(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2009-43276
(P2009-43276A)
(43)公開日　平成21年2月26日(2009.2.26)

(51)Int.Cl.		F I		テーマコード（参考）
G06F 13/38	(2006.01)	G06F 13/38	310A	5B077

審査請求 有　請求項の数 5　ＯＬ　（全 31 頁）

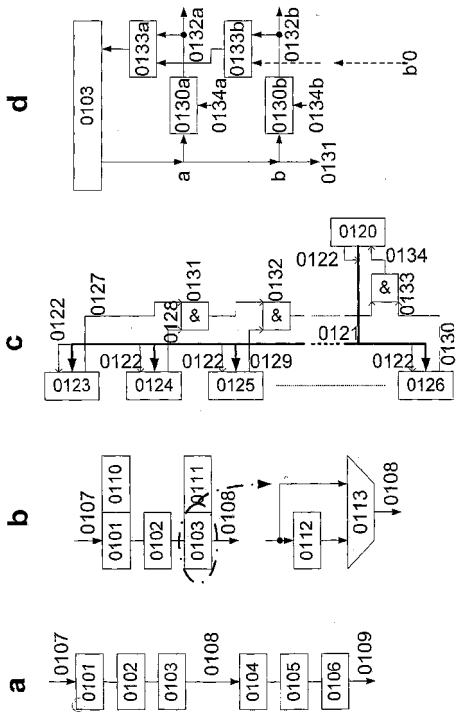
(21)出願番号	特願2008-249116 (P2008-249116)
(22)出願日	平成20年9月26日 (2008.9.26)
(62)分割の表示	特願2002-570104 (P2002-570104)の分割
原出願日	平成14年3月5日 (2002.3.5)
(31)優先権主張番号	101 10 530.4
(32)優先日	平成13年3月5日 (2001.3.5)
(33)優先権主張国	ドイツ(DE)
(31)優先権主張番号	101 11 014.6
(32)優先日	平成13年3月7日 (2001.3.7)
(33)優先権主張国	ドイツ(DE)
(31)優先権主張番号	PCT/EP01/06703
(32)優先日	平成13年6月13日 (2001.6.13)
(33)優先権主張国	欧州特許庁(EP)

(71)出願人　399122756
ペーアーツェーテー　イクスペーペー　テクノロジーズ　アクチエンゲゼルシャフト
ＰＡＣＴ　ＸＰＰ　Ｔｅｃｈｎｏｌｏｇｉｅｓ　ＡＧ
ドイツ連邦共和国　ミュンヘン　ムートマンシュトラーセ　１
Ｍｕｔｈｍａｎｎｓｔｒａｓｓｅ　１，Ｄ－８０９３９　Ｍｕｅｎｃｈｅｎ，　Ｇｅｒｍａｎｙ
(74)代理人　100061815
弁理士　矢野　敏雄
(74)代理人　100094798
弁理士　山崎　利臣

最終頁に続く

(54)【発明の名称】ＦＩＦＯ記憶方法

(57)【要約】　（修正有）
【課題】ＦＩＦＯの出し過程を、以前にデータ語が読出・書込みされている場合には再び開始することができるＦＩＦＯ記憶方法を提供する。
【解決手段】データ流を複数の独立した分岐路に分配し、続いて個々の分岐路を１つのデータ流に統合する。このとき個々のデータ流は時間的に正しい順序で再び統合される。このデータ流の相互の同期のために、ＦＩＦＯの書き込み読み出しに関するフロー制御を、コンフィギュレーション可能なプロトコルにより提供する。
【選択図】図１



【特許請求の範囲】
【請求項１】
読出し過程を、以前にデータ語が読出されている場合には再び開始することができる、ことを特徴とするＦＩＦＯ記憶方法。
【請求項２】
書込み過程を、以前にデータ語が書き込まれている場合には再び開始することができる、ことを特徴とするＦＩＦＯ記憶方法。
【請求項３】
保安レジスタがデータ語のアドレス位置を確保し、当該アドレスで過程を繰り替えることができる、請求項１または２記載の方法。
【請求項４】
ＦＩＦＯの空き状態または満杯状態は、保安レジスタとの比較によって検査される、請求項１から３までのいずれか１項記載の方法。
【請求項５】
保安レジスタは、各アドレスに任意にセットすることができる、請求項１から４までのいずれか１項記載の方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、送信器および受信器が多次元構成されている中でデータを管理および伝送する方法に関する。
【背景技術】
【０００２】
データ流を複数の独立した分岐路に分配し、続いて個々の分岐路を１つのデータ流に統合することは簡単に実行できるべきであり、このとき個々のデータ流は時間的に正しい順序で再び統合される。とりわけ再入可能なコードを処理するにはこの方法が重要である。本発明の方法はとりわけコンフィギュレーション可能なアーキテクチャに適し、コンフィギュレーションおよび再コンフィギュレーションの効率的な制御に重点が置かれる。
【発明の開示】
【発明が解決しようとする課題】
【０００３】
本発明の課題は、産業的使用のために新たなものを提供することである。
【課題を解決するための手段】
【０００４】
この課題は、読出し過程を、以前にデータ語が読出されている場合には再び開始することができる、ことを特徴とするＦＩＦＯ記憶方法および書込み過程を、以前にデータ語が書き込まれている場合には再び開始することができる、ことを特徴とするＦＩＦＯ記憶方法により解決される。有利な実施形態は従属請求項に記載されている。
【発明を実施するための最良の形態】
【０００５】
再コンフィギュレーション可能なアーキテクチャとは、コンフィギュレーション可能な機能および／またはネットワークを備える既存の構成ユニット（ＶＰＵ）であると理解されたい。これはとりわけ１次元または多次元配置された算術的及び／または論理的および／またはアナログおよび／または記憶可能および／または内部／外部でネットワーク化された複数の構成群であり、これらは直接またはバスシステムを介して相互に接続されている。
【０００６】
この構成ユニットにはとりわけシストリック・アレイ、ニューラルネットワーク、マルチプロセッサシステム、複数の計算機構および／または論理セルおよび／または通信／端末セル（ＩＯ）を備えるプロセッサ、ネットワーク構成ユニット、例えばクラスバースイ

ッチ、および公知のＦＰＧＡ、ＤＰＧＡ、Chameleon, XPUTER等の構成素子である。　上記のアーキテクチャは例として明確化のために使用され、以下、ＶＰＵと称する。このアーキテクチャは、任意の算術的、論理的セル（メモリも）および／またはメモリセルおよび／またはネットワークセルおよび／または通信／端末（ＩＯ）セル（ＰＡＥ）からなり、これらは一次元または多次元マトリクス（ＰＡ）に配置することができる。ここでこのマトリクスは種々異なって任意に構成されたセルを有することができ、ここではバスシステムもセルとして理解されたい。このマトリクスには全体としてまたはその一部としてコンフィギュレーションユニット（ＣＴ）が配属されており、これらはＰＡのネットワークおよび機能に影響を与える。

【実施例】

【０００７】

発明の説明

ＶＰＵのコンフィギュレーション可能なセルはデータを正しく処理するために相互に同期していなければならない。このために２つの異なるプロトコルが使用される。１つはデータトラフィックの同期化のためであり、もう一つはデータ処理のフロー制御のためである。データは有利には複数のコンフィギュレーション可能なバスシステムを介して伝送される。コンフィギュレーション可能なバスシステムとはここで、任意のＰＡＥがデータを送信し、受信器ＰＡＥへの接続、並びに受信器ＰＡＥが任意にコンフィギュレーション可能であることを意味する。

【０００８】

データトラフィックの同期化は有利にはハンドシェークプロトコルにより行われ、これはデータと共に伝送される。以下、簡単なハンドシェーク並びに複雑な方法を説明する。これらの有利な適用はそれぞれ実行すべきアプリケーションまたはアプリケーション量に依存する。

【０００９】

フロー制御は、ＰＡＥの状態を指示する信号（トリガ）により行われる。トリガはデータに依存せずに任意のコンフィギュレーション可能なバスシステムを介して実行することができる。すなわち種々異なる送信器および／または受信器を有することができ、同様にハンドシェークプロトコルを有する。トリガは送信側ＰＡＥの状態により発生される（例えばゼロフラグ、オーバフローフラグ、ネガティブフラグ）。これは個々の状態または組合せを転送することにより行われる。

【００１０】

ＶＰＵ内でデータ処理するセル（ＰＡＥ）は種々異なる処理状態を取ることができる。これはセルおよび／または到来するトリガ、または到来したトリガに依存する：

"not configured"：

　データ処理なし

"configured"：

　ＧＯ　到来するすべてのデータを計算する。

【００１１】

　ＳＴＯＰ　到来するデータが計算されない。

【００１２】

　ＳＴＥＰ　正確に１つの計算が実行される。

ＧＯ、ＳＴＯＰおよびＳＴＥＰは次に説明するトリガによりトリガされる。

【００１３】

ハンドシェーク同期

とりわけ簡単で、それでもなお高性能なハンドシェークプロトコルを次に説明する。このハンドシェークプロトコルは有利にはデータおよびトリガの伝送の際に使用される。ハンドシェークプロトコルの制御は有利には固定的のハードウエアに設定されており、ＶＰＵのデータ処理パラダイムの重要な構成部である。

【００１４】

送信器から任意のバスを介して送信された各情報と共にＲＤＹ信号も送信される。この信号は情報の有効性を指示する。

【００１５】

受信器はＲＤＹ信号の付された情報だけを処理し、他の情報はすべて無視する。

【００１６】

情報が送信器により処理され、受信器が新たな情報を受け取ると直ちに、受信器は受領信号（ＡＣＫ）を送信器に送信することにより、送信器が新たに情報を送信しても良いことを指示する。送信器は常に、これが新たなデータを送信する前にＡＣＫの到来を待機する。

【００１７】

２つの動作形式が区別される：

ａ）"depended"：情報を受け取らなければならないすべての入力側は、情報を処理する前に有効なＲＤＹを有する。その後に初めてＡＣＫが発生される。

ｂ）"independent"：情報を受け取る入力側が有効なＲＤＹを有すると直ちに、この入力側がデータを受け取ることができる場合、すなわち先行のデータが処理された場合、この所定の入力側に対してＡＣＫが発生される；それ以外の場合は、データの処理が待機される。

【００１８】

同期の実行およびデータ処理の制御は従来技術によれば固定的に実現されたステートマシン、微細顆粒状にコンフィギュレートされた状態マシン、または有利にはプログラミング可能なシーケンサにより実行できる。プログラミング可能なステートマシンは実行すべきフローに相応してコンフィギュレートされる。Alteraの構成素子ＥＰＳ４４８は例えばこの種のプログラミング可能なシーケンサを実現する。

【００１９】

ＶＰＵに対するハンドシェークプロトコルのタスクは、パイプライン形式のデータ処理の実行である。このデータ処理ではとりわけ各クロックサイクルでデータを各ＰＡＥで処理することができる。この要求はハンドシェークへの特別の負荷となる。ＲＤＹ／ＡＣＫプロトコルの例でこのタスクの問題と解決を示す：

図１ａはＶＰＵ内のパイプラインの構造を示す。データは、有利にはコンフィギュレート可能なバスシステム０１０７，０１０８，０１０９を介してレジスタ０１０１，０１０４に供給される。レジスタには場合によりデータ処理論理回路０１０２，０１０５が後置接続されている。このデータ処理論理回路には出力段０１０３，０１０６が配属されており、出力段は有利にはレジスタを有しており、結果を再びバスに接続する。バスシステム０１０７，０１０８，０１０９を介しても、データ処理論理回路によっても０１０２，０１０５、ＲＤＹ／ＡＣＫプロトコルが同期化のために伝送される。

【００２０】

２つの意味（セマンティック）がＲＤＹ／ＡＣＫプロトコルに対して重要である：

ａ）ＡＣＫは「受信器がデータを引き受ける」を意味する。この効果によりパイプラインは各クロックで動作する。しかしハードウエア技術的実現により、パイプラインストールの場合にはＡＣＫがパイプラインの停止されたすべての段を介して非同期に経過するという問題がある。このことにより甚だしい問題が時間特性に、とりわけＶＰＵが大きくおよび／またはクロック周波数が高い場合に生じる。

ｂ）ＡＣＫは「受信器がデータを引き受ける」を意味する。この効果により、ＡＣＫは常にそれぞれ次の段までだけ経過し、そこにはレジスタが存在している。このことにより発生する問題は、パイプラインがハードウエア技術的実現で必要なレジスタの遅延により各２つのクロックでしか動作しないことである。

【００２１】

この問題の解決は、２つの意味（セマンティック）を図１ｂのように組み合わせることである。図１ｂは段０１０１から０１０３を抜粋して示す。バスシステム０１０７，０１０８，０１０９にはプロトコルｂ）が使用される。これによりレジスタ１０１１０は到来

シアタＲＤＹを伝送されたデータの記録により入力レジスタへＡＣＫより１クロックだけ遅れて再びバスに送出する。この段０１１０はほぼバスプロトコルとプロトコルとの間のプロトコルコンバータとしてデータ処理論理回路内で動作する。

【００２２】

データ処理論理回路はプロトコルａ）を使用する。このプロトコルは後置接続されたプロトコルコンバータ０１１１により形成される。とりわけ０１１１では到来するデータがデータ処理論理回路により実際にバスシステムから取り出されたものであるか否かを予測しなければならない。このことは、付加的バッファレジスタ０１１２を、バスシステム上で伝送すべきデータに対する出力段０１０３，０１０６に挿入することにより解決される。データ処理論理回路により発生されたデータはバスシステムとバッファレジスタに同時に書き込まれる。バスがデータを取り出すことが出来なければ、すなわちバスシステムのＡＣＫがオフであれば、データはバッファレジスタに存在し、バスシステムが待機状態になると直ちにマルチプレクサ０１１３を介してバスシステムに切り替えられる。バスシステムが直ちにデータを取り出す準備が出来ていれば、データはマルチプレクサ０１１３を介してバスに直接転送される。バッファレジスタにより意味（セマンティック）ａ）による受領確認が可能である。なぜならバッファレジスタが空であれば、「受信器がデータを引き受ける」により受領確認することができるからである。なぜならバッファレジスタへの書込みによって、データの失われないことが保証されるからである。

【００２３】

トリガ

ＶＰＵ構成素子では簡単な情報の伝送のためにいわゆるトリガが使用される。トリガはセグメントに分割された一次元または多次元バスシステムによって伝送される。個々のセグメントにはドライバを、信号品質の改善のために装備することができる。複数のセグメントの接続を介して実現されるそれぞれのトリガ接続はユーザによりプログラミングすることができ、ＣＴを介してコンフィギュレートされる。

【００２４】

トリガは後続の情報またはこれらの任意の組合せを主に伝送するが、それだけを伝送するわけではない。

【００２５】

＊計算機構（ＡＬＵ）の状態情報、例えば

－キャリー

－ゼロによる割算

－否定

－アンダーフロー／オーバフロー

＊比較および／またはループの結果

＊ｎビット情報（小さなｎに対して）

＊内部または外部で発生される割込み要求

トリガは任意のセルにより発生され、任意のイベントにより個々のセルで制御される。とりわけトリガはＣＴ、またはセルアレイまたは構成素子の外部にある外部ユニットにより発生される。

【００２６】

トリガは主にＶＰＵ内でのフロー制御に、例えば比較および／またはループに対して用いられる。データ経路および／または分岐はトリガによりイネーブルまたはディスエーブルされる。トリガの別の重要な使用領域はシーケンスの同期化および制御、並びにそれらの情報交換；同様にセル内でのデータ処理の制御である。

【００２７】

トリガの管理およびデータ処理の制御は従来技術によれば、固定的に実現されたステートマシン、微細顆粒状にコンフィギュレートされた状態マシンまたは有利にはプログラミング可能なステートマシンにより実行される。プログラミング可能なステートマシンは実行すべきフローに相応してコンフィギュレートされる。Alteraの構成素子ＥＰＳ４４８は

例えばこの種のプログラミング可能なシーケンサを実現する。

【００２８】

基本方法

ＲＤＹ／ＡＣＫプロトコルによる単純な同期化方法は複雑なデータ流の処理を困難にする。なぜなら正しい順序を維持するためのコストが非常に高いからである。正しい実現はプログラマのタスクである。さらに実現のために付加的リソースが必要である。

【００２９】

以下、この課題を解決する簡単な方法を説明する。

【００３０】

伝送　１：ｎ

これは通常のありきたりな場合である：送信器はデータをバスに書き込む。データは、受領確認としてのＡＣＫがすべての受信器から到来するまでバスに安定して存在する（データが「立っている」）。ＲＤＹはパルスである。すなわち正確に１クロックの間だけ存在する。これによりデータが誤って多重に読み出されることがない。ＲＤＹが実現例に依存してマルチプレクサおよび／またはゲートおよび／または別の適切な伝送素子を制御し、これらがデータ伝送を制御する場合、この制御はこのデータ伝送の時間の間、記憶される（ＲｄｙＨｏｌｄ）。これによりゲートの位置および／またはマルチプレクサおよび／または別の適切な伝送素子がＲＦＹパルスの後も有効に留まり、これによりさらに有効なデータがバスに供給される。

【００３１】

受信器がデータを引き受けると直ちに、受信器はＡＣＫにより受領確認する。正しいデータは受信器による取り出しまでバスに存在することを再度述べておく。ＡＣＫは同様に有利にはパルスとして伝送される。ＲＤＹが以前に制御の記憶のために用いられたマルチプレクサおよび／またはゲートおよび／または別の適切な伝送素子（ＲｄｙＨｏｌｄ参照）をＡＣＫが通過すると、この制御は消去される。

【００３２】

１：ｎの伝送のために有利にはＡＣＫを、新たなＲＤＹが到来するまで保持する。複数の入力側への分岐路である各バスノードでは、到来するＡＣＫが相互に丸められる（ＡＮＤ）。ＡＣＫが持続しているので、最終的に持続するＡＣＫが送信器に生じ、これはすべての受信器のＡＣＫを代表する。ＡＮＤゲートによるＡＣＫチェーンの伝搬時間をできるだけ小さく維持するため、ツリー状のバス構造を選択すること、ないしは処理すべきプログラムのルーティング中に形成することが推奨される。

【００３３】

ＡＣＫが持続することは実現形態に依存して問題となることがある。すなわちＲＤＹが信号ｇｅＡＣＴになり、この信号に対して本来のＡＣＫが存在しないことが生じ得る。なぜなら古いＡＣＫが過度に長く存在するからである。これに対する解決手段は、ＡＣＫを基本的にパルスとし、分岐路で到来する各経路のＡＣＫを記憶するのである。すべての経路のＡＣＫが到来して初めて、ＡＣＫパルスを送信器の方向に転送し、同時に、記憶されているすべてのＡＣＫ（ＡｃｋＨｏｌｄ）および場合によりＲｄｙＨｏｌｄを消去するのである。

【００３４】

図１ｃはこの方法の基本を示す。送信器０１２０はデータをバスシステム０１２１を介して、ＲＤＹ０１２２と共に送出する。複数の受信器０１２３，０１２４，０１２５，０１２６がデータおよびこれに所属するＲＤＹ０１２２を受け取る。各受信器はＡＣＫ０１２７，０１２８，０１２９，０１３０を発生し、これらはそれぞれ適切なブール論理回路０１３１，０１３２，０１３３、例えば論理ＡＮＤ機能を介して結合され、送信器に導かれる０１３４。

【００３５】

図１ｄは、２つの受信器ａ，ｂを有する有利な構成を示す。出力段０１０３がデータと、これに所属するこの実施例ではパルス状のＲＤＹ０１３１を送出する。ＲｄｙＨｏｌｄ

段０１３０は目標ＰＡＥの前方でパルス状ＲＤＹを定常的ＲＤＹに変換する。定常的ＲＤＹはこの例ではブール値ｂ’１を有する。全ＲｄｙＨｏｌｄ段の内容は論理ＯＲ機能０１３３のチェーンを介して０１０３にフィードバックされる。目標ＰＡＥがデータの受け取りを確認すると、それぞれ到来したＡＣＫ０１３３によってだけそれぞれ相応するＲｄｙＨｏｌｄ段がリセットされる。フィードバックされた信号の意味（セマンティック）は、ｂ’１＝"いずれかのＰＡＥがデータを取り出さなかった"である。すべてのＲｄｙＨｏｌｄ段がリセットされると直ちに、ＯＲチェーン０１３３を介して情報ｂ’０＝"すべてのＰＡＥがデータを取り出した"が０１０３に到達する。このことはＡＣＫとして評価される。ＲｄｙＨｏｌｄ段の出力側０１３２はすでに説明したように、バススイッチの制御のために共に使用することができる。

【００３６】

ＯＲチェーンの最後の入力側には論理ｂ’０が印加され、これによりチェーンは正常に機能する。

【００３７】

伝送　ｎ：１

これは比較的複雑な場合である。（Ｆ１）一方では複数の送信器が１つの受信器でマルチプレクスされなければならない。（Ｆ２）他方では通常は、送信器の時間的順序を維持しなければならない。以下、この課題を解決するための複数の方法を説明する。ここでは基本的に有利な方法は存在しない。むしろシステムおよび実行すべきアルゴリズムに応じて、それぞれプログラミング、面倒さ、およびコストの観点から最適のものを選択すべきである。

【００３８】

単純なｎ：１伝送は、それぞれ複数のデータ経路をＰＡＥの入力側に導くことによって実現される。ＰＡＥはマルチプレクサ段としてコンフィギュレートされる。到来するトリガはマルチプレクサを制御し、多数のデータ経路からそれぞれ１つを選択する。必要であれば、マルチプレクサとしてコンフィギュレートされたＰＡＥからなるツリー構造を形成し、複数のデータ流をまとめることができる。この方法では、種々異なるデータ流を時間的に正確に分類するためにプログラミングに特別の注意を払わなければならない。とりわけすべてのデータ経路が同じ長さおよび／または同じ遅延を有し、これによりデータの正しい順序を保証しなければならない。

【００３９】

高性能の統合方法は次のとおりである：

まずＦ１は後置接続されたマルチプレクサを備える任意のアービタにより簡単に解決されると思われるので、考察をＦ２からはじめる。

【００４０】

時間的順序の維持は単純なアービタでは不可能である。図２は第１の可能な実現例を示す。ＦＩＦＯ０２０６が、伝送要求の時間的順序をバスシステム０２０８へ正しく配分し処理するために使用される。このために各送信器０２０１，０２０２，０２０３，０２０４には一義的な番号が割り当てられ、この番号がアドレスを表わす。各送信器はバスシステム０２０８へのデータ伝送を、自分のアドレスをバス０２０９，０２１０，０２１１，０２１２に指示することにより要求する。それぞれのアドレスはマルチプレクサ０２０５を介し、ＦＩＦＯで送信要求の順序に相応して記憶される。ＦＩＦＯはステップごとに処理され、それぞれのＦＩＦＯエントリーのアドレスは別のバス０２０７に指示される。このバスは送信器をアドレシングし、相応に適合するアドレスを有する送信器はバス０２０８へアクセスする。この種の方法のためには、ＶＰＵ技術の内部メモリをＦＩＦＯとして使用することができる。

【００４１】

しかし詳細に考察すると次の問題が発生する：複数の送信器が同時にバスへアクセスしようとすると直ちに、そのアドレスがＦＩＦＯに記憶される送信器を選択しなければならない。そして次のクロックでは次の送信器が選択される。この選択はアービタ０２０５に

より行うことができる。これにより同時性が解決され、このことは通常は問題とならない。リアルタイム適用に対しては優先付けられたアービタを使用することができる。しかしこの方法は簡単な例で失敗する：時点ｔで３つの送信器Ｓ１，Ｓ２，Ｓ３が受信器Ｅを要求すると、ｔでＳ１が、ｔ＋１でＳ２が、そしてｔ＋２でＳ３が記憶される。しかしｔ＋１でＳ４とＳ５が、ｔ＋２でさらにＳ６とＳ１が再び受信器を要求する。今や９つの要求が重なるので、処理は非常に急速に極端に複雑となり、甚だしい付加的ハードウエアコストを必要とする。

【００４２】

従って図２に示した方法は有利には単純なｎ：１伝送に対して使用すべきであり、この伝送では同時のバス要求は発生しない。

【００４３】

この考察によれば、１つの送信器を各クロック毎に記憶するのではなく、すべての送信器の集合が所定の時点で伝送を要求するのが有利であると思われる。それぞれ連続するクロックでそれぞれ新たな集合を記憶するのである。複数の送信器が同じクロックで伝送を要求する限り、これらはメモリの処理の際に仲裁される。

【００４４】

しかし複数の送信器アドレスの記憶は同様に非常に面倒である。以下の構成による簡単な実現が図３に示されている：

・付加的カウンタ（ＲＥＱＣＮＴ，０３０１）はクロックＴを計数する。クロックｔで伝送を要求する各送信器０２０１，０２０２，０２０３，０２０４は、クロックｔにおけるＲＥＱＣＮＴの値（ＲＥＱＣＮＴ（ｔ））を自分のアドレスとして記憶する。

・・・・

・クロックｔ＋ｎで伝送を要求する各送信器は、クロックｔ＋ｎにおけるＲＥＱＣＮＴの値（ＲＥＱＣＮＴ（ｔ＋ｎ））を自分のアドレスとして記憶する。

【００４５】

ＦＩＦＯ０２０６は次にＲＥＱＣＮＴの値（ｔｂ）を所定のクロックｔｂにおいて記憶する。

【００４６】

ＦＩＦＯは記憶したＲＥＱＣＮＴの値を送信要求として別個のバス０２０７に指示する。各送信器はこの値を、自分の記憶した値と比較する。値が同じであれば、送信器はデータを送出する。複数の送信器が同じ値を有する場合、すなわち同時にデータを伝送しようとする場合、伝送は適切なアービタ（ＣＨＮＡＲＢ，０３０２ｂ）により仲裁され、アービタにより制御されるマルチプレクサ（０３０２ａ）によってバスに切り替えられる。アービタの例として可能な構成を以下に説明する。

【００４７】

送信器がＲＥＱＣＮＴに応答しなくなると、すなわちアービタに仲裁のためのバス要求が存在しなくなると（０３０３）直ちに、ＦＩＦＯは次の値までさらに切り替わる。ＦＩＦＯが有効なエントリーを含まなくなると（ｅｍｐｔｙ）、値は無効としてマークされ、これにより間違ったバスアクセスが発生しなくなる。

【００４８】

有利な構成では、送信器０２０１，０２０２，０２０３，０２０４のバス要求が存在したＲＥＱＣＮＴの値だけがＦＩＦＯ０２０６に記憶される。このために各送信器は自分のバス要求０３１０，０３１１，０３１２，０３１３を通知する。これらは例えばＯＲ機能により論理的に結合される０３１４。発生したすべての送信器の送信要求０３１５はゲート０３１６に供給される。このゲートは実際にバス要求が存在していたＲＥＱＣＮＴの値だけをＦＩＦＯ０２０６に送出する。

【００４９】

前記の方法は図４の有利な実施例に相応して次のように最適化することができる：

ＲＥＱＣＮＴ０４１０により値（ＲＥＱＣＮＴ（ｔｂ））の線形シーケンスが発生される。これは、すべてのクロックｔではなく、送信器０３１５のバス要求が存在しているク

ロックが計数される場合である。ＲＥＱＣＮＴから発生し、空隙のない線形シーケンスによって、ＦＩＦＯを簡単なカウンタ（ＳＮＤＣＮＴ、０４０２）により置換することができる。この簡単なカウンタも同様に線形に計数し、その値０４０３は０２０７に相応してそれぞれの送信器を解除接続する。ここでＳＮＤＣＮＴは、送信器がＳＮＤＣＮＴの値に応答しなくなるまでさらに計数する。ＲＥＱＣＮＴの値がＳＮＤＣＮＴの値と等しくなると直ちに、ＳＮＤＣＮＴは計数をストップする。なぜなら最後の値に達したからである。

【００５０】

全体的実現に対しては、最大で必要なＲＥＱＣＮＴの幅が$\log_2$（送信器の数）であることが当てはまる。最大可能値を上回る場合、ＲＥＱＣＮＴとＳＮＤＣＮＴは再び最小値（通常は０）からカウントする。

【００５１】

アービタ

複数のアービタを従来技術によりＣＨＮＡＲＢとして使用することができる。適用に応じて優先付けられたアービタまたは優先付けられないアービタがより適する。ここで優先付けられたアービタでは、リアルタイムタスクの際に所定のタスクを優先することができる。

【００５２】

以下、シリアル・アービタについて説明する。シリアル・アービタはＶＰＵ技術において特に簡単に、リソースを節約して実現することができる。とりわけこのアービタは、優先度を以て動作するという利点を有し、これにより所定の伝送を優先的に処理することができる。

【００５３】

まずバスシステムの可能な基本構造を図５で説明する。ＶＰＵの構成素子は複数のデータバスシステム０５０２からなるネットワークを有し、ここで各ＰＡＥはデータ伝送のために少なくとも１つの端子をデータバスに有する。通常のように、ネットワークは複数の平行なデータバス０５０２から構成されており、これらのデータバスの各々は１つのデータ伝送のためにコンフィギュレートすることができる。残りのデータバスは別のデータ伝送のために自由に使用することができる。

【００５４】

さらにデータバスはセグメント化することができることを述べておく。すなわちコンフィギュレーション０５２１によりバスセグメント０５０２を、ゲートＧを介して隣接するバスセグメント０５２２へ接続することができる。ゲートＧはトランスミッションゲートから構成することができ、有利には信号増幅器および／またはレジスタを有する。

【００５５】

ＰＡＥ０５０１は有利にはマルチプレクサ０５０３を介してまたは同等の回路を介してデータをバス０５０２の１つから取り出す。マルチプレクサ構成の解除接続はコンフィギュレート可能である０５０４。

【００５６】

有利にはＰＡＥ０５１０から発生されたデータ（結果）が同様に依存せずにコンフィギュレート可能０５０５なマルチプレクサ回路を介してバス０５０２に接続される。

【００５７】

図５に示した回路はバスノードにより特徴付けられる。

【００５８】

バスノードに対する簡単なアービタは、図６に示すように実現することができる：

２つのＡＮＤゲート０６０１，０６０２により、簡単なシリアル・アービタの基本素子０６１０が形成される、図６ａ参照。この基本素子は入力側ＲＤＹ、０６０３を有し、この入力側によって、これがデータを伝送し、受信器バスへの解除接続を要求する入力バスを指示する。別の入力側（ＡＣＴＩＶＡＴＥ，０６０４）はこの実施例では論理１レベルによって、先行の基本素子のいずれもが瞬時にバスを仲裁しておらず、従ってこの基本素子による仲裁が許容されることを指示する。出力側ＲＤＹ＿ＯＵＴ０６０５は例えば後置

接続されたバスノードに、基本素子がバスアクセスを解除接続することを指示し（バス要求ＲＤＹが存在する場合）、ＡＣＴＩＶＡＴＥ＿ＯＵＴ０６０６は、基本素子が瞬時には解除接続を実行していないことを指示する。これは、バス要求（ＲＤＹ）が存在していないか、および／または先行のアービタ段が受信器バス（ＡＣＴＩＶＥ）を占有していないからである。

【００５９】

図６ｂに相応して、ＡＣＴＩＶＡＴＥとＡＣＴＩＶＡＴＥ＿ＯＵＴとを基本素子０６１０を介してシリアルにチェーン接続することにより、優先付けられたシリアル・アービタが発生する。ここで第１の基本素子は最上位の優先度を有し、そのＡＣＴＩＶＡＴＥ入力側は常にアクティベートされている。

【００６０】

すでに説明したプロトコルにより、同じＳＮＤＣＮＴ値内では各ＰＡＥが１つのデータ伝送だけを実行することが保証される。なぜなら後続のデータ伝送は別のＳＮＤＣＮＴ値を有することとなるからである。この条件はシリアル・アービタの障害のない機能に対して必要である。なぜならこれにより、優先付けに対して必要な解除接続要請（ＲＤＹ）の仲裁順序が保証されるからである。言い替えると、解除要求（ＲＤＹ）は仲裁中に後から、すでにＡＣＴＩＶＡＴＥ＿ＯＵＴによりバスアクセスの解除接続が不可能であることを指示する基本素子に発生することはできない。

【００６１】

局所性および伝搬時間

基本的に本発明の方法は長い区間を越えて使用することができる。システム周波数に依存する長さを越えると、データの伝送およびプロトコルの実行を１つのクロックで行うのは不可能である。

【００６２】

解決手段はデータ経路を正確に同じ長さに設定することと、統合を正確に１つの個所で実行することである。これによりプロトコルに対する全体的制御信号がローカルとなり、これによりシステム周波数を上昇することができる。データ経路を平衡化するためには、ＦＩＦＯ段が提案される。このＦＩＦＯ段はコンフィギュレート可能な遅延を有する遅延段（遅延線）として動作し、以下、詳細に説明する。

【００６３】

データ経路をツリー状に統合することのできる十分に理想的な解決手段は次のように構成することができる：

変更されたプロトコル、タイムスタンプ

前提条件は、１つのデータ経路が複数の分岐に分割され、後で再び統合されることである。このことは通常は、プログラム構造の「ＩＦ」または「ＣＡＳＥ」の場合のような分岐で行われる。図７ａにはＣＡＳＥに類似する構造が例として示されている。遅くとも分岐０７０１の前の最後のＰＡＥにＲＥＱＣＮＴ０７０２が配属され、このＲＥＱＣＮＴは各データ語に値（タイムスタンプ）を割り当てる。この値は以降常にデータ語と共に伝送される。ＲＥＧＣＮＴは線形に各データ語を計数する。これにより一義的な値によってデータ語のデータ流内での位置を検出することができる。データ語は以降、複数の異なるデータ経路０７０３，０７０４，０７０５に分岐する。各データ語にと共に、これに配属された値（タイムスタンプ）がデータ経路を通って導かれる。

【００６４】

合流されたデータ経路をさらに処理するＰＡＥ０７０８の前方で、マルチプレクサ０７０７がデータ語を再び正しい順序にソートする。このためにマルチプレクサには線形に計数するＳＮＤＣＮＴ０７０６が配属されている。各データ語に割り当てられた値（タイムスタンプ）はＳＮＤＣＮＴの値と比較される。それぞれ通過するデータ語はマルチプレクサにより選択される。所定の時点でデータ語が通過しなければ、選択は実行されない。ＳＮＤＣＮＴは、通過したデータ語が選択された場合だけさらに計数する。

【００６５】

できるだけ高いクロック周波数を達成するために、データ経路の統合は非常に局所的に実行すべきである。これにより線路長は最小になり、これと結び付いた伝搬時間も小さく維持される。

【００６６】

場合によりデータ経路の長さはレジスタ段（パイプライン）により、データ経路全体が共通の点で統合できるまで調整される。このとき、パイプラインの長さがほぼ同じであり、データ語環で大きな時間差が生じないように注意しなければならない。

【００６７】

マルチプレクサへのタイムスタンプの使用

１つのＰＡＥ（ＰＡＥ－Ｓ）の出力は複数のＰＡＥ（ＰＡＥ－Ｅ）にさらに導かれる。これらＰＡＥのうちの１つだけがデータを各クロックサイクルで処理する。ＰＡＥ－Ｅはそれぞれ異なってコンフィギュレートされた固定のアドレスを有し、このアドレスはそれぞれタイムスタンプバスと比較される。ＰＡＥ－Ｓは受信側ＰＡＥを次のようにして選択する。すなわちＰＡＥ－Ｓが、受信側ＰＡＥのアドレスをタイムスタンプバスに出力することにより選択する。このことによりデータがそれぞれ定められたＰＡＥがアドレシングされる。

【００６８】

推測実行とタスクスイッチ

古典的マイクロプロセッサにより推測実行の問題が公知である。この問題は、先行のデータ処理の結果に依存するデータを処理する場合に生じる。しかし依存データの処理はパフォーマンスの理由から前もって、所要の結果が存在する前に開始されるからである。結果が前もって仮定したものと異なると、間違った仮定に基づくデータの処理を新たに実行しなければならない（エラー推測）。一般的にこのことはＶＰＵで発生する。

【００６９】

再ソートおよび類似の方法によってこの問題を最小にすることができる。しかしその発生を除外することはできない。

【００７０】

類似の問題が、データ処理においてＰＡ内で上位のユニット（例えばオペレーティングシステムのタスクスケジューラ、リアルタイム要求等）によりデータ処理が、これが完全に実行される前に中断された場合にも発生する。この場合、パイプラインの状態を次のように確保しなければならない。すなわち、データ処理が再び、最後に生じた結果の計算のためのオペランドの個所の後から開始されるように確保しなければならない。

【００７１】

パイプライン内では２つの関連する状態が発生する：

ＲＤ　パイプラインの開始部では、新たなデータが仮定された、または要求されたことが指示される。

ＤＯＮＥ　パイプラインの終端部では、エラー推測の発生しなかったデータの正しい処理が指示される。

【００７２】

さらに状態ＭＩＳＳ＿ＰＲＥＤＩＣＴを使用することができる。この状態は、エラー推測が発生したことを指示する。補助的にこの状態は、状態ＤＯＮＥを適切な時点で反転することにより発生することもできる。

【００７３】

特殊なＦＩＦＯ

データがメモリに保持され、このメモリから処理のために読み出され、ないしは結果がこれに格納される方法が公知である。このために複数の独立したメモリが使用される。メモリは種々の動作形式で動作することができ、とりわけランダムアクセス、スタック動作モードまたはＦＩＦＯ動作モードを使用することができる。データはＶＰＵで線形処理され、これによりＦＩＦＯ動作モードが頻繁に優先的に使用される。例としてメモリのＦＩＦＯ動作モードに対する特別の拡張を紹介する。この拡張ＦＩＦＯ動作モードは推測を直

接サポートし、エラー推測の場合はエラー推測されたデータの再処理が可能である。さらにＦＩＦＯはラスクスイッチを任意の時点でサポートする。

【００７４】

まず拡張ＦＩＦＯ動作モードはメモリの例で実行される。このメモリへは所定のデータ処理の枠内で読み出しアクセスされる。例としてのＦＩＦＯが図８に示されている。書込み回路の構造は通常の書込みポインタＷＲ＿ＰＴＲ，０８０１に相応し、従来技術では各書込みアクセス０８１０によりさらに移動する。読み出し回路は例えば通常のカウンタＲＤ＿ＰＴＲ、０８０２を有し、このカウンタは各読み出されたワードの読み出し信号０８１１に相応して計数し、相応にメモリ０８０３の読み出しアドレスを変更する。従来技術に対して新規なのは付加的回路ＤＯＮＥ＿ＰＴＲ０８０４であり、この付加的回路は読み出されたデータを文書化せずに、読み出し、正しく処理する。言い替えると、エラーが発生詩なかったデータを処理し、それらの結果を計算の最後に出力し、正しい計算終了は信号０８１２により指示される。可能な回路を以下に説明する。

【００７５】

（従来技術による）ＦＵＬＬフラグ０８０５はＦＩＦＯが一杯であり、それ以上のデータを記憶することができないことを指示する。このＦＵＬＬフラグはＤＯＮＥ＿ＰＴＲとＷＲ＿ＷＴＲとの比較０８０６により発生される。このことにより、生じ得るエラー推測によりバックアクセスが必要となるデータが上書きされないことが保証される。

【００７６】

ＥＭＰＴＹフラグ０８０７は通常の構造に相応して、ＲＤ＿ＰＲＴとＷＲ＿ＰＴＲの比較０８０８により発生する。エラー推測ＭＩＳＳ＿ＰＲＥＤＩＣＴ、０８０９が発生すると、読み出しポインタには値ＤＯＮＥ＿ＰＴＲ＋１がロードされる。これによりデータ処理が再度、エラー推測をトリガした値から開始される。

【００７７】

ＤＯＮＥ＿ＰＴＲの２つの可能な構成を例として詳細に説明する：

ａ）カウンタによる実現

ＤＯＮＥ＿ＰＴＲはカウンタとして実現される。カウンタは回路のリセット時、またはデータ処理の開始時にＲＤ＿ＰＴＲに等しくセットされる。到来する信号（ＤＯＮＥ）によりデータが必要であることが指示される。すなわちエラー推測なしで処理さえたことが指示される。これによりＤＯＮＥ＿ＰＲＴは、処理中の次のデータ語を指示するように変更される。

ｂ）減算器による実現

データ処理するパイプラインの長さが常に正確に既知であり、長さが一定であること（すなわち長さの異なるパイプラインへの分岐が生じない）が保証されれば、減算器を使用することができる。配属されたレジスタでは、メモリの端子から生じ得るエラー推測を識別するまでのパイプラインの長さが記憶される。このことによりデータ処理はエラー推測後に、差により計算することのできるデータ語において再開されなければならない。

【００７８】

書込み側では、コンフィギュレーションのデータ処理の結果を確保するために相応に構成されたメモリが必要であり、ここで書込みポインタに対するＤＯＮＥ＿ＰＲＴの機能が実現される。これによりすでに（エラー）計算された結果をデータ処理の新たな実行の際に上書きすることができる。言い替えると、書込みポインタと読み出しポインタの機能が図面に示されたアドレスに相応して交換される。

【００７９】

データの処理が他のソース（例えばオペレーティングシステムのタスクスイッチ）により中断されると、ＤＯＮＥ＿ＰＴＲが十分に確保され、データ処理は後の時点でＤＯＮＥ＿ＰＴＲ＋１から再開される。

【００８０】

入力／出力段に対するＦＩＦＯ、例えば０１０１，０１０３

データ経路および／またはグラフの種々異なるエッジの状態、ないしデータ処理の種々

の分岐を平衡化するため、コンフィギュレート可能なＦＩＦＯをＰＡＥの出力端または入力端で使用するのが有利である。ＦＩＦＯは調整可能な待機時間を有しており、種々異なるエッジ／分岐の遅延、すなわちデータが少なくとも平行で長さの異なるデータ経路を介して伝搬する伝搬時間を相互に調整することができる。

【００８１】

ＶＰＵ内では発生するデータまたは発生するトリガに基づいてパイプラインが停止することがあるから、ＦＩＦＯも同様に遅延を調整すると有利である。以下に説明するＦＩＦＯは２つの課題を解決する：

ＦＩＦＯ段は例えば図９に示されており、次のように構成することができる：レジスタ０９０１にはマルチプレクサ０９０２が後置接続されている。レジスタはデータ０９０３と、その正確な存在、すなわち所属のＲＤＹ０９０４を記憶する。レジスタへの書込みは、ＦＩＦＯの出力端０９２０の近くに示された隣接するＦＩＦＯ段が一杯であり０９０５、ＲＤＹ０９０４がデータに対して印加されるときに行われる。マルチプレクサは到来するデータ０９０３を、データがレジスタに書き込まれ、ＦＩＦＯの入力端０９２１の近くに示された隣接するＦＩＦＯ段がこれにより一杯になるまで出力端へ直接伝送する０９０６。データのＦＩＦＯ段への取り込みは入力確認（ＩＡＣＫ）０９０８により確認される。データのＦＩＦＯからの取り出しは、出力確認（ＯＡＣＫ）０９０９により確認される。ＯＡＣＫは同時にすべてのＦＩＦＯ段に到達し、データをＦＩＦＯでそれぞれ１段だけさらに書き込ませる。

【００８２】

個々のＦＩＦＯ段は任意の長さのＦＩＦＯを構成するために図９ａに示すようにカスケード接続することができる。このためにすべてのＩＡＣＫ出力端は例えばＯＲ機能０９１０相互に論理結合される。

【００８３】

この機能は図１０ａ，ｂの例で説明する：

データ語の挿入

新たなデータ語が個々のＦＩＦＯ段のマルチプレクサを介してレジスタに導かれる。第１の一杯になったＦＩＦＯ段１００１はその前の段１００２に、記憶されたＲＤＹに基づいて、これ以上データを取り込むことができないことを通知する。その前の段１００２はＲＤＹを記憶していない。しかし後続の段１００１の満杯状態を識別する。従ってこの段はデータとＲＤＹ１００３を記憶する；そして送信器へのＡＣＫにより記憶を確認する。ＦＩＦＯ段のマルチプレクサ１００４は、これがデータ経路を後続の段に接続せず、レジスタの内容を伝送するように切り替わる。

【００８４】

データ語の除去

ＡＣＫ１０１１が最後のＦＩＦＯ段に到来すると、各先行の段のデータがそれぞれ後続の段に伝送される１０１０。このことは、グローバル書込みクロックを各段に印加することにより行われる。マルチプレクサ全体はすでにレジスタ占有に相応して調整されているから、ＦＩＦＯのすべてのデータは１つのセルだけ下方に移動する。

【００８５】

データ語の除去と同時の挿入

グローバル書込みクロックが印加されると、最初に空きになった段にはデータ語が記憶されない。この段のマルチプレクサはデータをさらに後続の段に転送するから、最初に一杯になった段１０１２はデータを記憶する。そのデータは前に述べたように同じくロックで後続の段に記憶される。言い替えると：新たに書き込むべきデータは自動的に最初に空きになったＦＩＦＯ段１０１２に移動する。すなわちかつては最後に一杯であったＦＩＦＯ段であり、ＡＣＫの除去により空になったＦＩＦＯ段に移動する。

【００８６】

コンフィギュレート可能なパイプライン

所定の適用に対しては、図９の例に示されたＦＩＦＯ段にあるスイッチ０９３０によっ

てＦＩＦＯの個々のマルチプレクサを切り替え、基本的に相応するレジスタがスイッチオンされるようにすると有利である。これにより固定の待機時間ないし遅延時間をデータ伝送の際にスイッチを介して調整可能にコンフィギュレートすることができる。

【００８７】

データ流の統合（マージ）

データ流を統合するために全体で３つの方法が使用される。これらの方法は適用に応じてそれぞれ適切に使用される：

ａ）ローカルマージ

ｂ）ツリーマージ

ｃ）メモリマージ

ローカル統合（ローカルマージ）

もっとも単純な変形はローカルマージである。ここではすべてのデータ流が有利にはただ１つのポイントまたは比較的ローカルに統合され、場合により直ちに分離される。ローカルＳＮＤＣＮＴはマルチプレクサを介して、そのタイムスタンプがＳＮＤＣＮＴの値に相応し、従って瞬時に予期されるデータ語を正確に選択する。２つの手段を図７ａと図７ｂに基づいて詳細に説明する。

ａ）カウンタＳＮＤＣＮＴ、０７０６はデータパケットが到来するたびにさらに計数する。各データ経路には比較器が後置接続されている。比較器は計数器状態をデータ経路のタイムスタンプとそれぞれ比較する。計数器状態とタイプスタンプの値が一致すると、データパケットがマルチプレクサを介して後続のＰＡＥに転送される。

ｂ）解決手段ａ）を次のように拡張する。すなわち将来のデータ経路としてそれぞれアクティブなデータ経路を選択した後に、この経路に例えばＣＴコンフィギュレート可能なルックアップテーブル０７０１０を介して、目標データ経路を配属するのである。将来のデータ経路は、データと共に到来したタイムスタンプを方法ａ）に相応してＳＮＤＣＮＴ、０７１１と比較し０７１２、一致したデータ経路をアドレシングし０７１４、マルチプレクサ０７１３を介して選択することにより求められる。アドレス０７１４はルックアップテーブル０７１０によって目標データ経路アドレス０７１５に割り当てられる。この目標データ経路アドレスはデマルチプレクサ０７１６を介して目標経路を選択する。前記構造がバスノードにおいて同様に実現されていれば、ルックアップテーブル０７１０を介してデータ接続を、バスノードに配属されたＰＡＥ０７１８でも形成することができる。これはＰＡＥの入力端へのゲート機能（透過ゲート）０７１７を介して行われる。

【００８８】

特に高性能な回路例が図７ｃに示されている。ＰＡＥ０７２０は３つのデータ入力端Ａ，Ｂ，Ｃを有し、例えばＸＰＵ１２８ＥＳにある。バスシステム０７３３はコンフィギュレート可能および／またはマルチプレクス可能であり、各クロックサイクルで選択可能にデータ入力端へ切り替えることができる。各バスシステムはデータ、ハンドシェークおよび所属のタイムスタンプ０７２１を伝送する。ＰＡＥ０７２０の入力端ＡとＣは、データチャネルのタイムスタンプをＰＡＥに転送する０７２２，０７２３ために使用される。個々のタイムスタンプは後で説明するＳＩＭＤバスシステムにより束ねられる。束ねられたタイムスタンプはＰＡＥで再び分離され、各タイムスタンプはそれぞれ個別に０７２５，０７２６，０７２７、ＰＡＥで実現された／コンフィギュレートされたＳＮＤＣＮＴ０７２４と比較される０７２８。比較結果は、入力側マルチプレクサ０７３０を制御するために使用される。これにより入力側マルチプレクサは、正しいタイムスタンプを有するバスシステムを集合レール０７３１に接続する。集合レールは有利には入力端Ｂと接続されており、データをＰＡＥに０７１７，０７１８に相応して転送することを可能にする。出力側デマルチプレクサ０７３２はデータを種々異なるバスシステムへさらに転送する。この出力側マルチプレクサも同様に結果により制御される。ここで有利には結果の再配列はフレキシブルな翻訳により、例えばルックアップテーブル０７２９により行われる。その結果、結果を自由にデマルチプレクサ０７３２を介して、選択されたバスシステムに割り当てることができる。

【００８９】
ツリー状の統合（ツリーマージ）
多くのアプリケーションでは、複数のポイントでデータ流の一部をマージすることが所望される。そこからツリー状の構造が生じる。このとき、データ語を選択するのに中央での決定が行われず、決定が複数のノードに分散されているという問題が生じる。従ってすべてのノードにＳＮＤＣＮＴのそれぞれの値を伝送する必要がある。しかしクロック周波数が高い場合、このことは複数のレジスタ段により伝送の際に発生する待機時間を伴う。そのためこの解決手段は有意な性能を示さない。しかし能力を改善するために、各ノードでのローカル決定をＳＮＤＣＮＴの値に依存しないで行うようにする。例えば単純なアプローチでは、それぞれもっとも小さなタイムスタンプを有するデータ語をノードで選択する。しかしこのアプローチは、データ経路がノードで１つのクロックごとにデータ語を送出しない場合に問題となる。この場合、どのデータ経路を優先すべきかの判断ができない。

【００９０】
以下のアルゴリズムはこの特性を改善する：
ａ）各ノードは固有のＳＮＤＣＮＴカウンタ$SNDCNT_K$を有している。
ｂ）各ノードは入力データ経路（Ｐ0...Ｐn）を有するべきである。
ｃ）各ノードは複数の出力データ経路を有することができ、これらは変換方法、例えば上位のコンフィギュレートＣＴによりコンフィギュレート可能なルックアップテーブルによって入力データ経路に依存して選択される。
ｄ）ルートノードはメインＳＮＤＣＮＴを有し、このメインＳＮＤＣＮＴにすべての$SNDCNT_K$が同期化される。

【００９１】
正しいデータ経路を選択するために、次のアルゴリズムが使用される：
Ｉ．データがすべてのＰn入力側データ経路で行列していれば次のことが当てはまる：
ａ）最小のタイムスタンプＴｓを有するデータ経路Ｐ(Ts)が選択される。
ｂ）Ｋ：＝Ｔｓ＋１；ＳＮＤＣＮＴ＞Ｔｓ＋１が割り当てられ、$SNDCNT_K$：＝ＳＮＤＣＮＴが成り立つ。
ＩＩ．すべてのＰｎ入力側データ経路にデータが発生していなければ、次のことが当てはまる：
ａ）タイムスタンプＴｓ＝＝$SNDCNT_K$の場合だけデータ経路を選択する。
ｂ）$SNDCNT_K$：＝ＳＮＤＣＮＴ＋１
ｃ）ＳＮＤＣＮＴ：＝ＳＮＤＣＮＴ＋１
ＩＩＩ．１つのクロックで割り当てが実行されない場合には次のことが当てはまる：
ａ）$SNDCNT_K$：＝ＳＮＤＣＮＴ
ＩＶ．ルートノードはＳＮＤＣＮＴを有する。このＳＮＤＣＮＴは、有効なデータ語を選択するたびにさらに増分計数し、ツリーのルートにおいてデータ語の正しい順序を保証する必要であれば（１から３参照）他のすべてのノードがＳＮＤＣＮＴの値に同期される。このとき待機時間が発生し、この待機時間はＳＮＤＣＮＴの区間を$SNDCNT_K$の後にブリッジオーバするために挿入すべきレジスタの数に相応する。

【００９２】
図１１は可能なツリーを示す。このツリーは例えばＰＡＥ上でＶＰＵ　ＸＰＵ１２８ＥＳのツリーと同じように構成される。ルートノード１１０１は累積ＳＮＤＣＮＴを有し、その値は出力端Ｈ、１１０２で得られる。入力端ＡとＣのデータ語は記述の方法に相応して選択され、それぞれデータ語が正しい順序で出力端Ｌに導かれる。

【００９３】
次の階層段１１０３のＰＡＥと、さらに上位の各階層段１１０４，１１０５も相応して動作する。しかし以下の相違がある：累積$SNDCNT_K$はローカルであり、それぞれの値は転送されない。$SNDCNT_K$は、その値が入力端Ｂに印加されるＳＮＤＣＮＴによる記述の方法に相応して同期化される。

【００９４】
ＳＮＤＣＮＴはすべてのノード間で、とりわけ個々の階層段の間で、レジスタを介してパイプラインされる。
【００９５】
メモリによる統合（メモリマージ）
この方法では、データ流の統合のためにメモリが使用される。ここでタイムスタンプの各値にはメモリスペースが割り当てられる。データはそのタイムスタンプの値に相応してメモリにファイルされる。言い替えると、タイムスタンプは割り当てられたデータに対するメモリセルのアドレスとして使用される。これによりタイムスタンプに対して線形なデータ空間が発生する。すなわちこのデータ空間はタイムスタンプに相応してソートされている。データ空間が完全になって初めて、すなわちすべてのデータが記憶されて初めて、メモリはさらなる処理のためにイネーブルされるか、または線形に読出される。このことは次のようにして簡単に検出される。すなわち、幾つのデータがメモリに書き込まれたかを計数することにより検出される。メモリが有しているデータ登録と同じだけのデータが書き込まれれば、メモリは一杯である。
【００９６】
基本原理の実施の際には次の問題が発生する：メモリが隙間なしに満たされる前に、タイムスタンプのオーバフローが発生することがある。オーバフローは次のように定義される：タイムスタンプが有限線形数値空間（ＴＳＲ）からの数である。タイムスタンプの設定は厳しく単調に行われる。これにより数値空間内で各設定されたタイムスタンプが一義的である。タイムスタンプの設定の際に数値空間の終了に達すれば、設定はＴＳＲの始めから継続される。このことにより不連続な個所が発生する。そうするとタイムスタンプの設定は先行のものに対してもはや一義的ではなくなる。基本的にはこの不連続個所が処理の際に考慮されることを保証すべきである。従って数値空間ＴＳＲは、最悪の場合でも同じタイムスタンプがデータ処理内で２つ発生することによる多義性が発生しないような大きさに選択すべきである。言い替えるとＴＳＲは、後続の処理パイプラインおよび／またはメモリ内で発生し得る最悪の場合でも、同じタイムスタンプが処理パイプラインおよび／またはメモリ内に存在しないような大きさでなければならない。
【００９７】
タイムスタンプのオーバフローが発生すると、メモリはいずれの場合でもこれに応答できなければならない。オーバフロー後には、一部ではオーバフロー前のタイムスタンプを有するデータ（古いデータ）と、一部ではオーバフロー後のタイムスタンプを有するデータ（新しいデータ）とがメモリに含まれていることを前提にしなければならない。新しいデータを古いデータのメモリ個所に書き込んではならない。なぜなら古いデータが未だ読出されていないからである。従って複数の（少なくとの２つの）独立したメモリブロックを設ける必要がある。これにより古いデータと新た恣意データを別個に書き込むことができる。
【００９８】
メモリブロックを管理するために任意の方法を使用することができる。２つの手段を詳細に説明する：
ａ）所定のタイムスタンプの古いデータがこのタイムスタンプの新たなデータよりも前に到来することが常に保証されるならば、古いデータに対するメモリセルが未だ空きであるか否かが検査される。未だ空きであるなら古いデータが存在し、メモリセルは書き込まれ、空きがなければ新しいデータが存在し、メモリセルは新しいデータに対して書き込まれる。
ｂ）所定のタイムスタンプ値の古いデータがこのタイムスタンプ値の新しいデータより前に到来することが保証されなければ、このタイムスタンプに識別子が付され、新しいデータと古いデータとが区別される。この識別子は１または複数のビットとすることができる。タイムスタンプがオーバフローする場合、識別子は線形に変化する。このことにより古いデータと新しいデータに一義的なタイムスタンプが付される。識別子に相応して、デー

タは複数のメモリブロックの１つに割り当てられる。
【００９９】
従って有利には、その最大数値がタイムスタンプの最大数値よりも格段に小さい識別子が使用される。有利な関係は次のとおりである：
識別子（最大）＜（最大タイムスタンプ／２）
幅広のグラフをパーティショニングするためのメモリの使用
公知にように、大きなアルゴリズムをパーティショニングする必要がある。すなわち複数の部分アルゴリズムに分割する必要がある。これによりアルゴリズムがＶＰＵのＰＡＥの所定の要求および量に適合することができる。パーティショニングは一方では性能効率の点から、他方ではもちろんアルゴリズムの正当性を維持するように実行すべきである。ここで重要な側面は、それぞれのデータ経路のデータと状態（トリガ）の管理である。以下に、管理を改善し、簡素化するための方法を紹介する。
【０１００】
多くの場合、データ流グラフを１つのエッジでだけ切断する（図２ａ参照）ことは不可能である。なぜならグラフが例えば広すぎるか、または多数のエッジ１２０１，１２０２，１２０３が切断個所１２０４に存在するからである。
【０１０１】
パーティショニングは本発明によれば、図１２ｂに相応してすべてのエッジに沿って切断することにより実行される。第１のコンフィギュレーション１２１３の各エッジのデータは別個のメモリ１２１１に書き込まれる。
【０１０２】
データと共に（または別個に）データ処理の関連する状態情報全体もエッジを介して（例えば図１２ｂ）伝搬し、メモリに書き込まれることを述べておく。状態情報はＶＰＵテクノロジーに例えばトリガによって表わされる。
【０１０３】
再コンフィギュレーション後に、データおよび／または状態情報は後続のコンフィギュレーション１２１４によりメモリから読出され、このコンフィギュレーションによりさらに処理される。
【０１０４】
メモリは第１のコンフィギュレーションのデータ経路として（もっぱら書込み動作で）動作し、また後続のコンフィギュレーションのデータ送信器として（もっぱら読出し動作で）動作する。メモリ１２１１自体は２つのコンフィギュレーションの一部／リソースである。
【０１０５】
データを正しく処理するために、データがメモリに書き込まれた時間順序を正しく識別することが必要である。基本的にこのことは次のようにして保証される。すなわちデータ流を、
ａ）メモリへの書込みの際にソートするか、および／または
ｂ）メモリから読み出しの際にソートするか、および／または
ｃ）ソート順序をデータと共に記憶し、後続のデータ処理の際に使用するのである。
【０１０６】
このためにメモリには制御ユニットが配属され、制御ユニットはデータ１２１０のメモリ１２１１への書込みの際とデータのメモリ１２１２からの読出しの際に、データ順序およびデータ依存性の管理を行う。構成に応じて種々異なる管理形式および相応の制御メカニズムを使用することができる。
【０１０７】
２つの可能な相応する方法を図１３に基づき詳細に説明する。メモリは、ＰＡＥからのアレイ１３１０，１３２０に配属されている：
ａ）図１３ａで、メモリはそのアドレスを例えば共通のアドレス発生器により同期して発生する。言い替えると、書込みアドレス１３０１はサイクルごとにさらに計数されるが、

この計数はメモリに実際に有効なデータが記憶されるか否かには依存しない。これにより多くのメモリ１３０３，１３０４は同じタイムベース、ないし書込み／読出しアドレスを有する。個々のフラグＶＯＩＤ，１３０２はメモリ中の各データメモリ個所ごとに、有効データがメモリアドレスに書き込まれたか否かを指示する。フラグＶＯＩＤは、データに配属されたＲＤＹ１３０５により発生することができる。相応してメモリの読み出しの際にデータＲＤＹ１３０６はフラグＶＯＩＤから発生される。データを後続のコンフィギュレーションにより読出すために、データの書込みに相応して共通の読出しアドレス１３０７が発生され、この読出しアドレスはサイクルごとにさらに転送される。

ｂ）より効率的なのは、図１３ｂの実施例に示すように、タイムスタンプを各データ語にすでに述べた方法に相応して割り当てることである。データ１３１７は所属のタイムスタンプ１３１１と共にそれぞれのメモリ個所に記憶される。このことによりメモリ中に隙間が発生せず、メモリは効率的に活用される。各メモリは独立の書込みポインタ１３１３，１３１４をデータ書込みコンフィギュレーションのために有し、読出しポインタ１３１５，１３１６を後続のデータ読出しコンフィギュレーションのために有する。公知の方法（例えば図７ａまたは図１１）に相応して、データ語の読出しの際に時間的に正しいデータ語が、配属され、共に記憶されたタイムスタンプ１３１２に基づいて選択される。

【０１０８】

データのメモリへのソート／メモリからのソートは、種々のそれぞれ適切な方法に従って例えば次のようにして行われる。

ａ）メモリスペースをタイムスタンプにより割り当てる。

ｂ）タイムスタンプに従ってデータ流へソートする。

ｃ）各クロックをＶＡＬＩＤフラグと共に記憶する。

ｄ）タイムスタンプを記憶し、これをメモリの読み出しの際に後続のアルゴリズムへさらに転送する。

【０１０９】

アプリケーションに独立して、複数の（またはすべての）データ経路を記憶の前でも本発明のマージ方法を介して統合することができる。これを実行するか否かは実質的に、使用されるリソースに依存する。少数のメモリしか使用できなければ、記憶前での統合が必要であり、所望される。少数のＰＡＥしか使用できなければ、有利にはそれ以上のＰＡＥが統合のために使用されない。

【０１１０】

タイムスタンプによる端末インタフェース（ＩＯ）の拡張

以下に、ＩＯチャネルに端末構成素子および／または外部メモリタイムについてのイムスタンプを割り当てる方法を説明する。割り当ては種々の目的を満たすことができ、例えばデータ流を送信器と受信器との間で正しくソートするため、および／またはデータ流のソースおよび／または宛先を一義的に選択するために行われる。

【０１１１】

次の実施例を、インタフェースセルの例で説明する。ここではＶＰＵ内部バスの収束、および種々のＶＰＵ間またはＶＰＵと端末（ＩＯ）との間のデータ交換方法を記述する。

【０１１２】

この方法の欠点は、データソースが受信器においてもはや同定不能であり、正しい時間的順序も保証されないことである。以下の新しい方法はこの問題を解決し、それぞれ適用固有の複数の方法を使用され、場合により組み合わされる。

ａ）データソースの同定

図１４は例として、２つのＶＰＵ１４１０，１４２０のコンフィギュレート可能なエレメントＰＡＥからのアレイＰＡ、１４０８間の同定を示す。アービタ１４０１がデータ送信側構成素子ＶＰＵ、１４１０において可能なデータソース１４０５の１つを選択し、これをマルチプレクサ１４０２を介してＩＯに接続する。データソース１４０３のアドレスはデータ１４０４と共にＩＯに送信される。データ受信側構成素子ＶＰＵ、１４１１は、データソースのアドレス１４０３に相応して相応の受信器１４０６をデマルチプレクサ１

４０７を介して選択する。有利には変換方法、例えばルックアップテーブルを使用して、伝送されるアドレス１４０３と受信器１４０６とをフレキシブルに割り当てることができる。ルックアップテーブルは上位のコンフィギュレーションユニットＣＴによりコンフィギュレートすることができる。マルチプレクサ１４０２に前置接続されたおよび／またはデマルチプレクサ１４０７に後置接続されたインタフェース構成群を、バスシステムのコンフィギュレート可能な接続のために使用することができることを述べておく。

ｂ）時間的順序の維持

ｂ１）最も簡単な方法は、タイムスタンプをＩＯに送信し、タイムスタンプを受理した受信器に評価を任せることである。

ｂ２）別のバージョンでは、タイムスタンプがアービタによりデコードされる。アービタは正しいタイムスタンプを有する送信器だけを選択し、ＩＯに送信する。受信器はデータを正しい順序で受け取る。

【０１１３】

ａ）およびｂ）による方法は、それぞれのアプリケーションの要求に相応して共通して、または個別に適用することができる。

【０１１４】

さらにこの方法はチャネル番号の設定および同定により拡張することができる。チャネル番号は所定の送信領域を表わす。例えばチャネル番号は、構成素子内のバス、構成素子、構成群の記述のような複数の同定から成ることができる。このことにより多数のＰＡＥおよび／または多数の構成素子の合同を使用する場合には簡単な同定が得られる。

【０１１５】

有利にはチャネル番号を使用する際にそれぞれ個々のデータ語を伝送せずに、複数のデータ語をデータパケットにまとめ、チャネル番号の記載の下で伝送する。個々のデータ語をまとめることは、例えば適切なメモリを使用して行うことができる。

【０１１６】

伝送されたアドレスおよび／またはタイムスタンプは有利に識別子または識別子の一部としてバスシステムで使用できることを述べておく。

【０１１７】

シークエンサの構造

タイムスタンプまたは同等の方法を使用することにより、ＰＡＥの群からの簡単なシークエンサ構造が可能となる。回路のバスおよび基本機能はコンフィギュレートされ、詳細機能およびデータアドレスはオペコードによって伝搬時間にフレキシブルに調整される。

【０１１８】

複数のこのシークエンサは同時にＰＡ（ＰＡＥからのアレイ）内で構成され、駆動される。

【０１１９】

ＶＰＵ内のシークエンサはアルゴリズムに相応して形成され、複数に分割された本発明のステップですでに与えられている。ここでは複数のＰＡＥからのシーケンサの構造が記載されている。このことは以下の説明に対する基礎的例として用いる。

【０１２０】

以下のシーケンサの構成は自由に適合される：

・ＩＯ／メモリの形式および量
・割込みの形式および量（例えばトリガを介して）
・命令セット
・レジスタの数および形式

簡単なシーケンサは例えば次の例から形成される：

１．算術機能および論理機能を実行するためのＡＬＵ
２．データを擬似的にレジスタセットとして記憶するためのメモリ
３．プログラムに対するコードソースとしてのメモリ

場合によりシーケンサはＩＯエレメントだけ拡張される。とりわけさらなるＰＡＥがデ

ータソースまたはデータ受信器として接続される。
【０１２１】
使用されるコードソースに応じて、ＰＡＥのオペコードをデータバスを介して直接セットすることができ、データソース／データ宛先を指示することができる。
【０１２２】
データソース／データ宛先のアドレスは例えばタイムスタンプ方法で伝送される。さらにバスをオペコードの伝送のために使用することができる。
【０１２３】
例としてのタイムスタンピングが図１４に示されており、シーケンサはプログラムを記憶するためのＲＡＭ、データ（ＡＬＵ）を計算するためのＰＡＥ１５０２、プログラムポインタ１５０３を計算するためのＰＡＥ、レジスタセットとしてのメモリ、および外部機器１５０５に対するＩＯから成る。
【０１２４】
配線によって２つのバスシステム、すなわちＡＬＵ　ＩＢＵＳ１５０６への入力バス、およびＡＬＵ　ＯＢＵＳ１５０７からの出力バスが発生する。バスにはそれぞれ４ビット幅のタイムスタンプが配属されており、このタイムスタンプがソースＩＢＵＳ－ＡＤＲ１５０８ないし宛先ＯＢＵＳ－ＡＤＲ１５０９をアドレシングする。
【０１２５】
１５０４からプログラムポインタ１５１０が１５０１に供給される。１５０１はオペコードをフィードバックする。オペコードは、ＡＬＵ１５１２およびプログラムポインタ１５１３に対するコマンド中にあり、データアドレス１５０８，１５０９を分割する。バスを分割するために次のＳＩＭＤ方法およびバスシステムを使用することができる。
【０１２６】
１５０２はアキュムレータマシンとして構成されており、例えば次の機能をサポートする：
ld＜reg＞　　レジスタからアキュムレータ１５２０にロード
add_sub＜reg＞　　加算／減算レジスタからアキュムレータへ
sl_sr　　アキュムレータシフト
rl_rr　　アキュムレータ回転
st＜reg＞　　アキュムレータをレジスタに書込み
コマンドに対しては３ビットが必要である。第４のビットがオペレーションの形式を指示する：加算または減算、右または左シフト。１５０２はＡＬＵ状態キャリーをトリガポート０に、ゼロをトリガポート１に送出する。
【０１２７】
＜reg＞は次のように符号化される：
０..７　　１５０４のデータレジスタ
８　　入力レジスタ１５２１　プログラムポインタ計算
９　　ＩＯデータ
１０　　ＩＯアドレス
アドレスに対しては４ビットが必要である。
【０１２８】
１５０３は次のオペレーションをプログラムポインタを介してサポートする：
jmp　　入力レジスタ２３２１のアドレスへジャンプ
jt0　　トリガ０がセットされているとき入力レジスタのアドレスへジャンプ
jt1　　トリガ１がセットされているとき入力レジスタのアドレスへジャンプ
jt2　　トリガ２がセットされているときに入力レジスタのアドレスへジャンプ
jmpr　　入力レジツタのアドレスプラスＰＰにジャンプ
コマンドに対しては３ビットが必要である。第４のビットはオペレーションの形式を維持する：加算または減算。
【０１２９】

オペコード１５１１は3つの群でそれぞれ４ビットごとに分割される：（１５０８，１５０９）、１５１２，１５１３，１５０８と１５０９は所定のインストラクションセットでは同じとすることができる。１５１２，１５１３は例えばＰＡＥのＣレジスタに供給され、ＰＡＥ内で命令としてデコードされる。

【０１３０】

シーケンサは複合構造に形成することができる。例えば＜reg＞＝１１，１２，１３，１４，１５によりさらなるデータソースをアドレシング可能であり、このデータソースは他のＰＡＥから発することもできる。同様にさらなるデータ経路がアドレシングされる。データソースおよびデータ受信器は任意であり、ＰＡＥとすることができる。

【０１３１】

図示された回路はオペコード１５１１の１２ビットしか必要としないことを述べておく。従って３２ビットアーキテクチャでは、２０ビットが基本回路の拡張のためのオプションとして使用される。

【０１３２】

ＳＩＭＤ計算機構とＳＩＭＤバスシステム

アルゴリズムを処理するために再コンフィギュレーション可能な技術を使用する場合、重大なパラドクスが発生する：一方ではできるだけ高い計算能力を得るために複合ＡＬＵが必要であり、その際に再コンフィギュレーションに対するコストは最小でなければならない。他方ではＡＬＵはできるだけ簡単にビットレベルでの効率的な処理を可能にしなければならない。さらに再コンフィギュレーションとデータ管理は、これが効率的かつ簡単にプログラミングされるよう、インテリジェントで高速に実行されなければならない。

【０１３３】

これまでの技術は、

ａ）再コンフィギュレーションサポート（ＦＰＧＡ）の少ない小さなＡＬＵを使用し、別途レベルでは効率であるか、

ｂ）再コンフィギュレーションサポートの少ない大きなＡＬＵ（Ｃａｍｅｌｅｏｎ）を使用するか、

ｃ）再コンフィギュレーションサポートとデータ管理（ＶＰＵ）を行う大きなＡＬＵと小さなＡＬＵを混合して使用していた。

【０１３４】

ＶＰＵ技術は高性能な技術であるから、これに基づく最適な方法を見出すべきである。この方法は同様に他のアーキテクチャに対しても使用できることを述べておく。

【０１３５】

再コンフィギュレーションを効率的に制御するための面積コストはＰＡＥ当たりで約１００００から４００００ゲート量であり比較的高い。このゲート量以下では簡単な伝搬制御しか実現されず、ＶＰＵのプログラミング性が非常に制限され、汎用プロセッサとしての使用が除外される。特別に高速なコンフィギュレーションを目的とするならば、付加的なメモリを設けなければならないが、これにより所要のゲート量はさらに上昇する。

【０１３６】

再コンフィギュレーションコストと計算能力との間で適切な関係を得るためには、大きなＡＬＵ（多数の機能性および／または大きなビット幅）を必然的に使用しなければならない。しかしＡＬＵが過度に大きくなると、チップ当たりの使用可能な平行計算能力が低下する。ＡＬＵが過度に小さいと（例えば４ビット）、面倒な機能（例えば３２ビット乗算）をコンフィギュレートするためのコストが過度に大きくなる。とりわけ配線コストが商品的に意味のない領域まで上昇する。

【０１３７】

１１．１　ＳＩＭＤ計算機構の使用

小さなビット幅の処理と、配線コストと、面倒な機能のコンフィギュレーションとの間で理想的な関係を得るために、ＳＩＭＤ計算機構の使用が提案される。ここでは幅ｍの計算機構が分割され、幅がｂ＝ｍ／ｎのｎ個のブロックが発生する。コンフィギュレ

ーションにより各計算機構は、計算機構を分割しないか、または複数のブロックに分割し、それぞれ同じ幅または異なる幅を有するようにするかが設定される。言い替えると、計算機構は、１つの計算機構内で異なるワード幅が同時にコンフィギュレートされるように（例えば３２ｎビット幅を１×１６ビット、１×８ビットそして２×４ビットに）分割することができる。データは次のようにＰＡＥ間で伝送される。すなわち分割されたデータ語（ＳＩＭＤ語）がビット幅ｍのデータ語にまとめられ、パケットとしてネットワークを介して伝送される。ネットワークは常に完全なパケットを伝送する。すなわちすべてのデータ語は１パケット内で有効であり、公知のハンドシェーク法に従って伝送される。

【０１３８】

１１．１．１　ＳＩＭＤ語の再ソート

ＳＩＭＤ計算機構を効率的に使用するためには、ＳＩＭＤ語を相互にバス内または種々異なるバス間でフレキシブルかつ効率的に再ソートする必要がある。

【０１３９】

図５ないし図７ｂ、ｃのバススイッチは次のように変更することができる。すなわち個々のＳＩＭＤ語のフレキシブルなネットワーキングが可能であるように変更することができる。このためにマルチプレクサは計算機構に相応して、コンフィギュレーションにより分割を定めることができるように構成される。言い替えると、ビット幅ｍのマルチプレクサをバスごとに使用するのではなく、ビット幅ｂ＝ｍ／ｎのｎ個の個々のマルチプレクサを使用するのである。これによりデータバスをｂビット幅に対してコンフィギュレートすることができる。バスのマトリクス構造（図５）によって、データを簡単に再ソートすることができる。これは図１６ｃに示されている。第１のＰＡＥはデータを第２のバス１６０１，１６０２を介して送信する。これらのバスはそれぞれ４つの部分バスに分割されている。バスシステム１６０３は個々の部分バスを、付加的にバスに存在する部分バスと接続する。第２のＰＡＥは、種々異なってソートされた部分バスをその２つの入力バス１６０４，１６０５で受け取る。

【０１４０】

例えば２重ＳＩＭＤ計算機構１６１４，１６１５を有する２つのＰＡＥ間でのバスのハンドシェークは図１６ａで論理的に結合され、新たに配列されたバス１６１１に対する共通のハンドシェーク１６１０が元のバスのハンドシェークから発生される。例えば新たにソートされたバスに対するＲＤＹは、このバスに対してデータを送出するバスのすべてのＲＤＹを論理ＡＮＤ結合することにより発生される。同様にデータを送出するバスのＡＣＫは、データをさらに処理するすべてのバスのＡＣＫをＡＮＤ結合することにより発生できる。

【０１４１】

共通のハンドシェークは、ＰＡＥ１６１２を管理するための制御ユニット１６１３を制御する。バス１６１１はＰＡＥを内部で２つの計算機構１６１４，１６１５に分割する。

【０１４２】

第１の変形実施例では、ハンドシェークの結合が各バスノード内で実行される。このことにより、ビット幅ｂのｎ個の部分バスから成るビット幅ｍのバスシステムにハンドシェークプロトコルを割り当てることが可能になる。

【０１４３】

別の有利な実施形態では、バスシステム全体が幅ｂに構成され、この幅ｂはＳＩＭＤ語の最小実現可能入／出力データ幅ｂに相当する。ＰＡＥデータ経路（ｍ）の幅に相応して、入／出力バスは幅ｂのｍ／ｂ＝ｎ個の部分バスからなる。例えば３つの３２ビット入力バスと、２つの３２ビット出力バスを備えるＰＡＥは、８の最小ＳＩＭＤ語幅の場合に実際には３×４の８ビット入力バスと、２×４の８ビット出力バスを有する。

【０１４４】

部分バスの各々には、ハンドシェーク信号と制御信号全体が配属される。

【０１４５】

ＰＡＥの出力端は、ｎ個の部分バス全体に対して同じ制御信号を送信する。到来するす

べての部分バスの受領信号は相互に論理的にＡＮＤ機能により結合される。バスシステムは各部分バスを自由に接続することができ、これはルートに依存しない。バスシステムとバスノードは、個々のバスのハンドシェーク信号をそれらのルーティング、構成およびソートに依存しないで処理し、結合する。ＰＡＥにデータが到来する場合、ｎ個の部分バス全体の制御信号が相互に結合され、一般的に有効な制御信号がデータ経路に対するバス制御信号として発生される。

【０１４６】

例えば定義に従い「依存性の」動作形式でRdyHold段を各個々のデータ経路に対して使用することができ、全RdyHold段が発生したデータをシグナリングして初めて、これらはＰＡＥにより引き取られる。

【０１４７】

定義による「独立性の」動作形式では、各部分バスのデータが個別にＰＡＥの入力レジスタに書き込まれ、受領される。これにより部分バスは直ちに次のデータ伝送に対して空きとなる。すべての部分バスのすべての所要データが入力レジスタに存在することは、ＰＡＥ内において、各部分バスに対し入力レジスタに記憶されたＲＤＹ信号を適切に論理結合することにより検知される。これに基づきおＰＡＥはデータ処理を開始する。

【０１４８】

ここから得られるこの方法の利点は、ＰＡＥのＳＩＭＤ特性が使用されるバスシステムに何ら影響を及ぼさないことである。図１６ｂに示したように、幅ｂの小さな複数のバス１６２０と所属のハンドシェーク１６２１が必要なだけである。配線回路自体は変化しない。ＰＡＥは制御線路をローカルに結合し、管理する。このことにより、制御線路を管理および／または結合するためのバスシステムでの付加的ハードウエアコストが省略される。

【図面の簡単な説明】

【０１４９】

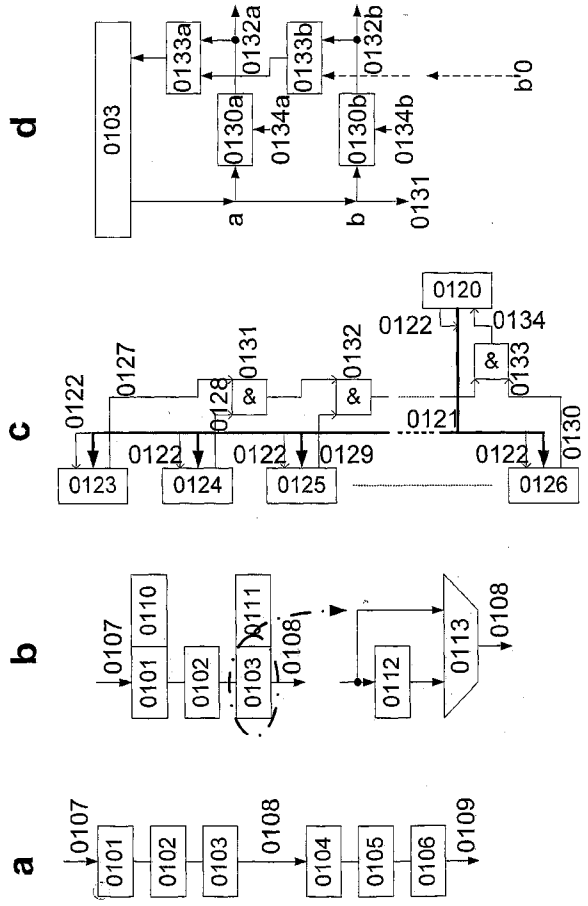
【図１】本発明の実施例の模式図である。
【図２】本発明の実施例の模式図である。
【図３】本発明の実施例の模式図である。
【図４】本発明の実施例の模式図である。
【図５】本発明の実施例の模式図である。
【図６】本発明の実施例の模式図である。
【図７ａ】本発明の実施例の模式図である。
【図７ｂ】本発明の実施例の模式図である。
【図７ｃ】本発明の実施例の模式図である。
【図８】本発明の実施例の模式図である。
【図９】本発明の実施例の模式図である。
【図１０】本発明の実施例の模式図である。
【図１１】本発明の実施例の模式図である。
【図１２】本発明の実施例の模式図である。
【図１３】本発明の実施例の模式図である。
【図１４】本発明の実施例の模式図である。
【図１５】本発明の実施例の模式図である。
【図１６】本発明の実施例の模式図である。

【符号の説明】

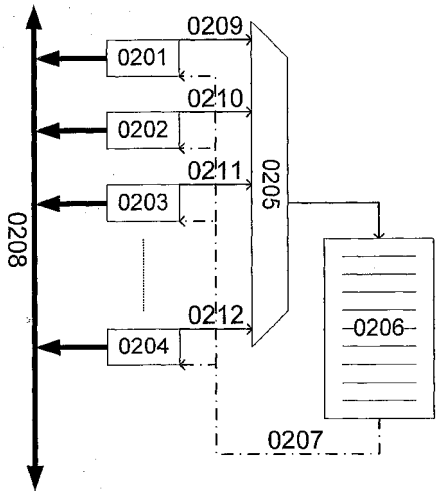
【０１５０】

０１０７，０１０８，０１０９　バスシステム、　０１０１，０１０４　レジスタ、０１０２，０１０５　データ処理論理回路、　０１０３，０１０６　出力段

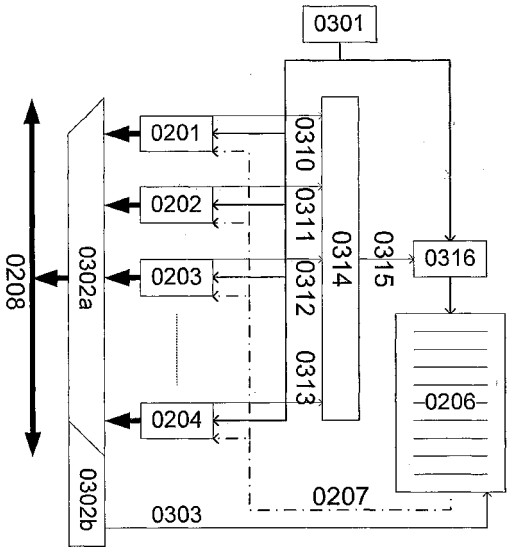
【図１】


d
0103
0133a
0132a
0133b
0132b
b0
0130a
0134a
0130b
0134b
a
b
0131
c
0120
0122
0134
0127
0131
0132
0133
0128
&
&
0121
0122
0122
0129
0122
0130
0123
0124
0125
0126
b
0107
0110
0101
0102
0111
0103
0108
0112
0113
0108
a
0107
0101
0102
0103
0108
0104
0105
0106
0109

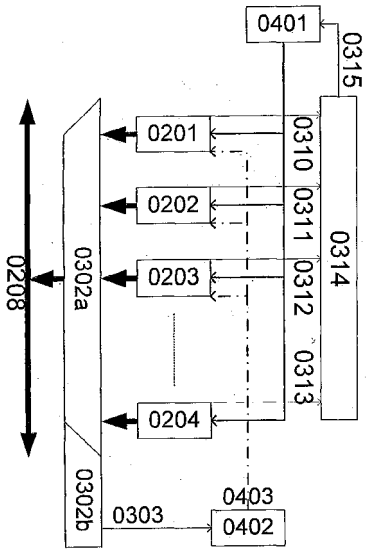
【図２】


0209
0201
0210
0202
0211
0203
0205
0208
0212
0204
0206
0207

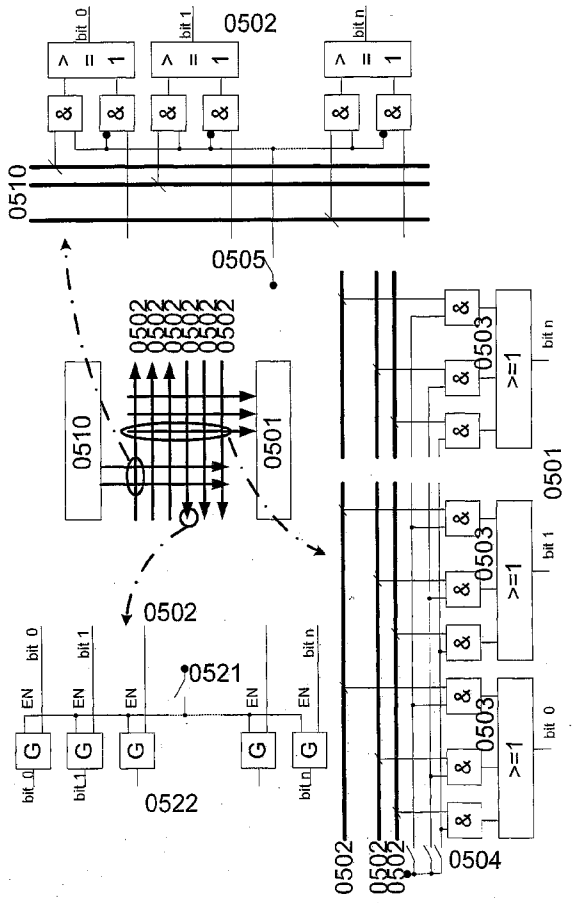
【図３】


0301
0201
0310
0202
0311
0203
0312
0314
0315
0316
0208
0302a
0313
0204
0206
0207
0303
0302b

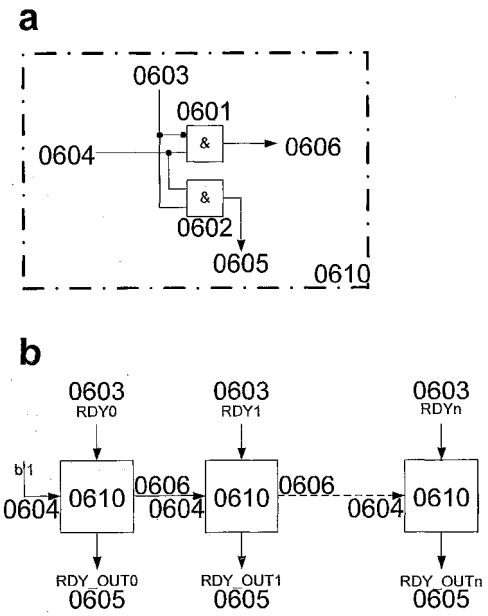
【図４】


0401
0315
0201
0310
0202
0311
0203
0312
0314
0208
0302a
0313
0204
0403
0303
0402
0302b

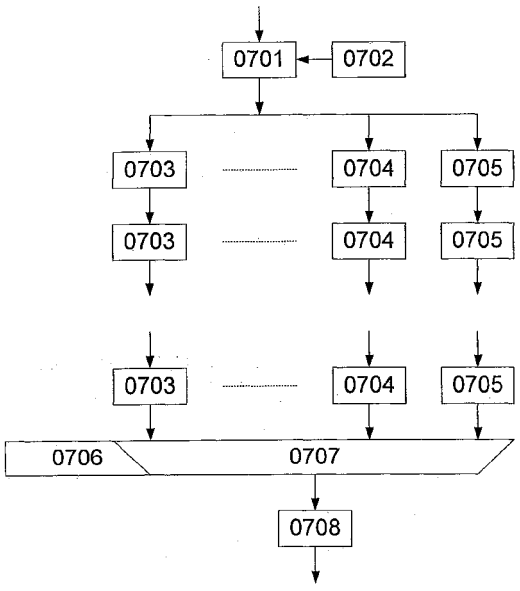
【図５】


0502
0510
0505
0501
0503
0504
0521
0522
bit 0
bit 1
bit n
EN
G

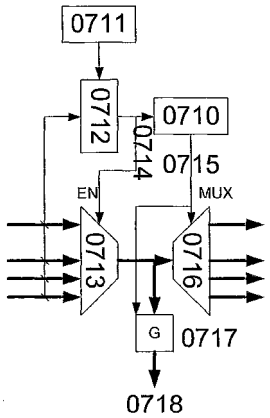
【図６】


a
0603
0601
0604
0606
0602
0605
0610
b
RDY0
RDY1
RDYn
bit1
RDY_OUT0
RDY_OUT1
RDY_OUTn

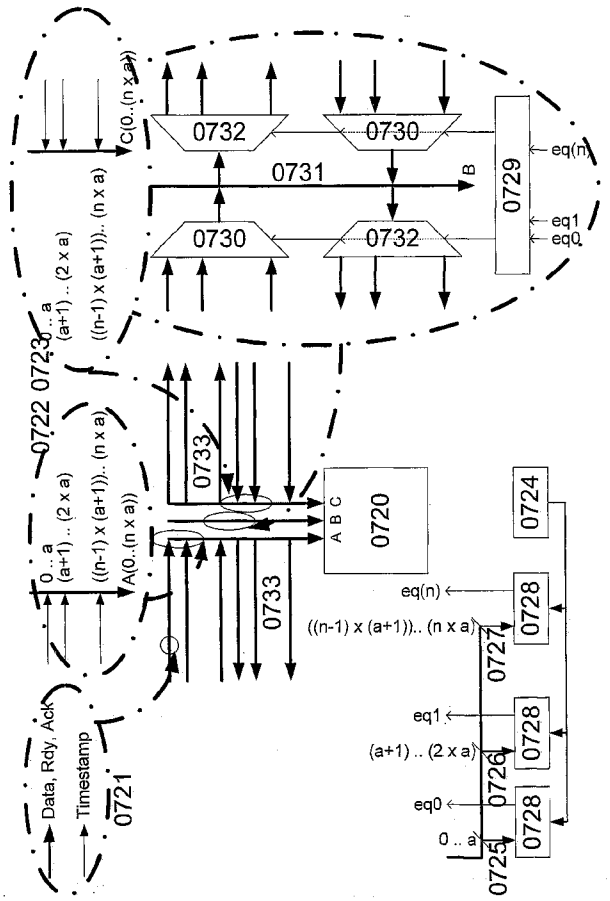
【図７ａ】


0701
0702
0703
0704
0705
0706
0707
0708

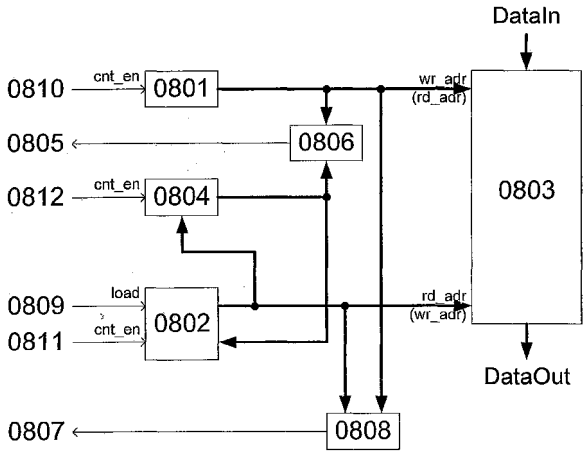
【図７ｂ】


0711
0712
0710
0714
0715
EN
MUX
0713
0716
G
0717
0718

【図７ｃ】



【図８】



【図９】

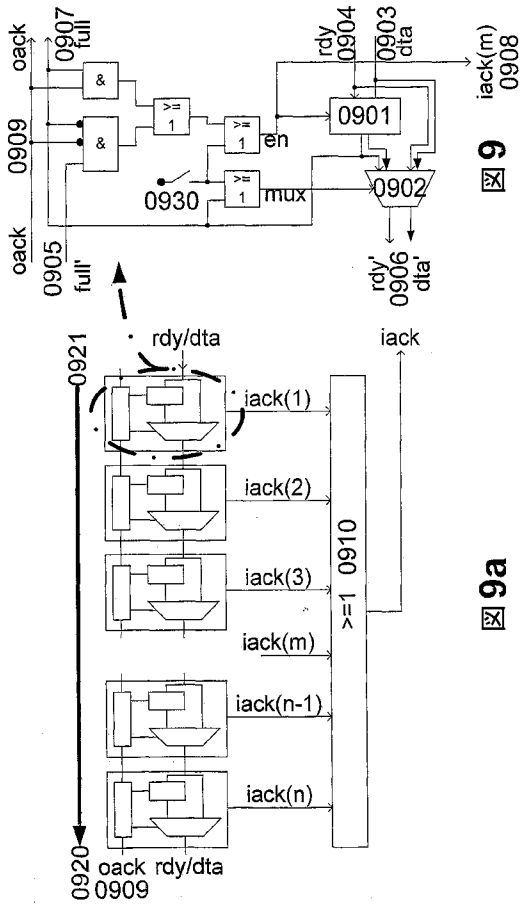
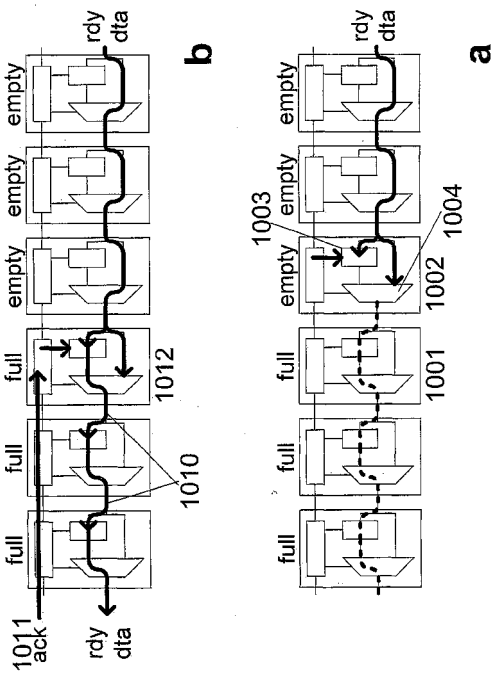


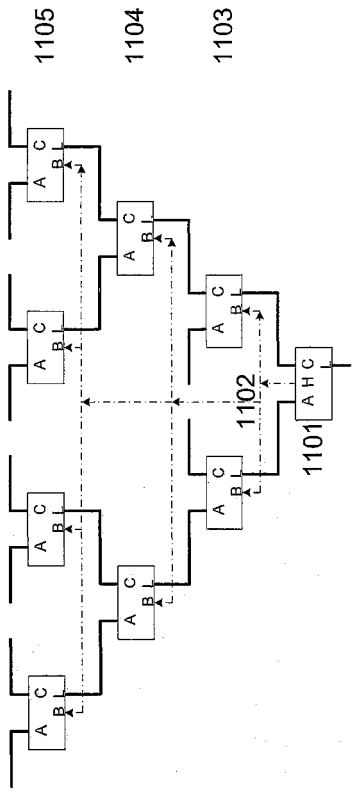
図９

図９ａ

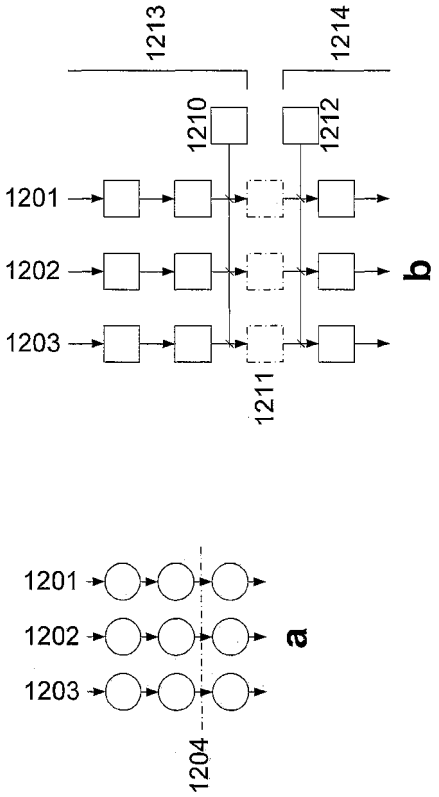
【図１０】



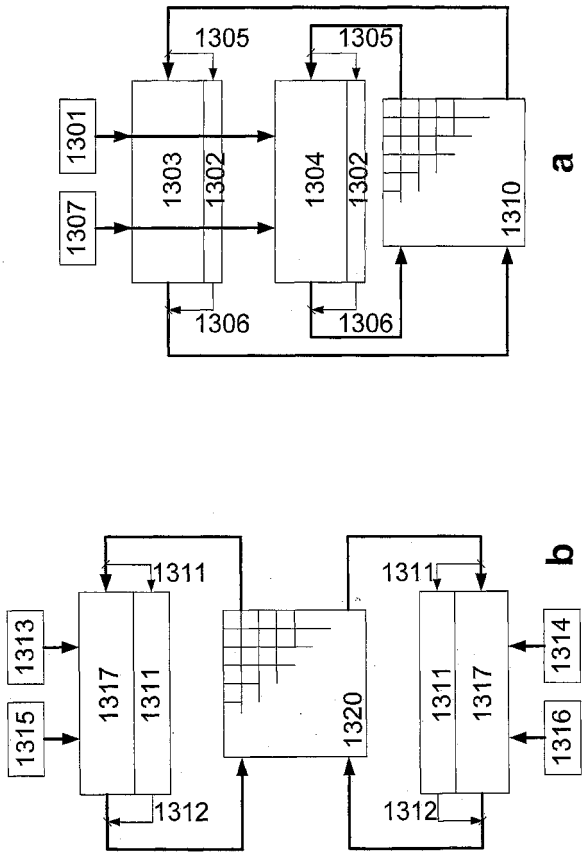
【図１１】



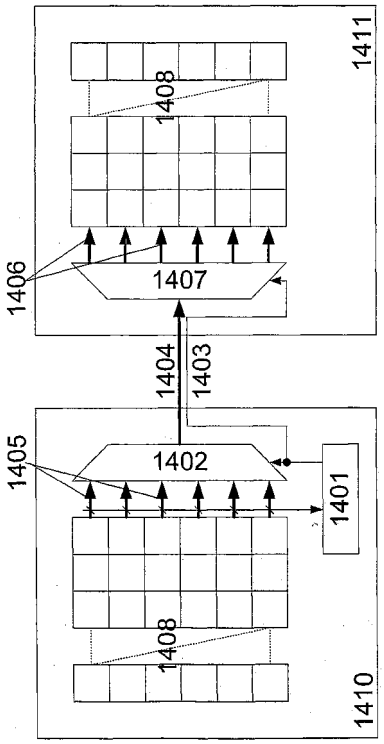
【図１２】



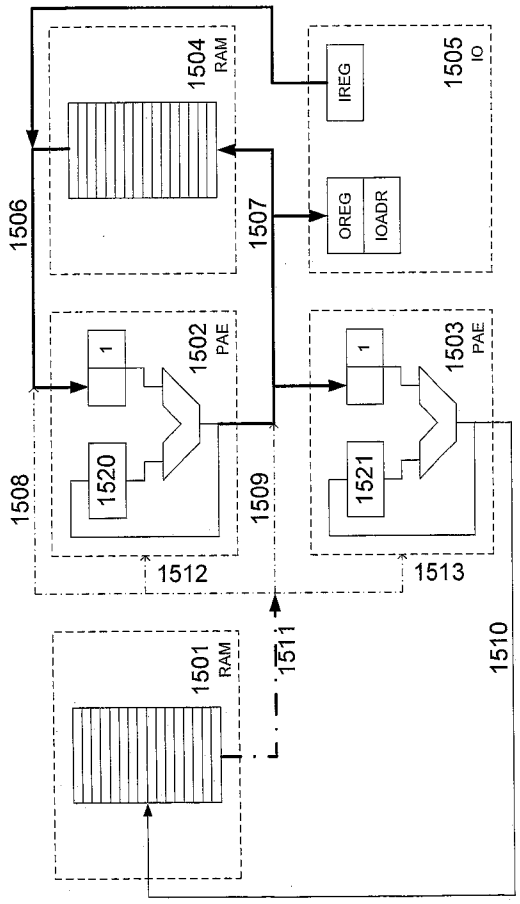
【図１３】



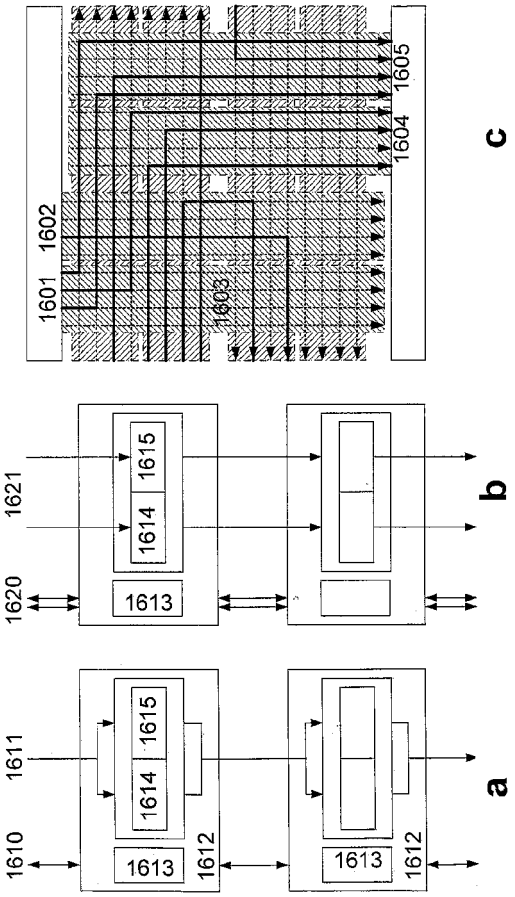
【図１４】



【図１５】


1504
RAM
1505
IO
IREG
OREG
IOADR
1506
1507
1502
PAE
1503
PAE
1
1520
1521
1508
1509
1512
1513
1511
1510
1501
RAM

【図１６】


1601
1602
1603
1604
1605
c
1621
1620
1615
1614
1613
b
1611
1610
1612
a

フロントページの続き

(31)優先権主張番号 101 29 237.6
(32)優先日 平成13年6月20日(2001.6.20)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 01115021.6
(32)優先日 平成13年6月20日(2001.6.20)
(33)優先権主張国 欧州特許庁(EP)
(31)優先権主張番号 101 35 210.7
(32)優先日 平成13年7月24日(2001.7.24)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 35 211.5
(32)優先日 平成13年7月24日(2001.7.24)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 PCT/EP01/08534
(32)優先日 平成13年7月24日(2001.7.24)
(33)優先権主張国 欧州特許庁(EP)
(31)優先権主張番号 101 39 170.6
(32)優先日 平成13年8月16日(2001.8.16)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 42 231.8
(32)優先日 平成13年8月29日(2001.8.29)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 42 894.4
(32)優先日 平成13年9月3日(2001.9.3)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 42 903.7
(32)優先日 平成13年9月3日(2001.9.3)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 42 904.5
(32)優先日 平成13年9月3日(2001.9.3)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 60/317,876
(32)優先日 平成13年9月7日(2001.9.7)
(33)優先権主張国 米国(US)
(31)優先権主張番号 101 44 732.9
(32)優先日 平成13年9月11日(2001.9.11)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 44 733.7
(32)優先日 平成13年9月11日(2001.9.11)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 45 792.8
(32)優先日 平成13年9月17日(2001.9.17)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 45 795.2
(32)優先日 平成13年9月17日(2001.9.17)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 46 132.1
(32)優先日 平成13年9月19日(2001.9.19)
(33)優先権主張国 ドイツ(DE)

(31)優先権主張番号 09/967,847
(32)優先日 平成13年9月28日(2001.9.28)
(33)優先権主張国 米国(US)
(31)優先権主張番号 PCT/EP01/11299
(32)優先日 平成13年9月30日(2001.9.30)
(33)優先権主張国 欧州特許庁(EP)
(31)優先権主張番号 PCT/EP01/11593
(32)優先日 平成13年10月8日(2001.10.8)
(33)優先権主張国 欧州特許庁(EP)
(31)優先権主張番号 101 54 259.3
(32)優先日 平成13年11月5日(2001.11.5)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 101 54 260.7
(32)優先日 平成13年11月5日(2001.11.5)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 01129923.7
(32)優先日 平成13年12月14日(2001.12.14)
(33)優先権主張国 欧州特許庁(EP)
(31)優先権主張番号 02001331.4
(32)優先日 平成14年1月18日(2002.1.18)
(33)優先権主張国 欧州特許庁(EP)
(31)優先権主張番号 102 02 044.2
(32)優先日 平成14年1月19日(2002.1.19)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 02 175.9
(32)優先日 平成14年1月20日(2002.1.20)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 06 653.1
(32)優先日 平成14年2月15日(2002.2.15)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 06 856.9
(32)優先日 平成14年2月18日(2002.2.18)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 06 857.7
(32)優先日 平成14年2月18日(2002.2.18)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 07 225.6
(32)優先日 平成14年2月21日(2002.2.21)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 07 224.8
(32)優先日 平成14年2月21日(2002.2.21)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 07 226.4
(32)優先日 平成14年2月21日(2002.2.21)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 08 435.1
(32)優先日 平成14年2月27日(2002.2.27)
(33)優先権主張国 ドイツ(DE)
(31)優先権主張番号 102 08 434.3
(32)優先日 平成14年2月27日(2002.2.27)

(33)優先権主張国　ドイツ(DE)

(74)代理人　100099483
弁理士　久野　琢也
(74)代理人　100110593
弁理士　杉本　博司
(74)代理人　100128679
弁理士　星　公弘
(74)代理人　100135633
弁理士　二宮　浩康
(74)代理人　100114890
弁理士　アインゼル・フェリックス＝ラインハルト
(72)発明者　マルティン　フォアバッハ
ドイツ連邦共和国　ミュンヘン　ゴットハルトシュトラーセ　１１７アー
(72)発明者　フォルカー　バウムガルテ
ドイツ連邦共和国　ミュンヘン　バルバロッサプラッツ　１４
(72)発明者　アルミン　ニュッケル
ドイツ連邦共和国　ノイポツ　ドロッセルヴェーク　４
(72)発明者　フランク　マイ
ドイツ連邦共和国　ミュンヘン　アン　デア　トゥーフブライヒェ　１２

Ｆターム(参考) 5B077 DD11