおよび書込みを行う。S S R A M コントローラ 4 1 8 は、ポート・コントローラの第 2 メモリに結合されるが、このメモリは、何らかの種類の同期スタティック・ランダム・アクセス・メモリ (S S R A M) であることが好ましい。 10

【0029】

M A C インターフェイス 4 0 2 は、データ・パケットを送受信するためにイーサネット物理層インターフェイス (P H Y) に接続可能である。一実施形態では、M A C インターフェイス 4 0 2 は、G M I I インターフェイスを備えたギガビット M A C を含むことができる。M A C インターフェイス 4 0 2 は、入口 D M A 4 0 4 に結合されて、M A C インターフェイスによって受信された入力データ・パケットを入口 D M A 4 0 4 に送ることができるようにする。

【0030】

入口 D M A 4 0 4 は、ヘッダ待ち行列 4 0 6、D R A M コントローラ 4 1 0、およびフリー・セル・マネージャ 4 1 6 に結合されている。入力データ・パケットを M A C インターフェイス 4 0 2 から受け取ると、入口 D M A 4 0 4 は、パケットの本体部分からパケットのヘッダ部分を分離する。次いで、パケットのヘッダ部分は、入口 D M A 4 0 4 からヘッダ待ち行列 4 0 6 に転送され、パケットの本体部分は、外部パケット・バッファにリンク・リストで記憶されるように D R A M コントローラ 4 1 0 に転送される。入口 D M A 4 0 4 はまた、パケット・プロセッサ 4 0 8 によって処理されているヘッダ部分に対して、メモリ中の予約された空間へのリンクを定義することも担い、このリンクは、ヘッダ部分が処理された後で入口 D M A が処理済みヘッダ部分をメモリに記憶するのに使用する。入口 D M A はまた、パケットの記憶が完了したときに出力向けスケジューラ 4 1 2 に通知することも担う。 20 30

【0031】

ヘッダ待ち行列 4 0 6 は、パケット・プロセッサ・ユニット 4 0 8 に結合されている。好ましい一実施形態では、ヘッダ待ち行列 4 0 6 は、パケット・プロセッサ・ユニット 4 0 8 によって先入れ先出し (F I F O) 処理されるように、入口 D M A 4 0 4 から受け取ったヘッダを記憶する。

【0032】

パケット・プロセッサ・ユニット 4 0 8 は、パケットを分析および処理する働きをする。一実施形態では、パケット・プロセッサは複数のパケット・プロセッサを含む。図 4 に示す実施形態では、パケット・プロセッサ・ユニットは 2 つのパケット・プロセッサ P₀ および P₁、4 0 8 a および 4 0 8 b を含み、これらは、ヘッダ待ち行列 4 0 6 中のヘッダを処理する際に交互になる。 40

【0033】

パケット・プロセッサ 4 0 8 は、入力パケットのヘッダ部分をどのように処理するか決定する。パケット・プロセッサ 4 0 8 によるヘッダ部分の処理は、ルックアップ・アドレス・テーブルを利用することによってパケットの宛先を決定し、パケットのヘッダ部分に内部タグ (宛先ベクトルを含む) を付け、次いで処理の完了をヘッダ待ち行列 4 0 6 に通知することを含む。アドレス・ルックアップは、外部 S S R A M に記憶された外部アドレス・テーブルにアクセスする (S S R A M コントローラ 4 1 8 を介して) ことによって行う (この S S R A M は、他の必要情報を記憶するのににも使用される)。

【0034】

出力向けスケジューラ 412 は、パケットが処理されたときに入口 DMA から通知を受け取る。出力向けスケジューラ 412 はまた、DRAM コントローラ 410 を介して外部パケット・バッファから処理済みパケットをリトリブできるように、DRAM コントローラ 410 に結合されている。処理済みパケットをパケット・バッファからリトリブすると、出力向けスケジューラ 412 は、ポート・コントローラ 400 の外に送るために、リトリブしたパケットをインターフェイス・トランスミッタ 414（または定義された優先順位に応じてネットワーク・インターフェイス 426）に転送する。

【0035】

DRAM コントローラ 410 は、入口 DMA 404 から外部パケット・バッファにデータを書き込むための書込みコマンドを実行する。DRAM コントローラ 410 はまた、出力向けスケジューラ 412 から読取りコマンドを受け取ると、パケット・バッファから出力向けスケジューラ 412 へのデータの読取りおよび転送も実行する。 10

【0036】

インターフェイス・トランスミッタ 414 は、出力向けスケジューラ 412 に結合されている。インターフェイス・トランスミッタは、別のポート・コントローラまたはスイッチ・ファブリック・コンポーネントへの接続を可能にする外部入出力を有する。出力向けスケジューラ 412 は、処理済みパケットをインターフェイス・トランスミッタ 414 から外部入出力を介してスイッチ・ファブリック・コンポーネントまたは別のポート・コントローラに送ることができるようにするために、処理済みパケットをインターフェイス・トランスミッタ 414 に送る。 20

【0037】

フリー・セル・マネージャ 416 は、入口 DMA 404 および出力向けスケジューラ 412 に結合されており、それにより DRAM コントローラ 410 と通信することができる。フリー・セル・マネージャ 416 は、外部パケット・バッファ空間を管理し、データ記憶に利用可能なメモリ中の空間について入口 DMA 404 に通知し、データ（すなわち処理済みパケット）が出力向けスケジューラ 412 によってリトリブされた後でメモリ空間を開放する。

【0038】

インターフェイス・レシーバ 420 は、スイッチ・ファブリック・コンポーネントまたは別のポート・コントローラに接続可能な外部入出力を有し、スイッチ・ファブリック・コンポーネントまたは他のポート・コントローラからパケットを受け取ることができる。出口コントローラ 422 は、インターフェイス・レシーバ 420 に結合されている。出口コントローラ 422 には、出口バッファ 424 および MAC インターフェイス 402 が結合されている。出口コントローラ 422 は、インターフェイス・レシーバから受け取った送信パケットを出口バッファ 424 に一時的に記憶し、次いでその後、PHY に送られるように、記憶したパケットを出口バッファ 424 から MAC インターフェイス 402 に転送する。 30

【0039】

ネットワーク・インターフェイス・ユニット 426 は、外部入出力または内部チャネルを介してホスト・プロセッサとデータを交換する。ネットワーク・インターフェイス・ユニット 426 は、ホスト・プロセッサによって作成された制御パケットを分析し、内部レジスタを読み取り、ルックアップ・テーブルの内容を変更する。好ましい一実施形態では、ポート・コントローラのあらゆる内部ユニットをホスト・プロセッサによってネットワーク・インターフェイス・ユニット 426 を介して制御できるように、あらゆる内部ユニットはネットワーク・インターフェイス（NPI）バスに接続される。 40

【0040】

SSRAM コントローラ 4018 は、第 2 メモリ（SSRAM）に結合され、SSRAM へのアクセスを制御する。ネットワーク・インターフェイス・ユニット 4026 およびパケット処理ユニット 408 が SSRAM コントローラに結合され、したがって SSRAM にアクセスする。SSRAM コントローラ 4018 は、ネットワーク・インターフェイス 50

・ユニットおよびパケット処理ユニットからの要求を調停する。

【0041】

(パケット処理)

ポート・コントローラ400中では、パケットは、受信された順序に従ってパケット・メモリ／バッファに保存することができる。しかし、本発明の好ましい一実施形態では、入力パケットのヘッダ・フィールドと本体フィールドとの保存順序を変更することのできる、よりフレキシブルなパケット・ソーティング構成を利用することができる。

【0042】

通常、イーサネット・ヘッダおよびTCP/IPヘッダを有するパケットを処理するときは、従来型のスイッチ中でイーサネット・フィールドまたはIPフィールドだけを使用して、切換え方向を決定することができる。対照的に、本明細書で述べるスイッチ・ファブリック・システムの実施形態では、イーサネットとIPフィールドの両方、ならびにTCPフィールドを使用して切換え方向を決定することができる。しかし、このような処理能力を用いると、このような逐次的なパケット保存順序ルーチンは、ヘッダの処理に十分な時間および／または効率を提供することが不可能（または非实际的）であることにより、パケット処理中に獲得できる情報の量を制限する恐れがある。ヘッダ・フィールドと本体フィールドの保存順序を変更することにより、処理中により多くの情報を獲得することができ、それにより、本体フィールドの保存中にヘッダ・フィールドを処理するための追加のタイミング余裕がもたらされ、効率を最大限にしてメモリ・オーバヘッドを縮小することができる。

10

20

【0043】

図5は、本発明の一実施形態による、スイッチ・ファブリックを介して送信するためのデータグラムを作成するプロセス500のフローチャートである。まず動作502で、ヘッダ部分および本体部分を有するデータグラムを受信する。動作504および506で、ヘッダ部分を本体部分から分離し、次いで処理のために転送する。動作508で、分離した本体部分をメモリに記憶する。動作510および512で、ヘッダ部分を処理後に受け取り、メモリに記憶する。

【0044】

本発明の一実施形態では、本体部分をメモリに記憶する前に、ヘッダ部分を記憶するためにメモリ中の空間を予約することができ、予約済み空間を突き止めるための、予約済み空間へのリンクもまた生成される。このような実施形態におけるオプションとして、処理済みヘッダを記憶することはさらに、リンクを利用して予約済み空間を突き止め、次いで処理済みヘッダを予約済み空間に記憶することを含むこともできる。

30

【0045】

本発明の一態様では、本体部分を複数のセルに分割することができる。このような態様では、本体部分をメモリに記憶することはさらに、各セルを逐次的にメモリに記憶し、逐次的に記憶したセル間のリンクを生成することを含むことができる。本発明の別の態様では、処理済みヘッダ部分は、パケットのルーティング先となるスイッチ・ファブリック・システム中の宛先を示す宛先ベクトルを含むことができる。このような態様の好ましい一実施形態では、宛先ベクトルは32ビットのデータを含むことができる。

40

【0046】

本発明の別の実施形態では、後でスイッチ・ファブリックを介して送信するために、データグラムを出力優先順位待ち行列に割り当てることができる。

【0047】

図6は、本発明の一実施形態による例示的なパケット600の概略図である。パケットは、パケット交換ネットワーク上の発信源と宛先との間でルーティングされるデータの単位として定義することができる。パケット600は、ヘッダ部分またはフィールド（「ヘッダ」）602と、本体部分またはフィールド（「本体」）604を含む。ヘッダ部分602は、切換えに使用される制御情報を含み、本体604は一般に、パケットの残りを含む。

【0048】

50

図 6 に示す例示的なパケット中では、ヘッダ部分 602 は処理済みであり、パケットの宛先を示す宛先ベクトル 606 と、パケットのソースを示すソース・ベクトル 608 と、パケットの長さを示す長さベクトル 610（これらは総称して「内部タグ」または「内部タグ・フィールド」とも呼ばれる）を含む。ヘッダ部分 602 は、パケットの MAC / TCP / IP および何らかのアプリケーション・データも含むことができる。

【0049】

図 7 は、本発明の一実施形態による、パケット 600 を処理するプロセス 700 のフローチャートである。動作 702 で、入力パケットを受信する。入力パケットは、ヘッダ部分および本体部分を含む。動作 704 で、パケットのヘッダ部分を本体部分から分離する。次いで動作 706 で、分離した本体部分をメモリに記憶する。動作 2308 で、好ましくは動作 2306 と同時に、ヘッダ部分を処理する。動作 2310 で、後続の（または第 2 の）パケットを受信し、この第 2 のパケットに対してプロセス 700 を繰り返す。第 1 の入力パケットのヘッダ部分を処理すると、次いで動作 712 で、処理済みヘッダ部分をメモリに記憶する。

10

【0050】

図 8 は、本発明の一実施形態による、図 7 で述べたパケット処理プロセス 700 をポート・コントローラ 400 中で示した第 1 の概略流れ図である。入力パケット 600 が、MAC インターフェイス 402 を介してポート・コントローラの中に受信され、MAC インターフェイス 402 は、パケットを入口 DMA 404 にルーティングする（矢印 802 参照）。入口 DMA 404 中で、パケットのヘッダ部分 602 がパケットの本体 604 から分離される（動作 704 参照）。次いで入口 DMA 404 は、ヘッダ部分 602 を処理のためにヘッダ待ち行列 406 に転送し（矢印 806 参照）、本体部分 604 を、DMA コントローラ 410 を介してメモリ（「パケット・メモリ」）804 に記憶する（矢印 808 および動作 706 参照）。

20

【0051】

ヘッダ部分 602 を処理するために（動作 708 参照）、ヘッダ待ち行列 406 は、パッケージ・プロセッサ・ユニット 408 中の利用可能なパッケージ・プロセッサにヘッダ部分 602 を転送する（矢印 810 参照）。ヘッダ部分がパケット・プロセッサ・ユニット 408 によって処理された後、処理済みヘッダ部分はヘッダ待ち行列 406 に戻される（矢印 812 参照）。

30

【0052】

図 9 は、本発明の一実施形態による、図 7 で述べたパケット処理プロセス 700 をポート・コントローラ 400 中で示した第 2 の概略流れ図である。処理済みヘッダ部分は、ヘッダ待ち行列 406 に戻った後、続いて入口 DMA 404 および DRAM コントローラ 410 を介してパケット・メモリ 804 に記憶される（矢印 902 参照）。

【0053】

入口 DMA 404 は、処理済みヘッダ部分を記憶すると、パケットの送信準備が今やできたことを出力向けスケジューラ 412 に通知し、パケットをパケット・メモリ 804 からリトリブするための情報を出力向けスケジューラに提供する（矢印 904 参照）。出力向けスケジューラ 412 は、パケットを送信する準備ができたとき、パケットをパケット・メモリ 804 からリトリブして、インターフェイス・トランスミッタ 414 に転送する。インターフェイス・トランスミッタ 414 は、ポート・コントローラ 400 からパケットを出力する（矢印 906 参照）。

40

【0054】

図 10 は、本発明の一実施形態による、ポート・コントローラに入力される複数の入力パケットの処理および記憶の一例を示す概略図である。この図では、第 1 の入力パケット 1002 はポート・コントローラのインGRESS DMA 404 によって受信され、その後に第 2 のパケット 1004、第 3 のパケット 1008、第 4 のパケット 1008 が続く。上記の通り、各パケットはヘッダ部分（1002a、1004a、1006a、1008a）と本体部分（1002b、1004b、1006b、1008b）から構成される。各パ

50

ケットが受信される際、イングレスDMA 404によって本体部分からヘッダ部分が分離される。次いで分離された本体部分はケット・メモリ804に記憶され、一方、ヘッダ部分はケット・プロセッサ・ユニット408による処理のために送られる（注意：簡約化のため、図10ではヘッダ・キュー406は省略した）。ヘッダ部分のそれぞれの処理が完了すると、そのヘッダ部分のそれぞれはイングレスDMA 404を介してケット・メモリ804に記憶される（簡約化のため、図10ではDRAMコントローラ410も省略したことに注意されたい）。

【0055】

図10に示すように、本体部分と処理済みヘッダ部分は、イングレスDMAによって受信された順番にケット・メモリに記憶される。例えば、第1のケット1002のヘッダ部分と本体部分を受信し分離する際、第1の本体部分1002bはメモリ804に記憶され、一方、第1のヘッダ部分1002aはケット・プロセッサ・ユニット408による処理のために転送される。第1のヘッダ部分1002aが処理されている間、イングレスDMA 404は第2のケット1004を受信し、第2のケットのヘッダ部分と本体部分とを分離する。次いで第2の本体部分1004bはメモリに記憶され、第2のヘッダ部分1004aは処理のために送られる。この時点で、第1のヘッダ部分1002aの処理は完了しており、処理済みの第1のヘッダ部分は処理から転送されてメモリ804に記憶される。

10

【0056】

次に、第2のヘッダ部分1004aが処理されている間、イングレスDMAは第3のケット1006を受信し、第3のケットのヘッダ部分と本体部分とを分離する。次いで第3の本体部分1006bはメモリに記憶され、一方、第3のヘッダ部分1006aは処理のために送られる。この時点までに、第2のヘッダ部分1004aの処理は完了しており、その処理済みヘッダ部分は処理から転送されて、メモリに記憶される。第3のヘッダ部分1006aが処理されている間、イングレスDMAは第4のケット1008を受信し、第4のケットのヘッダ部分と本体部分とを分離する。次いで第4の本体部分1008bはケット・メモリに記憶され、一方、第4のヘッダ部分1008aは処理のために送られる。この時点までに、第3のヘッダ部分1006aの処理は完了しており、その処理済みの第3のヘッダ部分は処理から転送されて、メモリに記憶される。最後に、第4のヘッダ部分1008aの処理が完了すると、これもまた処理から転送されて、メモリに記憶される。

20

30

【0057】

図11は、本発明の一実施形態による、ケットの本体部分からヘッダ部分を分離するプロセス704の流れ図である。オペレーションA4302では、入力ケットは受信された際にセルに分割される。入力ケットの第1のセルには、ヘッダ部分が含まれる。このヘッダ部分は、1つまたは複数のMAC/IP/TCPヘッダならびにアプリケーション・データを含むことができる。好ましい一実施形態では、各セルは128バイトのサイズを有することができる。オペレーション1104では、ヘッダ部分（すなわち、第1のセル）のためにメモリに空間が予約される。次いで、オペレーション1106でメモリ内の予約済み空間を指し示すポインタが生成される。次いでこのヘッダ部分は、オペレーション1108の処理に送られる。

40

【0058】

図12は、本発明の一実施形態による、ケットの本体部分をヘッダ部分からの分離後にメモリに記憶するプロセス706を示す流れ図である。オペレーション1202では、ケットの本体部分の第1の受信されたセル（すなわち、入力ケットの第2のセル）がメモリに記憶される。オペレーション1204では、そのセルをヘッダ部分（すなわち、ケットの第1のセル用に予約された空間）のために予約された空間にリンクするためのポインタが生成される。入力ケットの後続の各入力セルも、記憶されているセルのそれぞれを総合的にリンクするために、新たに記憶されたセルから前回記憶されたセルへのポインタと共にメモリに記憶される（判断1206およびオペレーション1208および12

50

10を参照のこと)。

【0059】

図13は、本発明の一実施形態による、メモリにパケットの処理済みヘッダ部分を記憶するプロセス712を示す流れ図である。オペレーション1302で、処理済みヘッダ部分が受信される。オペレーション1304では、処理済みヘッダ部分のためにメモリに予約された空間が、その予約済み空間を指し示すポインタを用いて探し出される(オペレーション1204を参照のこと)。次いで処理済みヘッダは、オペレーション1306で予約済みの空間に記憶される。

【0060】

図14は、本発明の一実施形態による、入力パケットのセルのメモリへの記憶を示す概略図である。入力パケット1400は、イングレスDMA404によって受信された際に複数のセル(C_0 1402、 C_1 1404、 C_2 1406、 C_3 1408、 C_4 1410)に分割される(図11のオペレーション1102を参照のこと)。図14に示すように、第1のセル C_0 1402はパケットのヘッダ部分を含んでおり、一方、残りのセル C_1 1404、 C_2 1406、 C_3 1408、 C_4 1410はパケットの本体部分を含んでいる。

【0061】

パケット・メモリ1412では、空間1414がそのヘッダを含んでいるので第1のセル C_0 1402のために予約されており、その予約済み空間を指し示すポインタが生成される(図11のオペレーション1104および1106を参照のこと)。一実施形態では、予約済み空間のサイズは、処理中にそのヘッダ部分に追加される可能性のある任意の追加データに対して十分な空間を提供するために、予約済みの第1のセル C_0 1402のサイズよりも大きい場合がある(図7の708を参照のこと)。次いで予約済み空間1414を指し示すポインタ1416が生成され、その結果、現在処理されているヘッダ部分を含んでいる第1のセル C_0 1402が予約済み空間に引き続いて記憶することができる(図11のオペレーション1106を参照のこと)。

【0062】

次に、入力パケット1400の本体部分の第1のセグメントを含んでいる第2の受信されたセル C_1 1404が、パケット・メモリ1412に記憶され(図12のオペレーション1202を参照のこと)、記憶されているセル C_1 1404を予約済み空間にリンクするためのポインタ1418が生成される(図12のオペレーション1204を参照のこと)。第3のセル C_2 1406が受信されると、これもまたパケット・メモリ1412に記憶することができ、記憶されているセル C_2 1406を前回記憶されたセル C_1 1404にリンクするためのポインタ1420が生成される(図12のオペレーション1206、1208、および1210を参照のこと)。同じ要領で、第4と第5のセル C_3 1408、 C_4 1410が受信される際に、これらはそれぞれパケット・メモリ1412に記憶することができ、ポインタ1422、1424を生成させて、セル C_3 を C_2 に、また、セル C_4 をセル C_3 にそれぞれリンクする。図12に示すプロセスによれば、第5のセル C_4 1410はパケット1400の最終セルなので、第5のセル C_4 はポインタを含んでいる(1426を参照のこと)。

【0063】

図15は、本発明の一実施形態による、ポート・コントローラからスイッチ・ファブリック・システムのスイッチ・ファブリックにパケットを転送するプロセス1500の流れ図である。オペレーション1502で、パケットがポート・コントローラで既に処理済みであることを示す通知が受信される。この処理済みパケットはオペレーション1504および1506で出力優先順位キューに割り当てられ、リンクされる。その後、オペレーション1508でそのパケットはリンクを介して取り出され、オペレーション1510でポート・コントローラからスイッチ・ファブリックに転送される。

【0064】

本発明の一態様では、パケットは、リンクされたリスト内の割り当てられた出力優先順位

10

20

30

40

50

キューにリンクすることができる。

【0065】

本発明の一実施形態では、後続の処理されるパケットを同じ出力優先順位キューに割り当て、リンクすることができる。任意選択で、ポインタは、そのパケットを後続の処理されるパケットにリンクすることができる。本発明の別の実施形態では、そのパケットはメモリに記憶し、また、メモリから取り出すことができる。さらなる任意選択として、パケットを複数のリンクされたセルでメモリに記憶することができる。

【0066】

本発明の別の態様では、処理済みパケットは、スイッチ・ファブリックを介してアクセス可能なパケットに対する1つまたは複数の宛先を示す宛先ベクトルを含むことができる。 10

【0067】

本発明は、ポート・コントローラのメモリにパケットを記憶するためのデータ構造を含むこともできる。このデータ構造は、複数のキューから構成することができ、各キューはそれ自体に関連付けられたヘッド／開始アドレスを有する。各パケットをそのキューのうちの1つに割り当てて、1つまたは複数のパケットをメモリに記憶することができる。キューごとに、開始アドレスはそのキューの第1のパケットへのリンクを有することができ、そのキュー内の各パケットは、そのキュー内の次の後続パケットへのリンクを有することができる。一態様では、各パケットは、複数のリンクされたセルでメモリに記憶することができる。好ましい一実施形態では、ポインタは、パケットのセルをそのパケットの後続セルにリンクし、パケットの最終セルはヌル・ポインタを有する。 20

【0068】

図16は、本発明の一実施形態による、エグレスのためにキューに入れられたパケットのリンクされたリストを生成するプロセス1600の流れ図である。オペレーション1602では、イングレスDMA404は、出力指向スケジューラ412にパケットのヘッダ部分が既に処理済みであることを通知する。その後、イングレスDMAは、オペレーション1984で、出力指向スケジューラから戻されてきた、パケットがどの出力キュー（ポート・コントローラからのエグレス用）に割り当てられているかに関する情報を受信する。オペレーション1606で、ポインタが生成され、パケット・メモリに現在記憶されているヘッダ部分を割り当てられた出力キューにリンクする。割り当てられたキューが、既にそれに割り当てられている別のパケットを有しない場合、ポインタは、処理されたヘッダ部分を割り当てられたキューの開始アドレスにリンクする。 30

【0069】

一方、割り当てられたキューが、それに割り当てられた1つまたは複数の別のパケットを既に有している場合、ポインタは、処理済みヘッダ部分をそのキュー内のその直前に配置されているもう1つのパケットの記憶されたヘッダ部分にリンクする。オペレーション3802、3804および3806は、ポート・コントローラによって処理される各ヘッダ部分に対して反復される（オペレーション1608を参照のこと）。

【0070】

図17は、本発明の一実施形態により、パケット・メモリ804に記憶されているパケットをそれらの割り当てられている出力優先順位キューにリンクすることを示す概略図である。出力指向スケジューラ412は、例示の実施形態では4つのキューから成る、複数の出力優先順位キュー1702、1704、1706、1708を有する。好ましい一実施形態では、出力指向スケジューラとイングレスDMAのどちらも各キューの開始アドレスを認識している。 40

【0071】

各優先順位キューにはパケット・メモリに記憶されている複数のパケットが割り当てられており、特定キューに割り当てられている複数のパケットは、それらが割り当てられているキューの下に示される。各キューは先入れ先出し（FIFO）なので、記憶されている各パケットは、特定キュー内の1つのパケットから同じ割り当てられたキュー内の次に続くパケットの開始へのリンクを提供する次のパケット・ポインタ（「NPP」）を有する 50

。補足的な矢印は、次のセルが保存されているアドレスを各セルがどのように指し示すかを示している。例えば、図17のキュー01702を参照すると、キュー01702に割り当てられたパケット・メモリに記憶されている第1のパケット1710は、キュー内の次の記憶されているパケット1714の開始を指し示すポインタ1712を有し、パケット1714は、そのキュー内の第3の記憶されているパケット1718の開始を指し示すポインタ1716を有する、という具合である。

【0072】

図18は、本発明の一実施形態による、図17に示されるパケット・メモリに記憶される例示的パケット1710を示す概略図である。例示的パケット1710は、第1のセル1802がパケットの処理済みヘッダ部分を含み、残りのセル1804、1806、1808、1810がパケットの本体を含むように、複数のセルに分割される。セル1802、1804、1806、1808は、それぞれに、次に続くセルの開始を指し示す次のセル・ポインタ（「NCP」）1812、1814、1816、1818を有する。各NCPは、パケットの最終セルの最後の1つを除いて、次のセルを指し示す。最後のNCPは、いかなる値をも有することができるが、これは出力指向スケジューラによって無効であるといみなされる可能性がある。何故ならば、出力指向スケジューラは、内部タグ情報からパケットのサイズを認識しているからである。

【0073】

一実施形態では、イングレスDMA404は、リンク／ポインタの生成に応答可能である。好ましい一実施形態では、ポインタのサイズは16ビット（2バイト）であってよい。

【0074】

上記の通り、第1のセル1802は、パケットのサイズと、そのパケットがスイッチ・ファブリック・システムから出力される宛先ポートを示すことのできる内部タグを含む処理済みヘッダ部分を有する。また、第1のセル1802は、キュー01702内の次に記憶されているパケット1714を指し示す次のパケット・ポインタ1712を含む。

【0075】

各キューは先入れ先出し（FIFO）であってよい。出力指向スケジューラ412が、そのFIFOキューのうちの1つに割り当てられたパケットをパケット・メモリから取り出す準備ができると、出力指向スケジューラは、パケットの次のセルを探すために各セルのNCPを読み取ることができる。一実施形態では、最終セルのNCPは、出力指向スケジューラが取り出しているパケットのサイズの知識に基づいてその出力指向スケジューラから無視される場合がある。

【0076】

図19は、本発明の一実施形態による、ポート・コントローラ400を介してスイッチ・システムから抜けようとしているパケットを処理するプロセスを示す概略的な流れ図である。このプロセスでは、パケットは、ポート・コントローラ400によって受信される前に既に処理が完了している。抜けようとしているパケットがインターフェイス・レシーバ420によって最初にポート・コントローラ内に受信される。インターフェイス・レシーバ420は、そのパケットをエグレス・コントローラ422に送り、次いでエグレス・コントローラ422は、そのパケットをエグレス・バッファ424に転送することができる。エグレス・コントローラが、抜けようとしているパケットをポート・コントローラから送出する準備ができると、エグレス・コントローラ422は、エグレス・バッファ424からパケットを取り出し、そのパケットを、ポート・コントローラから送出するためにMACインターフェイス402に転送する（矢印1904を参照のこと）。

【0077】

（スイッチ・ファブリック）

図20は、本発明の一実施形態によるスイッチ・ファブリック構成要素2000のアーキテクチャを示す概略的ブロック図である。スイッチ・ファブリック構成要素2000には、インターフェイス・レシーバ2002、スイッチ・マトリックス2004、インターフェイス・トランスミッタ2006、中央コントローラ2008、フリー・セル・マネージ

ャ 2010、キュー・マネージャ 2012、および外部インターフェイス 2014 が含まれる。

【0078】

より明瞭かつ容易に理解できるように、図 20 には 1 つのインターフェイス・レシーバ 2002 と 1 つのインターフェイス・トランスミッタ 2006 しか示さない。しかし、スイッチ・ファブリック構成要素が複数のインターフェイス・レシーバと複数のインターフェイス・トランスミッタを含むことができるということを理解されたい。例えば、図 1 に示されている 8×8 のスイッチ・ファブリック構成要素（例えば 104a）では、各スイッチ・ファブリック構成要素は 8 つのインターフェイス・レシーバと 8 つのインターフェイス・トランスミッタとを有する。同様に、図 2 に示されているスイッチ・ファブリック構成要素 204 は、4 つのインターフェイス・レシーバと 4 つのインターフェイス・トランスミッタとを含んでいる。 10

【0079】

インターフェイス・レシーバ 2002 は、パケットを受信するように働く。このインターフェイス・レシーバは、別のスイッチ・ファブリック構成要素のインターフェイス・トランスミッタ、またはポート・コントローラのインターフェイス・トランスミッタ 4014 への 1 つの接続用外部 I/O を有する。このインターフェイス・レシーバは、受信したパケットをインターフェイス・レシーバ 2002 からスイッチ・マトリックス 2004 に転送することを可能にするために、スイッチ・マトリックス 2004 に結合されている。インターフェイス・レシーバ 2002 の機能は、ポート・コントローラのインターフェイス・レシーバ 420 の機能と類似である。インターフェイス・レシーバ 2002 は、外部 I/O によってリンクを構成し、制御のために中央コントローラ 2008 と対話する。 20

【0080】

スイッチ・マトリックス 2004 は、スイッチ・ファブリック構成要素内のパケットへのデータ・パスを提供する。スイッチ・マトリックスは、1 つまたは複数のインターフェイス・レシーバ 2002 と 1 つまたは複数のインターフェイス・トランスミッタ 2006 とに結合されている。スイッチ・マトリックスは、インターフェイス・レシーバからデータ（すなわち、パケット）を受信する。各入力パケットは、スイッチ・ファブリック構成要素からの送出的のためにパケットをインターフェイス・トランスミッタ 2006 に転送できるようになるまでは、スイッチ・マトリックスのメモリ内に記憶される。好ましい一実施形態では、スイッチ・マトリックスは 64 K バイトのメモリを有する。 30

【0081】

上記の通り、インターフェイス・トランスミッタ 2006 は、スイッチ・マトリックス 2004 に結合されている。インターフェイス・トランスミッタは、別のスイッチ・ファブリック構成要素のインターフェイス・レシーバまたは外部 I/O を介したポート・コントローラのインターフェイス・レシーバに接続可能である。インターフェイス・トランスミッタ 2006 は、スイッチ・マトリックスからのパケットをスイッチ・ファブリック構成要素から送出する。各インターフェイス・トランスミッタ 2006 は、別のスイッチ・ファブリック構成要素のインターフェイス・レシーバまたはポート・コントローラに外部的に接続されることができ、外部 I/O によってリンクを構成することができる。各インターフェイス・トランスミッタ 2006 は、制御のために中央コントローラ 2008 と対話する。 40

【0082】

中央コントローラ 2008 は、スイッチ・ファブリック構成要素の全てのユニットに接続されており、それを制御する。中央コントローラ 2008 は、イングレス・ユニット 2008a、マトリックス・ユニット 2008b、およびエグレス・ユニット 2008c の 3 つの構成要素を含むことができる。中央コントローラ 2008 は、フリー・セル・マネージャ 2010 によって提供されるフリー・セル・アドレスに従って、スイッチング・マトリックスのメモリ内の入力パケットを記憶すること管理する。中央コントローラ 2008 は、また、パケットの宛先ベクトルから宛先ポートを読み取り、そのパケットをキュー・ 50

マネージャ 2012 によってキューに記憶する。さらに、中央コントローラは、キュー・マネージャ 2012 の情報に従ってパケットをスイッチ・マトリックスからインターフェイス・トランスミッタ 2006 に転送するための信号を生成することもできる。

【0083】

フリー・セル・マネージャ 2010 は、中央コントローラとキュー・マネージャ 2012 に結合されている。フリー・セル・マネージャ 2010 は、中央コントローラ 2008 を介して提供された情報に基づいてスイッチング・マトリックスのメモリ内のフリー・セルを管理する。具体的には、フリー・セル・マネージャ 2010 は、スイッチ・マトリックス内の占有されたセルとフリー・セルに関する情報を表形式で維持し、この情報を中央コントローラ 2008 とキュー・マネージャ 2012 に提供する。

10

【0084】

キュー・マネージャ 2012 は、中央コントローラ 2008 とフリー・セル・マネージャ 2010 とに結合されている。キュー・マネージャ 2012 は、入力パケットに関する情報を受信し、出力ポートに従って FIFO でリンクされるリストとしてその情報を記憶する。キュー・マネージャ 2012 は、抜けようとしているパケットのキューも管理する。

【0085】

外部インターフェイス 2014 は、中央コントローラとスイッチ・マトリックスに接続されている。外部インターフェイスの機能は、ポート・コントローラのネットワーク・インターフェイスと同様である。外部インターフェイスは、制御パケットを外部 I/O を介して内部チャンネルと交換し、次いでスイッチ・ファブリック構成要素の内部レジスタを維持するためにこれらのパケットを分析する。

20

【0086】

図 21 は、本発明の一実施形態による、スイッチ・ファブリックを介してデータグラムを送信するプロセス 2100 の流れ図である。受信したデータグラムから宛先ベクトルが抽出される（オペレーション 2102 および 2104 を参照のこと）。宛先ベクトルは、データグラムの 1 つまたは複数の宛先（すなわち、単一キャストまたはマルチキャスト）を示す。オペレーション 2106 で、宛先ベクトルに基づいてデータグラムに対してポート・ビットマップが生成される。オペレーション 2108 では、このデータグラムは引き続いてポート・ビットマップを利用して宛先に向けて送信される。

【0087】

本発明の一態様では、宛先ベクトルは 32 ビットのデータを含むことができる。本発明の別の態様では、ポート・ビットマップは 8 ビットのデータを含むことができる。

30

【0088】

本発明の一実施形態では、ポート・ビットマップを生成することは、宛先ベクトルからデータグラムの全ての宛先を獲得すること、および、次いでそのデータグラムが宛先に到達するにはそのスイッチ・ファブリック構成要素のどのスイッチ・ポートを介して経路指定することができるかを決定するためにスイッチング・テーブルを利用することを含むことができる。次いでデータグラムがその対象とする宛先に到達するために、そこを介するよう経路指定することができる全ての決定されたスイッチ・ポートを示すためにポート・ビットマップを生成することができる。

40

【0089】

本発明の別の実施形態では、データグラムを転送することは、ポート・ビットマップに基づいてマスクを作成すること、および、データグラムの送信に際してマスクを用いてポート・ビットマップを変更することを含むことができる。このような一実施形態では、マスクを用いてポート・ビットマップを変更するステップは、データグラムがそこに向けて送信されている宛先に関連付けられたデータのポート・ビットマップをクリアすることも含むことができる。

【0090】

本発明のさらなる実施形態では、データグラムが宛先に到達するためにそこを介してデータグラムを経路指定することができるスイッチ・ポートの可用性を判定することもできる

50

。次いでデータグラムは使用可能と判定されたスイッチ・ポートを介して送信することができる。

【0091】

図22は、本発明の実施形態によるスイッチ・ファブリックを介してパケットを経路指定するためのスイッチング・テーブルの実施を示す8行8列のスイッチ・ファブリック構成要素を含むスイッチ・ファブリック2200の概略ブロック図である。この実施形態の下で議論される基礎となっている原理は、本明細書によるいかなるスイッチ・ファブリックにおいても利用することが可能である。この議論のため、スイッチ・ファブリックおよびスイッチ・ファブリック構成要素は、すべて双方向であることに留意することが重要である。

10

【0092】

図22に示すとおり、スイッチ・ファブリック2200は、SF1 2202a、SF2 2202b、SF3 2202c、SF4 2202dとして示される8行8列のスイッチ・ファブリック構成要素を含む。各スイッチ・ファブリック構成要素は、8つのポート（「スイッチ・ポート」）を有する。例えば、図22では、SF1 2202aの8つのスイッチ・ポートが、1/1、2/1、3/1、4/1、5/1、6/1、7/1、8/1（「1の1、1の2、1の3...」等）として示される。SF2、SF3、SF4の8つのスイッチ・ポートも同様に、図22で示される。例えば、SF2 2202bのポートは、n/2（ただし、nは、1から8までの整数）を使用して示され、SF3 2202cのポートは、n/3を使用して示され、またSF4 2202dのポートは、n

20

【0093】

図22の示されているスイッチ・ファブリックでは、各スイッチ・ファブリック構成要素が、2つの隣接するスイッチ・ファブリック構成要素に結合された2つのポートを有している。例えば、SF1 2202aのポート5/1および6/1は、それぞれ、SF2 2202bのポート2/2および1/2に結合され、またSF1 2202aのポート7/1および8/1は、それぞれ、SF3 2202cの4/3および3/3に結合される。

【0094】

図22の参照を続けると、スイッチ・ファブリック2200は、複数のデバイス・ポートも含む。この議論では、デバイス・ポートは、外部デバイス（例えば、ポート・コントローラなど）に結合することができ、スイッチ・ファブリック内の別のスイッチ・ファブリック構成要素に結合されていないスイッチ・ファブリック構成要素のポートと定義することができる。図22に示す例としての実施形態では、スイッチ・ファブリックは、十六（16）のデバイス・ポート（これらには、図22で1ないし16の番号が付けられている）を有し、各スイッチ・ファブリック構成要素は、4つのデバイス・ポートを有している。例えば、SF1 2202aは、デバイス・ポート1、2、3、4を有し、SF2 2202bは、デバイス・ポート5、6、7、8を有し、SF3 2202cは、デバイス・ポート9、10、11、12を有し、またSF4 2202dは、デバイス・ポート13、14、15、16を有する。

30

40

【0095】

（スイッチング・テーブル）

本発明の実施形態では、スイッチング・テーブルを利用して、スイッチ・ファブリックの柔軟性を高めるのを促進することができる。スイッチング・テーブルを各スイッチ・ファブリック構成要素に関連付けて、スイッチ・ファブリック構成要素の各ポートを介してどのデバイスに到達できるかを記録することができる。言い換えれば、各スイッチング・テーブルは、特定のスイッチ・ファブリック構成要素のどのポートを使用して（すなわち、アクセス可能であって）当該の特定のスイッチ・ファブリック構成要素からパケットを伝送することができるかに関する情報を提供する。

【0096】

50

図 2 3、2 4、2 5、2 6 は、本発明の実施形態による図 2 2 に描いたスイッチ・ファブリック 2 2 0 0 に関する例としてのスイッチング・テーブルを示す概略図である。詳細には、図 2 3 に示す例としてのスイッチング・テーブル 2 3 0 0 は、スイッチ・ファブリック構成要素 S F 1 2 2 0 2 a に対するものであり、図 2 4 に示す例としてのスイッチング・テーブル 2 4 0 0 は、スイッチ・ファブリック構成要素 S F 2 2 2 0 2 b に関するものであり、図 2 5 に示す例としてのスイッチング・テーブル 2 5 0 0 は、スイッチ・ファブリック構成要素 S F 3 2 2 0 2 c に対するものであり、また図 2 6 に示す例としてのスイッチング・テーブル 2 6 0 0 は、スイッチ・ファブリック構成要素 S F 4 2 2 0 2 d に対するものである。

【0097】

各スイッチング・テーブルは、スイッチ・ファブリックのデバイス・ポートに対する複数の列 2 3 0 2、および特定のスイッチ・ファブリック構成要素のポートに対する複数の行 2 3 0 4 を有する格子（例えば、複数のビットマップ・アレイ）として表すことができる。列と行は一緒に、スイッチング・テーブルのビットマップ・アレイを形成する。この格子の表現では、ビットマップ・アレイは、スイッチング・テーブル格子の列と行によって形成された 1 行のボックスを含む。各格子の列と行によって形成されたボックス内には、1 および 0 がある。ボックス内の「1」は、特定のデバイス・ポートが、特定の行のポートを介してアクセス可能であることを示し、一方、ボックス内の「0」は、特定のデバイス・ポートが、特定の行のポートを介してアクセス可能ではないことを示す。そのような表現を使用して、特定のポートに関連付けられたアレイに基づいてビットマップ、スイッチ・ファブリック構成要素のポートを生成することができる。

【0098】

例えば、スイッチ・ファブリック構成要素 S F 1 2 2 0 2 a に対するスイッチング・テーブル 2 3 0 0 を表す図 2 3 を参照すると、ポート 1 / 1 に対して指定された一番上の行を見ることにより、ポート 1 / 1 に対するビットマップ・アレイを確認することができる。このアレイでは、デバイス・ポート 1 に対する列の下ボックス内に「1」が存在し、また残りの列の下ボックスのそれぞれの中に「0」が存在する。このことは、デバイス・ポート 1 だけが、S F 1 2 2 0 2 a のポート 1 / 1 を介してアクセス可能であることを示す（すなわち、デバイス・ポート 1 を通る伝送を介してその宛先がアクセス可能である S F 1 のパケットは、ポート 1 / 1 を介するように経路指定しなければならない）。同様に、ポート 5 / 1 に対して指定された行（すなわち、アレイ）を参照すると、デバイス・ポート 5、6、7、8、9、10 に対する列の下ボックスのそれぞれの中に「1」が存在し、またデバイス・ポート 1、2、3、4、11、12、13、14、15、16 の列の下ボックスのそれぞれの中に「0」が存在する。このことは、デバイス・ポート 5、6、7、8、9、10 が、S F 1 2 2 0 2 a のポート 5 / 1 を介する S F 1 内のパケットに関してアクセス可能であることを示す。

【0099】

図 2 2 および 2 3 - d に示す例としての実施形態では、スイッチング・テーブルが、スイッチ・ファブリック構成要素のどのポートからのアクセスも、6 つのデバイス・ポート未満に制限するように生成されていることに留意されたい。これは、図 2 2 のスイッチ・ファブリック 2 2 0 0 を介してパケットの伝達の理解を簡単にするのを助けるために行っている。図 2 2 のスイッチ・ファブリック 2 2 0 0 の場合、スイッチ・ファブリック構成要素のポートは、最高で 12 のポートに対するアクセスを有するのが可能である（例えば、ポート 5 / 1 を介して伝送される S F 1 内のパケットは、スイッチング・テーブルがそのように生成された場合、デバイス・ポート 5 ないし 16 に対するアクセスを有するのが可能である）ことを理解されたい。

【0100】

また、図 2 3 - d に示すスイッチング・テーブルのそれぞれは、最高で 32 デバイス・ポートに対する列が含まれる好ましい実施形態に従って生成されていることにも留意されたい。スイッチング・テーブルのこの好ましい実施形態は、32 のポート・コントローラが

10

20

30

40

50

接続された 3 2 のデバイス・ポートを備えたスイッチ・ファブリックを有する図 1 に示すスイッチ・ファブリックの好ましい実施形態におけるスイッチング・テーブルの使用を可能にするのを助けることである。図 2 2 に示す例としてのスイッチ・ファブリック 2 2 0 0 は、1 6 だけのデバイス・ポートを有する（例えば、1 6 のポート・コントローラをそこに接続して）ので、列 1 7 ないし 3 2 内のボックスはすべて、4 つのスイッチング・テーブル 2 3 - d のそれぞれの中に「0」を含む。

【0 1 0 1】

図 2 7 は、本発明の実施形態によるスイッチ・ファブリック構成要素を介するパケットを処理するための処理 2 7 0 0 に関する流れ図である。オペレーション 2 7 0 2 で、パケットが、スイッチ・ファブリック構成要素によって受信される。オペレーション 2 7 0 4 で、宛先ベクトルが、受信されたパケットから抽出される。次に、オペレーション 2 7 0 6 で、スイッチ・ファブリック構成要素のスイッチング・テーブルに基づき、抽出された宛先ベクトルからポート・ビットマップが生成される。オペレーション 2 7 0 8 で、スイッチ・ファブリック構成要素の論理が、ポート・ビットマップ内に示されたスイッチ・ファブリック構成要素のポートのどれかがアイドルであるかどうかを検査して判定する。ポート・ビットマップによって示されるポートのどれかが、アイドルであると判明した場合には、オペレーション 2 7 1 0 で、ポート・ビットマップに基づき、識別されたアイドルなポートのためのマスクが作成される。オペレーション 2 7 1 2 で、次に、スイッチ・ファブリックの識別されたアイドルなポートを介して発信パケットが伝送され、パケットがそれを介して伝送されたばかりのポートに関連する情報が、ポート・ビットマップから消去される。消去プロセスは、識別されたポートにおける発信パケットの宛先ベクトルに対しても実行されて、発信パケットの宛先ベクトルが変更される。判定 2 7 1 4 で、ポート・ビットマップ内に別の宛先が含まれると判定された場合には、追加の宛先のそれぞれに対してオペレーション 2 7 1 0 および 2 7 1 2 が反復される。

【0 1 0 2】

パケットがスイッチ・ファブリック構成要素に入った際、パケットの宛先ベクトルおよびスイッチ・ファブリック構成要素のスイッチング・テーブルを使用して、スイッチ・ファブリック構成要素のどのポートを介してパケットが出るべきかが判定される。一実施形態では、判定結果をリンクされたリストとして記憶することが可能である。一態様では、判定結果は、ポート・ビットマップを含むことが可能である。別の態様では、判定結果は、宛先ベクトルを含むことが可能である。

【0 1 0 3】

図 2 8 は、本発明の実施形態による、判定結果がポート・ビットマップを含むスイッチ・ファブリック構成要素を介してパケットを処理するためのプロセス 2 8 0 0 を示す概略図である。スイッチ・ファブリック構成要素内へのインGRESに先立ち、ポート・コントローラ 2 8 0 4 においてパケットの処理中にパケット 2 8 0 2 内にタグが挿入される。タグは、パケットが宛先とする 1 つまたは複数のデバイス・ポートを示す宛先ベクトル 2 8 0 6 を含む。好ましい実施形態では、宛先ベクトル 2 8 0 6 は、3 2 ビットの長さ（すなわち、「3 2 ビットの宛先ベクトル」）を有することが可能である。ポート・コントローラは、スイッチ・ファブリック構成要素 2 8 0 8 のデバイス・ポートに結合されて、挿入されたタグを有するパケットをスイッチ・ファブリック構成要素に転送することを可能にする。

【0 1 0 4】

スイッチ・ファブリック構成要素 2 8 0 8 内へのパケット（挿入されたタグを有する）2 8 0 2 のインGRESの後、タグが抽出されて宛先ベクトル 2 8 0 6 が獲得される。次に、宛先ベクトル 2 8 0 6 およびスイッチ・ファブリック構成要素 2 8 0 8 のスイッチング・テーブル 2 8 1 2 に基づき、ポート・ビットマップ 2 8 1 0 が生成される。好ましい実施形態では、ポート・ビットマップは、8 ビットの長さ（すなわち、「8 ビットのポート・ビットマップ」）を有する。次に、パケットが、スイッチ・ファブリック構成要素のスイッチ・マトリックスのメモリ内のセルの中に記憶される。生成されたポート・ビットマッ

プ 2 8 1 0 を使用して、リンクされたリスト 2 8 1 4 の中でパケットが中に記憶されたメモリのセルがリンクされる。また、ポート・ビットマップ 2 8 1 0 も、スイッチ・ファブリック構成要素 2 8 0 8 内のリンクされたリスト 2 8 1 4 の中で記憶される。

【0105】

適切な出力ポートが、利用可能であると判定されたとき、パケットは、出力ポートを介してスイッチ・ファブリック構成要素を出ることができる。適切な出力ポートが利用可能であると判定された後、適切なセルに関する情報（ポート・ビットマップを含む）が、リンクされたリスト 2 8 1 4 から読み取られてポート・ビットマップ 2 8 1 0 が獲得される。次に、スイッチング・テーブル 2 8 1 2 を使用して、イグレスのポートのためのマスク 2 8 1 6 が生成される。好ましい実施形態では、マスク 2 8 1 6 は、8 ビットの長さ（すなわち、「8 ビットのビット・マスク」）を有する。次に、マスク 2 8 1 6 を使用してポート・ビットマップ 2 8 1 0 が変更され、出力ポートを介するパケットの送信に関連する情報が、ポート・ビットマップから除去されるようにする。次に、変更されたポート・ビットマップ 1 0 1 8 が、リンクされたリストの中に記憶される。変更されたポート・ビットマップ 2 8 1 8 が、すべて「0」のビット（すなわち、「1」が全くない）を含む場合には、スイッチ・マトリックス内の対応するメモリ・セルが、解放されている。そのような実施形態の好ましい態様では、リンクされたリストは、ポート・ビットマップを記憶するための縦 8 ビット／語、横 2 5 6 語のメモリを含むことが可能である。そのような実施形態の別の好ましい態様では、スイッチング・テーブルは、マスクを生成する論理（すなわち、回路）を含むことが可能である。この実施形態のさらに別の好ましい態様では、スイッチング・テーブルの行は、互いに排他的であることが可能である。

【0106】

図 2 8 に示すプロセス 2 8 0 0 に対する好ましいオプションとして、パケットがスイッチ・ファブリック構成要素の発信ポートを出る際にパケットから宛先ベクトルを消去することができる。そのようなオプションを実施することの利点は、図 2 2 に描いたスイッチ・ファブリック 2 2 0 0 を使用する以下の実施例で見ることができる。この実施例では、スイッチ・ファブリック 2 2 0 0 に入るパケットが、S F 1 2 2 0 2 a 内のデバイス・ポート 2 および S F 2 2 2 0 2 b 内のデバイス・ポート 7 という 2 つの宛先を有するものと想定する。図 3 0 は、本実施例に従ってデバイス・ポート 1 から S F 1 2 2 0 2 に入り、S F 1 2 2 0 2 a 内のデバイス・ポート 2 および S F 2 2 2 0 2 b 内のデバイス・ポート 7 という宛先を有するパケットに関する宛先ベクトル 3 0 0 0（詳細には、3 2 ビットの宛先ベクトル）を示す概略図である。

【0107】

（ポート・ビットマップ）

図 2 9 は、本発明の実施形態によるスイッチ・ファブリック構成要素内のポート・ビットマップ（詳細には、8 ビットのポート・ビットマップ）の例としての利用を示す概略図である。この例は、図 2 2 に示す例としてのスイッチ・ファブリック 2 2 0 0 に基づき、スイッチ・ファブリック構成要素 S F 1 2 2 0 2 a 内で生じる。この例では、パケットが、デバイス・ポート 1 に結合されたポート・コントローラを介して S F 1 によって受信され、（1）デバイス・ポート 2（S F 1 2 2 0 2 a にある）および（2）デバイス・ポート 9（S F 4 2 2 0 2 d にある）という 2 つの宛先を有する。

【0108】

S F 1 2 2 0 2 a 内で、パケットから宛先ベクトルが抽出される。図 2 9 に示すとおり、抽出された宛先ベクトル 2 9 0 2 は、パケットが目指す宛先ポート（デバイス・ポート 2 および 9）を示すビット 2 2 9 0 4 内およびビット 9 2 9 0 6 内に「1」を有する。抽出された宛先ベクトル 2 9 0 2 の情報に基づき、スイッチ・ファブリック構成要素は、スイッチング・テーブルを使用して、パケットが目指す宛先に着信するのに、そのスイッチ・ポートのどれを介するようにパケットを経路指定することができるかを判定する。図 2 3 に示すスイッチ・ファブリック構成要素 S F 1 に対するスイッチング・テーブル 2 3 0 0 を参照すると、デバイス・ポート 2 は、S F 1 のスイッチ・ポート 2 / 1 を介して

アクセスすることができ（行 2 / 1 参照）、またデバイス・ポート 9 は、S F 1 のスイッチ・ポート 5 / 1 および 6 / 1 を介してアクセスできる（行 5 / 1 および 6 / 1 参照）ことが分かる。

【0109】

ポート・ビットマップは、スイッチ・ファブリック構成要素によって受信されたパケットの抽出された宛先ベクトルからの情報および情報、スイッチ・ファブリック構成要素のスイッチング・テーブルに基づいてスイッチ・ファブリック構成要素によって生成されるビットマップである。好ましくは、ポート・ビットマップは、スイッチ・ファブリック構成要素の中央コントローラによって生成される。ポート・ビットマップは、パケットが目指す宛先（すなわち、デバイス・ポート）に着信するため、スイッチ・ファブリック構成要素のスイッチ・ポートのどれを介するように経路指定される必要があるかを示す。ポート・ビットマップは、複数のビットを含み、各ビットは、関連スイッチ・ファブリック構成要素の対応するスイッチ・ポートに関連付けられている。例えば、4つのスイッチ・ポートを有するスイッチ・ファブリック構成要素では、そのスイッチ・ファブリックに対するポート・ビットマップは、（少なくとも）4ビットを有し、各ビットは、スイッチ・ファブリック構成要素の4つのスイッチ・ポートのどれかに関連付けられている。別の例として、図 2 2 に示す実施形態では、スイッチ・ファブリック構成要素のそれぞれが、8つのスイッチ・ポートを有する。したがって、各スイッチ・ファブリック構成要素に対して生成されるポート・ビットマップは、少なくとも8ビットを有し、8ビットのそれぞれは、それぞれのスイッチ・ファブリック構成要素の対応するスイッチ・ポートに関連付けられている。好ましい実施形態では、ポート・ビットマップは、八（8）ビットを含む。

【0110】

ポート・ビットマップのビットは、それぞれ、「1」または「0」を含むことが可能である。一実施形態では、ポート・ビットマップのあるビットが「1」を含む場合には、そのことは、その特定のビットに関連するスイッチ・ポートが、スイッチ・ファブリック構成要素からパケットを出力するためのポートとして働くことができるのを示す。反対に、ポート・ビットマップのあるビットが「0」を含む場合には、そのことは、その特定のビットに関連するスイッチ・ポートが、スイッチ・ファブリック構成要素からパケットを出力するためのポートとして働くことができないのを示す。

【0111】

再び図 2 9 を参照すると、受信されたパケットのスイッチ・ファブリック要素 S F 1 により、抽出された宛先ベクトル 2 9 0 2 およびスイッチ・ファブリック要素 S F 1 の切換えテーブル 2 3 0 0 に基づいて生成することのできるポート・ビットマップ 2 9 0 8 が示されている。このポート・ビットマップ 2 9 0 8 は、それぞれ、S F 1 のスイッチ・ポート 2 / 1、5 / 1、および 6 / 1 に関連するビット 2、5、および 6 2 9 1 0、2 9 1 2、2 9 1 4 に「1」を含む。

【0112】

この例では、スイッチ・ポート 5 / 1 または 6 / 1 がスイッチ・ポート 2 / 1 よりも前に利用可能になる（すなわち、アイドル状態になる）と判定されるものと仮定する。この場合、この例ではまず、パケットがスイッチ・ポート 5 / 1 または 6 / 1 を介して装置ポートの所期の宛先に向かって（S F 2 を介し、次いで S F 4 を介して）出力される。図 2 7 に示されたフローチャートに示されているように、引き続き図 2 9 を参照するとわかるように、第 1 のマスク 2 9 1 6 は、ポート・ビットマップおよび出力スイッチ・ポートに基づいて生成される。この例では、この第 1 のマスクは、パケットがスイッチ・ポート 5 / 1 または 6 / 1 を介して出力されるときにポート・ビットマップにおける処理されるビットを示すビット 5 および 6 2 9 1 8、2 9 2 0 に「1」を含む。

【0113】

引き続きこの例で説明する。次に、第 1 のマスク 2 9 1 6 を使用して、処理されるビットがポート・ビットマップから削除される。一般に、削除では、ポート・ビットマップ内の、「1」を含むビットが削除され、この場合、マスクの対応するビットも「1」を含む。

言い換えれば、削除では、ポート・ビットマップ内の、「1」を含むビットが「1」から「0」に変化し、この場合、マスクの対応するビットも「1」を含む。たとえば、図29には、処理されるビットが、第1のマスクを使用して削除された後の（修正された）ポート・ビットマップ2908'のビットが示されている。図示のように、修正されたポート・ビットマップにおいて、「1」はビット2 2910にのみ残っており、「0」はビット6 2912、2914にあり、ビット2は除去されておらず（すなわち、スイッチ・ポート2/1はまだ処理されていない）、ビット5および6が除去されている（すなわち、スイッチ・ポート5/1または6/1が処理されている）ことがわかる。

【0114】

次に、スイッチ・ポート2/1が利用可能である（すなわち、アイドル状態である）と判定されると、スイッチ・ポート2/1の処理に基づいてポート・ビットマップ用の第2のマスク2916が生成される。このマスクはビット2 2922に「1」を有し、スイッチ・ポート2/1を介したパケットの出力を示している。この場合、スイッチ・ポート2/1（すなわち、装置ポート2）からパケットが出力されるとき、第2のマスク2916を使用してポート・ビットマップを除去することができる。図13Aには、第2のマスク2916によって削除された後の修正されたポート・ビットマップ2908"が示されている。図29に示されているように、修正されたポート・ビットマップ2906"のすべてのビットが「0」を含み、それにより、ポート・ビットマップには、パケットを出力する宛先が残っていないことがわかる。

【0115】

図30は、本発明の実施形態による例示的な宛先ベクトル3000、宛先ベクトルの発信マスク3002、および修正された宛先ベクトル3004の概略図である。図30に示されているように、この例では、装置ポート2 3006、3008のビットは「1」を含み、それにより、関連するパケットについての装置2および7の宛先を示している。SF1において、パケットは（装置ポート7に到達するように）2/1（すなわち、装置ポート2）を介し、また5/1または6/1を介してSF2に送信される。次に、SF2に送信されたパケットが、宛先ベクトル3000を修正せずに送信された場合、SF2は、修正されていない宛先ベクトル3000を抽出し、SF2の宛先の1つが装置ポート2である場合と同様にパケットを処理し、したがって、パケットのコピーを、装置ポート2を通して送信できるようにSF1に送り返す。

【0116】

この問題を解消するには、スイッチ・ファブリック要素の各出力ポートの端部に削除プロセスを挿入することができる。削除プロセスでは、発信ポートを通して到達できる装置に対応するビットを除いて、発信パケットの宛先ベクトルのすべてのビットが削除される。この削除プロセスでは、発信マスク3002が生成され、次いで発信マスク3002を使用して発信パケットの宛先ベクトルが修正される。発信マスク3002は、特定の発信ポートを通してアクセスできるすべてのビットに「1」を含む（残りのビットは「0」である）。引き続きこの例で説明すると、SF1におけるパケットは、SF1のいずれかのポート5/1または6/1から装置ポート7に到達することができる（図23のSF1の切換えテーブルを参照されたい）。図23に示されているSF1の切換えテーブルを参照すると、ポート5/1または5/6を介して到達できる装置ポートは装置ポート5～10であることがわかる。したがって、生成された発信マスク3002は、図30に示されているように、ビット5から10に「1」を含み、残りのビットに「0」を含む。次いで、この発信マスクを使用し、発信マスクの「1」でカバーされていないビット内の任意の「1」の宛先ベクトルを削除することにより、ポート5/1または6/1からSF2に向かうパケットの宛先を修正することができる。したがって、図30に示されているこの例の修正された宛先ベクトル3004は、装置ポート2用のビットの「1」が発信マスクの「1」によってマスクされていないので装置ポート7 3010用のビットに1つの「1」を有するに過ぎない。このような削除プロセスを用いた場合、修正された宛先ベクトルは、SF2に、パケットのコピーを、装置ポート2に送信されるようにSF1に送り返させる

情報を含まない。好ましい実施形態では、スイッチ・ファブリック要素の各出力ポートの発信マスクが、それぞれの各ポートに対応する切換えテーブルの関連する行と同じであってよいことに留意されたい。このような好ましい実施形態の一態様では、スイッチ・ファブリック要素の各出力ポートは、それぞれの切換え要素の切換えテーブルの関連する行と同じ内容を含むレジスタを有してよい。

【0117】

図31は、本発明に実施形態によって判定結果が宛先ベクトルを含む、スイッチ・ファブリック要素を通してパケットを処理する他のプロセス3100を示す概略図である。図28に示されている実施形態と同様に、パケット3102が、スイッチ・ファブリック要素に着信する前に、スイッチ・ファブリック要素に結合されたポート・コントローラで処理されている間に、宛先ベクトル（好ましくは、32ビット宛先ベクトル）を示すタグがパケット3102に挿入される。パケット3102がスイッチ・ファブリック要素に着信すると、タグが抽出され、宛先ベクトル3104が得られる。図28に示されたプロセス2800と同様に、次いで、宛先ベクトル3104およびスイッチ・ファブリック要素3100の切換えテーブル3108に基づいてポート・ビットマップ3106が生成される。好ましい実施形態では、ポート・ビットマップ3106は8ビット・ポート・ビットマップを備える。生成されたポート・ビットマップ3106は、連係リスト3110内で、パケットが記憶されているメモリのセルをリンクするために使用される。しかし、8ビット・ポート・ビットマップ3106は、連係リスト3110には格納されない。その代わり、（32ビット）宛先ベクトル3104が連係リスト3110に格納される。

【0118】

引き続き図31を参照するとわかるように、パケットの発信にあたって適切な出力ポートが利用可能であることが判定されると、連係リスト3110からセル情報が読み取られ、宛先ベクトル3104が得られる。切換えテーブルを使用して、発信ポートの宛先マスク3112が生成される。宛先マスクの長さは32ビットである（すなわち、「32ビット宛先マスク」）ことが好ましい。一態様では、32ビット宛先マスクは、発信ポートに対応する切換えテーブルの行を含む。次に、宛先マスク3112を使用して宛先ベクトル3114が修正され、宛先ベクトル3114はその後、連係リスト3110に格納される。図28の修正されたポート・マスク2818と同様に、修正された宛先ベクトル3114がすべての「0」ビットを含む場合、対応するメモリ・セルが解放される。

【0119】

このような実施形態3100の好ましい態様では、連係リストは、宛先ベクトル3104を格納するための32ビット／語×256語メモリを含んでよい。このような実施形態の他の好ましい態様では、図28に示されている実施形態の切換えテーブル2812とは異なり、宛先マスクが切換えテーブル3108の行に過ぎないので、切換えテーブル3108が、宛先マスクを生成する論理（すなわち、回路）を含む必要はない。この実施形態の他の好ましい態様では、切換えテーブルの行が互いに排他的である必要はない。

【0120】

（後方伝搬）

図32は、本発明の実施形態によるスイッチ・ファブリック中の切換えテーブルを更新するプロセス3200のフローチャートである。動作3202において、送信側のソースと受信側のスイッチ・ファブリック要素との間のハンドシェーキングなしで、スイッチ・ファブリック要素によって1つまたは複数のパケットが受信される。各ステータス・パケットは、出力ポートのステータスに関する情報を含む。次いで、動作3204において、受信されたステータス・パケットのステータス情報に基づいて切換えテーブルが更新される。

【0121】

本発明の態様では、スイッチ・ファブリック・システムによるイーサネット・パケットの送信専用のパスとは別の、ステータス・パケットの送信専用のパス／チャネル／コンジットを介してステータス・パケットを受信することができる。本発明の他の態様では、ステ

ータス・パケットを定期的に受信することができ、切換えテーブルをリアルタイムで更新することができる。

【0122】

本発明の実施形態では、受信されたステータス・パケットの組み合わされたステータス情報に基づいて、組み合わされたステータス・パケットを生成することができる。

【0123】

次いで、組み合わされたステータス・パケットを他のスイッチ・ファブリック要素に送信することができる。このような実施形態の一態様では、組み合わされたステータス・パケットを外側層スイッチ・ファブリック要素から中央層スイッチ・ファブリック要素に送信することができる。

10

【0124】

本発明の一態様では、受信されるステータス・パケットをポート・コントローラから生成することができる。本発明の他の態様では、受信されるステータス・パケットをスイッチ・ファブリック要素から生成することができる。本発明の他の態様では、ステータス・パケットは、各ビットがスイッチ・ファブリック要素のポートに関連する32ビットのデータを含んでよい。

【0125】

一実施形態において、スイッチ・ファブリック・ポート・コントローラを備えるスイッチ・システムは、パケットがスイッチ・ファブリックを通して宛先ポート・コントローラに至るパスを、ハンドシェーキング・プロトコルを使用して判定する。しかし、ハンドシェーキング・プロトコルを使用すると、(1)各着信パケットから、スイッチ・ファブリックを通るパスを判定しなければならず、(2)ハンドシェーキング・プロトコルが、パケットがスイッチ・ファブリックを通して伝搬するのと同じチャネルまたはコンジットを使用して実行されるため、スイッチ・システムの帯域幅が小さくなる可能性がある。

20

【0126】

図33は、本発明の実施形態によるスイッチ・ファブリックにおけるステータス情報および制御情報の後方伝搬の例示的な構成3300を示す概略図である。図33に示されている例示的なスイッチ・ファブリック3300は、図1に示されているマルチポート・スイッチ・アーキテクチャ100のClosネットワーク・トポロジに配置することが好ましい12個のスイッチ・ファブリック要素(SF1からSF12)を備えている。例示的なスイッチ・ファブリック3300は、32個のポート・コントローラ(PC01からPC32)と、外側層スイッチ・ファブリック要素(SF1~4およびSF9~12)のそれぞれに結合された4つのポート・コントローラとをさらに有している。

30

【0127】

後方伝搬機構3300において、各ポート・コントローラは、それぞれのポート・コントローラを、関連する、外側層スイッチ・ファブリック要素に結合する専用コンジット(たとえば、3302)を有している。同様に、各外側層スイッチ・ファブリック要素(SF1~4およびSF9~12)は、それぞれの外側層スイッチ・ファブリック要素を各中央層スイッチ・ファブリック要素に結合する専用コンジット(たとえば、3304)を有している。SF1が各ポート・コントローラからステータス情報をどのように受信するかの観点から後方伝搬プロセスを説明するうえで図を明確にするために、図33には、外側層スイッチ・ファブリック要素SF2~4およびSF9~12からの専用コンジットしか示されていない。外側層スイッチ・ファブリック要素(すなわち、SF1~4およびSF9~12)がそれぞれ、各中央スイッチ・ファブリック要素(すなわち、SF5~8)の専用コンジットを有することに留意されたい。各中央スイッチ・ファブリック要素(すなわち、SF5~8)は、それぞれの中央スイッチ・ファブリック要素を各外側スイッチ・ファブリック要素(すなわち、SF1~4およびSF9~12)に結合する専用コンジット(たとえば、3306)を有している。この場合も、SF1が各ポート・コントローラからステータス情報をどのように受信するかに関して後方伝搬プロセスを説明するうえで図を明確にするために、中央スイッチ・ファブリック要素SF5から外側層スイッチ・ファブ

40

50

リック要素 S F 1 への専用コンジット 3 3 0 6 しか示されていない。しかし、外側層スイッチ・ファブリック要素が、図 3 3 の例で説明したのと同様に各中央層スイッチ・ファブリック要素からステータス情報を受信できることを認識されたい。

【0 1 2 8】

引き続き図 3 3 を参照するとわかるように、専用コンジット（たとえば、3 3 0 2、3 3 0 4、3 3 0 6）を介すことにより、パケット（および他の制御情報）を受信するポート・コントローラの可用性に関するステータス情報をスイッチ・ファブリックを通して伝搬させることができる。専用コンジット 3 3 0 2、3 3 0 4、3 3 0 6 の矢印は、スイッチ・システム全体からスイッチ・ファブリック要素 S F 1 へのステータス情報の伝搬を示している。好ましい実施形態では、専用パスはそれぞれ、1 サイクル当たり 2 ビットの情報を送信できるようにする帯域幅を有してよい。このような好ましい実施形態では、帯域幅の 1 ビットを制御情報の送信に使用し、他のビットを、スイッチ・システムのポート・コントローラの可用性に関するステータス情報の送信に使用してよい。

【0 1 2 9】

図 3 3 には、S F 5 を介してステータス情報を受信するように中央層スイッチ・ファブリック要素 S F 5 に結合されたスイッチ・ファブリック要素 S F 1 が示されているが、中央層スイッチ・ファブリック要素 S F 6、S F 7、および S F 8 はそれぞれ、ステータス情報を S F 1 に与えるように S F 1 に結合された専用コンジットを有してよい。この冗長性は、S F 5 がステータス情報を S F 1 に送信できない場合でもポート・コントローラに関する情報を引き続き受信することができるので有利である。

【0 1 3 0】

一般に、出力ポート・ステータスは、スイッチ・ファブリックを通して伝播するときにスイッチ・ファブリックのローカル出力ポート・ステータスと併合され、それによってスイッチ・ファブリックの全体的なパケット・ステータスに関する与えられた情報を蓄積することができる。したがって、このような機構を用いた場合、スイッチ・システム内の各要素はポートのリアルタイムに更新されたパケット・ステータスを維持し、このローカル情報を使用して、スイッチ・システムへのパケットの入口点とスイッチ・システム内のパケットの宛先との間のハンドシェーキング・プロセスなしにパケットの流れを制御することができる。

【0 1 3 1】

図 3 4 は、本発明の実施形態による、スイッチ・ファブリック・システム内の切換えテーブルを更新するプロセス 3 4 0 0 のフローチャートである。好ましい実施形態では、このプロセスは、明細書の、図 3 3 に関する部分に記載された後方処理専用コンジットを有するスイッチ・システムで実行される。図 3 4 に示されているように、プロセスの一部は、スイッチ・ファブリック・システムのポート・コントローラで実行され、他の部分は、外側層スイッチ・ファブリック要素および中央層スイッチ・ファブリック要素で実行される。

【0 1 3 2】

動作 3 4 0 2 において、ポート・コントローラでステータス・パケットが実行される。ステータス・パケットは、発信パケットの受信に関するポート・コントローラの現在のステータス／可用性を示す情報を含む。生成されたステータス・パケットは、動作 3 4 0 4 で、ポート・コントローラに関連する外側層スイッチ・ファブリック要素に送信される。たとえば、図 3 3 を参照するとわかるように、ポート・コントローラ P 3 2 - 0 8 はスイッチ・ファブリック要素 S F 2 に関連付けされている。したがって、ポート・コントローラ P 3 2 - 0 8 は、生成したステータス・パケットをスイッチ・ファブリック要素 S F 2 に送信する。好ましい実施形態では、ポート・コントローラは、ステータス・パケットを、関連する外側層スイッチ・ファブリック要素に、該ポート・コントローラと該スイッチ・ファブリック要素を連結する専用コンジット（たとえば、専用コンジット 3 3 0 2）を介して送信する。

【0 1 3 3】

10

20

30

40

50

動作 3 4 0 6 で、外側層スイッチ・ファブリック要素は、関連するポート・コントローラから着信ステータス・パケットを受信し収集する。受信されたステータス・パケットからの情報は、次いで動作 3 4 0 8 で、外側層スイッチ・ファブリック要素によって生成されたステータス・パケットにおいて組み合わされる。動作 3 4 1 0 で組み合わされたステータス・パケットは、次いで中央層スイッチ・ファブリック要素に送信される。外側層スイッチ・ファブリック要素は、組み合わされたステータス・パケットを中央層スイッチ・ファブリック要素に、これらのスイッチ・ファブリック要素を連結する専用コンジット（たとえば、S F 2 と S F 5 を連結する専用コンジット 3 3 0 4）を介して送信することが好ましい。外側層スイッチ・ファブリック要素はさらに、ポート・コントローラから受信したステータス・パケットの情報を使用して、それ自体の切換えステータスを更新する（動作 3 4 1 2 参照）。 10

【0 1 3 4】

中央層スイッチ・ファブリック要素は、動作 3 4 1 4 で、外側層スイッチ・ファブリック要素によって送信されたステータス・パケットを受信し収集する。たとえば、図 3 3 に示されている例示的な機構を参照する。動作 3 4 1 6 で、中央層スイッチ・ファブリック要素は、外側層スイッチ・ファブリック要素から受信されたステータス・パケットに含まれるステータス情報を組み合わせたステータス・パケットを生成する。たとえば、図 3 3 に示されている例示的な実施形態では、中央層スイッチ・ファブリック要素 S F 5 は、各パケットがそれぞれの各スイッチ・ファブリック要素に関連するポート・コントローラについてのステータス情報を持つステータス・パケットを外側層スイッチ・ファブリック要素 S F 2 ～ S F 1 2 から受信する（たとえば、スイッチ・ファブリック要素 S F 2 からのステータス・パケットは、ポート・コントローラ P 3 2 - 0 8 に関する情報を有し、他のパケットについても同様である）。 20

【0 1 3 5】

動作 3 4 1 8 で、中央層スイッチ・ファブリック要素によって生成されたステータス・パケットは、受信側の外側層ファブリック要素（たとえば、図 3 3 に示されている例では、スイッチ・ファブリック要素 S F 1）に送信される。中央層スイッチ・ファブリック要素は、それが生成した組み合わされたステータス・パケットを受信側の外側層スイッチ・ファブリック要素に、これらのスイッチ・ファブリック要素を連結する専用コンジット（S F 5 と S F 1 を連結する専用コンジット 3 3 0 6）を介して送信するのが好ましい。さらに、中央層スイッチ・ファブリック要素は、（ポート・コントローラのステータスに関する）外側層スイッチ・ファブリック要素から受信したステータス・パケットの情報を使用して、それ自体の切換えテーブルを更新する（動作 3 4 2 0 参照）。 30

【0 1 3 6】

受信側の外側層スイッチ・ファブリック要素（たとえば、図 3 3 に示されている例ではスイッチ・ファブリック要素 S F 1）は、動作 3 4 2 2 で、中央層スイッチ・ファブリック要素からステータス・パケットを受信した後、受信されたステータス・パケットに含まれるステータス情報に基づいて、動作 3 4 2 4 で、切換えテーブルを更新する。

【0 1 3 7】

したがって、図 3 3 および図 3 4 に示されている後方伝搬機構および手順の下では、パケットの受信および出力に関するすべての装置ポートの可用性（すなわち、出力ステータス）を定期的かつ／または連続的に（好ましくはリアルタイムで）通知する専用コンジットを設けてよい。さらに、この後方伝搬機構は、ステータス情報をすべてのスイッチ・ファブリック要素に送信するにあたって必ずしもハンドシェーキング・プロトコルを使用しなくてもよい。 40

【0 1 3 8】

図 3 5 は、本発明の好ましい実施形態による、図 3 4 に示されているプロセスで使用できるステータス・パケット 3 5 0 0 の概略図である。このステータス・パケットは複数のビットを含み、好ましい実施形態では、3 2 ビット含んでよい。ステータス・パケットの各ビットは、スイッチ・システムの特定のポート・コントローラと関連付けることができる 50

。たとえば、図 3 3 に示されているスイッチ・システムに基づいて、ステータス・パケットのビット 0 1 3 5 0 2 をポート・コントローラ P C 0 1 と関連付けし、ステータス・パケットのビット 0 2 3 5 0 4 をポート・コントローラ P 5 と関連付けし、他のビットについても同様に関連付けすることができる。ステータス・パケットの各ビットは、スイッチ・システムを通して送信されるパケットの受信／出力に関する、該ビットに関連するポート・コントローラの可用性（すなわち、ステータス）を示すために使用することができる。

【0 1 3 9】

一例として、「1」を含むビットは、関連するポート・コントローラが、パケットの受信／出力に利用できる（すなわち、このポート・コントローラのステータスは「利用可能」である）ことを示すことができる。逆に、「0」を含むビットは、関連するポート・コントローラが、パケットの受信／出力に利用できない（すなわち、このポート・コントローラのステータスは「利用不能」である）ことを示すことができる。したがって、図 3 5 に示されている例示的なステータス・パケット 3 5 0 0 において、ビット 0 1 および 0 2 3 5 0 2、3 5 0 4 に含まれる「1」は、ポート・コントローラ P C 0 1 および P 5 が利用可能であることを示し、一方、ビット 0 9、1 6、および 1 7 3 5 0 6、3 5 0 8、3 5 1 0 に含まれる「0」は、ポート・コントローラ P C 0 9、P C 1 6、P C 1 7 が利用不能であることを示す。

【0 1 4 0】

（マルチバンク・バッファ）

図 3 6 は、スイッチ・ファブリックのスイッチ・マトリクスに、入力データグラムを保存するためのプロセス 3 6 0 0 のフローチャートである。スイッチ・マトリクスは、各バッファが 1 組の部分有する 1 組のバッファを有する。データグラムのデータは動作 3 6 0 2 に受信され、バッファ部分は、動作 3 6 0 4 においてデータで連続的に満たされる。動作 3 6 0 6 において、周期的に、データの転送は、バッファからスイッチ・マトリクスに可能になる。バッファ部分が満たされるシーケンスにおいて、データの転送が可能になる各周期において、バッファ部分の 1 つにおけるデータは、動作 3 6 0 8 において、スイッチ・マトリクスに転送されてもよい。

【0 1 4 1】

本発明の 1 つの態様において、バッファ部分が、データで満たされている、または、データグラムの終端（すなわち、データグラムのテールまたは最終バイト）を含んでいる場合、バッファ部分におけるデータは、スイッチ・マトリクスに転送されるだけでよい。本発明の他の態様において、スイッチ・マトリクスは、転送されたデータを保存するための複数のメモリ・バンクで構成することもできる。このような実施形態において、メモリ・バンクは、バッファ部分から転送されたデータを受信する際に交替することができる。

【0 1 4 2】

本発明のさらなる態様において、バッファ部分は、受信したデータを保存するために同等の保存容量をそれぞれ有することができる。好ましい実施形態において、各バッファ部分は、入来データを保存するために、1 6 バイトの保存容量を有する。他の好ましい態様において、データの転送は、1 6 サイクル毎に可能としてもよい。

【0 1 4 3】

図 3 7 は、本発明の実施形態によるスイッチ・ファブリック・コンポーネント 3 7 0 0 内のパケット（すなわち、データグラム）3 7 0 2 の入来および保存を示す模式的ダイアグラムである。上記に検討したように、スイッチ・ファブリック・コンポーネント 3 7 0 0 は、V ユニット 1 ～ 9 として示される 8 個の入来レシーバ（例えば、図 1 および 2 2 に示すスイッチ・ファブリック・コンポーネントのような）を有する図示された実施形態を備えた、複数のインターフェイス・レシーバ 3 7 0 4 を含むことができる。

【0 1 4 4】

各入来レシーバは、スイッチ・マトリクス 3 7 0 6 に接続され、これにより、特定の入来レシーバから、スイッチ・マトリクス 3 7 0 6 への入来パケットの転送を可能にする。特

10

20

30

40

50

に、入来レシーバは、関連するセットの入力バッファに結合されている。一実施形態において、入力バッファの各セットは、2つの入力バッファ3708、37100から構成される。一実施形態において、各入力バッファは、少なくとも16バイトの保存容量を有することができる。好ましい実施形態において、各入力バッファは、32バイトの保存容量を有することができる。好ましい選択肢として、各入力バッファは、2つの部分（例えば、入力バッファ3710の2つの部分3712、3714）に分割してもよい。このような選択肢について、各入力バッファの2つの部分3712、3714が、同等のデータ保持容量（例えば、32バイト容量の好ましい実施形態におけるもの）を有することが好ましい。

【0145】

10

中央制御装置3716は、入力バッファに保存されたデータのスイッチ・マトリクスメモリ3718への転送を制御する。一実施形態において、スイッチ・マトリクスメモリは、内部の複数のメモリ・バンクから構成することができる。図37に示すように、各入力バッファが2つの部分から構成されている好ましい実施形態において、スイッチ・マトリクスメモリは、各メモリ・バンクが、各入力バッファの2つの部分の1つからデータを受信するための専用であること（例えば、メモリ・バンク3720が、入力バッファ部分3712の専用であり、メモリ・バンク3722が入力バッファ部分3714の専用であるなど）が好ましい、2つのメモリ・バンク3720、3722で構成することができる。

【0146】

20

図38は、本発明の実施形態による、1組の2部分入力バッファ3708、3710、および、2メモリ・バンク・スイッチ・マトリクス3706を有するスイッチ・ファブリック・コンポーネント3700内のパケットの入来と保存を示すシナリオの模式的ダイアグラムである。2つのうちの第1の入力バッファ3708は、1組の部分（「部分A」3802および「部分B」3804）から構成される一方、第2の入力バッファ3710は、他の1組の部分（「部分C」3712および「部分D」3714）から構成される。各入力バッファ3708、3710は、32バイトの容量を有し、各部分3712、3714、3802、3804は、16バイトの容量を有している。図38を参照すると、バッファの各々における各バイトは、矩形で表される。黒い矩形は、特定のバイトがデータで満たされていることを表す。

30

【0147】

入来レシーバ3704は、各サイクルの入力バッファの1つのバイトに、それが受信中の入来パケットの1バイトを転送することができる。このデータ転送の間、入来レシーバは、入力バッファのバイトを連続的に満たす。同様に、入力バッファの部分から、メモリへのデータの転送を可能にするために、中央制御装置は、周期的に、16サイクル・カウンタ毎に（メモリ・バンク1 3720またはメモリ・バンク2 3722のいずれかの）メモリにアクセスすることを許可する。このシナリオにおいて、メモリのアクセスは、先ず、サイクル・カウンタ7で可能になり、続いて、サイクル・カウンタ23、39、55、71、87、103、119、135、151などで可能になる。さらに、入力バッファからメモリ・バンクへのデータの転送は、入力バッファの部分がデータで完全に満たされた（すなわち、16バイトのデータ）、または、パケットの終端バイト（すなわち、パケットの最終バイト）を含む場合にのみ行われる。

40

【0148】

さらに、メモリ・バンクは、入力バッファからのパケット・データの受信中に交替する。例えば、第1のメモリ・アクセスにおいて、データはメモリ・バンク1に転送してもよく、第2のメモリ・アクセスにおいて、データはメモリ・バンク2に転送してもよく、また、第3のメモリ・アクセスにおいて、データはメモリ・バンク3に転送してもよい、などである。

【0149】

図38に示すシナリオにおいて、97バイトの長さを有する第1の入来パケット3806

50

は、スイッチ・ファブリック 3700 の入来レシーバ 3704 に転送されており、この後、4 バイトのスペース 3808 が続き、その後、97 バイトの長さを有する第 2 の入来パケット 3810 が続く。サイクル・カウンタ 1 では、第 1 のパケットの第 1 のバイトは、入来レシーバによって受信され、第 1 の入力バッファの部分 A 3802 の第 1 のバイト 3812 を満たすために転送される。

【0150】

サイクル 7 において、部分 A 3812 の 16 バイトのうちの 7 つは、第 1 パケット 3806 のデータの最初の 7 バイトで満たされ、中央制御装置は、このシナリオにおいて初めてメモリ・バンクへのアクセスを許可する。しかし、入力バッファのどの部分も完全に満たされていず、それらのどれもパケットの終端バイトを含んでいないため、サイクル 7 において、入力バッファとメモリ・バンクとの間では、パケット・データの転送はない。 10

【0151】

サイクル・カウンタ 16 において、第 1 の入力バッファの部分 A 3812 の全ての 16 のバイトは、パケット 1 の最初の 16 入来バイトで満たされている。しかし、この時点において、次のメモリ・アクセスは、サイクル・カウンタ 23 まで行われない。

【0152】

図 P11 は、本発明の実施形態によるサイクル・カウンタ 23 における図 38 に示すシナリオの模式的ダイアグラムである。サイクル・カウンタ 23 までに、パケット 1 のさらに 7 つのバイトが、入力バッファ 1 の部分 B 3804 に転送されている。同様に、図 P11 に示すように、サイクル・カウンタ 23 において、メモリ・アクセスが中央制御装置によって可能となり、入力バッファ 1 の部分 A が（サイクル・カウンタ 16 以来）満たされているため、部分 A 3802 に含まれるデータの 16 バイトはメモリ・バンク 1 3720 に読み出され、書き込まれ（矢印 P1102 を参照）、続いて、部分 A 3802 から消去される。 20

【0153】

サイクル・カウンタ 39 における次のメモリ・アクセスの時までに、入力バッファ 1 の部分 B の全ては、パケット 1 からのデータで（サイクル・カウンタ 32 以来）満たされており、パケット 1 のバイト 33～39 は、入力バッファ 2 3710 の部分 C 3712 に転送されている。したがって、サイクル・カウンタ 39 において、部分 B のデータのバイトの全てがメモリ・バンク 2 3722 に転送される（すなわち、読み出しと書き込みの機能を実行する）。 30

【0154】

サイクル・カウンタ 55 における次のメモリ・アクセスの時に、入力バッファ 2 の部分 C の全ては、パケット 1 からのデータで（サイクル・カウンタ 48 以来）満たされており、パケット 1 のバイト 49～55 は、入力バッファ 2 3710 の部分 D 3714 に転送されている。したがって、サイクル・カウンタ 55 において、部分 C のデータのバイトの全ては、メモリ・バンク 1 3720 に（メモリ・バンクがデータの受信に対する利用可能性において交替するため）転送される（すなわち、読み出しと書き込みの機能を実行する）。 40

【0155】

サイクル・カウンタ 71 における次のメモリ・アクセスの時に、入力バッファ 2 の部分 D の全ては、パケット 1 からのデータで（サイクル・カウンタ 64 以来）満たされており、パケット 1 のバイト 65～71 は、入力バッファ 1 3708 の部分 A 3802 に転送されている。同様に、サイクル・カウンタ 71 において、部分 D のデータのバイトの全ては、メモリ・バンク 2 3722 に転送される。 40

【0156】

次のメモリ・アクセスは、サイクル・カウンタ 87 において行われ、この時、入力バッファ 1 の部分 A の全ては、パケット 1 からのデータで（サイクル・カウンタ 80 以来）満たされており、パケット 1 のバイト 81～87 は入力バッファ 1 3708 の部分 B 3804 に転送されている。同様に、部分 A のデータのバイトの全ては、サイクル・カウンタ 8 50

7でメモリ・バンク1 3720に転送される。

【0157】

図40は、本発明の実施形態によるサイクル・カウント103における図38に示すシナリオの模式的ダイアグラムである。図40は、サイクル・カウント88とサイクル・カウント103との間のシナリオにおいて、何が起こるかについての以下の検討のために使用される。サイクル・カウント96において、入力バッファ1の部分Bの全ては、パケット1のバイト81～96で満たされている。サイクル・カウント97において、パケット1の最終バイト（バイト97）は、入力バッファBの部分Cの第1のバイト4002に転送される。パケット1のバイト97はパケット1の終端バイトであるため、中央制御装置は、入力バッファ2をデータで満たすことを終了する。

10

【0158】

サイクル98、99、100、101の間、パケット1と2の間の4バイトのスペースのために、データは入来レシーバによって受信されない。サイクル・カウント102において、パケット2の第1のバイトは、入来レシーバによって受信され、入力バッファ1の部分Aの第1のバイトに転送される。この時点で、入力バッファ1の部分Bがパケット1のバイト81～96を未だに含んでいることを念頭に置くことは重要である。入力バッファ1に転送された、サイクル・カウント102において受信されたパケット1の入来バイトは、同じく入力バッファ1に同時に保存されているパケット1からのデータのいずれをも上書きしない。これは、パケット2 4002の第1の入来バイトが、入力バッファ1の部分Aに保存される一方、パケット1の保存されたバイト（バイト81～96）は、入力

20

【0159】

サイクル・カウント103において、次のメモリ・アクセスが行われる。パケット1のバイト81～96は、メモリ・バンク2 3722に転送され、入力バッファ1の部分Bは消去される。同様に、サイクル・カウント103において、パケット2の第2の入来バイト4006は、入力バッファ1の部分Bに転送される。パケット1 4002のバイト97は、入力バッファ2の部分C 4002に未だに残っている。サイクル・カウント119において、続く次のメモリ・アクセスが中央制御装置によって許可される。この時点において、バイト97がパケットの終端バイトであるため、パケット1 4002のバイト97は、メモリ・バンク1に転送される。同様に、サイクル・カウント119においてメモ

30

【0160】

図41は、本発明の実施形態による図38で使用されているデュアルの2部分バッファの代わりに、小さなデュアル入力バッファ4102、4104が使用されている図38に示すシナリオに類似する流れ4100の模式図である。同様に、このシナリオ4100において、単一のバンク・メモリ4106は、デュアル・メモリ・バンクの代わりにスイッチ・マトリクスに設けられている。図38に示すシナリオにおけるように、各入力バッファ4102、4104は、それぞれのバッファ内に矩形で示す各バイトを備えた32バイトを保持するための容量を有する。黒い矩形は、特定のバイトがデータで満たされていることを示す。中央制御装置は、同様に、周期的に、入力バッファからメモリ・バンクへのデータの転送を可能にするために、（ちょうど図38のシナリオにあるように）このシナリオP800において、16サイクル・カウント毎にアクセスを許可する。この現在のシナリオにおいて、メモリ・アクセスも、同様に、先ず、サイクル・カウント7で可能となり、続いて、サイクル・カウント23、39、55、71、87、103、119、135、151などで可能となる。入力バッファが、データ（すなわち、32バイトのデータ）で完全に満たされているか、パケットの終端バイト（すなわち、パケットの最終バイト）を含んでいる場合にのみ、さらに、このシナリオ4100において、入力バッファからメモリ・バッファへのデータの転送が行われる。

40

【0161】

50

さらに、図 38 に示すシナリオにおけるように、このシナリオ 4100 において、97 バイトの長さを有する第 1 の入来パケット 3806 は、入来レシーバに転送され、次に、4 バイトのスペースが続き、次に、97 バイトの長さを有する第 2 の入来パケットが続く。サイクル・カウンタ 1 で、第 1 パケットの第 1 バイトは、入来レシーバによって受信され、第 1 入力バッファ 4102 を満たすために転送される。

【0162】

図 38 に示す配置と比較する時、この配置に生じる問題としては、小さなデュアル入力バッファが、様々な大きさの入来パケットを扱えないことである。この問題は、サイクル・カウンタ 102 におけるシナリオ 4100 の状況を示す図 41 に示す。サイクル・カウンタ 96 において、第 1 の入力バッファ 4102 は、第 1 の入来パケットのバイト 65 ~ 96 で完全に満たされている。第 1 入力バッファが満たされていず、パケットの終端バイトを含んでいなかったため、第 1 の入力バッファ 4102 のデータの転送は、サイクル・カウンタ 71 および 87 において、メモリ・アクセスで可能とされた。サイクル・カウンタ 97 において、第 1 の入来パケットの最終バイト（バイト 97）4108 が受信され、第 2 の入力バッファ 4104 の第 1 のバイトに転送される。しかし、次のメモリ・アクセスは、サイクル・カウンタ 103 において可能となる。したがって、第 1 の入力バッファの第 1 のパケットのデータは、サイクル・カウンタ 103 まで転送することができない。

【0163】

サイクル・カウンタ 98 ~ 101 において、入来パケット間にスペースが発生し、そのため、これらのサイクルの間には入力バッファへの伝送されるデータはない。サイクル・カウンタ 102 において、第 2 のパケットの第 1 のバイト 4110 は、入来レシーバによって受信され、（標準パケット転送プロトコルに従って）第 1 の入力バッファに転送され、したがって、第 1 の入力バッファに保存された第 1 のパケットのバイト 4112（バイト 65）を上書きし、これによって、第 1 のパケットのデータを、これがメモリ・バンク 4106 に転送される前に、損傷する。第 1 のパケットのデータをそのような状態に損傷する問題を回避するための 1 つの選択肢は、第 3 の入力バッファを追加することである。他の選択肢は、図 38 に示すデュアルの 2 部分入力バッファ配置を使用することである。デュアルの 2 部分入力バッファ配置を使用することによって、スイッチ・マトリクス／スイッチ・ファブリック・コンポーネントの全チップ面積のハードウェア面での削減を達成することができる。

【0164】

図 42 は、本発明の実施形態によるスイッチ・マトリクス・メモリ・バンクのためのメモリ・アクセスのサイズを示す模式的ダイアグラムである。入力バッファ 4204 への入力パケット（すなわち、データグラム）4202 に対するデータ伝送速度（「R x 速度」）は、R（ビット／秒）と表すことができる。メモリ 4206 へのアクセスは、各入力バッファ（および、各出力バッファに対しても同じであってもよい）に対して T（秒）毎に許可される。本発明の実施形態によれば、メモリに対する最小メモリ・アクセス・サイズは、RT（ビット）である。好ましい実施形態において、様々な大きさのパケットを扱うために十分なメモリ・アクセス・サイズは、2RT（ビット）である。

【0165】

本発明によるシステムの一実施形態は、IBM 互換のパーソナル・コンピュータ、Apple Macintosh コンピュータ、または、UNIX ベースのワーク・ステーションなどのコンピュータの流れにおいて実施することができる。代表的なハードウェア・コンフィギュレーションは、図 43 に示し、この図は、マイクロプロセッサなどの中央処理ユニット 4310、および、システム・バス 4312 を介して相互接続されたいくつかの他のユニットを有する実施形態によるワーク・ステーションの典型的なハードウェア・コンフィギュレーションを示す。このワーク・ステーションは、さらに、ランダム・アクセス・メモリ（RAM）4314、リード・オンリー・メモリ（ROM）4316、ディスク記憶ユニット 4320 などの周辺機器をバス 4312 に接続するための I/O アダプタ 4318、キーボード 4324 を接続するためのユーザ・インターフェイス・アダプタ 4

10

20

30

40

50

3 2 2、マウス 4 3 2 6、スピーカ 4 3 2 8、マイクロフォン 4 3 3 2、および／または、バス 4 3 1 2 へのタッチ・スクリーン（図示せず）などの他のユーザ・インターフェイス・デバイス、ワーク・ステーションを（例えば、データ処理ネットワークなどの）通信ネットワーク 4 3 3 5 へ接続するための通信アダプタ 4 3 3 4、および、バス 4 3 1 2 をディスプレイ・デバイス 4 3 3 8 に接続するためのディスプレイ・アダプタ 4 3 3 6 をさらに含む。ワーク・ステーションは、これに常駐で、Microsoft Windows NT または Windows / 9 5 のオペレーティング・システム（OS）、IBM OS / 2 オペレーティング・システム、MAC OS、または、UNIX オペレーティング・システムを典型的に有する。当業者には、本発明の実施形態が、上述したもの以外のプラットフォームおよびオペレーティング・システム上に実装することもできることを理解する。実施形態は、J A V A、C、および、C ++ 言語を使用して書くこともでき、オブジェクト志向のプログラミング方法論を利用することもできる。

10

【0 1 6 6】

様々な実施形態が上記に説明された一方、それらが実施例の方法でのみ提示され、限定とはならないことが理解される。したがって、好ましい実施形態の柔軟性および範囲は、上記に説明された実施形態のいずれによっても限定されるべきではないが、冒頭の特許請求の範囲およびそれに同等のものに従ってのみ定義されるべきである。

【図面の簡単な説明】

【図 1】

本発明の実施形態による、マルチポート・スイッチ・アーキテクチャの概略的なブロック図である。

20

【図 2】

本発明の実施形態による、他のマルチ・ポート・スイッチ・アーキテクチャの概略的なブロック図である。

【図 3】

本発明の実施形態による、ギガビット・アップリンクの概略的なブロック図である。

【図 4】

本発明の実施形態による、ポート・コントローラのアーキテクチャの概略的なブロック図である。

【図 5】

本発明の実施形態による、スイッチ・ファブリックを通る送信のデータグラムを準備するプロセスのフローチャートである。

30

【図 6】

本発明の実施形態による、例示的なパケットの概略的な表示である。

【図 7】

本発明の実施形態による、パケットを処理するプロセスのフローチャートである。

【図 8】

本発明の実施形態による、ポート・コントローラにおいて図 7 に示したパケットを処理するプロセスを示す第 1 の概略的な流れ図である。

【図 9】

本発明の実施形態による、ポート・コントローラにおいて図 7 に示したパケットを処理するプロセスを示す第 2 の概略的な流れ図である。

40

【図 1 0】

本発明の実施形態による、ポート・コントローラに入る複数の入力パケットの処理および記憶の例を表す概略図である。

【図 1 1】

本発明の実施形態による、パケットのボディ部分からヘッダ部分を分離するプロセスのフローチャートである。

【図 1 2】

本発明の実施形態による、ヘッダ部分から分離した後、パケットのボディ部分をメモリに

50

記憶するプロセスのフローチャートである。

【図 1 3】

本発明の実施形態による、パケットの処理したヘッダ部分をメモリに記憶するプロセスのフローチャートである。

【図 1 4】

本発明の実施形態による、入力パケットのセルをメモリに記憶することを表す概略図である。

【図 1 5】

本発明の実施形態による、スイッチ・ファブリック・システムにおいて、ポート・コントローラからスイッチ・ファブリックまでパケットを転送するプロセスのフローチャートである。 10

【図 1 6】

本発明の実施形態による、エグレスのためにキューに入れたパケットの連結リストを生成するプロセスのフローチャートである。

【図 1 7】

本発明の実施形態による、パケット・メモリに記憶されたパケットを割り当てられた出力優先キューに連結することを示す概略図である。

【図 1 8】

本発明の実施形態による、図 1 7 に示したパケット・メモリに記憶された例示的なパケットを示す概略図である。 20

【図 1 9】

本発明の実施形態による、ポート・コントローラを介してスイッチ・システムを出るパケットを処理するプロセスを示す概略的な流れ図である。

【図 2 0】

本発明の実施形態による、スイッチ・ファブリック構成要素のアーキテクチャの概略的なブロック図である。

【図 2 1】

本発明の実施形態による、スイッチ・ファブリックを通るデータグラムを送信するプロセスのフローチャートである。

【図 2 2】

本発明の実施形態による、スイッチ・ファブリックを通るパケットを経路指定するスイッチング・テーブルの実装を示すための、4つの8×8のスイッチ・ファブリック構成要素を備えるスイッチ・ファブリックの概略的なブロック図である。 30

【図 2 3、図 2 4、図 2 5、図 2 6】

本発明の実施形態による、図 2 2 に示したスイッチ・ファブリックのための例示的なスイッチング・テーブルの概略図である。

【図 2 7】

本発明の実施形態による、スイッチ・ファブリック構成要素を通るパケットを処理するプロセスのフローチャートである。

【図 2 8】

本発明の実施形態による、決定結果がポート・ビットマップを備えるスイッチ・ファブリック構成要素を通るパケットを処理するプロセスを示す概略図である。 40

【図 2 9】

本発明の実施形態による、スイッチ・ファブリック構成要素におけるポート・ビットマップ（特に、8ビットのポート・ビットマップ）の例示的な使用を示す概略図である。

【図 3 0】

本発明の実施形態による、例示的な宛先ベクトル、宛先ベクトルのためのエグレス・マスク、および変更した宛先ベクトルの概略図である。

【図 3 1】

本発明の実施形態による、決定結果が宛先ベクトルを備えるスイッチ・ファブリック構成 50

要素を通るパケットを処理する他のプロセスを示す概略図である。

【図 3 2】

本発明の実施形態による、スイッチ・ファブリックのスイッチング・テーブルを更新するプロセスのフローチャートである。

【図 3 3】

本発明の実施形態による、スイッチ・ファブリックにおける状況と制御の情報の後方伝播のための例示的な構成を示す概略図である。

【図 3 4】

本発明の実施形態による、スイッチ・ファブリック・システムにおいてスイッチング・テーブルを更新するプロセスのフローチャートである。

10

【図 3 5】

本発明の好ましい実施形態による、図 3 4 に記述したプロセスにおいて使用することが可能である状況パケットの概略図である。

【図 3 6】

本発明の実施形態による、入力データグラムをスイッチ・ファブリックのスイッチ・マトリックスに記憶するプロセスのフローチャートである。

【図 3 7】

本発明の実施形態による、スイッチ・ファブリック構成要素におけるパケットのイングレスと記憶（すなわち、データグラム）を示す概略図である。

【図 3 8】

20

本発明の実施形態による、2つの部分の入力バッファと2つのメモリ・バンクのスイッチ・マトリックスの対を有するスイッチ・ファブリック構成要素におけるパケットのイングレスと記憶を示すシナリオの概略図である。

【図 3 9】

本発明の実施形態による、サイクル・カウント 2 3 における図 3 8 に記述したシナリオの概略図である。

【図 4 0】

本発明の実施形態による、サイクル・カウント 1 0 3 における図 3 8 に記述したシナリオの概略図である。

【図 4 1】

30

本発明の実施形態による、図 3 8 で使用した二重の2つの部分のバッファの代わりにトリビアルな二重入力バッファを使用した、図 3 8 に提示したシナリオに類似のシナリオの概略図である。

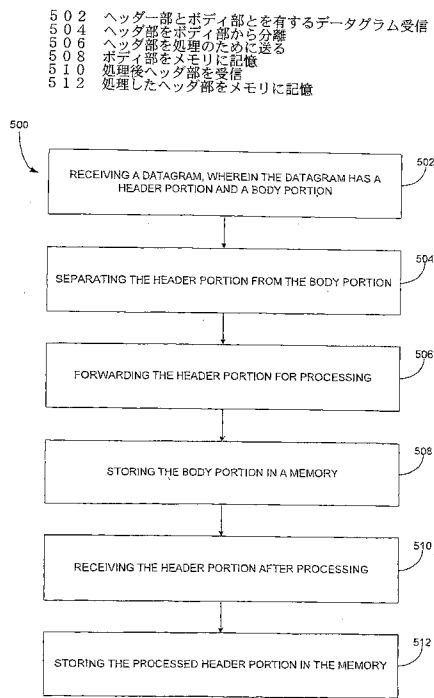
【図 4 2】

本発明の実施形態による、スイッチ・マトリックス・メモリ・バンクに対するメモリ・アクセス・サイズを示す概略図である。

【図 4 3】

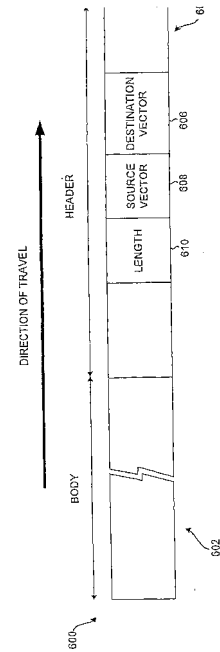
本発明の一実施形態のハードウェア実装に関する概略図である。

【図 5】



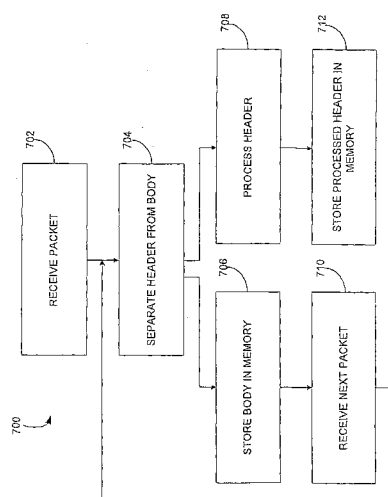
【図 6】

DIRECTION OF TRAVEL 移動方向、BODY ボディ、HEADER ヘッダ
 606 宛先ベクトル、608 ソース・ベクトル、610 長さ

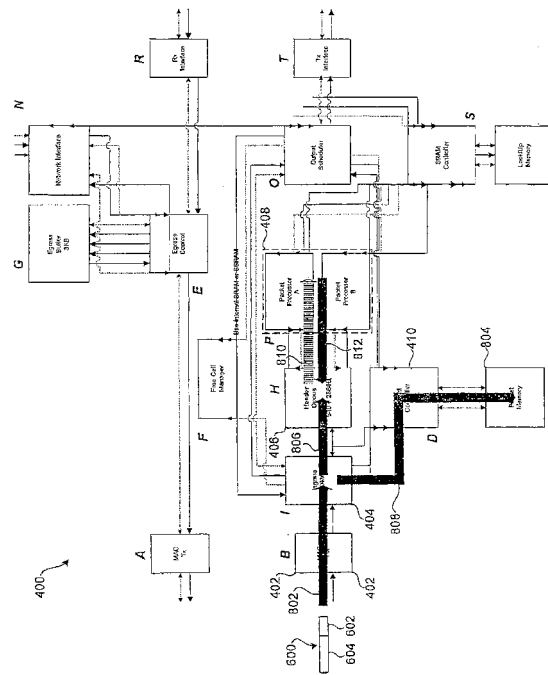


【図 7】

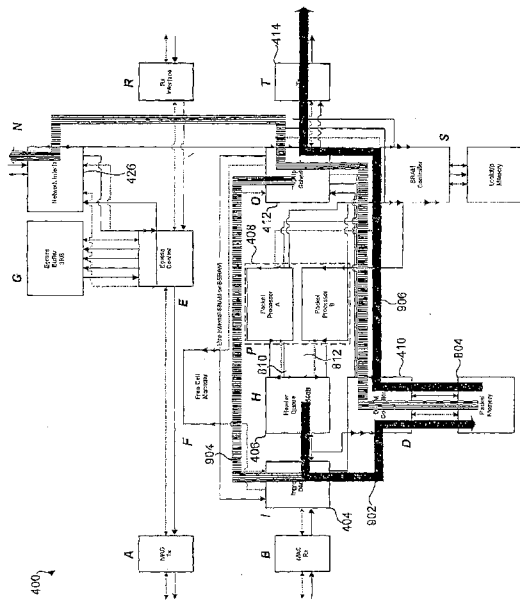
702 受信パケット、704 ボディからヘッダ分離、706 ボディをメモリに記憶
 708 ヘッダ処理、710 次にパケット受信、712 処理したヘッダをメモリに記憶



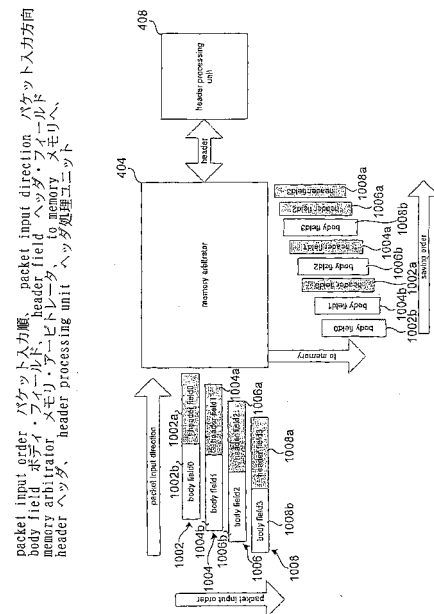
【図 8】



【図 9】

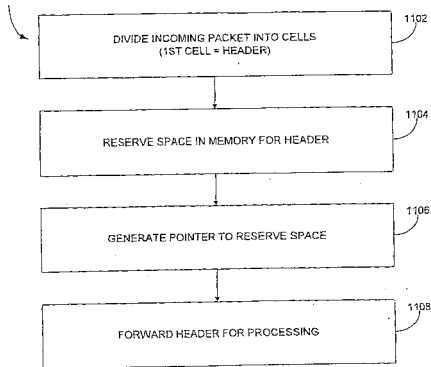


【図 10】



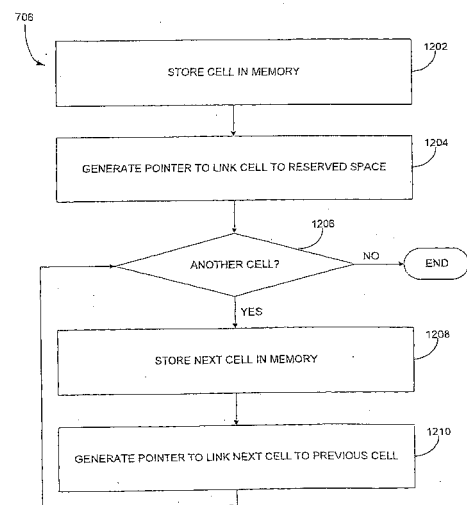
【図 11】

- 1102 入ってきたパケットをセルに分ける
- 1104 ヘッダ用にメモリにスペースを確保
- 1106 確保したスペースへポインタを生成
- 1108 処理のためにヘッダを送る



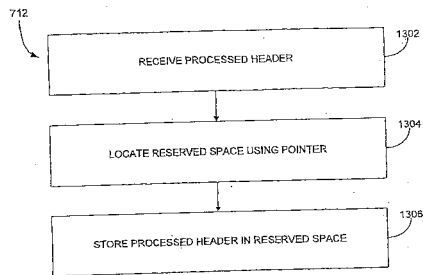
【図 12】

- 1202 メモリにセルを記憶
- 1204 確保されたスペースにセルをリンクするポインタを生成
- 1206 他のセルがあるか
- 1208 次のセルをメモリに記憶
- 1210 先にセルへ次のセルをリンクするポインタを生成

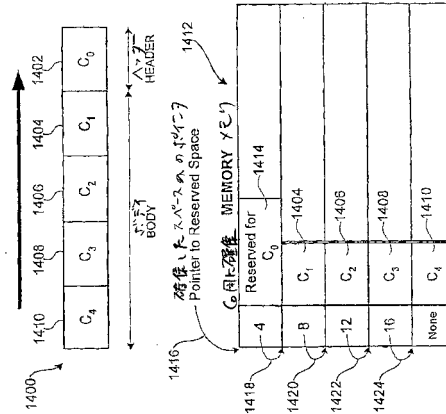


【図 13】

- 1302 処理したヘッダを受信
 1304 ポインタを用いて確保したスペースを突き止める
 1306 処理したヘッダを確保したスペースに記憶

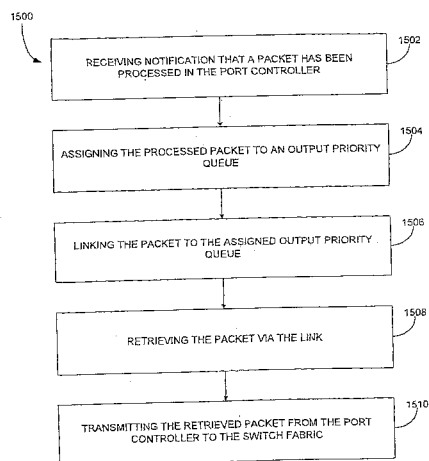


【図 14】



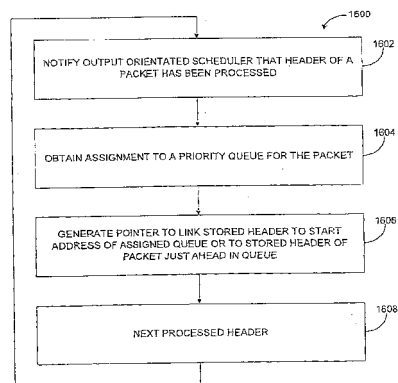
【図 15】

- 1502 ポート・コントローラでパケットが処理されたという通知を受信
 1504 処理されたパケットに出力優先キューを割り当てる
 1506 割り当てられた優先キューにパケットをリンクする
 1508 そのリンクを経てパケットを検索する
 1510 ポート・コントローラからスイッチ・ファイバに検索されたパケットを転送する

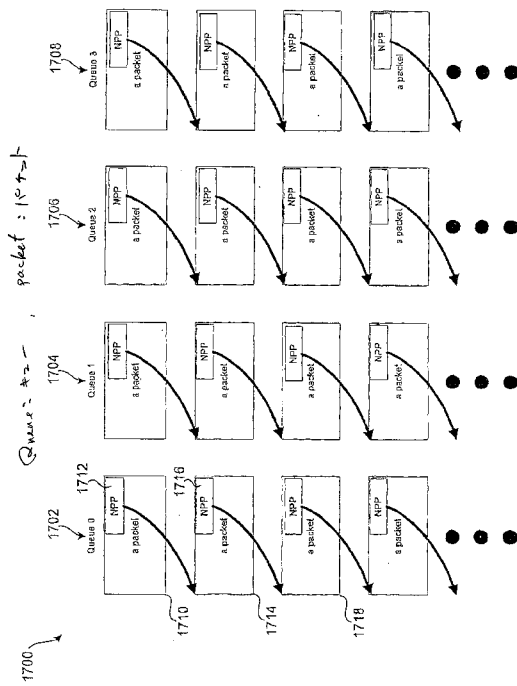


【図 16】

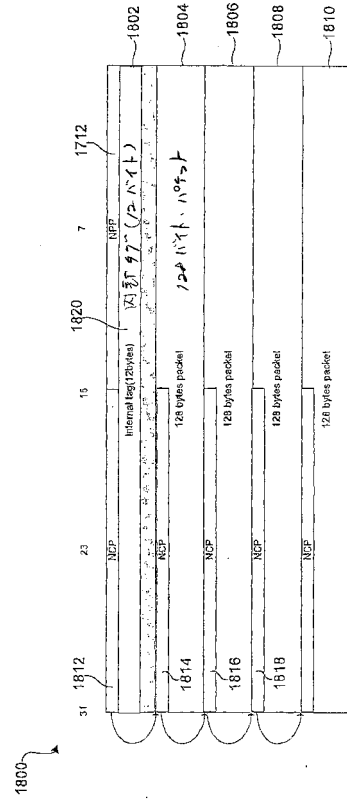
- 1602 パケットのヘッダが処理されたことを出力向けスケジューラに知らせる
 1604 パケットへ優先キューを割り当てる
 1606 割り当てられたキューの開始アドレス又はキューへ向けられたパケットの記憶されたヘッダへ記憶されたヘッダをリンクするポインタを生成
 1608 次の処理されたヘッダ



【図 17】

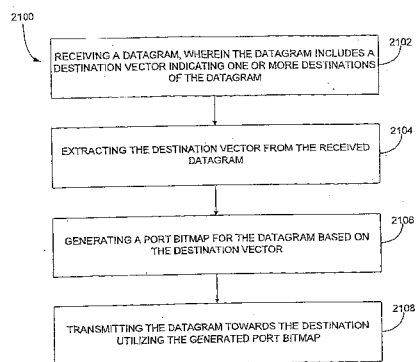


【図 18】

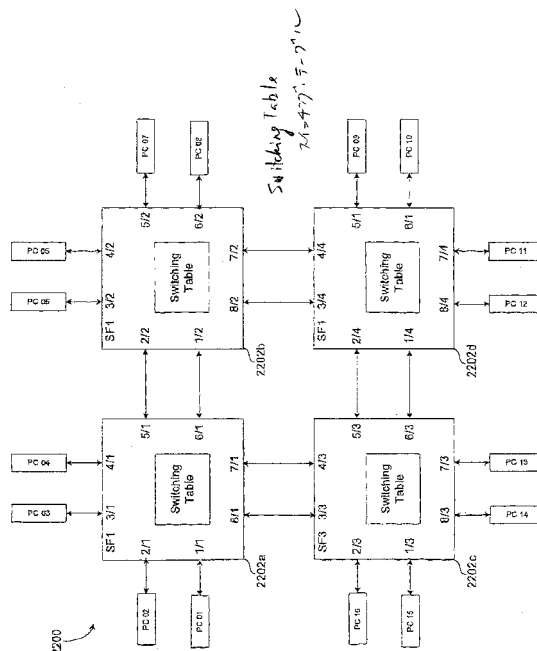


【图 2-1】

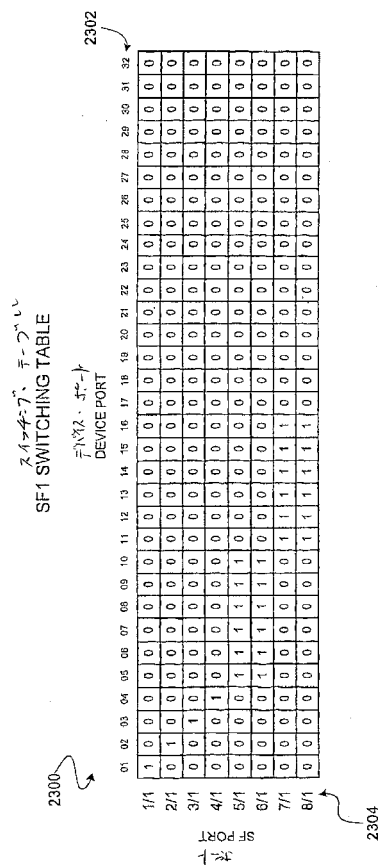
- 2102 一つ又はそれ以上の宛先を指示する宛先ベクトルを含むデータグラムを受信
2104 受信したデータグラムから宛先ベクタを抽出する
2106 宛先ベクトルに基づいてデータグラムのポート・ビットマップを生成
2108 生成されたビットマップを用いて宛先にデータグラムを転送



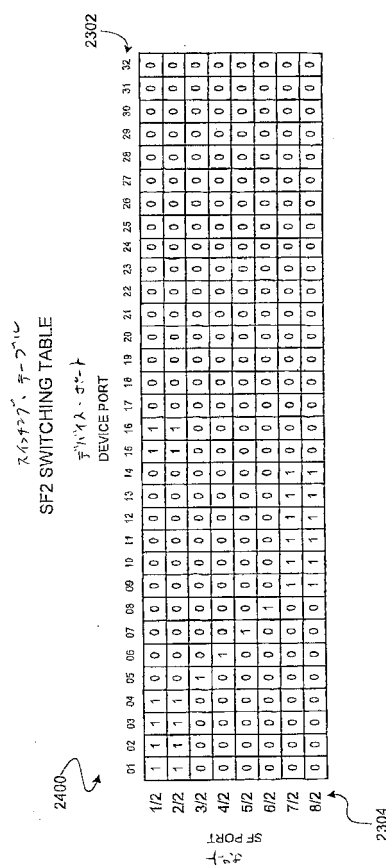
【图 2 2】



【图 2 3】



【図 2 4】



【図 25】

2500

2302

2304

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

2302

2304

23

【図 26】

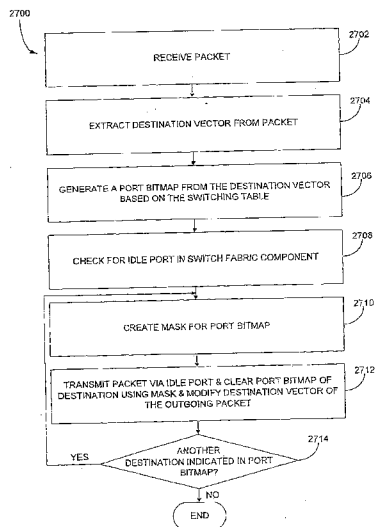
SF4 SWITCHING TABLE
ポート・ビットマップ

2302	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	2304
1/4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2/4
2/4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3/4
3/4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4/4
4/4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5/4
5/4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6/4
6/4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7/4
7/4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8/4

2600

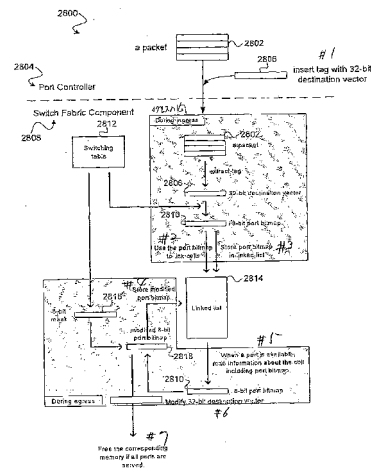
【図 27】

2702 パケットを受信
2704 パケットから宛先ベクトルを抽出
2706 宛先ベクトルからスイッチング・テーブルに基づいてポート・ビットマップを生成
2708 スイッチ・ファブリック・コンポーネントのアイドル・ポートをチェク
2710 ポート・ビットマップに対するマスクを生成
2712 送り出すパケットのマスク及び変形宛先ベクトルを用いてアイドル・ポート
又は宛先のクリア・ポート・ビットマップ群でパケットを転送する
2714 ポート・ビットマップで指示された他の宛先があるか

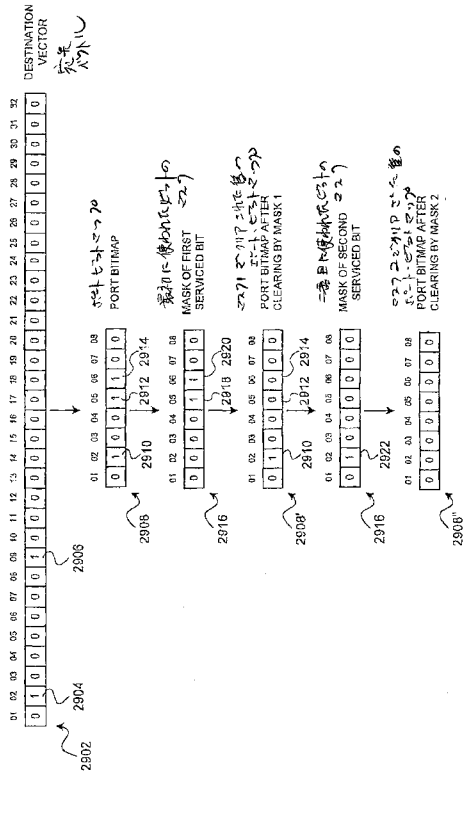


【図 28】

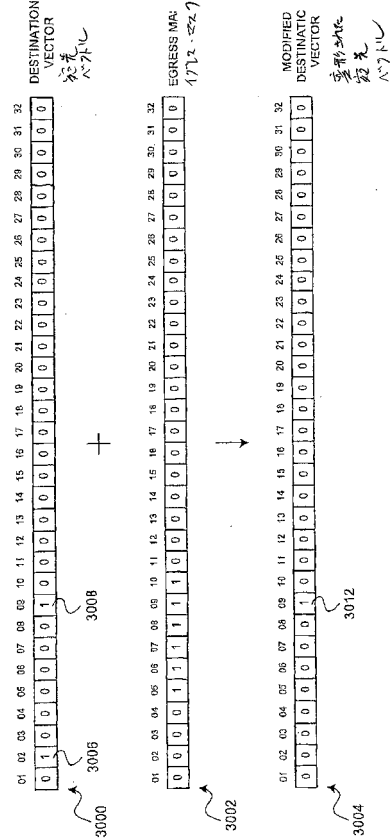
2802 パケット
2804 ポート・コントローラ
2806 32ビットの宛先ベクトル
2808 スイッチ・ファブリック・コンポーネント
2810 8ビット・ポート・ビットマップ
2812 32ビット宛先ベクトルを備えたタグを挿入
2814 リンク済みリストにポート・ビットマップを記録
2816 8ビット・ポート・ビットマップ
2818 変形8ビット・ポート・ビットマップ
2820 8ビット・ポート・ビットマップ
2822 32ビット宛先ベクトルを備えたタグを挿入
2824 変形ポート・ビットマップを記録
2826 全てのポートが使用されているとき対応するメモリをフリーにする



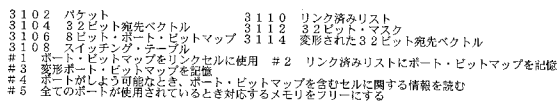
【图 29】



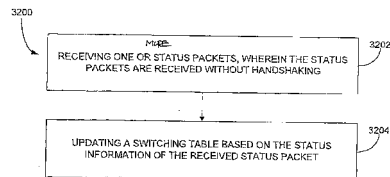
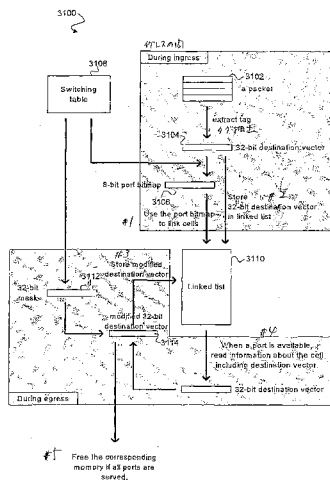
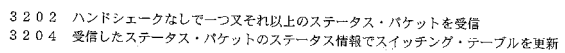
【図 30】



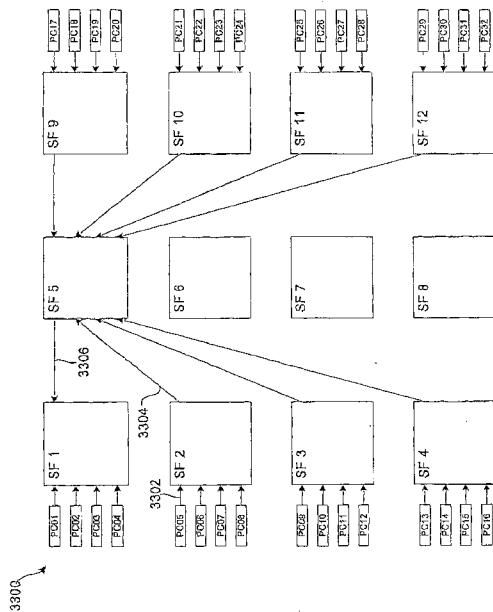
【 図 3 1 】



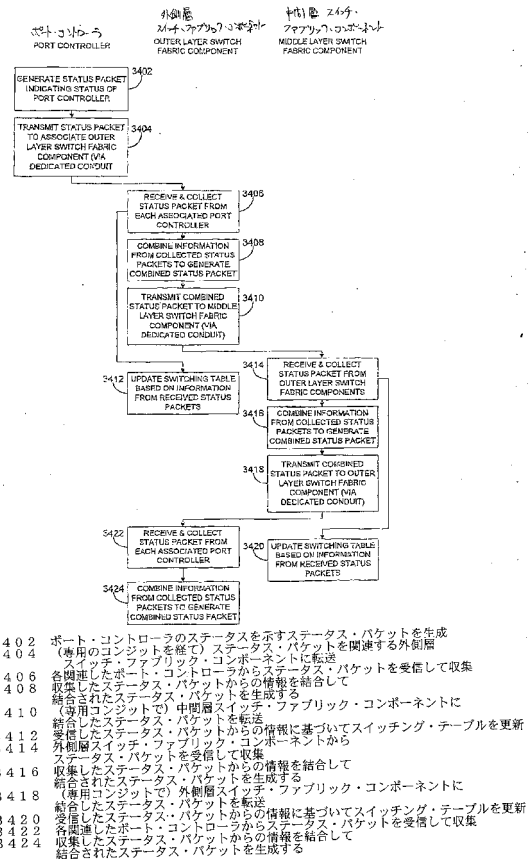
【図 3 2】



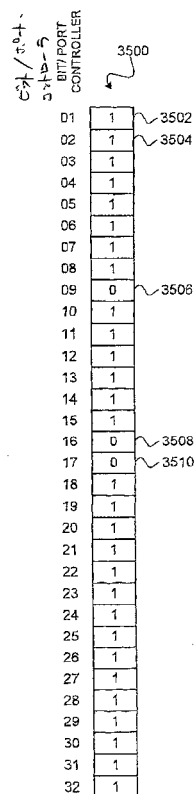
【図 33】



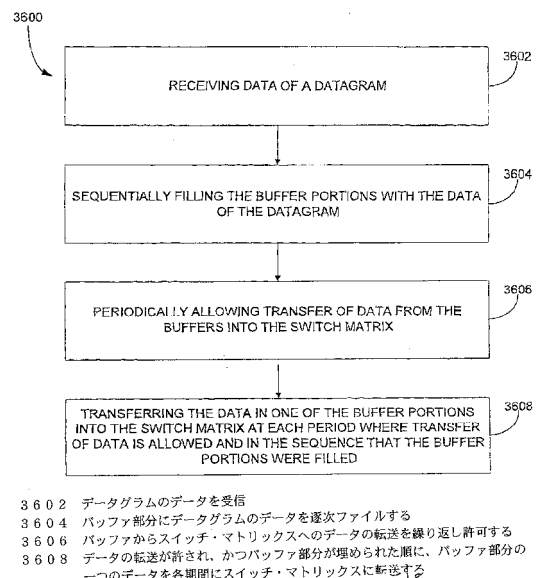
【図 34】



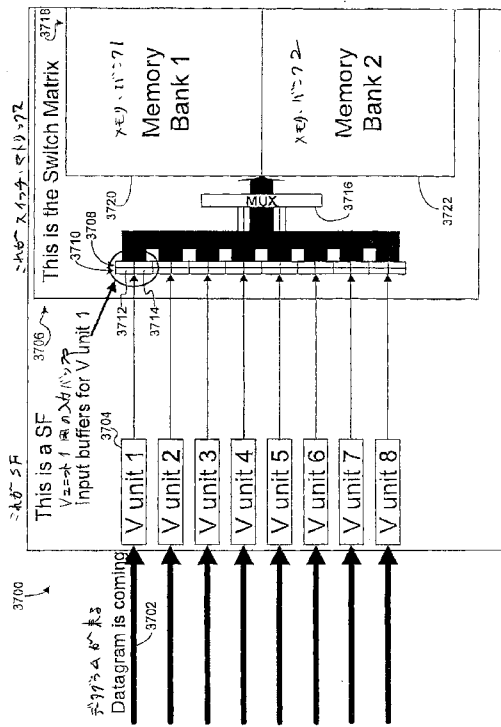
【図 35】



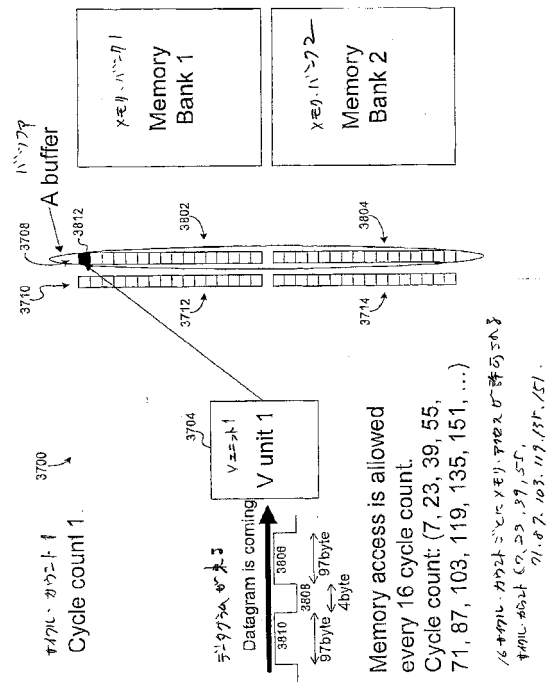
【図 36】



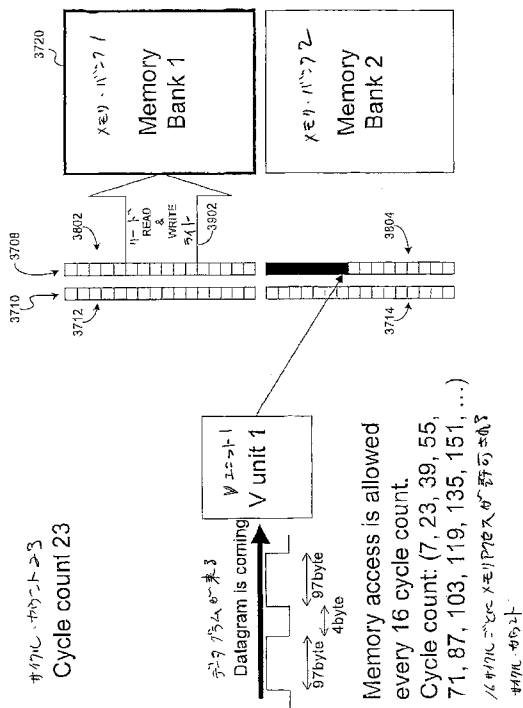
【图 3 7】



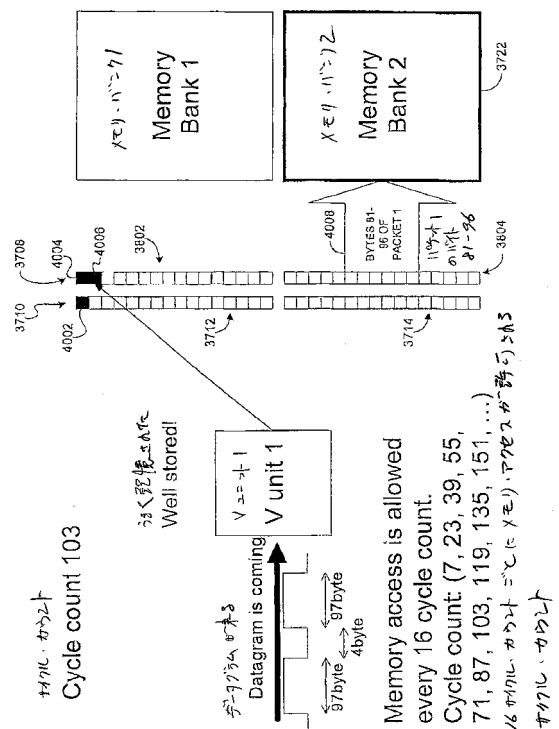
【 図 3 8 】



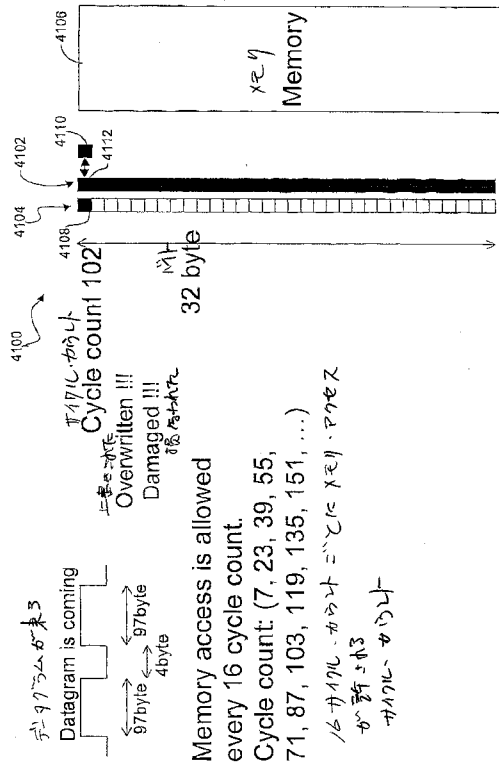
【 ㄨ 3 9 】



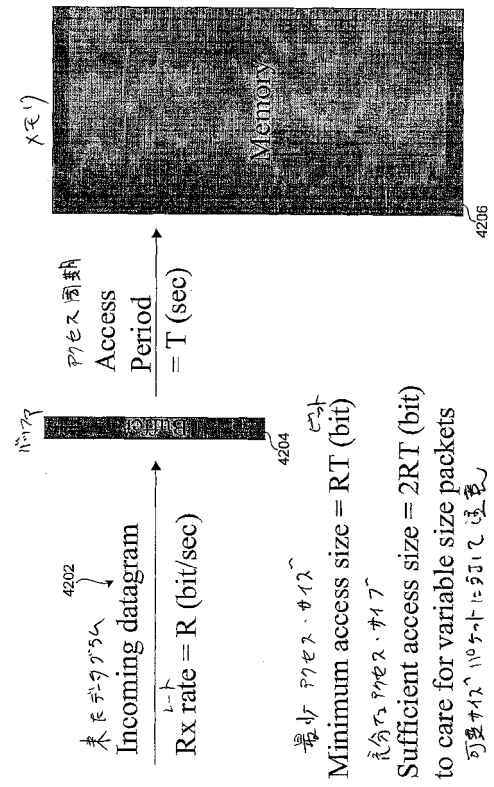
【 図 4 0 】



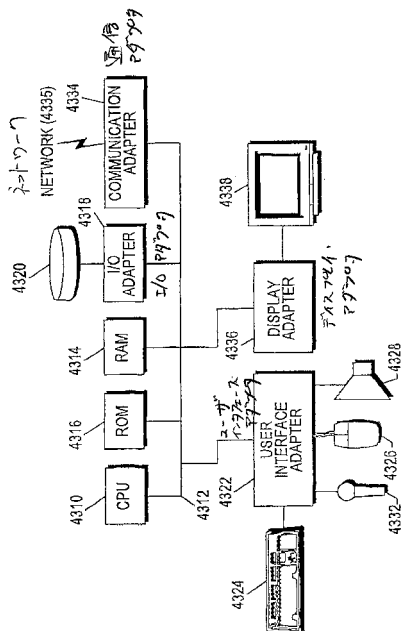
【図 4 1】



【図 4 2】



【図 4 3】



【国際公開パンフレット】

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau(43) International Publication Date
21 February 2002 (21.02.2002)

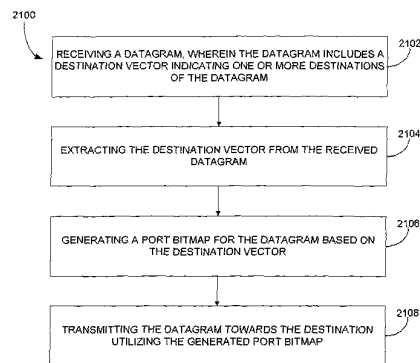
PCT

(10) International Publication Number
WO 02/15496 A1

- (51) International Patent Classification: **H04L 12/56**
Kaisti, 373-1 Kosung-dong, Yusong-gu, 305-701 Taejeon (KR).
- (21) International Application Number: PCT/US01/25083
- (22) International Filing Date: 9 August 2001 (09.08.2001)
- (74) Agent: **BHAKAR, Vidya, R.**, Squire, Sanders & Dempsey L.L.P., 600 Hansen Way, Palo Alto, CA 94304-1043 (US).
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data:
60/225,034 11 August 2000 (11.08.2000) US
09/670,007 25 September 2000 (25.09.2000) US
- (71) Applicant: **PAION COMPANY, LIMITED** (KR/US);
500 Airport Blvd., Suite 333, Burlingame, CA 94101 (US).
- (81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.
- (84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).
- (72) Inventors: **CHANG, Yoo-Sung**; Chips Building, Third Floor, Kaisti, 373-1 Kosung-Dong, Yusong-gu, 305-701 Taejeon (KR). **YI, Jo-Hwan**; Chips Building, Third Floor, Kaisti, 373-1 Kosung-dong, Yusong-gu, 305-701 Taejeon (KR). **KANG, Moo-Kyung**; Chips Building, Third Floor,

[Continued on next page]

(54) Title: NOVEL SWITCH FABRIC CHIPSET SYSTEM AND METHOD



(57) Abstract: A system, method and article of manufacture are provided for sending a datagram (2100) through a switch fabric (2000). From a received datagram (2102), a destination vector (2104) is extracted. The destination vector (2104) indicates one or more destinations (i.e., unicast or multicast) of the datagram (2102). A port bitmap (2106) is generated for the datagram (2102) based on the destination vector (2104). The datagram (2108) is subsequently transmitted toward the destination utilizing the port bitmap (2106).

WO 02/15496 A1

WO 02/15496 A1



Published:
— with international search report

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 02/15496

PCT/US01/25083

NOVEL SWITCH FABRIC CHIPSET SYSTEM AND METHOD

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims the benefit of United States Provisional Application No. 60/225034, filed August 11, 2000, entitled "GIGABIT ETHERNET IP SWITCH CHIPSET" and which is incorporated herein by reference in its entirety for all purposes.

15 **FIELD OF THE INVENTION**

The present invention relates generally to data routing systems and, more particularly, to routing packets through a network switch.

20 **BACKGROUND OF THE INVENTION**

Ethernet is a widely-installed local area network technology. Specified in a standard, Institute of Electrical and Electronic Engineers (IEEE) 802.3, Ethernet was originally developed by Xerox and then developed further by Xerox, DEC, and Intel.

25 An Ethernet LAN typically uses coaxial cable or special grades of twisted pair wires. A commonly installed Ethernet systems are called 10BASE-T and provide transmission speeds up to 10 Mbps. Devices are connected to the cable and compete for access using a Carrier Sense Multiple Access with Collision Detection (CSMA/CD) protocol. Fast Ethernet or 100BASE-T provides transmission speeds

WO 02/15496

2

PCT/US01/25083

up to 100 megabits per second and may be used for LAN backbone systems, supporting workstations with 10BASE-T cards. Gigabit Ethernet provides an even higher level of backbone support at 1000 megabits per second (1 gigabit or 1 billion bits per second).

5

Fast Ethernet is a local area network transmission standard that provides a data rate of 100 megabits per second (referred to as "100BASE-T"). Workstations with existing 10 megabit per second (10BASE-T) Ethernet card can be connected to a Fast Ethernet network. (The 100 megabits per second is a shared data rate; input to each workstation is constrained by the 10 Mbps card). Gigabit Ethernet is a local area network transmission standard that provides a data rate of 1 billion bits per second (one gigabit). Gigabit Ethernet is defined in the IEEE 802.3 standard. Gigabit Ethernet may be used as an enterprise backbone. Gigabit Ethernet may be carried on optical fibers (with very short distances possible on copper media). Existing Ethernet LANs with 10 and 100 Mbps cards can feed into a Gigabit Ethernet backbone.

10

15

Data may be sent over a packet switching network using digital signals. In a packet switching network, users can share the same paths at the same time and the route a data unit travels can be varied as conditions change. In packet-switching, a message is divided into packets, which are units of a certain number of bytes. The network addresses of the sender and of the destination are added to the packet. Each network point looks at the packet to see where to send it next. Packets in the same message may travel different routes and may not arrive in the same order that they were sent. At the destination, the packets in a message are collected and reassembled into the original message.

20

25

Layering is the organization of programming into separate steps that are performed sequentially, defined by specific interface for passing the result of each step to the next program or layer until the overall function, such as the sending or receiving of

30

- some amount of information, is completed. Communication programs are often layered. The reference model for communication programs, Open System Interconnection (OSI) is a layered set of protocols in which two programs, one at either end of a communications exchange, use an identical set of layers. OSI
- 5 includes of seven layers, each reflecting a different function that has to be performed in order for program-to-program communication to take place between computers. Transmission Control Protocol and Internet Protocol (TCP/IP) is an example of a two-layer set of programs that provide transport and network address functions for Internet communication.
- 10
- A switch is a internetworking device that selects a path or circuit for sending a unit of data to its next destination. A switch may also include the function of a router and determine the route and specifically what adjacent network point the data should be sent to. Relative to the layered Open Systems Interconnection (OSI)
- 15 communication model, a switch may be associated with the data link layer - Layer 2. Switches may also be capable of performing the routing functions of the network layer - Layer 3. Layer 3 switches are also sometimes called IP switches.
- The Data Link Layer of OSI – Layer 2 - is concerned with moving data across the
- 20 physical links in the network. In a network, a switch may be a device that redirects data messages at the Layer 2 level, using the destination Media Access Control (MAC) address to determine where to direct the message. The Data-Link Layer contains two sublayers that are described in the IEEE-802 LAN standards: Media Access Control (MAC), and Logical Link Control (LLC). The Data Link Layer
- 25 assures that an initial connection has been set up, divides output data into data frames, and handles the acknowledgements from a receiver that the data arrived successfully. It also ensures that incoming data has been received successfully by analyzing bit patterns at special places in the frames.
- 30 The Network Layer of OSI – Layer 3- is concerned with knowing the address of the neighboring nodes in the network, selecting routes and quality of service, and

WO 02/15496

4

PCT/US01/25083

recognizing and forwarding to the transport layer incoming messages for local host domains. A switch may be a Layer 3 device and perform layer 3 functions. The IP address is a layer 3 address.

- 5 Switching, does by its nature, involves transient data flows. At Gigabit speeds, these transients occur extremely fast, and in a highly random manner. In the past, it has been difficult to determine in sufficiently fast enough speeds which components of a switch data flows should be directed to so as maximize data flow through the switch and reduce blocking traffic in the switch.

SUMMARY OF THE INVENTION

A system, method and article of manufacture are provided for sending a datagram through a switch fabric. From a received datagram, a destination vector is extracted. The destination vector indicates one or more destinations (i.e., unicast or multicast) of the datagram. A port bitmap is generated for the datagram based on the destination vector. The datagram is subsequently transmitted towards the destination utilizing the port bitmap.

In an aspect of the present invention, the destination vector may comprise 32 bits of data. In another aspect of the present invention, the port bitmap may comprise 8 bits of data. In an embodiment of the present invention, generating the port bitmap may further include obtaining all destinations of the datagram from the destination vector and then utilizing a switching table to determine through which switch ports of the switch fabric component can the datagram be routed to reach the destinations. The port bitmap may then be generated to indicate therein all of the determined switch ports through which the can the datagram be routed to reach its intended destinations.

In another embodiment of the present invention, transmitting the datagram may also include creating a mask based on the port bitmap, and modifying the port bitmap using the mask upon transmission of datagram. In such an embodiment, modifying the port bitmap using the mask may also include clearing the port bitmap of a data associated with the destination that the datagram was transmitted towards. In a further embodiment of the present invention, the availability of a switch port through which the datagram may be routed to reach the destinations may also be determined. The datagram may then be transmitted through a switch port determined to be available.

WO 02/15496

6

PCT/US01/25083

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other features, aspects and advantages are better understood from the following detailed description, appended claims, and accompanying drawings

5 where:

Figure 1 is a schematic block diagram of a multi-port switch architecture in accordance with an embodiment of the present invention;

10 Figure 2 is a schematic block diagram of another multi-port switch architecture in accordance with an embodiment of the present invention;

Figure 3 is a schematic block diagram of a gigabit up-linking in accordance with an embodiment of the present invention;

15

Figure 4 is a schematic block diagram of an architecture of a port controller in accordance with an embodiment of the present invention;

20 Figure 5 is a flowchart of a process for preparing a datagram for transmission through a switch fabric in accordance with an embodiment of the present invention;

Figure 6 is a schematic representation of an illustrative packet in accordance with an embodiment of the present invention;

25 Figure 7 is a flowchart for a process for processing packets in accordance with an embodiment of the present invention;

30 Figure 8 is a first schematic flow diagram illustrating the process for processing packets set forth in Figure 7 in a port controller in accordance with an embodiment of the present invention;

WO 02/15496

7

PCT/US01/25083

Figure 9 is a second schematic flow diagram illustrating the process for processing packets set forth in Figure 7 in a port controller in accordance with an embodiment of the present invention;

- 5 Figure 10 is a schematic diagram representing an illustration of the processing and storing of a plurality of incoming packets entering a port controller in accordance with an embodiment of the present invention;

- 10 Figure 11 is a flowchart for a process for separating a header portion from a body portion of a packet in accordance with an embodiment of the present invention;

Figure 12 is a flowchart for a process for storing a body portion of a packet in memory after separation from the header portion in accordance with an embodiment of the present invention;

- 15 Figure 13 is a flowchart for a process for storing a processed header portion of a packet in memory in accordance with an embodiment of the present invention;

- 20 Figure 14 is a schematic diagram representing the storing of cells of an incoming packet into memory in accordance with an embodiment of the present invention;

Figure 15 is a flowchart of a process for transferring a packet from a port controller to a switch fabric in a switch fabric system in accordance with an embodiment of the present invention;

- 25 Figure 16 is a flowchart of a process for generating linked lists of packets queued for egress in accordance with an embodiment of the present invention;

- 30 Figure 17 is a schematic diagram depicting the linking of packets stored in the packet memory to their assigned output priority queues in accordance with an embodiment of the present invention;

WO 02/15496

8

PCT/US01/25083

Figure 18 is a schematic diagram illustrating an exemplary packet stored in the packet memory illustrated in Figure 17 in accordance with an embodiment of the present invention;

5

Figure 19 is a schematic flow diagram illustrating a process for processing a packet exiting a switch system via a port controller in accordance with an embodiment of the present invention;

10 Figure 20 is a schematic block diagram of an architecture of a switch fabric component in accordance with an embodiment of the present invention;

Figure 21 is a flowchart of a process for sending a datagram through a switch fabric in accordance with an embodiment of the present invention;

15

Figure 22 is a schematic block diagram of a switch fabric comprising four 8 by 8 switch fabric components to illustrate the implementation of switching tables for routing packets through the switch fabric in accordance with an embodiment of the present invention;

20

Figures 23, 24, 25, 26 are schematic illustrations of exemplary switching tables for the switch fabric depicted in Figure 22 in accordance with an embodiment of the present invention;

25 Figure 27 is a flowchart for a process for processing a packet through a switch fabric component in accordance with an embodiment of the present invention;

Figure 28 is a schematic diagram illustrating a process for processing a packet through a switch fabric component where the determination result comprises a port bitmap in accordance with an embodiment of the present invention;

30

WO 02/15496

9

PCT/US01/25083

Figure 29 is a schematic diagram illustrating an exemplary utilization of a port bitmap (in particular, an 8 bit port bitmap) in a switch fabric component in accordance with an embodiment of the present invention;

- 5 Figure 30 is a schematic representation of an exemplary destination vector, an egress mask for the destination vector and a modified destination vector in accordance with an embodiment of the present invention;

- 10 Figure 31 is a schematic diagram illustrating another process for processing a packet through a switch fabric component where the determination result comprises a destination vector in accordance with an embodiment of the present invention;

- 15 Figure 32 is a flowchart of a process for updating a switching table in a switch fabric in accordance with an embodiment of the present invention;

- Figure 33 is a schematic diagram illustrating an exemplary arrangement for back propagation of status and control information in a switch fabric in accordance with an embodiment of the present invention;

- 20 Figure 34 is a flowchart of a process for updating switching tables in a switch fabric system in accordance with an embodiment of the present invention;

- 25 Figure 35 is a schematic representation of a status packet that may be utilized in the process set forth in Figure 34 in accordance with a preferred embodiment of the present invention;

- 30 Figure 36 is a flowchart of a process for storing an incoming datagram in a switch matrix of a switch fabric in accordance with an embodiment of the present invention;

WO 02/15496

10

PCT/US01/25083

Figure 37 is a schematic diagram illustrating the ingress and storage of packets (i.e., datagrams) in a switch fabric component in accordance with an embodiment of the present invention;

5 Figure 38 is a schematic diagram of a scenario illustrating the ingress and storage of packets in a switch fabric component having a pair of two-portion input buffers and a two-memory-bank switch matrix in accordance with an embodiment of the present invention;

10 Figure 39 is a schematic diagram of the scenario set forth in Figure 38 at cycle count 23 in accordance with an embodiment of the present invention;

Figure 40 is a schematic diagram of the scenario set forth in Figure 38 at cycle count 103 in accordance with an embodiment of the present invention;

15

Figure 41 is a schematic illustration of a scenario similar to scenario presented in Figure 38 where trivial dual input buffers are used instead of the dual two-portion buffers used in Figure 38 in accordance with an embodiment of the present invention;

20

Figure 42 is a schematic diagram illustrating memory access size for a switch matrix memory bank in accordance with an embodiment of the present invention; and

25 Figure 43 is a schematic diagram of a hardware implementation of one embodiment of the present invention.

DETAILED DESCRIPTION

An Ethernet switch fabric chipset system is disclosed utilizing a switch fabric topology and a multiprocessor port controller capable of supporting wire-speed, non-blocking, Layer 2 and Layer 3 switching of 32 Gigabit Ethernet data streams, and which is expandable to what is known in the art as network processor. For a basic understanding of switching technology, the reader is directed to: The Switch Book: the complete guide to LAN switching technology by Rich Seifert (John Wiley & Sons, Inc. 2000) which is incorporated herein by reference in its entirety for all purposes.

With the system set forth herein, determinations as to which of the switch fabric components transient data flows can be directed may be made sufficiently fast enough to maximize data flow through the switch fabric system while preventing blockage by other data flow traffic. This is accomplished by making as much of the determination as possible before the transient occurs and reducing the decision to consideration of simple the next switch fabric component in the path, rather than the entire path through the switch fabric. As part of this solution, a separate control path may be provided in addition to the data path to provide for sending status and control information back through the switch fabric and to help reduce or eliminate the need for handshaking in the control path. As another part of this solution, a switching table may be provided inside each switch fabric component which can be updated utilizing the control path.

25

SYSTEM CONFIGURATION

Figure 1 is a schematic block diagram of a multi-port switch architecture 100 in accordance with an embodiment of the present invention. In general, this multi-port switch architecture 100 comprises a plurality of port controllers (32 port controllers in the illustrated embodiment) 102 interconnected to one another via a switch fabric

104, and a network control processor 106 connected to the port controllers 102 and the switch fabric 104. As illustrated in Figure 1, the connections (108, 110, 112 for example) between the elements are bi-directional.

- 5 With further reference to Figure 1, in this particular embodiment of the multi-port switch architecture 100, the switch fabric may comprise a plurality of switch fabric components (12 switch fabric components in the illustrated embodiment) 104a, 104b, 104c, 104d, 104e, 104f, 104g, 104h, 104i, 104j, 104k, 104l connected together via bi-directional conduits (e.g., conduit 110) in what is known as a Clos
- 10 network arrangement/topology. The particular arrangement depicted in Figure 1 is a known as a three-stage or layer Clos network topology having a middle stage of switch fabric components (104e, 104f, 104g, 104h) connecting a pair of outer stages of switch fabric components (104a, 104b, 104c, 104d and 104i, 104j, 104k, 104l).
- 15 With the Clos network topology for the switch fabric 104, each outer stage switch fabric component 104a, 104b, 104c, 104d and 104i, 104j, 104k, 104l is directly coupled via independent conduits to each of the middle stage switch fabric components 104e, 104f, 104g, 104h. For example, in Figure 1, switch fabric component 104a has four conduits coupling it to the four middle stage switch fabric
- 20 components 104e, 104f, 104g, 104h. Conversely, with the Clos network topology for the switch fabric 104, each of the middle stage switch fabric components 104e, 104f, 104g, 104h is directly coupled via independent conduits to each of the outer stage switch fabric components 104a, 104b, 104c, 104d, 104i, 104j, 104k, 104l. Thus, in the embodiment illustrated in Figure 1, each of the middle stage switch
- 25 fabric components 104e, 104f, 104g, 104h has a direct and independent connection to each of the outer stage switch fabric components 104a, 104b, 104c, 104d, 104i, 104j, 104k, 104l.

- It should also be noted that for purposes of a single packet traversing the switch
- 30 fabric, one may consider the particular embodiment illustrated in Figure 1, as having

WO 02/15496

13

PCT/US01/25083

an input stage comprising a single outer stage switch fabric component, a middle stage comprising the middle stage switch fabric components and an output stage comprising the remaining outer stage switch fabric components due to the bi-directional connectivity between the switch fabric components.

5

Each outer layer switch fabric component 104a, 104b, 104c, 104d, 104i, 104j, 104k, 104l has a plurality of the port controllers directly coupled/connected thereto. In the embodiment illustrated in Figure 1, each of the outer layer switch fabric components has four port controllers coupled to it by four conduits, one conduit (e.g., conduit 108) for each port controller. For example, switch fabric 104a has port controllers 102a, 102b, 102c, 102d coupled to it by four separate bi-directional conduits. In one aspect of this embodiment, the port controllers 102 may all comprise gigabit port controllers.

10

15

A feature of the Clos network topology of the switch fabric is that it is a non-blocking topology: there is a path from every port controller to all of the other port controllers not directly coupled to the same switch fabric component through every one of the middle stage switch fabric components 104e, 104f, 104g, 104h. For example, the port controllers 102a, 102b, 102c, 102d connected to switch fabric component 104a are connected to the port controllers coupled to all of the other outer stage switch fabric components 104b, 104c, 104d, 104i, 104j, 104k, 104l via each of the middle stage switch fabric components 104e, 104f, 104g, 104h.

20

25

In a preferred embodiment of the configuration 100 shown in Figure 1, there may be 16 Gigabit Ethernet port controllers provided to each outer stage of switch fabric components for a total of 32 Gigabit Ethernet port controllers. In this preferred embodiment, maximum throughput demand from each set of 16 Gigabit Ethernet port controllers may be 16×2 (duplex) Gigabit Ethernet Data Streams = 32 Gbps + Overhead for a maximum throughput demand in both directions of: $[(32 \text{ Gbps} + \text{Overhead}) + (32 \text{ Gbps} + \text{Overhead})] = 64 \text{ Gbps} + \text{Overhead}$. In such a preferred embodiment, each of the switch fabric components in the 32-Gigabit Ethernet port

30

WO 02/15496

14

PCT/US01/25083

configuration, may be a modified 8-Port Crossbar Switch Fabric with a throughput capability of 16 Gbps plus Overhead. Since the middle stage of the switch fabric comprises contains 4 of these devices, throughput of the middle stage may total 64 Gbps + Overhead.

5

Figure 2 is a schematic block diagram of another multi-port switch architecture 200 in accordance with an embodiment of the present invention. In this embodiment, the multi-port switch architecture 200 comprises four port controllers 202a, 202b, 202c, 202d interconnected together via a switch fabric comprising a single switch fabric component 204. In one aspect of such an embodiment, the port controllers 202a, 202b, 202c, 202d may comprise fast Ethernet port controllers having a similar architecture as that of a port controller as that as illustrated in Figure 1.

Figure 3 is a schematic block diagram of a gigabit up-linking 300 in accordance with an embodiment of the present invention. The up-linking 300 comprises first and second port controllers 302a, 302b bi-directionally connected together. In one aspect of such an embodiment, one of the port controllers may comprises a fast Ethernet port controller 302a such as the type used in the multi-port switch architecture 200 depicted in Figure 2 and the other port controller 302b may comprise a gigabit port controller such as the type used in the multi-port switch architecture 100 depicted in Figure 1.

PORT CONTROLLER

25

Each port controller implements a technique known as pipelining. Pipelining is an implementation technique in which multiple instructions are overlapped in execution. A pipeline may be defined as a series of stages, where some work is done at each stage. In general, the port controller has a first pipeline for receiving Ethernet packets from the physical interface (PHY), processes them, and transmits them to the Switch Fabric. The port controller also has a similar but less complex

30

second pipeline for receiving Ethernet Packets from the switch fabric and transmits them to the physical interface (PHY).

In general, there may be at least two preferred versions of the port controller. A first preferred version may provide wire-speed packet processing and other port controller functions, for a single (duplexed) Gigabit Ethernet line. The second preferred version may provide the same functionality for 8-Fast Ethernet (or Ethernet) lines. These devices may be configured with the switch fabric to provide wire speed Layer 2 / Layer 3 switching for up to 32 Gigabit Ethernet Lines, or up to 256 Fast Ethernet Lines.

Figure 4 is a schematic block diagram of an architecture of a port controller 400 in accordance with an embodiment of the present invention. The port controller 400 includes a media access control (MAC) unit 402, an ingress direct memory access (Ingress DMA or IDMA) unit 404, a header queue 406, a packet processor unit 408, a dynamic random access memory (DRAM) controller 410, an output orientated scheduler 412, an interface transmitter 414, a free cell manager 416, a synchronous static random access memory (SSRAM) controller 418, an interface receiver 420, an egress controller 422, an egress buffer 424, and a network interface 426. The DRAM controller 410 is coupled to a first memory called an external packet buffer/memory, which is preferably some sort of dynamic random access memory such as synchronous DRAM (SDRAM). The DRAM controller 410 controls the first memory/external packet buffer and reads/writes data from/to the external packet buffer. The SSRAM controller 418 is coupled to a second memory of the port controller, preferably some sort of synchronous static random access memory (SSRAM).

The MAC interface 402 is connectable to an Ethernet physical layer interface (PHY) to receive and transmit data packets. In one embodiment, the MAC interface 402 may comprise a Gigabit MAC with GMII interface. The MAC interface 402 is

coupled to the Ingress DMA 404 to permit transmitting of incoming data packets received by the MAC interface to the Ingress DMA 404.

5 The Ingress DMA 404 is coupled to the header queue 406, the DRAM controller 410 and the free cell manager 416. Upon receipt of an incoming data packet from MAC interface 402, the Ingress DMA 404 separates the header portion of the packet from the body portion of the packet. The header portion of the packet is then transferred by the Ingress DMA 404 to the header queue 406 while the body portion of the packet is transferred to the DRAM controller 410 for storage in the external packet
10 buffer in a linked list. The Ingress DMA 404 is also responsible for defining a link to a space reserved in memory to the header portion being processed by the packet processor 408 with which the Ingress DMA uses to store the processed header portion in memory after the header portion has been processed. The Ingress DMA also is responsible for notifying the output oriented scheduler 412 when the storage
15 of a packet is completed.

The header queue 406 is coupled to the packet processor unit 408. In a preferred embodiment, the header queue 406 stores the headers it receives from the from Ingress DMA 404 for first in, first out (FIFO) processing by the packet processor
20 unit 408.

The packet processor unit 408 serves to analyzes and processes packets. In one embodiment, the packet processor comprises a plurality of packet processors. While in the embodiment illustrated in Figure 4, the packet processor unit comprises two
25 packet processors – P₀ and P₁ 408a, 408b, which alternate in the processing of headers in the header queue 406.

The packet processors 408 determines how to process the header portions of incoming packets. Processing of the header portions by the packet processor 408 includes
30 determining the destination of a packet by utilizing a lookup address table, attaching an

WO 02/15496

17

PCT/US01/25083

internal tag (which includes a destination vector) to the header portion of the packet, and then notifying the header queue 406 of the completion of processing. The address lookup is done by accessing (via the SSRAM controller 418) an external address table stored in the external SSRAM (The SSRAM may also be used to store other necessary information).

The output oriented scheduler 412 receives a notification from the Ingress DMA when a packet has been processed. The output oriented scheduler 412 is also coupled to the DRAM controller 410 so that the output oriented scheduler 412 can retrieve processed packets from external packet buffer via the DRAM controller 410. Upon retrieving a processed packet from the packet buffer, the output orientated scheduler 412 transfers the retrieved packet to the interface transmitter 414 (or the network interface 426 depending to the defined priority) for transmission out of the port controller 400.

The DRAM controller 410 executes a writing command to write data from the Ingress DMA 404 into the external packet buffer. The DRAM controller 410 also executes reads and transfers of data from the packet buffer to the output orientated scheduler 412 upon receipt of read commands from the output oriented scheduler 412.

The interface transmitter 414 is coupled to the output orientated scheduler 412. The interface transmitter has an external I/O to permit connection to another port controller or a switch fabric component. The output orientated scheduler 412 sends processed packets to the interface transmitter 414 to permit transmission of the processed packet by the interface transmitter 414 to a switch fabric component or another port controller via its external I/O.

The free cell manager 416 is coupled to the Ingress DMA 404 and the output orientated scheduler 412 and thereby in communication with the DRAM controller

410. The free cell manager 416 manages the external packet buffer space and notifies the Ingress DMA 404 of available space in memory for storing data and frees up memory space after data (i.e., processed packets) is retrieved by the output oriented scheduler 412.

5

The interface receiver 420 has an external I/O connectable to a switch fabric component or another port controller to permit receipt of packets from the switch fabric component or other port controller. The egress controller 422 is coupled to the interface receiver 420. Coupled to the egress controller 422 are the egress buffer 424 and MAC interface 402. The egress controller 422 temporally stores outgoing packets it received from the interface receiver in the egress buffer 424 and then subsequently transfers the stored packets from the egress buffer 424 to the MAC interface 402 for transmission to the PHY.

10

15 The network interface unit 426 exchanges data with the host processor through an external I/O or an internal channel. The network interface unit 426 analyzes the control packet made by the host processor, reads internal registers and modifies the content of a lookup table. In a preferred embodiment, every internal unit of the port controller is connected to the network interface (NPI) bus so that all of the internal units may be controlled by the host processor via network interface unit 426.

20

The SSRAM controller 4018 is coupled to the second memory (the SSRAM) and controls access to the SSRAM. The network interface unit 4026 and the packet processing unit 408 are coupled to SSRAM controller and thereby access the SSRAM. The SSRAM controller 4018 arbitrates requests from network interface unit and the packet processing unit.

25

PACKET PROCESSING

30

In the port controller 400, packets may be saved in the packet memory/buffer according to the order they are received. However, in a preferred embodiment of the present invention, a more flexible packet storing arrangement may be utilized which allows the changing the saving order between header fields and body fields of incoming packets.

Typically, when processing a packet with an Ethernet header and a TCP/IP header, only the Ethernet Field or IP Field may be used in a conventional switch to determine switching directions. In contrast, in the embodiment of the switch fabric system set forth herein, both the Ethernet and IP fields as well as the TCP fields may be used to determine switching directions. However, with such a processing capability, such a sequential packet saving order routine may limit the amount of information that can be acquired during processing of a packet by being incapable (or impractical) of providing sufficient time and/or efficiency to process the header. By changing the saving order between header fields and body fields, more information may be acquired during processing and provide an extra timing margin for processing header fields during saving body fields while maximizing efficiency and reducing memory overhead.

Figure 5 is a flowchart of a process 500 for preparing a datagram for transmission through a switch fabric in accordance with an embodiment of the present invention. A datagram having a header portion and a body portion is first received in operation 502. The header portion is separated from the body portion and then forwarded for processing in operations 504 and 506. In operation 508, the separated body portion is stored in a memory. After processing, the header portion is received and stored in the memory in operations 510 and 512.

In an embodiment of the present invention, a space in the memory may be reserved for storing the header portion prior to storing the body portion in the memory with a link to the reserved space also being generated for locating the reserved space. As an option in such an embodiment, storing the processed header may further include

utilizing the link to locate the reserved space and then storing the processed header in the reserved space.

5 In an aspect of the present invention, the body portion may be divided into a plurality of cells. In such an aspect, storing the body portion in the memory may further include sequentially storing each cell in the memory, and generating links between the sequentially stored cells. In another aspect of the present invention, the processed header portion may include a destination vector indicating a destination in the switch fabric system to which the packet is to be routed. In a preferred
10 embodiment of such an aspect, the destination vector may comprise 32 bits of data.

In another embodiment of the present invention, the datagram may be assigned to an output priority queue for subsequent transmission through the switch fabric.

15 Figure 6 is a schematic representation of an illustrative packet 600 in accordance with an embodiment of the present invention. A packet may be defined as a unit of data that is routed between an origin and a destination on a packet switched network. A packet 600 comprises a header portion or field ("header") 602 and a body portion or field ("body") 604. The header portion 602 includes control information that is
20 used for switching while the body 604 generally comprises the remainder of the packet.

In the illustrative packet depicted in Figure 6, the header portion 602 has been processed and includes a destination vector 606 for indicating the destination of the
25 packet, a source vector 608 for indicating the source of the packet, and a length vector 610 indicating the length of the packet (collectively, they are also known as an "internal tag" or "internal tag field"). The header portion 602 may also include the MAC/TCP/IP of the packet and some application data as well.

30 Figure 7 is a flowchart for a process 700 for processing packets 600 in accordance with an embodiment of the present invention. In operation 702, an incoming packet

- is received. The incoming packet includes a header portion and a body portion. The header portion of the packet is separated from the body portion in operation 704. The separated body portion is then stored in memory in operation 706. In operation 2308, the header portion processed, preferably concurrent to operation 2306. In operation 2310, a subsequent (or second) packet is received with the process 700 repeating for this second packet. Once the header portion of the first incoming packet has been processed, the processed header portion is then stored in the memory in operation 712.
- Figure 8 is a first schematic flow diagram illustrating the process 700 for processing packets set forth in Figure 7 in a port controller 400 in accordance with an embodiment of the present invention. An incoming packet 600 is received into the port controller via the MAC interface 402 which routes the packet to the Ingress DMA 404 (see arrow 802). The header portion 602 of the packet is separated from the body 604 of the packet in the Ingress DMA 404 (see operation 704). The Ingress DMA 404 then forwards the header portion 602 to the header queue 406 for processing (see arrow 806) and stores the body portion 604 in memory ("packet memory") 804 via the DRAM controller 410 (see arrow 808 and operation 706).
- For processing of the header portion 602 (see operation 708), the header queue 406 transfers the header portion 602 to an available package processor in the package processor unit 408 (see arrow 810). After the header portion has been processed by the packet processor unit 408, the processed header portion is returned to the header queue 406 (see arrow 812).
- Figure 9 is a second schematic flow diagram illustrating the process 700 for processing packets set forth in Figure 7 in a port controller 400 in accordance with an embodiment of the present invention. After returning to the header queue 406, the processed header portion is then subsequently stored in the packet memory 804 via the Ingress DMA 404 and DRAM controller 410 (see arrow 902).

Upon storing of the processed header portion, the Ingress DMA 404 notifies the output orientated scheduler 412 that the packet is now ready to be transmitted and provides the output orientated scheduler with information for retrieving the packet
5 from the packet memory 804 (see arrow 904). When the output orientated scheduler 412 is ready to transmit the packet, the output orientated scheduler 412 retrieves the packet from the packet memory 804 and forwards the packet to the interface transmitter 414 which outputs the packet from the port controller 400 (see arrow 906).

10

Figure 10 is a schematic diagram representing an illustration of the processing and storing of a plurality of incoming packets entering a port controller in accordance with an embodiment of the present invention. In this illustration, a first incoming packet 1002 is received by Ingress DMA 404 of a port controller followed by a
15 second packet 1004, a third packet 1008 and a fourth packet 1008. As mentioned previously, each packet comprises a header portion (1002a, 1004a, 1006a, 1008a) and body portion (1002b, 1004b, 1006b, 1008b). As each packet is received the header portion from the body portion by the Ingress DMA 404. The separated body portions are then stored in the packet memory 804 while the header portions are
20 forwarded for processing by the packet processor unit 408 (Note: header queue 406 has been left out of Figure 10 for simplification purposes). Once each of the header portions have been processed, they too are then stored in the packet memory 804 via the Ingress DMA 404 (Note the DRAM controller 410 has also been left out of Figure 10 for simplification purposes).

25

As illustrated in Figure 10, the body portions and processed header portions are stored in the packet memory in the order that they are received by the Ingress DMA. For example, upon receipt and separation of the header and body portions of the first packet 1002, the first body portion 1002b is stored in memory 804 while the first
30 header portion 1002a is transferred for processing by the packet processor unit 408.

While the first header portion 1002a is being processed, the Ingress DMA 404 receives the second packet 1004 and separates header and body portions of the second packet. The second body portion 1004b is then stored in memory while the second header portion 1004a is forwarded for processing. At this point in time, 5 processing of the first header portion 1002a has been completed and the processed first header portion is transferred from processing and stored in memory 804.

Next, while the second header portion 1004a is being processed, the Ingress DMA receives the third packet 1006 and separates header and body portions of the third 10 packet. The third body portion 1006b is then stored in the memory while the third header portion 1006a is forwarded for processing. By this time, processing of the second header portion 1004a has been completed and the processed second header portion is transferred from processing and stored in memory. As the third header portion 1006a is being processed, the Ingress DMA receives the fourth packet 1008 15 and separates header and body portions of the fourth packet. The fourth body portion 1008b is then stored in the packet memory while the fourth header portion 1008a is forwarded for processing. By this time, processing of the third header portion 1006a has been completed and the processed third header portion is transferred from processing and stored in memory. Finally, when the fourth header 20 portion 1008a has been processed, it too is transferred from processing and stored in memory.

Figure 11 is a flowchart for a process 704 for separating a header portion from a body portion of a packet in accordance with an embodiment of the present invention. 25 In operation A4302, an incoming packet is divided into cells as it is received. Included in the first cell of the incoming packet is the header portion. The header portion may include the MAC/IP/TCP header(s) as well as application data. In a preferred embodiment, each cell may have a size of 128 bytes. In operation 1104, space is reserved in memory for the header portion (i.e., the first cell). A pointer to

the reserved space in memory is then generated in operation 1106. The header portion is then forwarded to for processing in operation 1108.

Figure 12 is a flowchart for a process 706 for storing a body portion of a packet in memory after separation from the header portion in accordance with an embodiment of the present invention. In operation 1202, the first received cell of the body portion of a packet (i.e., the second cell of the incoming packet) is stored in memory. A pointer is generated in operation 1204 to link the cell to the space reserved for the header portion (i.e., the space reserved for the first cell of the packet). Each subsequently incoming cell of the incoming packet is also stored in memory with a pointer from newly stored cell to the previously stored cell of the packet to link each of the stored cells together (see decision 1206 and operations 1208 and 1210).

Figure 13 is a flowchart for a process 712 for storing a processed header portion of a packet in memory in accordance with an embodiment of the present invention. The processed header portion is received in operation 1302. In operation 1304, the space reserved in memory for the processed header portion is located using the pointer to the reserved space (see operation 1204). The processed header is then stored in the reserved space in operation 1306.

Figure 14 is a schematic diagram representing the storing of cells of an incoming packet into memory in accordance with an embodiment of the present invention. The incoming packet 1400 is divided into a plurality of cells (C₀ 1402, C₁ 1404, C₂ 1406, C₃ 1408, C₄ 1410) as it is received by the Ingress DMA 404 (see operation 1102 of Figure 11). As illustrated in Figure 14, the first cell C₀ 1402 contains the header portion of the packet while the remaining cells C₁ 1404, C₂ 1406, C₃ 1408, C₄ 1410 contain the body portion of the packet.

In the packet memory 1412, a space 1414 is reserved for the first cell C₀ 1402 since it contains the header and a pointer to the reserved space is generated (see operations

1104 and 1106 of Figure 11). In one embodiment, the size of the reserved space may be larger than the size of the received first cell C₀ 1402 to provide sufficient space for any additional data that may be added to the header portion during processing (see 708 of Figure 7). A pointer 1416 to the reserved space 1414 is then
5 generated so that the first cell C₀ 1402 containing the now processed header portion can be subsequently stored in the reserved space (see operation 1106 of Figure 11).

Next, the second received cell C₁ 1404, which contains a first segment of the body portion of the incoming packet 1400 is stored in the packet memory 1412 (see
10 operation 1202 of Figure 12) and a pointer 1418 is generated to link the stored cell C₁ 1404 to the reserved space (see operation 1204 of Figure 12). Once the third cell C₂ 1406 is received, it too may be stored in the packet memory 1412 and a pointer 1420 generated to link the stored cell C₂ 1406 to the previously stored cell C₁ 1404 (see operations 1206, 1208, and 1210 of Figure 12). In a similar fashion, as the
15 fourth and fifth cells C₃ 1408, C₄ 1410 are received, they each may be stored in the packet memory 1412 and have pointers 1422, 1424 generated linking cell C₃ to cell C₂ and C₄ to cell C₃ respectively. In accordance with the process set forth in Figure 12, since the fifth cell C₄ 1410 is the last cell of the packet 1400, the fifth cell C₄ does contain a pointer (see 1426).

20 Figure 15 is a flowchart of a process 1500 for transferring a packet from a port controller to a switch fabric in a switch fabric system in accordance with an embodiment of the present invention. Notification is received in operation 1502 indicating that a packet has been processed in the port controller. The processed
25 packet is assigned and linked to an output priority queue in operations 1504 and 1506. Subsequently, the packet is retrieved via the link in operation 1508 and transmitted from the port controller to the switch fabric in operation 1510.

In an aspect of the present invention, the packet may be linked to the assigned output
30 priority queue in a linked list.

In an embodiment of the present invention, a subsequently processed packet may be assigned and linked to the same output priority queue. As an option, a pointer may link the packet to the subsequently processed packet. In another embodiment of the present invention, the packet may be stored in and retrieved from a memory. As a further option, the packet may be stored in the memory in a plurality of linked cells.

In another aspect of the present invention, the processed packet may include a destination vector indicating one or more destinations for the packet accessible via the switch fabric.

The present invention may also include a data structure for storing the packets in the memory of a port controller. The data structure may comprise a plurality of queues with each queue having a head/start address associated therewith. One or more packets are stored in the memory with each packet assigned to one of the queues. For each queue, the start address may have a link to a first of the packets in the queue and each packet in the queue may have a link to the next subsequent packet in the queue. In one aspect, each packet may be stored in the memory in a plurality of linked cells. In a preferred embodiment, a pointer links a cell of the packet to a subsequent cell of the packet with the last cell of the packet having a Null pointer.

Figure 16 is a flowchart of a process 1600 for generating linked lists of packets queued for egress in accordance with an embodiment of the present invention. In operation 1602, the Ingress DMA 404 notifies the output orientated scheduler 412 that a header portion of a packet has been processed. Subsequently, the Ingress DMA receives back from the output orientated scheduler information regarding to which output queue (for egress from the port controller) the packet has been assigned in operation 1984. A pointer is generated in operation 1606 linking the header portion now stored in the packet memory to the assigned output queue. If the assigned queue does not have any other packets already assigned to it, then the pointer links the processed header portion to the start address of the assigned queue.

On the other hand if the assigned queue does already have one or more other packets assigned to it, then the pointer links the processed header portion to stored header portion of another packet placed just before it in the queue. Operations 3802, 3804, and 3806 are repeated for each header portion processed by the port controller (see operation 1608).

Figure 17 is a schematic diagram depicting the linking of packets stored in the packet memory 804 to their assigned output priority queues in accordance with an embodiment of the present invention. The output orientated scheduler 412 has a plurality of output priority queues 1702, 1704, 1706, 1708 which in the illustrated embodiment comprise four queues. In a preferred embodiment, both the output orientated scheduler and the ingress DMA know the start address of each queue.

Assigned to each priority queue are a plurality of packets stored in the packet memory, with the packets assigned to a particular queue depicted beneath the queue to which they are assigned. Because each queue is a first-in, first-out(FIFO), each stored packet has a next packet pointer ("NPP") providing a link from one packet in a particular queue to the start of the packet following next in the same assigned queue. Additional arrows show how each cell points at the address where its next cell is saved. For example, with reference to queue 0 1702 in Figure 17, the first packet 1710 stored in the packet memory assigned to queue 0 1702 has a pointer 1712 to the start of the stored packet 1714 next in the queue which in turn has a pointer 1716 to the start of the stored packet 1718 third in the queue and so on.

Figure 18 is a schematic diagram illustrating an exemplary packet 1710 stored in the packet memory illustrated in Figure 17 in accordance with an embodiment of the present invention. The exemplary packet 1710 is divided into a plurality of cells with the first cell 1802 includes the processed header portion of the packet and the remaining cells 1804, 1806, 1808, 1810 comprising the body of the packet. Cells 1802, 1804, 1806, 1808 each have a next cell pointer ("NCP") 1812, 1814, 1816,

1818 to the start of the next subsequent cell. Every NCP points at the next cell except the final one of the last cell of a packet. The final NCP can have any value but it may be considered as invalid by the output oriented scheduler because output oriented scheduler knows the size of the packet from internal tag information.

5

In one embodiment, the ingress DMA 404 is responsible for generating the links/pointers. In a preferred embodiment the size of a pointer may be 16 bits (2 bytes).

- 10 As mentioned previously, the first cell 1802 has the processed header portion which includes an internal tag which may indicate the size of the packet and the destination port from which the packet is to be output from the switch fabric system. Also, the first cell 1802 includes the next packet pointer 1712 pointing to the start of next stored packet 1714 in queue 0 1702.

15

Each queue may be a first-in, first-out (FIFO). When the output orientated scheduler 412 is ready to retrieve a packet assigned to one of its FIFO queues from the packet memory, the output orientated scheduler may read the NCP in each cell to find the next cell of the packet. In one embodiment, the NCP of the last cell may be ignored by the output orientated scheduler based on its knowledge of the size of the packet it is retrieving.

20

- Figure 19 is a schematic flow diagram illustrating a process for processing a packet exiting a switch system via a port controller 400 in accordance with an embodiment of the present invention. In this process, the packet has already been processed prior to being received by the port controller 400. The exiting packet is first received into the port controller by the interface receiver 420. The interface receiver 420 forwards the packet to egress controller 422 which may then transfer the packet to the egress buffer 424. When the egress controller is ready to send the exiting packet out of the port controller, the egress controller 422 retrieves the packet from the egress buffer

30

424 and transfers the packet to the MAC interface 402 for transmitting the packet out of the port controller (see arrow 1904).

5

SWITCH FABRIC

Figure 20 is a schematic block diagram of an architecture of a switch fabric component 2000 in accordance with an embodiment of the present invention. The switch fabric component 2000 includes an interface receiver 2002, a switch matrix 2004, an interface transmitter 2006, a central controller 2008, a free cell manager 2010, a queue manager 2012, and an external interface 2014.

For the sake of clarity and ease of comprehension, only one interface receiver 2002 and one interface transmitter 2006 are depicted in Figure 20. However it should be understood, that a switch fabric component may include more than one interface receiver and more than one interface transmitter. For example, in the 8x8 switch fabric components (e.g., 104a) illustrated in Figure 1, each switch fabric component has eight interface receivers and eight interface transmitters. Similarly, the switch fabric component 204 illustrated in Figure 2 includes four interface receivers and four interface transmitters.

The interface receiver 2002 serves to receive packets. The interface receiver has an external I/O for connection to an interface transmitter of another switch fabric component or to an interface transmitter 4014 of a port controller. The interface receiver is coupled to the switch matrix 2004 to permit transferring of a received packet from the interface receiver 2002 to the switch matrix 2004. The function of the interface receiver 2002 is similar to that of interface receiver 420 of a port controller. The interface receiver 2002 configures the link through external I/O and also interacts with central controller 2008 for control.

30

The switch matrix **2004** provides a data path for packets in the switch fabric component. The switch matrix is coupled to both the interface receiver(s) **2002** and the interface transmitter(s) **2006**. The switch matrix receives data (i.e., packets) from the interface receivers. Each incoming packets is stored in memory of the switch matrix until the packet can be transferred to an interface transmitter **2006** for transmission out of the switch fabric component. In one preferred embodiment, the switch matrix has 64 Kbytes of memory.

As mentioned above, the interface transmitter **2006** is coupled to the switch matrix **2004**. The interface transmitter is connectable to the interface receiver of another switch fabric component or that of a port controller via an external I/O. The interface transmitter **2006** transmits packets from the switch matrix out of the switch fabric component. Each interface transmitter **2006** may be connected externally to an interface receiver of another switch fabric component or a port controller and may configure the link through external I/O. Each interface transmitter **2006** also interacts with central controller **2008** for control.

The central controller **2008** is connected to and controls every unit in the switch fabric component. The central controller **2008** may include three components: an ingress unit **2008a**, a matrix unit **2008b** and an egress unit **2008c**. The central controller **2008** manages the storing of incoming packets in the memory of the switching matrix according to free cell addresses provided by the free cell manager **2010**. The central controller **2008** also reads the destination port from the destination vector of a packet and stores the packet in a queue through the queue manager **2012**. In addition, the central controller may generate signals to transfer packets from the switch matrix to an interface transmitter **2006** according to the information of the queue manager **2012**.

The free cell manager **2010** is coupled to the central controller, and the queue manager **2012**. The free cell manager **2010** manages the free cells in the memory of

the switching matrix based on information provided via the central controller **2008**.
In particular, the free cell manager **2010** maintains the information about the
occupied cells and the free cells in the switch matrix in a table and provides this
information to central controller **2008** and the queue manager **2012**.

5

The queue manager **2012** is coupled to the central controller **2008** and the free cell
manager **2010**. The queue manager **2012** receives information about incoming
packets and stores the information as a linked list by FIFO according to output ports.
The queue manager **2012** also manages the queues of exiting packets.

10

The external interface **2014** is connected to the central controller and the switch
matrix. The function of the external interface is the same as the network interface of
the port controller. The external interface exchanges control packets with the
internal channel via an external I/O, and then analyzes these packets to maintain
internal registers in switch fabric component.

15

Figure **21** is a flowchart of a process **2100** for sending a datagram through a switch
fabric in accordance with an embodiment of the present invention. From a received
datagram, a destination vector is extracted (see operations **2102** and **2104**). The
destination vector indicates one or more destinations (i.e., Uni.-cast or multicast) of
the datagram. A port bitmap is generated for the datagram based on the destination
vector in operation **2106**. The datagram is subsequently transmitted in operation
2108 towards the destination utilizing the port bitmap.

20

In an aspect of the present invention, the destination vector may comprise 32 bits of
data. In another aspect of the present invention, the port bitmap may comprise 8 bits
of data.

25

In an embodiment of the present invention, generating the port bitmap may further
include obtaining all destinations of the datagram from the destination vector and
then utilizing a switching table to determine through which switch ports of the

30

switch fabric component can the datagram be routed to reach the destinations. The port bitmap may then be generated to indicate therein all of the determined switch ports through which the can the datagram be routed to reach its intended destinations.

5

In another embodiment of the present invention, transmitting the datagram may also include creating a mask based on the port bitmap, and modifying the port bitmap using the mask upon transmission of datagram. In such an embodiment, modifying the port bitmap using the mask may also include clearing the port bitmap of a data

10

associated with the destination that the datagram was transmitted towards.

In a further embodiment of the present invention, the availability of a switch port through which the datagram may be routed to reach the destinations may also be determined. The datagram may then be transmitted through a switch port

15

determined to be available.

Figure 22 is a schematic block diagram of a switch fabric 2200 comprising four 8 by 8 switch fabric components to illustrate the implementation of switching tables for routing packets through the switch fabric in accordance with an embodiment of the

20

present invention. The underlying principles discussed under this implementation may be utilized with any switch fabric in accordance with the present specification. For purposes of this discussion, is important to remember that the switch fabric and the switch fabric components are all bi-directional.

25

As illustrated in Figure 22, the switch fabric 2200 comprises four 8 by 8 switch fabric components denoted as: SF1 2202a, SF2 2202b, SF3 2202c, and SF4 2202d. Each switch fabric component has eight ports ("switch ports"). For example, in Figure 22, the eight switch ports of SF1 2202a are indicated as: 1/1, 2/1, 3/1, 4/1, 5/1, 6/1, 7/1, and 8/1 ("one-of-one, two-of-one, three-of-one,..." and so on). The

30

eight switch ports of SF2, SF3, and SF4 are similarly denoted in Figure 22. For example, the ports of SF2 2202b are denoted using n/2, (where n is an integer

between 1 and 8), the ports of SF3 2202c are denoted using n/3, and the ports of SF4 2202d are denoted using n/4 (where n is an integer between 1 and 8).

- In the illustrated switch fabric of Figure 22, each switch fabric component has two ports coupled to the two adjacent switch fabric components. For example, ports 5/1 and 6/1 of SF1 2202a are coupled to ports 2/2 and 1/2 of SF2 2202b, respectively, and ports 7/1 and 8/1 of SF1 2202a are coupled to ports 4/3 and 3/3 of SF3 2202c, respectively.
- With continuing reference to Figure 22, the switch fabric 2200 also includes a plurality of device ports. For purposes of this discussion, a device port may be defined as a port of a switch fabric component which may be coupled to an external device, (such as, for example, a port controller) and are not connected to another switch fabric component in the switch fabric. In the exemplary embodiment illustrated in Figure 22, the switch fabric has sixteen (16) device ports (these are numbered 1 through 16 in Figure 22) with each switch fabric component having four device ports. For example, SF1 2202a has device ports 1, 2, 3, and 4, SF2 2202b has device ports 5, 6, 7, and 8, SF3 2202c has device ports 9, 10, 11, and 12, and SF4 2202d has device ports 13, 14, 15, and 16.

SWITCHING TABLES

- In an embodiment of the present invention, switching tables may be utilized to help enhance flexibility of the switch fabric. A switching table may be associated with each switch fabric component to record which devices may be reached through each port of a switch fabric component. In other words, each switching table provides information as to which ports of a particular switch fabric component may be used (i.e., are accessible) to transmit a packet from out of the particular switch fabric component.

Figures 23, 24, 25, 26 are schematic illustrations of exemplary switching tables for the switch fabric 2200 depicted in Figure 22 in accordance with an embodiment of the present invention. In particular, the exemplary switching table 2300 illustrated in Figure 23 is for switch fabric component SF1 2202a, the exemplary switching table 2400 illustrated in Figure 24 is for switch fabric component SF2 2202b, the exemplary switching table 2500 illustrated in Figure 25 is for switch fabric component SF3 2202c, and the exemplary switching table 2600 illustrated in Figure 26 is for switch fabric component SF4 2202d.

Each switching table may be represented as a grid (e.g., a plurality of bitmap arrays) having a plurality of columns 2302 for the device ports of the switch fabric and a plurality of rows 2304 for the ports of the particular switch fabric component. Together the columns and rows form the bitmap arrays of the switching table. In this grid representation, a bitmap array comprises a row of boxes formed by the columns and rows of the switching table grid. In the boxes formed by the columns and rows of each grid are 1's 0's. A "1" in a box indicates that a particular device port is accessible via the port of the particular row while a "0" in a box indicates that a particular device port is not accessible via the port of the particular row. Using such a representation, one can generate a bitmap a port of a switch fabric component based on the array associated with the particular port.

For example, with reference to Figure 23 which represents the switching table 2300 for switch fabric component SF1 2202a, one can ascertain the bitmap array for port 1/1 by looking at the top row which is designated for port 1/1. In this array, there is a "1" in the box under the column for device port 1, and "0" in each of the boxes under the rest of the columns. This indicates that only device port 1 is accessible via port 1/1 of SF1 2202a (i.e., a packet in SF1 whose destination is accessible via transmission through device port 1 must be routed through port 1/1). Similarly, referring to the row (i.e., array) which is designated for port 5/1, we see that there is a "1" in each of the boxes under the columns for device ports 5, 6, 7, 8, 9, and 10, and "0" in each of the boxes under the columns for device ports 1, 2, 3, 4, 11, 12, 13,

14, 15, and 16. This indicates that device ports 5, 6, 7, 8, 9, and 10 are accessible for a packet in SF1 via port 5/1 of SF1 2202a.

It should be noted that in the exemplary illustrative embodiment illustrated in
5 Figures 22, and 23-d, the switching tables have been generated to limit access from any port of a switch fabric component to no more than six device ports. This has been done to help simply understanding of the travel of a packet through the switch fabric 2200 of Figure 22. It should be understood that for the switch fabric 2200 in Figure 22, that a port of a switch fabric component could have access to up to twelve
10 ports (for example, a packet in SF1 transmitted via port 5/1 could have access to device ports 5 through 16 if the switching tables were so generated).

It should also be noted that each of the switching tables illustrated in Figures 23-d have been generated in accordance with a preferred embodiment where columns for
15 up to 32 device ports have been included. This preferred embodiment of the switching tables is to help enable the use of switching tables in the preferred embodiment of the switch fabric illustrated in Figure 1 which has a switch fabric with 32 device ports with 32 port controllers connected thereto. Because the exemplary switch fabric 2200 illustrated in Figure 22 only has 16 device ports (for
20 e.g., connecting 16 port controllers thereto), the boxes in columns 17 through 32 all contain "0's" in each of the four switching tables 23-d.

Figure 27 is a flowchart for a process 2700 for processing a packet through a switch
25 fabric component in accordance with an embodiment of the present invention. In operation 2702, a packet is received by the switch fabric component. A destination vector is extracted from the received packet in operation 2704. A port bitmap is then generated from the extracted destination vector based on the switching table of the switch fabric component in operation 2706. In operation 2708, logic of the switch
30 fabric component checks to determine whether one of the ports of the switch fabric component indicated in the port bitmap is idle. If one of the ports indicated by the

port bitmap is found to be idle, then a mask for the identified idle port is created based on the port bitmap in operation 2710. In operation 2712, the outgoing packet is then transmitted via the identified idle port of the switch fabric and the port bitmap is cleared of the information relating to the port through which the packet
5 was just transmitted. A clearing process is also executed on the destination vector of the outgoing packet at the identified port to modify the destination vector of the outgoing packet. In decision 2714, if it determined that another destination is included in the port bitmap, then operations 2710 and 2712 are repeated for each additional destination.

10

When a packet enters a switch fabric component, the destination vector of the packet and the switching table of the switch fabric component are used to determine through which ports of the switch fabric component the packet should exit. In one embodiment, the determination result may be stored as a linked list. In one aspect,
15 the determination result may comprise a port bitmap. In another aspect, the determination result may comprise the destination vector.

Figure 28 is a schematic diagram illustrating a process 2800 for processing a packet through a switch fabric component where the determination result comprises a port
20 bitmap in accordance with an embodiment of the present invention. Prior to ingress into the switch fabric component, a tag is inserted into the packet 2802 during processing of the packet in the port controller 2804. The tag includes a destination vector 2806 indicating one or more device ports the packet is destined. In a preferred embodiment, the destination vector 2806 may have a length of 32 bits (i.e., a "32-bit
25 destination vector"). The port controller is coupled to a device port of the switch fabric component 2808 to permit transfer of the packet with the inserted tag into the switch fabric component.

Upon ingress of the packet (with inserted tag) 2802 into the switch fabric
30 component 2808, the tag is extracted to obtain the destination vector 2806. A port bitmap 2810 is then generated based on the destination vector 2806 and the

switching table **2812** of the switch fabric component **2808**. In a preferred embodiment, the port bitmap has a length of 8 bits (i.e., an "8-bit port bitmap"). The packet is then stored in cells in the memory of the switch matrix of the switch fabric component. The generated port bitmap **2810** is used to link in a linked list **2814** the
5 cells of memory in which the packet is stored. The port bitmap **2810** is also stored in the linked list **2814** in the switch fabric component **2808**.

When a proper output port is determined to be available, the packet may egress the switch fabric component via the output port. Upon the determination of the
10 availability of a proper output port, the information about the appropriate cells (including the port bitmap) is read from the linked list **2814** to obtain the port bitmap **2810**. The switching table **2812** is then used to produce a mask **2816** for the port of egress. In a preferred embodiment, the mask **2816** has a length of 8 bits (i.e., an "8-bit mask"). Next, the mask **2816** is used to modify the port bitmap **2810** so that
15 information relating to the sending of the packet through the output port is removed from the port bitmap. The modified port bitmap **1018** is then stored in the linked list **2814**. If the modified port bitmap **2818** contains all "0" bits (i.e., no "1's"), then the corresponding memory cell in the switch matrix is freed. In a preferred aspect of such an embodiment, the linked list may include 8 bit/word by 256 word memory for
20 storing port bitmaps. In another preferred aspect of such an embodiment, the switching table may include logic (i.e., circuits) to produce masks. In even another preferred aspect of this embodiment, the rows of the switching table may be exclusive with respect to one another.

25 As a preferred option to the process **2800** illustrated in Figure **28**, the destination vector may be cleared from a packet as the packet exits an outgoing port of a switch fabric component. A benefit to implementing such an option may be seen in the following illustration using the switch fabric **2200** depicted in Figure **22**: In this illustration, suppose a packet entering the switch fabric **2200** has two destinations:
30 device port 2 in SF1 **2202a** and device port 7 in SF2 **2202b**. Figure **30** is a

schematic representation of a destination vector **3000** (in particular, a 32 bit destination vector) for a packet entering SF1 **2202** from device port 1 and having destinations of device port 2 in SF1 **2202a** and device port 7 in SF2 **2202b** in accordance with the present illustration.

5

PORT BITMAP

Figure 29 is a schematic diagram illustrating an exemplary utilization of a port
10 bitmap (in particular, an 8 bit port bitmap) in a switch fabric component in accordance with an embodiment of the present invention. This example is based upon the exemplary switch fabric **2200** illustrated in Figure 22 and takes place in the switch fabric component SF1 **2202a**. In this example, a packet is received by SF1 via a port controller coupled to device port 1 and has 2 destinations: (1) device port
15 2 (located at SF1 **2202a**), and (2) device port 9 (located at SF4 **2202d**).

In SF1 **2202a**, the destination vector is extracted from the packet. As illustrated in Figure 29, the extracted destination vector **2902** has a "1" in bits 2 and 9 **2904**, **2906** indicating the intended destination ports of the packet (device ports 2 and 9). Based
20 on the information of the extracted destination vector **2902**, the switch fabric component then uses its switching table to determine through which of its switch ports the packet can be routed to arrive at the packet's intended destinations. With reference to the switching table **2300** for switch fabric component SF1 illustrated in Figure 23, we see that device port 2 can be accessed through switch port 2/1 of SF1
25 (see the row 2/1) and device port 9 can be accessed through switch ports 5/1 and 6/1 of SF1 (see rows 5/1 and 6/1).

A port bitmap is a bitmap generated by a switch fabric component based on information from the extracted destination vector of a packet received by the switch
30 fabric component and information the switching table of the switch fabric component. Preferably, the port bitmap is generated by the central controller of a

switch fabric component. The port bitmap indicates through which of the switch ports of the switch fabric component the packet needs to be routed through in order for the packet to reach its intended destination(s) (i.e., device port(s)). A port bit map comprises a plurality of bits with each bit being associated with a corresponding switch port of the associated switch fabric component. For example, in a switch fabric component having four switch ports, the port bitmap for that switch fabric has (at least) four bits, with each bit being associated with one of the four switch ports of the switch fabric component. As another example, in the embodiment illustrated in Figure 22, each of the switch fabric components has eight switch ports. Therefore, the port bitmaps generated for each switch fabric component has at least eight bits, with each of the eight bits associated with a corresponding switch port of the respective switch fabric component. In a preferred embodiment, the port bitmap comprises eight (8) bits.

The bits of a port bitmap may each contain either a "1" or "0". In one embodiment, if a bit of the port bitmap contains a "1", then that indicates that the switch port associated with that particular bit may serve as a port for outputting the packet from the switch fabric component. Conversely, if a bit of the port bitmap contains a "0", then that indicates that the switch port associated with that particular bit may not serve as a port for outputting the packet from the switch fabric component.

Returning to Figure 29, we see the port bitmap 2908 that may be generated by switch fabric component SF1 for the received packet based on the extracted destination vector 2902 and the switching table 2300 of SF1. This port bitmap 2908 includes "1's" in bits 2, 5 and 6 2910, 2912, 2914 which are associated with switch ports 2/1, 5/1 and 6/1 of SF1 respectively.

In the present illustration, suppose either switch port 5/1 or 6/1 is determined to be available (i.e., idle) before switch port 2/1. Then, in this example, the packet would first be output via switch port 5/1 or 6/1 towards its intended destination of device port 9 (via SF2 and then SF4). As set forth in the flowchart illustrated in Figure 27

and with continuing reference to Figure 29, a first mask 2916 is generated based on the port bitmap and the outputting switch port. In the present example, this first mask includes "1's" in bits 5 and 6 2918, 2920 which indicate the serviced bits of the port bitmap when the packet is outputted via switch ports 5/1 or 6/1.

5

Continuing with the present example, the port bitmap is then cleared of the served bits using the first mask 2916. In general, a clearing involves the clearing those bits in the port bitmap containing "1's" where the corresponding bits of the mask also contain "1's". In other words, a clearing involves the changing from "1's" to "0's"

10

in the bits in the port bitmap containing "1's" where the corresponding bits of the mask also contain "1's". For example, Figure 29 depicts the bits of the (modified) port bitmap 2908' after being cleared of the serviced bits using the first mask. As illustrated, the modified port bitmap has only a "1" remaining in bit 2 2910 and "0's" in bits 5 and 6 2912, 2914 indicating that that bit 2 has not been cleared (i.e.,

15

switch port 2/1 has not yet been serviced) and that bit 5 and 6 have been cleared (i.e., switch port 5/1 or 6/1 has been serviced).

Next, when switch port 2/1 is determined to be available (i.e., idle), a second mask 2916 is generated for the port bitmap based upon the servicing of switch port 2/1.

20

This mask has a "1" in bit 2 2922 indicating the outputting of the packet via switch port 2/1. As the packet is then outputted from switch port 2/1 (i.e., device port 2), the port bitmap can be cleared using the second mask 2916. Figure 13A shows the modified port bitmap 2908" after being cleared by the second mask 2916. As

25

illustrated in Figure 29, all of the bits of the modified port bitmap 2908" contain "0's" thereby indicating that no more destinations remain in the port bitmap for outputting the packet.

Figure 30 is a schematic representation of an exemplary destination vector 3000, an egress mask 3002 for the destination vector and a modified destination vector 3004

30

in accordance with an embodiment of the present invention. As illustrated in Figure

- 30, in the present illustration, the bits for device ports 2 and 7 **3006, 3008** contain "1's" thereby indicating that the destinations of devices 2 and 7 for the associated packet. In SF1, the packet is transmitted out via 2/1 (i.e., device port 2) and also via 5/1 or 6/1 to SF2 (in order to reach device port 7). Now, if the packet that was sent
- 5 to SF2 was transmitted without any modification to the destination vector **3000**, SF2 will then extract the unmodified destination vector **3000** and process the packet as if one of the destinations from SF2 is device port 2 and therefore send a copy of the packet back to SF1 for transmission out through device port 2.
- 10 To cure this problem, a clearing process may be inserted at the end of each output port of a switch fabric component. The clearing process clears all bits of the destination vector of an outgoing packet except for the bits corresponding to the devices reachable through the outgoing port. This clearing process involves the generation of an egress mask **3002** and then using the egress mask **3002** to modify
- 15 the destination vector of the exiting packet. The egress mask **3002** contains "1's" in all of the bits accessible through that particular egress port (and "0's" in the remaining bits). Continuing with the present illustration, the packet in SF1 can reach device port 7 from either port 5/1 or 6/1 of SF1 (see the switching table for SF1 in Figure 23). With reference to the switching table for SF1 shown in Figure
- 20 23, we can see that the device ports that can be reached via ports 5/1 or 5/6 are device ports 5 – 10. Therefore, the generated egress mask **3002** contains "1's" in bits 5 through 10 and "0's" in the remaining bits as illustrated in Figure 30. This egress mask may then be used to modify the destination of the packet leaving SF1 from ports 5/1 or 6/1 for SF2 by clearing the destination vector of any "1's" in bits
- 25 not covered by the "1's" of the egress mask. Thus, the modified destination vector **3004** for the present illustration depicted in Figure 30 has only a single "1" in the bit for device port 7 **3010** since the "1" in the bit for device port 2 was not "masked" by the "1's" of the egress mask. With such a clearing process, the modified destination vector does not include any information that would cause SF2 to send a copy of the
- 30 packet back to SF1 for egress to device port 2. It should be noted that in a preferred embodiment, that the egress mask of each output port of a switch fabric component

may be the same as the associated row of the switching table corresponding to each respective port. In one aspect of such a preferred embodiment, each output port of a switch fabric component may have a register which contains the same contents as associated row of the switching table of its respective switching component.

5

Figure 31 is a schematic diagram illustrating another process 3100 for processing a packet through a switch fabric component where the determination result comprises a destination vector in accordance with an embodiment of the present invention.

Like the embodiment illustrated in Figure 28, the tag including the destination vector

10 (preferably, a 32-bit destination vector) is inserted into the packet 3102 during processing of the packet in the port controller coupled to the switch fabric component prior to ingress into the switch fabric component. Upon ingress of the packet 3102 into the switch fabric component, the tag is extracted to obtain the destination vector 3104. As in the process 2800 set forth in Figure 28, a port
15 bitmap 3106 is then generated based on the destination vector 3104 and the switching table 3108 of the switch fabric component 3100. In a preferred embodiment, the port bitmap 3106 comprises an 8-bit port bitmap. The generated port bitmap 3106 is used to link in a linked list 3110 the cells of memory in which the packet is stored. However, the 8-bit port bitmap 3106 is not stored in the linked
20 list 3110. Instead, the (32-bit) destination vector 3104 is stored in the linked list 3110.

With continuing reference to Figure 31, when a proper output port is determined to be available for egress of the packet, the cell information is read from the linked list

25 3110 and the destination vector 3104 is obtained. The switching table is used to generate a destination mask 3112 for the egress port. Preferably, the destination mask has a length of 32 bits (i.e., a "32 bit destination mask"). In one aspect, the 32-bit destination mask comprises a row of the switching table corresponding to the egress port. Next, the destination mask 3112 is used to modify the destination vector
30 3114 which is then subsequently stored in the linked list 3110. If, like the modified

port mask 2818 in Figure 28, the modified destination vector 3114 contains all "0" bits, the corresponding memory cell is freed.

- 5 In a preferred aspect of such an embodiment 3100, the linked list may include 32 bit/word by 256 word memory for storing the destination vector 3104. In another preferred aspect of such an embodiment, unlike the switching table 2812 of the embodiment depicted in Figure 28, the switching table 3108 may not need to include logic (i.e., circuits) to produce destination masks since a destination mask is simply a row of the switching table 3108. In even another preferred aspect of this
- 10 embodiment, the rows of the switching table need not be exclusive with respect to one another.

Back Propagation

- 15 Figure 32 is a flowchart of a process 3200 for updating a switching table in a switch fabric in accordance with an embodiment of the present invention. In operation 3202, one or more status packets are received by a switch fabric component without handshaking between the transmitting source(s) and the receiving switch fabric
- 20 component. Each status packet includes information relating to a status of an output port. A switching table is then updated based on the status information of the received status packet in operation 3204.

- 25 In an aspect of the present invention, the status packets may be received via paths/channels/conduits dedicated for transmitting the status packets and separate from paths dedicated for transmitting Ethernet packets through the switch fabric system. In another aspect of the present invention, the status packets may be received periodically and the switching table may be updated in real time.
- 30 In an embodiment of the present invention, a combined status packet may be generated based on the combined status information of the received status packets.

The combined status packet may then be transmitted to other switch fabric components. In one aspect of such an embodiment, the combined status packet may be transmitted from an outer layer switch fabric component to a middle layer switch fabric component.

5

In one aspect of the present invention, the received status packet may be generated from a port controller. In a further aspect of the present invention, the received status packet may be generated by a switch fabric component. In yet another aspect of the present invention, the status packet may comprise 32 bits of data with each bit being associated with a port of the switch fabric component.

10

In one embodiment, a switch system comprising a switch fabric port controller may utilize a handshaking protocol to determine the path by which a packet travels through the switch fabric to its destination port controller(s). However, the utilization of the handshaking protocol may lead to a reduction in the bandwidth of the switch system because: (1) the determination of path through the switch fabric must be performed for each incoming packet, and/or (2) the handshaking protocol is executed using the same channels or conduits by which the packet is propagated through the switch fabric.

15

20

Figure 33 is a schematic diagram illustrating an exemplary arrangement 3300 for back propagation of status and control information in a switch fabric in accordance with an embodiment of the present invention. The exemplary switch fabric 3300 depicted in Figure 33 comprises twelve (12) switch fabric components (SF 1 through SF 12) preferably arranged in the Clos network topology of the multi-port switch architecture 100 illustrated in Figure 1. The exemplary switch fabric 3300 also has thirty-two (32) port controllers (PC01 through PC32), with four port controllers coupled to each of outer layer switch fabric component (SF 1-4 and SF 9-12).

25

30

In the back propagation arrangement 3300, each port controller has a dedicated conduit (e.g., 3302) coupling the respective port controller to its associated outer

layer switch fabric component. Similarly, each outer layer switch fabric component (SF 1-4 and SF 9-12) has a dedicated conduit (e.g., 3304) coupling the respective outer layer switch fabric component to each middle layer switch fabric component (SF 5-8). For sake of the clarity in explaining the back propagation process from the perspective of how SF 1 receives status information from each of the port controllers, only the dedicated conduits from the outer switch fabric components SF 2 through 4 and SF 9 through 12 are illustrated in Figure 33. It should be understood that each of the outer switch fabric components (i.e., SF 1-4 and SF 9-12) has a dedicated conduit to each of the middle switch fabric components (i.e., SF 5-8).

Each of the middle switch fabric components (i.e., SF 5-8) has a dedicated conduit (e.g., 3306) coupling the respective middle switch fabric component to each of the outer switch fabric components (i.e., SF 1-4 and SF 9-12). Again, for sake of the clarity in explaining the back propagation process with respect to how SF 1 receives status information from each of the port controllers, only the dedicated conduit 3306 from middle switch fabric component SF 5 to outer switch fabric component SF 1 is illustrated. However, it should be recognized that an outer layer switch fabric component may be receiving status information from each of the middle layer switch fabric components in a similar fashion as is set forth in the illustration in Figure 33.

With continuing reference to Figure 33, via these dedicated conduits (e.g., 3302, 3304, 3306), status information about the availability of the port controllers receiving a packet (and other control information) may be propagated through the switch fabric. The arrow heads of the dedicated conduits 3302, 3304, 3306 indicate the propagation of the status information from throughout the switch system to switch fabric component SF 1. In a preferred embodiment, the dedicated paths may each have a bandwidth that permits the transmission of 2 bits of information per cycle. In such a preferred embodiment, one bit of the bandwidth may be utilized for transmitting control information and the other bit may be utilized to transmit status information regarding the availability of the port controllers of the switch system.

It should be noted that while Figure 33 depicts switch fabric component SF 1 coupled to the dedicated conduit of middle layer switch fabric component SF 5 to receive status information via SF 5, middle layer switch fabric components SF 6, SF 7, and SF 8 each may have a dedicated conduit coupled to SF 1 to provide status information to SF 1. This redundancy may be beneficial because it allows SF 1 to continue to receive status information about the port controllers even if SF 5 is unable to transmit status information to SF 1.

In general, when the output-port status is propagated backward through the switch fabric, it may be merged with the local output-port status of the switch fabric and thereby eventually building up provided information about the overall path status of the switch fabric. Consequently, with such an arrangement, each component in the switch system may keep real-time updated path status information of the ports, and use this local information to control packet flow without a handshaking protocol between the entry point of the packet into the switch system and the output destination of the packet in the switch system.

Figure 34 is a flowchart of a process 3400 for updating switching tables in a switch fabric system in accordance with an embodiment of the present invention. In a preferred embodiment, the process is executed in a switch system having back processing dedicated conduits as set forth in the portion of the specification relating to Figure 33. As shown in Figure 34, portions of the process are executed in the port controllers of the switch fabric system while other portions of the process are executed in the outer layer and middle layer switch fabric components.

In operation 3402, a status packet is generated in a port controller. The status packet includes information which indicates the current status/availability of the port controller for receiving outgoing packets. The generated status packet is transmitted to the outer layer switch fabric component associated with the port controller in operation 3404. For example, with reference to Figure 33, port controllers P32-08 are associated with switch fabric component SF 2. Therefore, port controllers P32-

WO 02/15496

47

PCT/US01/25083

08 transmit the status packets they generate to switch fabric component SF 2. In the preferred embodiment, the port controller transmits the status packet to the associated outer layer switch fabric component via the dedicated conduit connecting them together (e.g., dedicated conduit 3302).

5

In operation 3406, the outer layer switch fabric component receives and collects the incoming status packets from its associated port controllers. The information from the received status packets is then combined in a status packet generated by the outer layer switch fabric component in operation 3408. The combined status packet is

10

then transmitted to a middle layer switch fabric component in operation 3410. Preferably, the outer layer switch fabric component transmits the combined status packet to the middle layer switch fabric component via the dedicated conduit connecting them together (e.g., dedicated conduit 3304 connecting SF 2 to SF 5).

15

The outer layer switch fabric component also uses the information from the status packet it receives from its port controllers to update its own switching table (see operation 3412).

The middle layer switch fabric component receives and collects the status packets transmitted by the outer layer switch fabric components in operation 3414. For

20

example, with reference to the illustrative arrangement set forth in Figure 33, In operation 3416, the middle layer switch fabric component generates a status packet that combines the status information included in the status packets received from the outer layer switch fabric components. For example, in the exemplary embodiment illustrated in Figure 33, middle layer switch fabric component SF 5 receives status

25

packets from outer layer switch fabric components SF 2- SF 12, with each of these packets having status information about the port controllers associated with each respective switch fabric component (e.g., the status packet from switch fabric component SF 2 has status information about port controllers P32-08, and so on).

30

In operation 3418, the status packet generated by the middle layer switch fabric component is transmitted to a receiving outer layer fabric component (e.g., in the

- example shown in Figure 33, switch fabric component SF 1). Preferably, the middle layer switch fabric component transmits the combined status packet it generated to the receiving outer layer switch fabric component via the dedicated conduit connecting them together (e.g., dedicated conduit 3306 connecting SF 5 to SF 1).
- 5 Additionally, the middle layer switch fabric component uses the information from the status packet it receives from the outer layer switch fabric components (regarding the status of the port controllers) to update its own switching table (see operation 3420).
- 10 After receiving the status packet from the middle layer switch fabric component in operation 3422, the receiving outer layer switch fabric component (e.g., switch fabric component SF 1 in the example set forth in Figure 33) updates its switching table in operation 3424 based on the status information included in the received status packet.
- 15 Thus, under the back propagation arrangement and procedures set forth in Figures 33 and 34, the dedicated conduits may be provided to periodically and/or continually broadcast (preferably in real time) notifications of the availability (i.e., output status) of all of device ports for receiving and outputting packets. Additionally, this back
- 20 propagation arrangement does not necessarily need to utilize a handshaking protocol in order to transmit the status information to all of the switch fabric components.
- Figure 35 is a schematic representation of a status packet 3500 that may be utilized in the process set forth in Figure 34 in accordance with a preferred embodiment of
- 25 the present invention. The status packet comprises a plurality of bits, and in a preferred embodiment the status packet may comprise 32 bits. Each bit of the status packet may be associated with a particular port controller of the switch system. For example, based on the switch system illustrated in Figure 33, bit 01 3502 of the status packet may be associated with port controller PC01, bit 02 3504 of the status
- 30 packet may be associated with port controller P5, and so on. Each bit of the status

packet may be used to indicate the availability (i.e., status) of its associated port controller for receiving/outputting a packet sent through the switch system.

As an illustrative example, a bit comprising a "1" may indicate that the associated port controller is available for receiving/outputting a packet (i.e., the status of the port controller is "available"). Conversely, a bit comprising a "0" may indicate that the associated port controller is not available for receiving/outputting a packet (i.e., the status of the port controller is "unavailable"). Thus, in the exemplary status packet 3500 set forth in Figure 35, the "1's" contained in bits 01 and 02 3502, 3504 indicate that port controllers PC01 and P5 are available while the "0's" contained in bits 09, 16, and 17 3506, 3508, 3510 indicate that port controllers PC09, PC16, PC17 are unavailable.

15

Multi-Bank Buffer

Figure 36 is a flowchart of a process 3600 for storing an incoming datagram in a switch matrix of a switch fabric. The switch matrix has a pair of buffers with each buffer having a pair of portions. Data of a datagram is received in operation 3602 and the buffer portions are sequentially filled with the data in operation 3604. Periodically, transfer of data is allowed from the buffers into the switch matrix in operation 3606. At each period where transfer of data is allowed in the sequence that the buffer portions were filled, the data in one of the buffer portions may be transferred into the switch matrix in operation 3608.

25

In an aspect of the present invention, data in a buffer portion may only be transferred into the switch matrix if the buffer portion is filled with data or contains an end of a datagram (i.e., the tail or last byte of the datagram). In another aspect of the present invention, the switch matrix may comprise a plurality of memory banks for storing the transferred data. In such an embodiment, the memory banks may alternate in receiving data transferred from the buffer portions.

30

In a further aspect of the present invention, the buffer portions may each have equal storage capacities for storing received data. In a preferred embodiment, each buffer portion has a storage capacity of 16 bytes for storing the incoming data. In another
5 preferred embodiment, wherein transfer of data may be allowed every 16 cycles.

Figure 37 is a schematic diagram illustrating the ingress and storage of packets (i.e., datagrams) 3702 in a switch fabric component 3700 in accordance with an embodiment of the present invention. As discussed previously, the switch fabric
10 component 3700 may include a plurality of interface receivers 3704, with the illustrated embodiment having eight ingress receivers (e.g., such as the switch fabric components illustrated in Figure 1 and 22) which are labeled as V unit 1-9.

Each ingress receiver is connected to the switch matrix 3706 thereby permitting
15 transmission of an incoming packet from the particular ingress receiver into the switch matrix 3706. In particular, ingress receiver is coupled to an associated set of input buffers. In one embodiment, each set of input buffers comprises two input buffers 3708, 37100. In one embodiment, each input buffer may have a storage capacity of least to 16 bytes. In a preferred embodiment, each input buffer may have
20 a storage capacity of 32 bytes. As a preferred option, each input buffer may be divided into two portions (e.g., the two portions 3712, 3714 of input buffer 3710). With such an option, it is preferred that the two portions 3712, 3714 of each input buffer have equal data holding capacities (e.g., in the 32 byte capacity preferred embodiment).

25 The central controller 3716 controls the transfer of data stored in the input buffers into memory 3718 of the switch matrix. In one embodiment, the memory of the switch matrix may comprise a plurality of memory banks therein. As illustrated in Figure 37, in the preferred embodiment where each input buffer comprise two
30 portions, the memory of the switch matrix may comprise two memory banks 3720,

3722 with each memory bank preferably being dedicated for receiving data from one of the two portions of each input buffer (e.g., memory bank 3720 is dedicated to input buffer portion 3712 and memory bank 3722 is dedicated to input buffer portion 3714).

5

Figure 38 is a schematic diagram of a scenario illustrating the ingress and storage of packets in a switch fabric component 3700 having a pair of two-portion input buffers 3708, 3710 and a two-memory-bank switch matrix 3706 in accordance with an embodiment of the present invention. A first of the two input buffers 3708

10

comprises a pair of portions ("portion A" 3802 and "portion B" 3804) while the second input buffer 3710 comprises another pair of portions ("portion C" 3712 and "portion D" 3714). Each input buffer 3708, 3710 has a capacity of 32 bytes with each portion 3712, 3714, 3802, 3804 having a capacity of 16 bytes. With reference to Figure 38, each byte in each of the buffers is represented by a square. A blacked square represents that the particular byte is filled with data.

15

The ingress receiver 3704 may transfer 1 byte of an incoming packet it is receiving into a byte of one of the input buffers each cycle. The ingress receiver fills the bytes of an input buffer sequentially during this transfer of data. Also, the central controller periodically permits access to the memory every 16-cycle count to allow transfer of data from a portion of an input buffer into the memory (either memory bank 1 3720 or memory bank 2 3722). In the present scenario, memory access is first allowed on cycle count 7 and then allowed on cycle counts, 23, 39, 55, 71, 87, 103, 119, 135, 151, and so on. Additionally, transfer of data from the input buffers to a memory bank only occurs if a portion of the input buffers has been completely filled with data (i.e., 16 bytes of data) or contains the ending byte of a packet (i.e., the last byte of the packet).

20

25

Further, the memory banks alternate in receiving packet data from the input buffers. For example, in a first memory access, data may be transferred into memory bank 1,

30

in a second memory access, data may be transferred into memory bank 2, and in a third memory access, data may be transferred into memory bank 3, and so on.

5 In the scenario presented in Figure 38, a first incoming packet 3806 having a length of 97 bytes is being transmitted to an ingress receiver 3704 of the switch fabric 3700 followed by a space 3808 of 4 bytes which is then followed by a second incoming packet 3810 having a length of 97 bytes. On cycle count 1, a first byte of the first packet is received by the ingress receiver and transferred to fill a first byte 3812 of portion A 3802 of the first input buffer.

10

At cycle 7, seven of sixteen bytes of portion A 3812 are filled with the first seven bytes of data of the first packet 3806 and the central controller allows access to the memory banks for the first time in this scenario. However, since none of the portions of the input buffers have been completely filled and none of them contain an ending byte of a packet, no transfer of packet data occurs between the input buffers and memory banks at cycle 7.

15

At cycle count 16, all sixteen bytes of portion A 3812 of the first input buffer are filled with the first sixteen incoming bytes of packet 1. However, at this point, the next memory access will not occur until cycle count 23.

20

Figure P11 is a schematic diagram of the scenario set forth in Figure 38 at cycle count 23 in accordance with an embodiment of the present invention. By cycle count 23, seven more bytes of packet 1 have been transferred into portion B 3804 of the input buffer 1. Also, as illustrated in Figure P11, in cycle count 23, memory access is allowed by the central controller and since portion A of input buffer 1 has been filled (since cycle count 16), the 16 bytes of data contained in portion A 3802 is read and written (see arrow P1102) into memory bank 1 3720 and then cleared from portion A 3802.

25

30

By the time of the next memory access at cycle count 39, all of portion B of the input buffer 1 has been filled with data from packet 1 (since cycle count 32) and bytes 33-39 of packet 1 have been transferred to portion C 3712 of input buffer 2 3710.

Therefore, at cycle count 39, all of the bytes of data in portion B are transferred (i.e.,
5 executing read and write functions) into memory bank 2 3722.

At the time of the next memory access at cycle count 55, all of portion C of input buffer 2 has been filled with data from packet 1 (since cycle count 48) and bytes 49-55 of packet 1 have been transferred to portion D 3714 of input buffer 2 3710.

10 Therefore, at cycle count 55, all of the bytes of data in portion C are transferred (i.e., executing read and write functions) into memory bank 1 3720 (since the memory banks alternate in availability for the receipt of data).

At the time of the next memory access at cycle count 71, all of portion D of input
15 buffer 2 has been filled with data from packet 1 (since cycle count 64) and bytes 65-71 of packet 1 have been transferred to portion A 3802 of input buffer 1 3708. Also at cycle count 71, all of the bytes of data in portion D are transferred into memory bank 2 3722.

20 The next memory access occurs at cycle count 87, at which time all of portion A of input buffer 1 has been filled with data from packet 1 (since cycle count 80) and bytes 81-87 of packet 1 have been transferred to portion B 3804 of input buffer 1 3708. Also, all of the bytes of data in portion A are transferred into memory bank 1 3720 at cycle count 87.

25 Figure 40 is a schematic diagram of the scenario set forth in Figure 38 at cycle count 103 in accordance with an embodiment of the present invention. Figure 40 will be used for the following discussion of what takes places in the scenario between cycle counts 88 and cycle count 103. At cycle count 96, all of portion B of input buffer 1
30 has been filled with bytes 81-96 of packet 1. At cycle count 97, the last byte of packet 1 (byte 97) is transferred into the first byte 4002 of portion C of input buffer

B. Since byte 97 of packet 1 is the ending byte of packet 1, the central controller ends the filling of data into of input buffer 2.

During cycles 98, 99, 100, 101, no data is received by the ingress receiver because of the 4 byte space between packets 1 and 2. At cycle count 102, the first byte of packet 2 is received by the ingress receiver and transferred to the first byte of portion A of input buffer 1. It is important to keep in mind that at this point, portion B of input buffer 1 still contains bytes 81-96 of packet 1. The incoming byte of packet 1 received at cycle count 102 that was transferred to input buffer 1 does not overwrite any of the data from packet 1 that is also concurrently stored in input buffer 1 since the incoming first byte of packet 2 **4002** is stored in portion A of input buffer 1 while the stored bytes of packet 1 (bytes 81-96) are stored in portion B of input buffer 1.

At cycle count 103, the next memory access occurs. Bytes 81-96 of packet 1 are transferred to memory bank 2 **3722** and portion B of input buffer 1 is cleared. Also during cycle count 103, the second incoming byte **4006** of packet 2 is transferred into portion B of input buffer 1. Byte 97 of packet 1 **4002** still remains in portion C **4002** of input buffer 2. At cycle count 119, the next subsequent memory access is allowed by the central controller. At this point, byte 97 of packet 1 **4002** is transferred into memory band 1 since byte 97 is an ending byte of a packet. Also transferred into memory bank 1 at cycle count 119 are the first 16 bytes of packet 2 which had completed filled portion A at cycle count 117.

Figure **41** is a schematic illustration of a scenario **4100** similar to scenario presented in Figure **38** where trivial dual input buffers **4102**, **4104** are used instead of the dual two-portion buffers used in Figure **38** in accordance with an embodiment of the present invention. Also, in this scenario **4100**, a single bank memory **4106** is provided in the switch matrix instead of a dual memory bank. Like in the scenario set forth in Figure **38**, each input buffer **4102**, **4104** has a capacity for holding 32 bytes with each byte represented by a square in the respective buffer. A blacked square represents that the particular byte is filled with data. The central controller

- periodically also permits access to the memory every 16-cycle count in this scenario **P800** (just as in the scenario of Figure 38) to allow transfer of data from an input buffer into the memory bank. In this present scenario, memory access is also first allowed on cycle count 7 and then allowed on cycle counts, 23, 39, 55, 71, 87, 103, 119, 135, 151, and so on. Additionally in this scenario **4100**, transfer of data from the input buffers to a memory bank only occurs if an input buffer has been completely filled with data (i.e., 32 bytes of data) or contains the ending byte of a packet (i.e., the last byte of the packet).
- 10 Furthermore, as in the scenario set forth in Figure 38, in this scenario **4100**, a first incoming packet **3806** having a length of 97 bytes is transmitted to the ingress receiver followed by a space of 4 bytes which is then followed by a second incoming packet having a length of 97 bytes. On cycle count 1, a first byte of the first packet is received by the ingress receiver and transferred to fill a byte of the first input
- 15 buffer **4102**.

- As problem with this arrangement when compare to the arrangement set forth in Figure 38 is that the trivial dual input buffers cannot handle variably-sized incoming packets. This problem is illustrated in Figure 41: which shows the situation of the
- 20 scenario **4100** at cycle count 102. At cycle count 96, the first input buffer **4102** is completely filled with bytes 65-96 of the first incoming packet. Because the first input buffer was not filled and did not contain an ending byte of a packet, transfer of the data in the first input buffer **4102** was not allowed in the memory accesses at cycle count 71 and 87. At cycle count 97, the last byte (byte 97) **4108** of the first
- 25 incoming packet is received and transferred to the first byte of the second input buffer **4104**. However, the next memory access will be allowed at cycle count 103. Therefore, the data of the first packet in the first input buffer cannot be transferred until cycle count 103.
- 30 Between cycle counts 98-101, the space between the incoming packets occurs so no data is transferred into the input buffers during these cycles. At cycle count 102, the

first byte 4110 of the second packet is received by the ingress receiver and transferred into the first input buffer (according to standard packet transfer protocols) and therefore overwrites a byte 4112 (byte 65) of the first packet stored in the first input buffer thereby damaging the data of the first packet before it can be
5 transferred into the memory bank 4106. One option to avoid the problem of damaging the data of the first packet such a situation, is to add a third input buffer. Another option is to use the dual two-portion input buffers arrangement as set forth in Figure 38. By using the dual two-portion input buffers arrangement, a hardware reduction of the total chip area of the switch matrix/switch fabric component may be
10 achieved.

Figure 42 is a schematic diagram illustrating memory access size for a switch matrix memory bank in accordance with an embodiment of the present invention. The data transmission rate ("Rx rate") for an incoming packet (i.e., datagram) 4202 into a
15 input buffer 4204 may be expressed as R (bit/sec). Access to the memory 4206 is permitted every T(sec) period ("Access Period") for each input buffer (and may also be the same for each output buffer). In accordance with an embodiment of the present invention, a minimum memory access size for a memory is RT (bit). In a preferred embodiment, a memory access size sufficient to care for variably sized
20 packets is 2RT (bit).

One embodiment of a system in accordance with the present invention may be practiced in the context of a computer such as an IBM compatible personal computer, Apple Macintosh computer or UNIX based workstation. A representative
25 hardware environment is depicted in Figure 43, which illustrates a typical hardware configuration of a workstation in accordance with an embodiment having a central processing unit 4310, such as a microprocessor, and a number of other units interconnected via a system bus 4312. The workstation further includes a Random Access Memory (RAM) 4314, Read Only Memory (ROM) 4316, an I/O adapter
30 4318 for connecting peripheral devices such as disk storage units 4320 to the bus 4312, a user interface adapter 4322 for connecting a keyboard 4324, a mouse 4326, a

speaker 4328, a microphone 4332, and/or other user interface devices such as a touch screen (not shown) to the bus 4312, communication adapter 4334 for connecting the workstation to a communication network 4335 (e.g., a data processing network) and a display adapter 4336 for connecting the bus 4312 to a display device 4338. The workstation typically has resident thereon an operating system such as the Microsoft Windows NT or Windows/95 Operating System (OS), the IBM OS/2 operating system, the MAC OS, or UNIX operating system. Those skilled in the art will appreciate that embodiments present invention may also be implemented on platforms and operating systems other than those mentioned. An embodiment may be written using JAVA, C, and the C++ language and utilizes object oriented programming methodology.

While various embodiments have been described above, it should be understood that they have been presented by way of example only, and not limitation. Thus, the breadth and scope of a preferred embodiment should not be limited by any of the above described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

CLAIMS

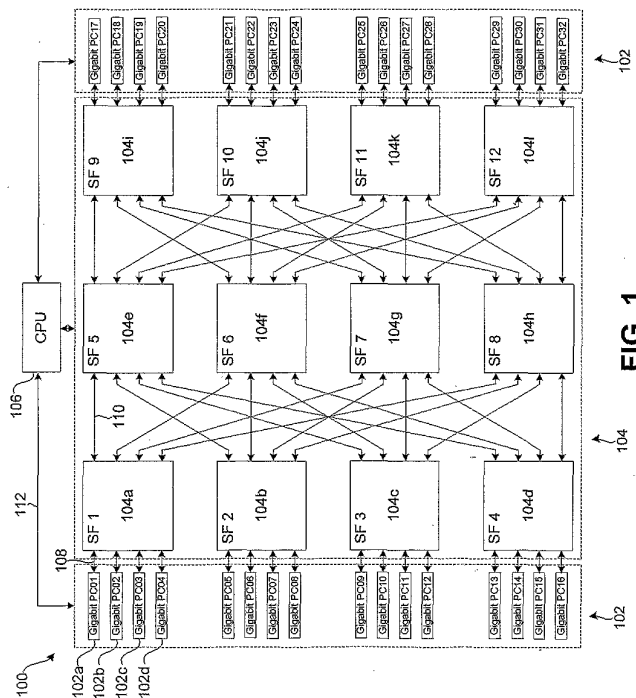
What is claimed is:

- 1 1. A method for sending a datagram through a switch fabric, comprising:
 - 2 (a) receiving a datagram, wherein the datagram includes a destination vector
3 indicating one or more destinations of the datagram;
 - 4 (b) extracting the destination vector from the received datagram;
 - 5 (c) generating a port bitmap for the datagram based on the destination vector;
6 and
 - 7 (d) transmitting the datagram towards the destination utilizing the generated port
8 bitmap.
- 1 2. The method of claim 1, wherein the destination vector comprises 32 bits of
2 data.
- 1 3. The method of claim 1, wherein the port bitmap comprises 8 bits of data.
- 1 4. The method of claim 1, wherein generating the port bitmap further
2 comprises: obtaining the one or more destinations of the datagram from the
3 destination vector, utilizing a switching table to determine through which
4 switch ports of the switch fabric can the datagram be routed to reach the one
5 or more destinations, and generating the port bitmap to indicate therein the
6 determined switch ports.
- 1 5. The method of claim 1, wherein transmitting the datagram further comprises:
2 creating a mask based on the port bitmap, and modifying the port bitmap
3 using the mask upon transmission of datagram.

- 1 6. The method of claim 5, wherein modifying the port bitmap using the mask
2 includes clearing the port bitmap of a data associated with the destination
3 that the datagram was transmitted towards.
- 1 7. The method of claim 1, further comprising *determining* the availability of a
2 switch port through which the datagram may be routed to reach the one or
3 more destinations, and wherein the datagram is transmitted through a switch
4 port determined to be available.
- 1 8. A system for sending a datagram through a switch fabric, comprising:
2 (a) logic for receiving a datagram, wherein the datagram includes a destination
3 vector indicating one or more destinations of the datagram;
4 (b) logic for extracting the destination vector from the received datagram;
5 (c) logic for generating a port bitmap for the datagram based on the destination
6 vector; and
7 (d) logic for transmitting the datagram towards the destination utilizing the
8 generated port bitmap.
- 1 9. The system of claim 8, wherein the destination vector comprises 32 bits of
2 data.
- 1 10. The system of claim 8, wherein the port bitmap comprises 8 bits of data.
- 1 11. The system of claim 8, wherein the logic for generating the port bitmap
2 further includes logic for obtaining the one or more destinations of the
3 datagram from the destination vector; logic for utilizing a switching table to
4 determine through which switch ports of the switch fabric can the datagram
5 be routed to reach the one or more destinations, and logic for generating the
6 port bitmap to indicate therein the determined switch ports.

- 1 12. The system of claim 8, wherein the logic for transmitting the datagram
2 further includes logic for creating a mask based on the port bitmap, and logic
3 for modifying the port bitmap using the mask upon transmission of datagram.
- 1 13. The system of claim 12, wherein the logic for modifying the port bitmap
2 using the mask includes logic for clearing the port bitmap of a data
3 associated with the destination that the datagram was transmitted towards.
- 1 14. The system of claim 8, further comprising logic for determining the
2 availability of a switch port through which the datagram may be routed to
3 reach the one or more destinations, and wherein the datagram is transmitted
4 through a switch port determined to be available.
- 1 15. A computer program product for sending a datagram through a switch fabric,
2 comprising:
3 (a) computer code for receiving a datagram, wherein the datagram includes a
4 destination vector indicating one or more destinations of the datagram;
5 (b) computer code for extracting the destination vector from the received
6 datagram;
7 (c) computer code for generating a port bitmap for the datagram based on the
8 destination vector; and
9 (d) computer code for transmitting the datagram towards the destination utilizing
10 the generated port bitmap.
- 1 16. The computer program product of claim 15, wherein the destination vector
2 comprises 32 bits of data and wherein the port bitmap comprises 8 bits of
3 data.
- 1 17. The computer program product of claim 15, wherein the computer code for
2 generating the port bitmap further includes computer code for obtaining the
3 one or more destinations of the datagram from the destination vector,

- 4 computer code for utilizing a switching table to determine through which
5 switch ports of the switch fabric can the datagram be routed to reach the one
6 or more destinations, and computer code for generating the port bitmap to
7 indicate therein the determined switch ports.
- 1 18. The computer program product of claim 15, wherein the computer code for
2 transmitting the datagram further includes computer code for creating a mask
3 based on the port bitmap, and computer code for modifying the port bitmap
4 using the mask upon transmission of datagram.
- 1 19. The computer program product of claim 18, wherein the computer code for
2 modifying the port bitmap using the mask includes computer code for
3 clearing the port bitmap of a data associated with the destination that the
4 datagram was transmitted towards.
- 1 20. The computer program product of claim 15, further comprising computer
2 code for determining the availability of a switch port through which the
3 datagram may be routed to reach the one or more destinations, and wherein
4 the datagram is transmitted through a switch port determined to be available.



4x4 Gigabit Switch

200

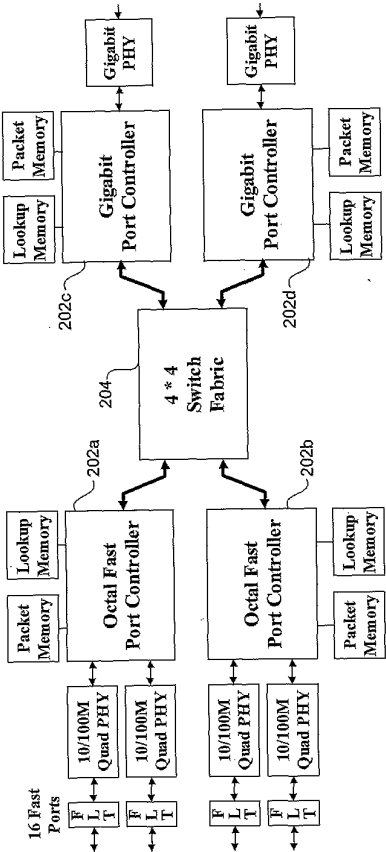


FIG. 2

❑ Gigabit Up-linking

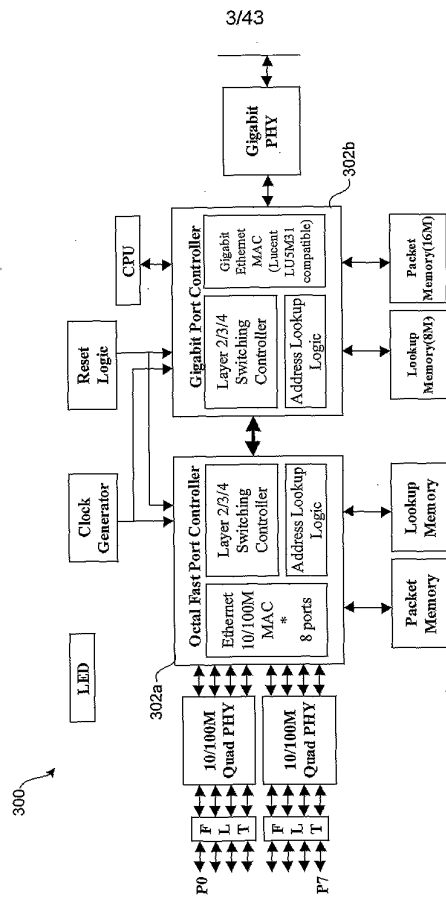


FIG. 3

WO 02/15496

PCT/US01/25083

4/43

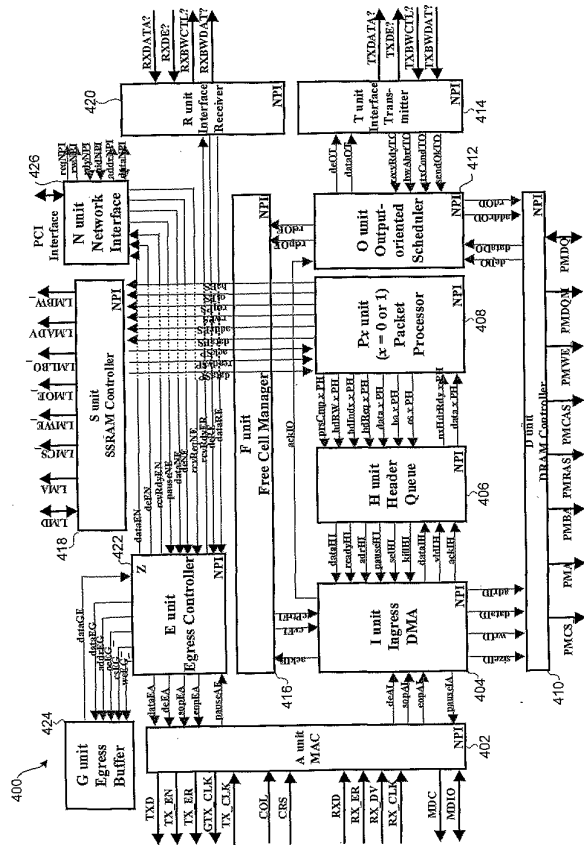


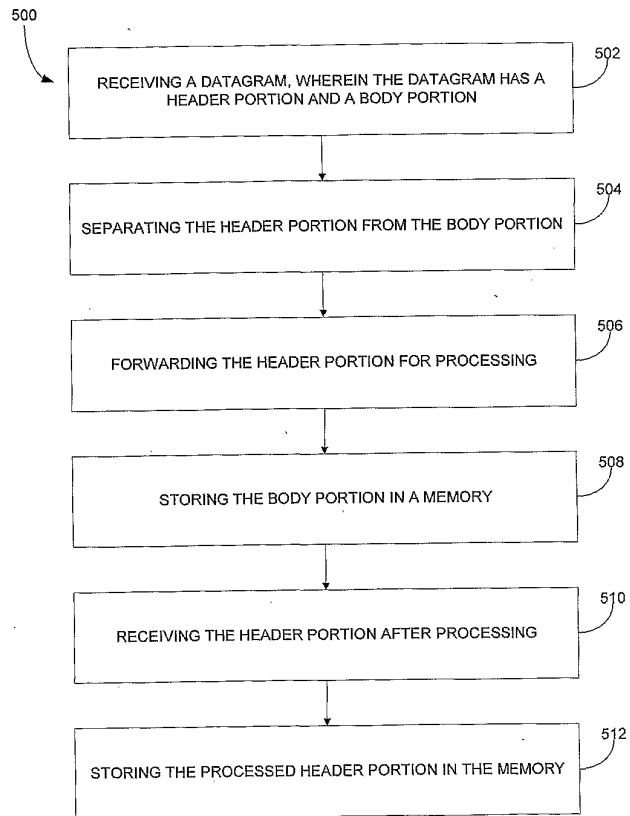
FIG. 4

SUBSTITUTE SHEET (RULE 26)

WO 02/15496

PCT/US01/25083

5/43

**FIG. 5**

SUBSTITUTE SHEET (RULE 26)

6/43

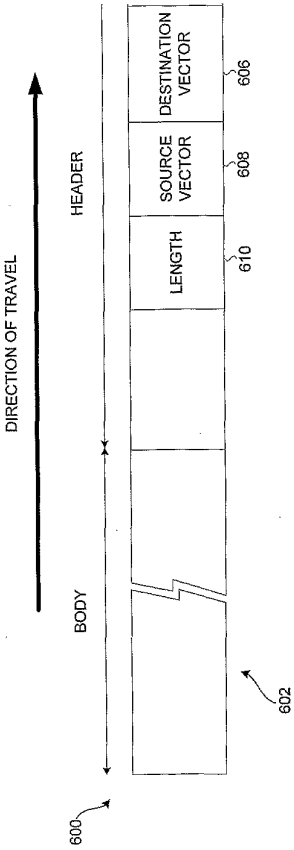


FIG. 6

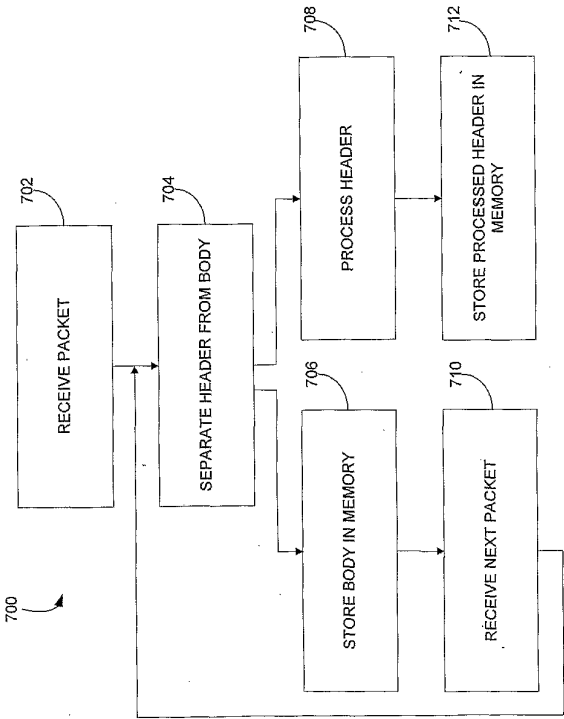


FIG. 7

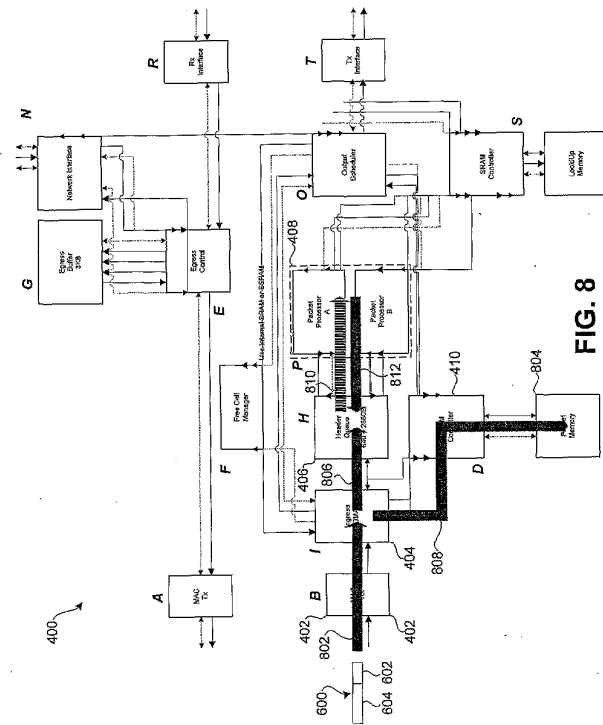


FIG. 8

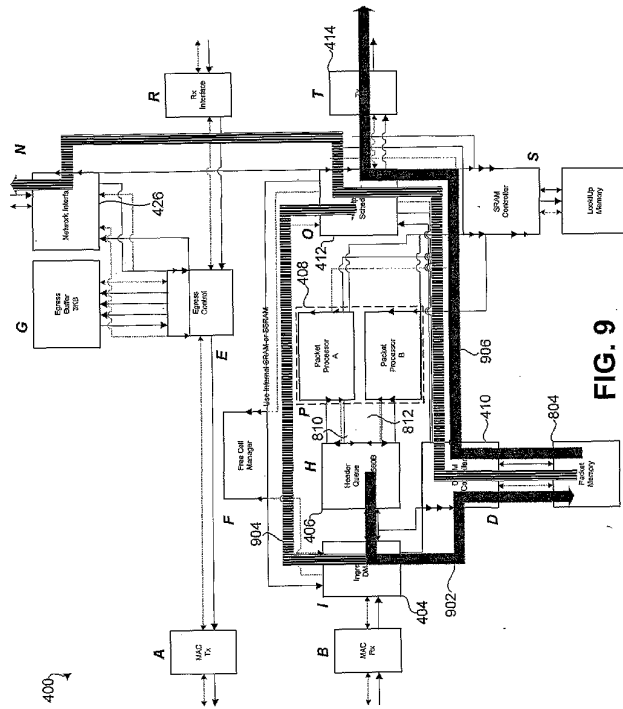


FIG. 9

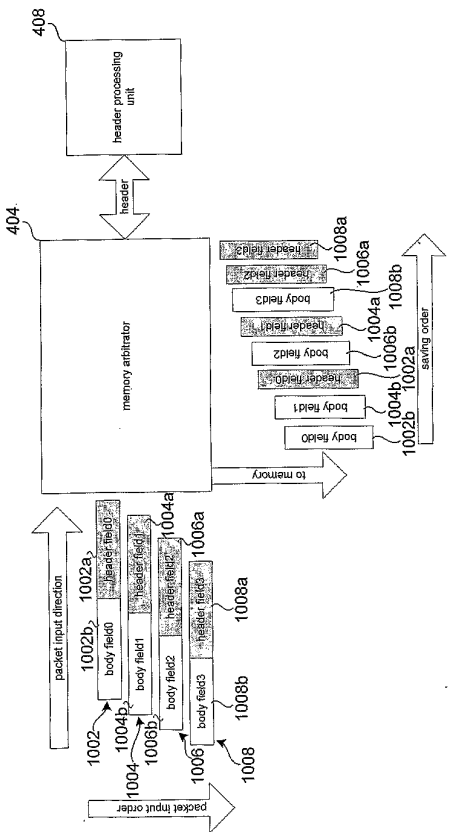
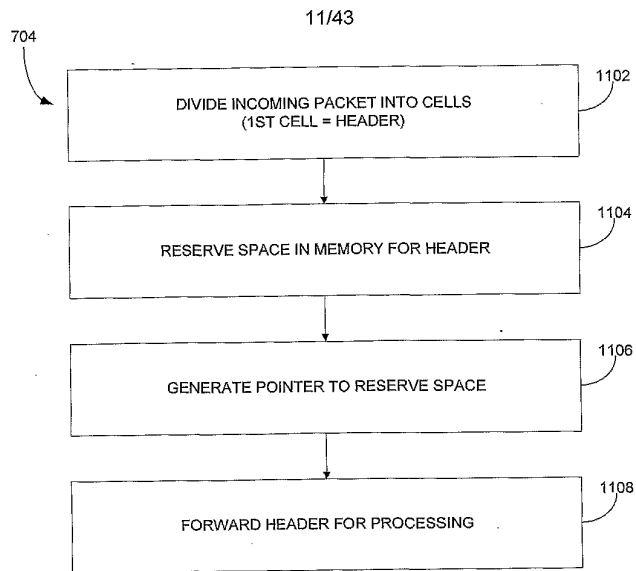


FIG. 10

WO 02/15496

PCT/US01/25083

**FIG. 11**

WO 02/15496

PCT/US01/25083

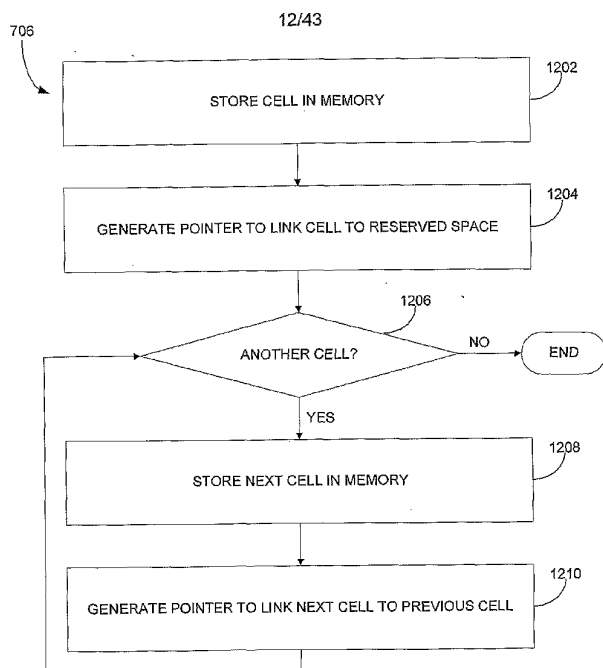
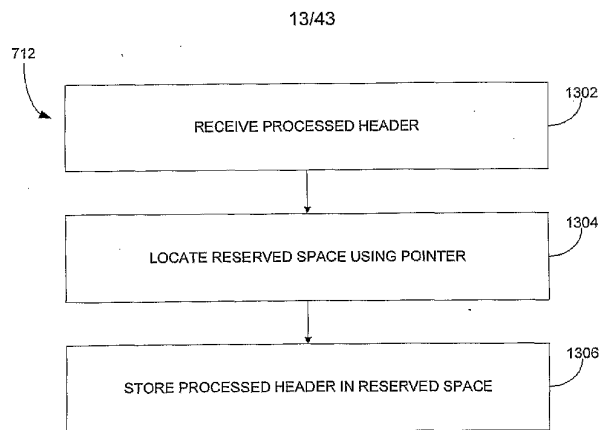


FIG. 12

WO 02/15496

PCT/US01/25083

**FIG. 13**

The diagram illustrates a memory structure with a header and a body. The header (1410) contains a pointer to reserved space (1416) and a reserved space (1414). The body (1418) contains reserved space (1416) and reserved space (1418). The reserved space (1418) is divided into four segments: 4, 8, 12, and 16. The reserved space (1416) is divided into four segments: C₀, C₁, C₂, and C₃. The reserved space (1414) is divided into four segments: C₄, C₅, C₆, and C₇. The reserved space (1410) is divided into four segments: C₈, C₉, C₁₀, and C₁₁.

FIG. 14

WO 02/15496

PCT/US01/25083

15/43

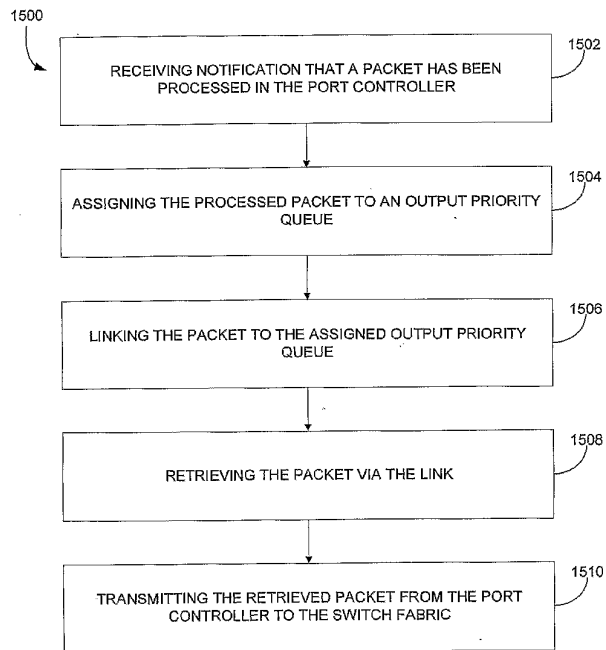


FIG. 15

WO 02/15496

PCT/US01/25083

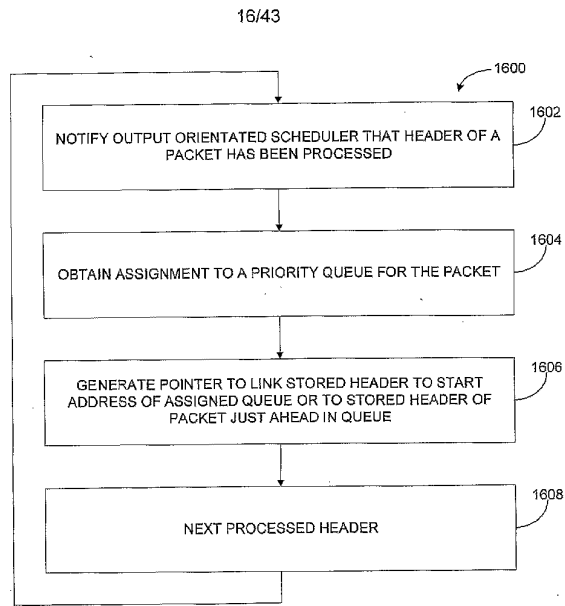


FIG. 16

17/43

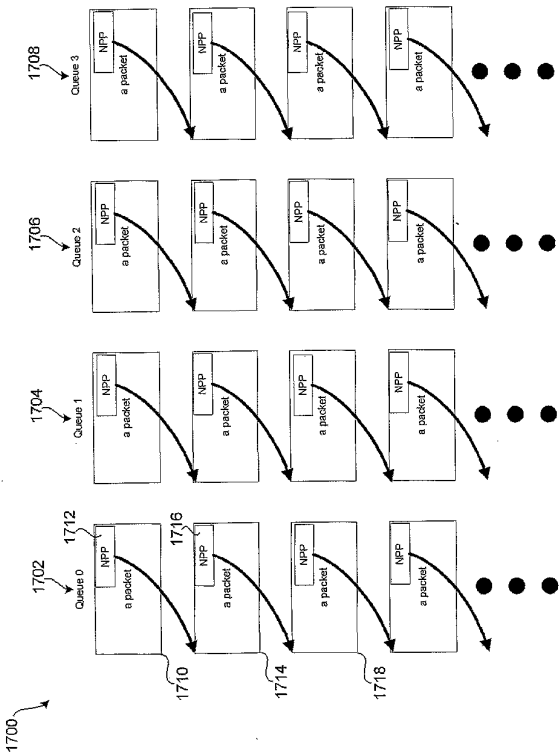


FIG. 17

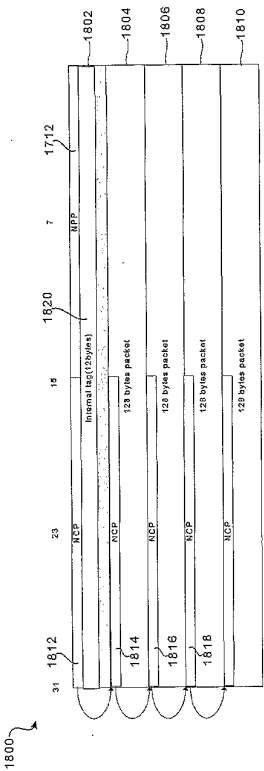
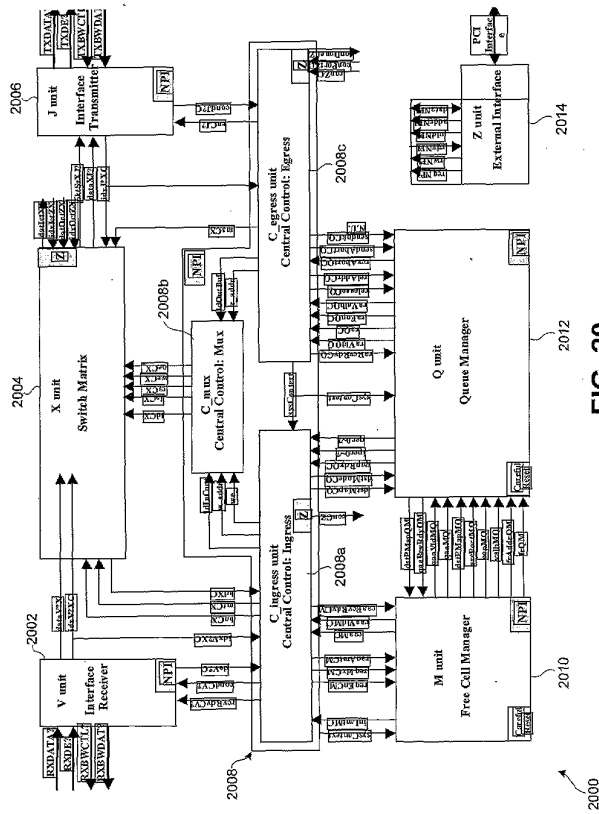


FIG. 18

[illegible]

FIG. 19

FIG. 20



WO 02/15496

PCT/US01/25083

21/43

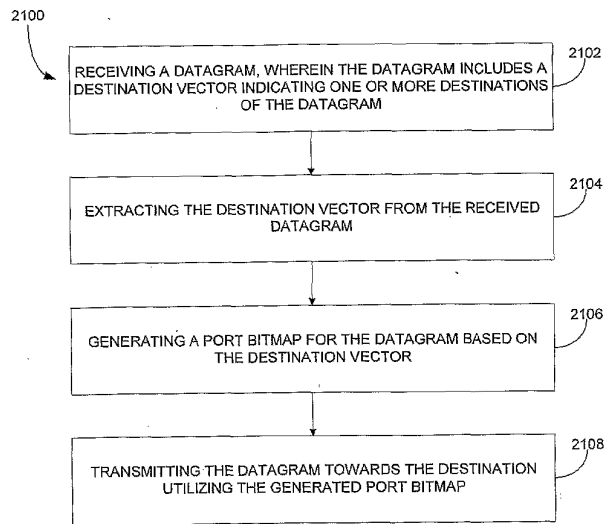
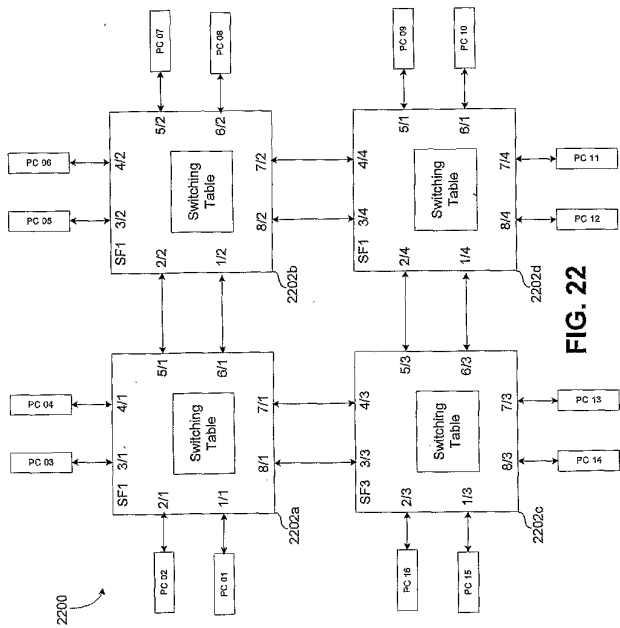


FIG. 21

22/43



SF1 SWITCHING TABLE

2300

DEVICE PORT

23/43

2302

	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1/1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
2/1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3/1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4/1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5/1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
6/1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
7/1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
8/1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

SF PORT

2304

FIG. 23

SF2 SWITCHING TABLE

	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1/2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2/2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3/2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4/2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5/2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6/2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7/2	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8/2	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SF PORT

DEVICE PORT

FIG. 24

FIG. 24

2500

SF3 SWITCHING TABLE

25/43

2302

	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1/3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2/3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3/3	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4/3	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5/3	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6/3	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7/3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8/3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2304

FIG. 25

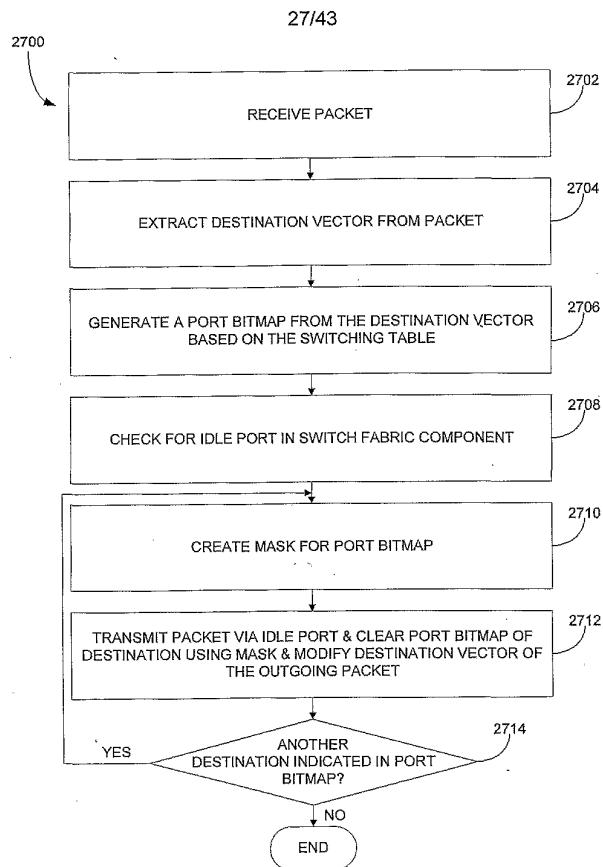
FIG. 25

[illegible]

FIG. 26

WO 02/15496

PCT/US01/25083

**FIG. 27**

SUBSTITUTE SHEET (RULE 26)

WO 02/15496

PCT/US01/25083

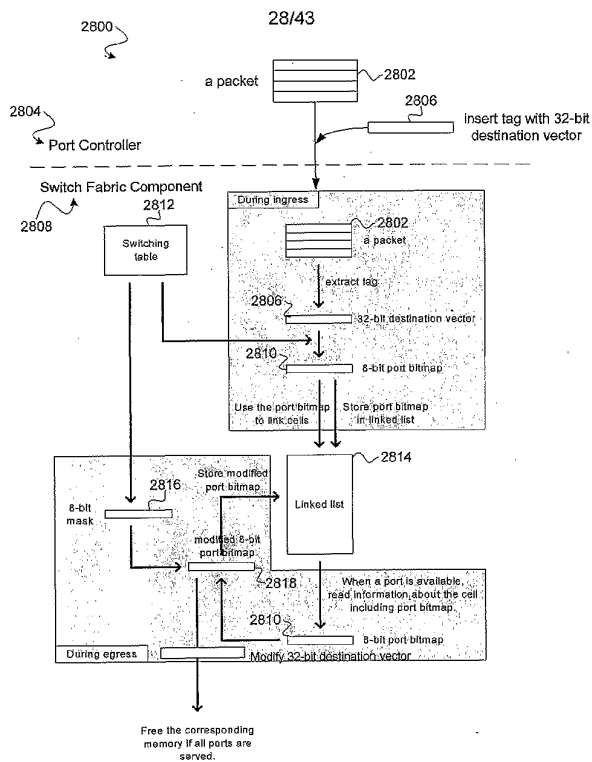


FIG. 28

SUBSTITUTE SHEET (RULE 26)

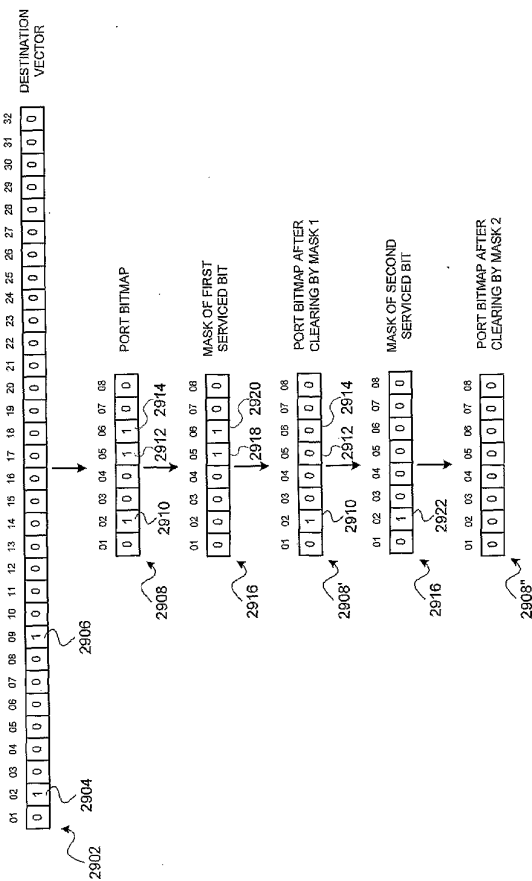


FIG. 29

WO 02/15496

PCT/US01/25083

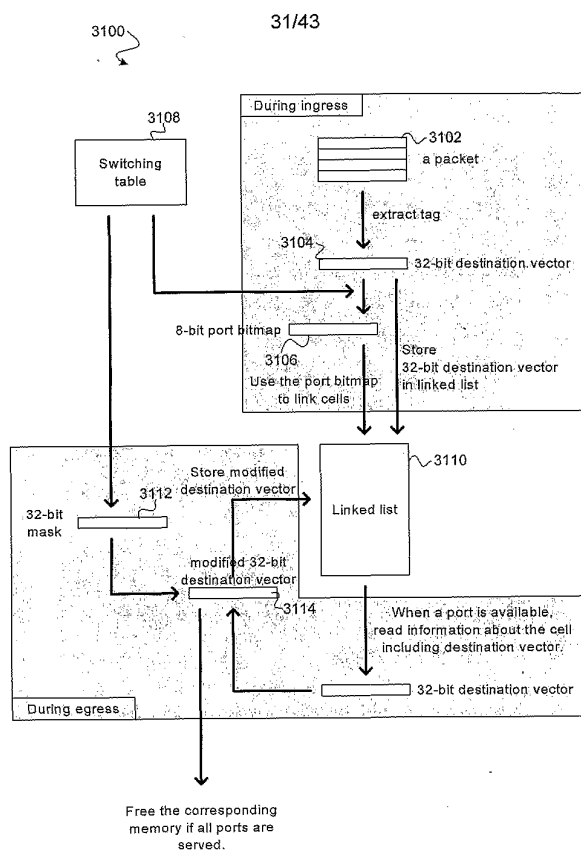


FIG. 31

WO 02/15496

PCT/US01/25083

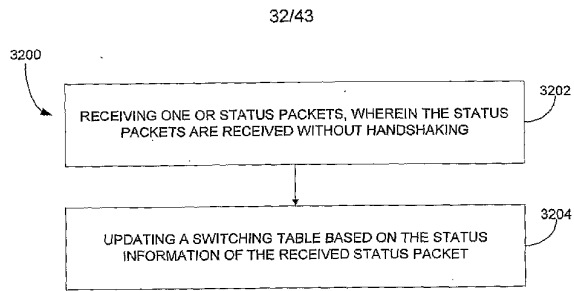


FIG. 32

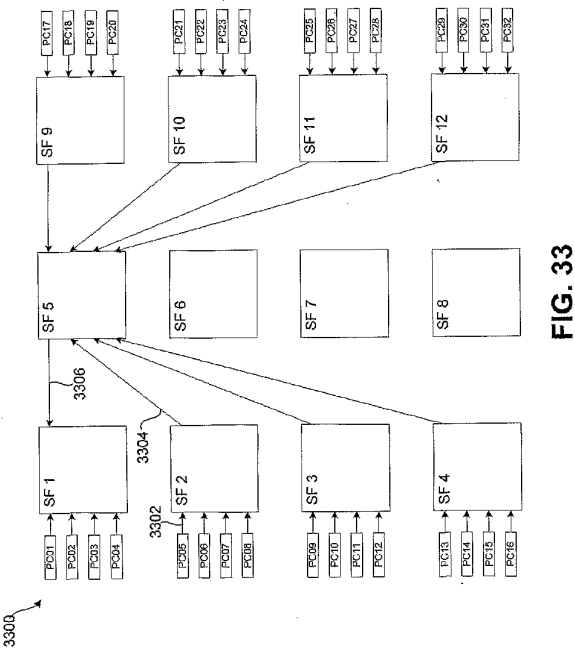
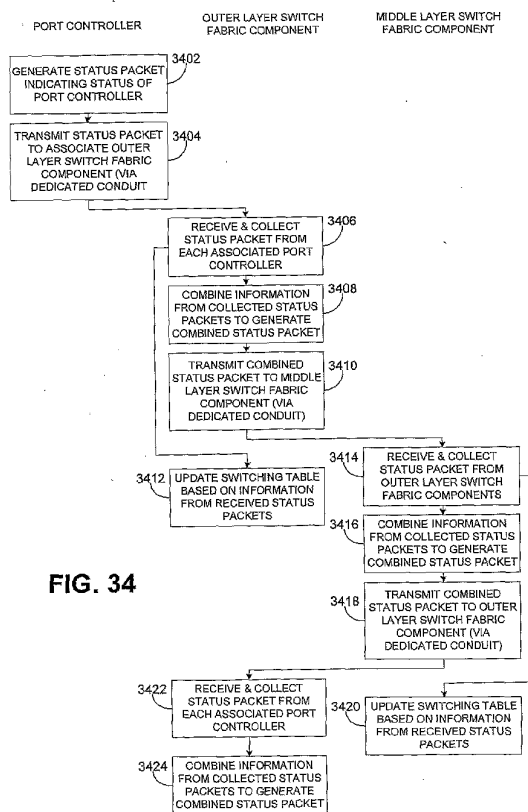


FIG. 33

WO 02/15496

PCT/US01/25083

34/43



SUBSTITUTE SHEET (RULE 26)

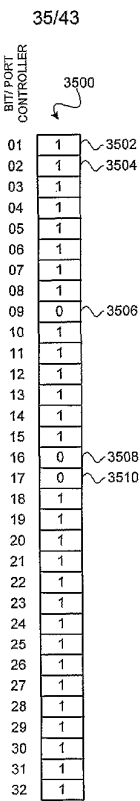


FIG. 35

WO 02/15496

PCT/US01/25083

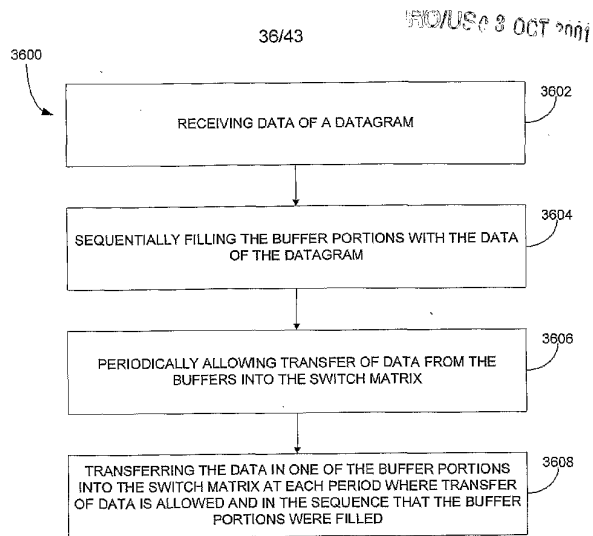


FIG. 36

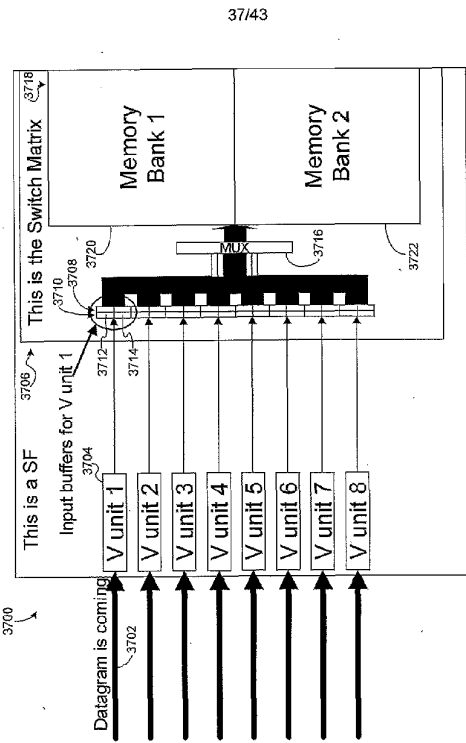


FIG. 37

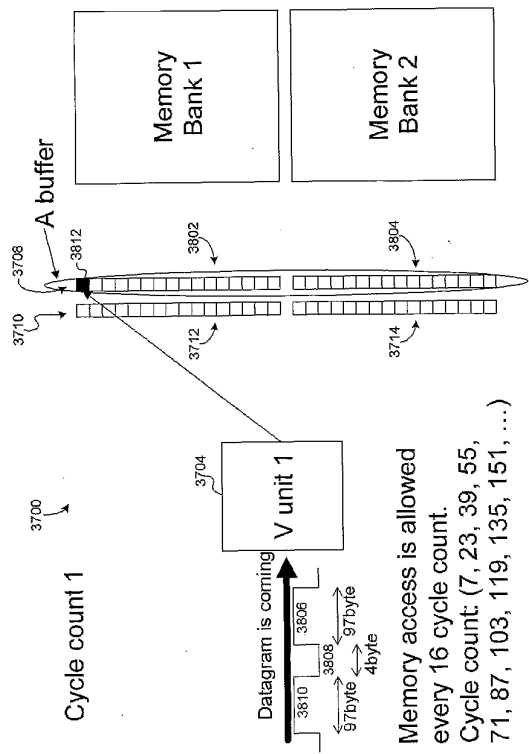
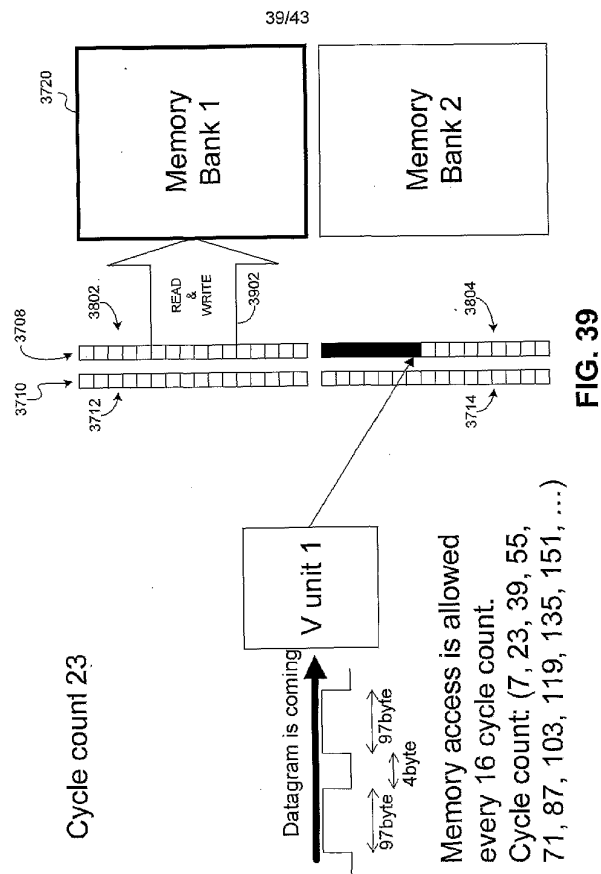


FIG. 38



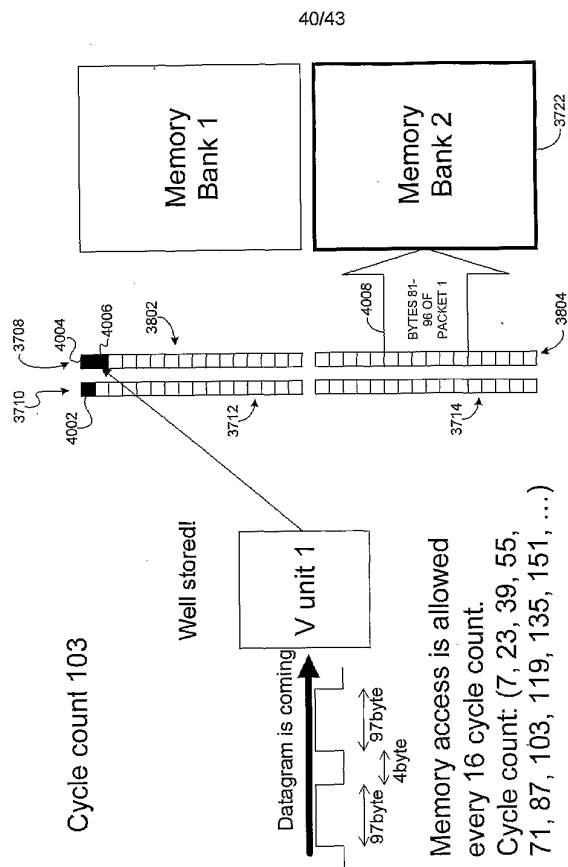
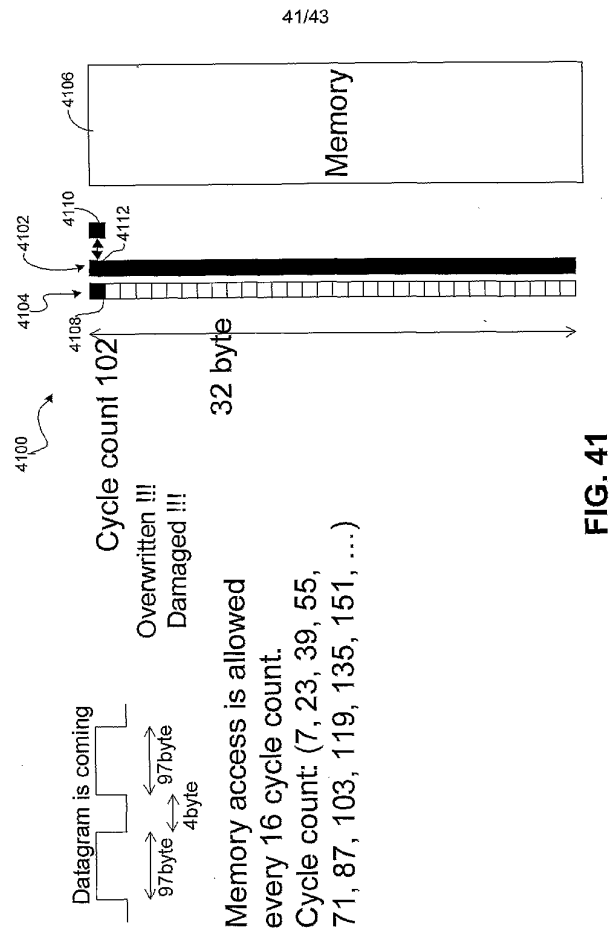


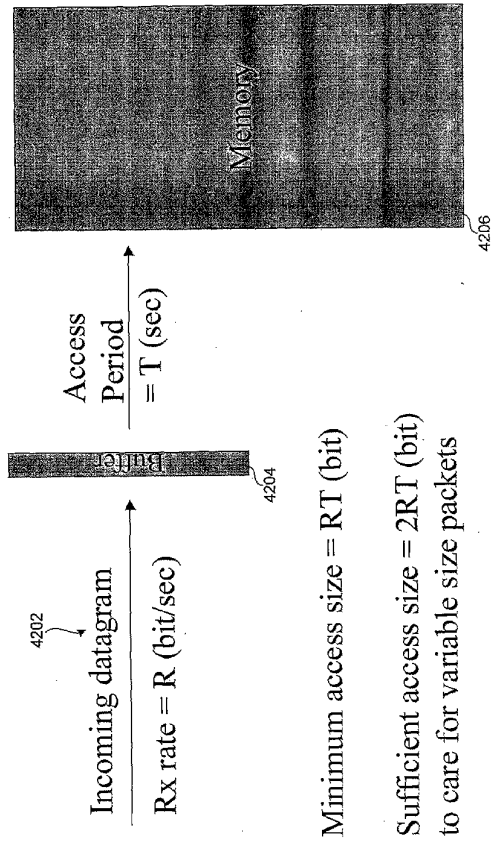
FIG. 40



WO 02/15496

PCT/US01/25083

42/43

**FIG. 42**

43/43

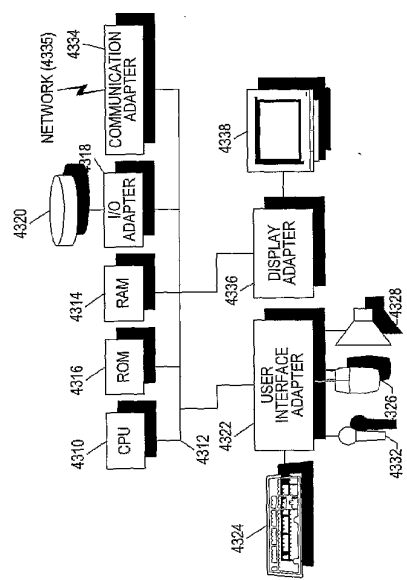


FIG. 43

【国際調査報告】

INTERNATIONAL SEARCH REPORT		International application No. PCT/US01/26093
A. CLASSIFICATION OF SUBJECT MATTER IPC(7) : H04L 19/58 US CL : 370/389 According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) U.S. : 370/288,389,389, 413,416 Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,732,087 A (LAUER et al) 24 March 1998, See Figs. 4 and 5.	1-20
Y	US 5,835,491 A (DAVIS et al) 10 November 1998, See Figs. 2,4,11,12.	1-20
Y	US 6,067,300 A (BAUMERT et al) 25 May 2000, See Figs. 4 and 5.	1-20
Y	US 6,084,858 A (MATTHEWS et al) 04 July 2000, See Cols. 6-10.	1-20
☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.		
* Special categories of cited documents "A" document defining the general state of the art which is not considered to be of particular relevance "B" earlier document published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, each combination being obvious to a person skilled in the art "Z" document member of the same patent family		
Date of the actual completion of the international search 18 OCTOBER 2001		Date of mailing of the international search report 16 NOV 2001
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 805-8230		Authorized officer SALVATORE CANGIALOSI Telephone No. (703) 805-1987

Form PCT/ISA/210 (second sheet) (July 1998)*

フロントページの続き

(81)指定国 AP(GH,GM,KE,LS,MW,MZ,SD,SL,SZ,TZ,UG,ZW),EA(AM,AZ,BY,KG,KZ,MD,RU,TJ,TM),EP(AT,BE,CH,CY,DE,DK,ES,FI,FR,GB,GR,IE,IT,LU,MC,NL,PT,SE,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AT,AU,AZ,BA,BB,BG,BR,BY,BZ,CA,CH,CN,CO,CR,CU,CZ,DE,DK,DM,DZ,EC,EE,ES,FI,GB,GD,GE,GH,GM,HR,HU,ID,IL,IN,IS,JP,KE,KG,KP,KR,KZ,LC,LK,LR,LS,LT,LU,LV,MA,MD,MG,MK,MN,MW,MX,MZ,NO,NZ,PL,PT,RO,RU,SD,SE,SG,SI,SK,SL,TJ,TM,TR,TT,TZ,UA,UG,UZ,VN,YU,ZA,ZW

(特許庁注：以下のものは登録商標)

E T H E R N E T

イーサネット

U N I X

J A V A

(72)発明者 イィ, ジューホワン

大韓民国・305-701・タエジョン・ユソンク・クスンードン・373-1・チップス ビ
ルディング・サード フロア・ケイエアイエスティ

(72)発明者 カン, ムーキューン

大韓民国・305-701・タエジョン・ユソンク・クスンードン・373-1・チップス ビ
ルディング・サード フロア・ケイエアイエスティ

Fターム(参考) 5K030 HA08 KA03 KX17 KX29