(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3587095号
(P3587095)
(45)発行日　平成16年11月10日(2004.11.10)　(24)登録日　平成16年8月20日(2004.8.20)

(51)Int.Cl.[7]　F I
H03K 19/173　H03K 19/173 101
H01L 21/82　H01L 21/82 A
H01L 21/82 C

請求項の数 7　(全 21 頁)

(21)出願番号　特願平11-238384
(22)出願日　平成11年8月25日(1999.8.25)
(65)公開番号　特開2001-68993(P2001-68993A)
(43)公開日　平成13年3月16日(2001.3.16)
審査請求日　平成15年10月27日(2003.10.27)

(73)特許権者　000005496
富士ゼロックス株式会社
東京都港区赤坂二丁目１７番２２号
(74)代理人　100101948
弁理士　柳澤　正夫
(72)発明者　佐藤　嘉秀
神奈川県足柄上郡中井町境４３０　グリーンテクなかい　富士ゼロックス株式会社内

審査官　彦田　克文

(56)参考文献　特開平０８－０４６５０９（ＪＰ，Ａ）
特開平１０－３３５４６２（ＪＰ，Ａ）
特開平０８－３１６３２９（ＪＰ，Ａ）

最終頁に続く

(54)【発明の名称】情報処理装置

(57)【特許請求の範囲】

【請求項１】

回路情報を変更することによって機能を随時変更し再構成することが可能なプログラマブル論理回路を備えた情報処理装置において、予め使用する回路情報と順番を保持する回路情報保持手段と、与えられた処理データから前記プログラマブル論理回路の再構成に使用する最初の回路情報を特定するとともに特定された回路情報から前記回路情報保持手段に保持されている順番に前記回路情報により前記プログラマブル論理回路を再構成して処理回路を構成し該処理回路に前記処理データの処理を行わせるインタフェース手段を有していることを特徴とする情報処理装置。

【請求項２】

前記処理データのヘッダ部には、あらかじめ規定された処理ステップの順番、前記順番に対応した再構成する回路情報を格納した前記回路情報保持手段のアドレス情報、前記順番に対応した回路で処理されたデータを格納するための記憶手段のアドレス情報、前記順番に対応して前記プログラマブル論理回路に連続的に同時再構成可能な回路数の情報が付加されていることを特徴とする請求項１に記載の情報処理装置。

【請求項３】

前記インタフェース手段は、前記プログラマブル論理回路の構成可能な最大回路規模と再構成する回路規模との関係において、連続する処理回路が同時に再構成できる領域が確保できない場合に、処理ステップの順番に対応して回路の再構成と構成した処理回路による処理と、該処理回路で処理された中間処理データの前記記憶手段への入出力を順次行いな

がら連続的に処理が行われるように制御することを特徴とする請求項２に記載の情報処理装置。
【請求項４】
前記インタフェース手段は、前記プログラマブル論理回路の構成可能な最大回路規模と再構成する回路規模との関係において、複数の処理回路の再構成を行ってから、該処理回路における処理が連続して実行され、最後の処理回路による結果を中間処理データとして前記記憶手段に格納し、続けて複数の処理回路の再構成と複数の処理回路における処理が順次実行されるように制御することを特徴とする請求項２に記載の情報処理装置。
【請求項５】
前記インタフェース手段は、前記プログラマブル論理回路の構成可能な最大回路規模と再構成する回路規模との関係において、複数の処理回路の再構成を、同時に再構成できる数だけ連続的に行いながら、最初の処理回路の再構成が終了した時点で最初の処理が実行されるとともに、処理の実行と並列的に次の回路の再構成を行い、処理結果を次に再構成された処理回路に受け渡して連続して処理が行われるように制御することを特徴とする請求項２に記載の情報処理装置。
【請求項６】
前記処理回路における処理結果は中間処理データとして前記記憶手段に格納可能であることを特徴とする請求項４または請求項５に記載の情報処理装置。
【請求項７】
前記処理回路における処理結果は、予め規定した処理ステップの処理終了ごとに中間処理データとして前記記憶手段に格納可能であることを特徴とする請求項４または請求項５に記載の情報処理装置。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、アプリケーションプログラムによる処理の一部分を、回路構成を再構成できるプログラマブル論理回路で処理することが可能である情報処理装置に関する。特に、回路の再構成と処理を連続的に実行する方法に関するものである。
【０００２】
【従来の技術】
デジタル回路装置、特に特定用途向け集積回路（ＡＳＩＣ）の分野において、製品の開発期間を短縮するために、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）やプログラマブルロジックデバイス（ＰＬＤ）などで構成されたプログラマブル論理回路が広く使われている。これらのプログラマブル論理回路は、論理回路を記述する回路情報を読み込ませることで、内部の論理回路と論理回路間の結線を自由に構成することができる。このため、プログラマブル論理回路を用いることで、従来は回路設計の終了後に数週間から数か月を必要とした集積回路の作製時間が不要となるメリットがある。特に、例えば米国特許第４，７００，１８７号明細書等に記載されているような電気的に再構成可能なプログラマブル論理装置は、一度作製した回路を必要に応じて自由に何度でも変更できるという利点があり、ますます広く使われるようになってきている。
【０００３】
ところで、最近の論理回路はますます複雑になってきており、ひとつのプログラマブル論理回路では実現できない規模にまで回路規模が大きくなっている。この問題を解決するためのひとつの方法として、異なる時間に異なる論理回路を実現するために、プログラマブル論理回路を処理の途中で再構成することが提案されている。この方法を用いることにより、携帯情報端末のように装置が小型であるために内蔵できる回路規模に制約がある場合でも、様々な処理を比較的高速に行うことができるという利点がある。
【０００４】
しかし、プログラマブル論理回路を再構成するときには、回路全体の回路情報を再度読み込ませる必要があるため、再構成に時間がかかるという欠点がある。さらに、処理の途中

で再構成することは、処理を一時中断し、その時のデータをプログラマブル論理回路の外部の記憶装置に待避させ、新たな回路情報を読み込んで再構成し、再構成前のデータと再構成に伴う新しいデータを入力するという余分な処理が必要になる。

【０００５】

この問題を解決するものとして、例えば米国アトメル社の「ＣＯＮＦＩＧＵＲＡＢＬＥ ＬＯＧＩＣ」という名のデータブックに記載されているプログラマブル論理回路、および米国ザイリンクス社の「ＴＨＥ ＰＲＯＧＲＡＭＭＡＢＬＥＬＯＧＩＣ」という名のデータブックに記載されているプログラマブル論理回路等がある。これらのプログラマブル論理回路は、データを記憶するためのデータ記憶装置を有し、回路の動作中でも外部の記憶装置から回路情報の一部を読み込んで部分的に再構成を行うことができる。これによって、再構成するための時間を最小に留めるようにしている。

【０００６】

このようなプログラマブル論理回路を情報処理システムに用いる場合には、例えばＣＰＵなどの制御装置がアプリケーション等の処理とプログラマブル論理回路の再構成とを高速かつ効率的に行わなければならないという問題がある。すなわち制御装置は、所望の論理回路を構成するための回路情報を格納先から取り出し、必要に応じて複数の回路情報を合成し、プログラマブル論理回路内に所望の論理回路を再構成する。これと並行して、アプリケーションの処理を実行する必要があり、両者の処理を高速かつ効率的に行わねばならない。

【０００７】

以上に述べた複数の回路情報によりプログラマブル論理回路を再構成しながら処理を行う情報システムは、ネットワークに接続して利用することができる。その例として、特開平１０－７８９３２号公報に公開される「リコンフィグラブル・ネットワークコンピュータ」がある。

【０００８】

図１７は、従来の情報処理システムの一例を示すブロック図である。図中、５１はアプリケーションサーバ、５２，５３はクライアントコンピュータ、５４は通信ネットワーク、５５はメインプロセッサ、５６は拡張ハードウェア、５７はアプリケーションプログラム、５８は拡張コード、５９はメインプロセッサコード、６０はＯＳ、６１はコード選択機能である。この情報処理システムは、通信ネットワーク５４によって接続された複数のコンピュータで構成されている。

【０００９】

アプリケーションサーバ５１は、アプリケーションプログラムを配布するコンピュータである。また、クライアントコンピュータ５２，５３は、アプリケーションサーバ５１からアプリケーションプログラム５７をダウンロードして実行する。クライアントコンピュータ５２，５３には、メインプロセッサ５５が搭載されており、ＯＳ６０の管理下においてアプリケーションプログラム５７を実行することができる。また、クライアントコンピュータ５２は、メインプロセッサ５５とは別に拡張ハードウェア５６を有している。拡張ハードウェア５６は、上述のプログラマブル論理回路が搭載されており、プログラムにより機能を随時変更し、再構成することが可能である。

【００１０】

アプリケーションサーバ５１に格納されたアプリケーションプログラム５７においては、その一部の機能については、拡張ハードウェア５６のプログラムコード（拡張コード５８）と、クライアントコンピュータのメインプロセッサ５５のコード（メインプロセッサコード５９）が含まれている。

【００１１】

また、クライアントコンピュータ５２，５３のＯＳ６０には、それぞれのハードウェア構成に適したコードを選択するコード選択機能６１を有している。このコード選択機能６１が、拡張ハードウェア５６を実装しているか否かを判断し、クライアントコンピュータ５２のように拡張ハードウェア５６が実装されている場合には、アプリケーションプログラ

ム５７の中から拡張コード５８を取り出して拡張ハードウェア５６により実行する。またクライアントコンピュータ５３のように拡張ハードウェアを持たない場合には、メインプロセッサコード５９を選択して実行する。

【００１２】

別の構成では、拡張ハードウェア５６で実現する機能を、クライアントコンピュータ上に後から動的に追加／削除が可能なＯＳ６０の拡張機能あるいは動的ライブラリとして実現し、アプリケーションプログラム５７がＯＳ６０に対し、処理中に利用する拡張機能あるいは動的ライブラリの種類を登録する。ＯＳ６０は、拡張機能あるいは動的ライブラリがクライアントコンピュータに上に存在する場合にはそれを用い、存在しない場合には通信ネットワーク５４上のアプリケーションサーバ５１から必要とする拡張機能あるいは動的ライブラリを転送して利用する。

【００１３】

また、メインプロセッサコード５９と拡張コード５８は、一体となっているのではなく、アプリケーションプログラム５７またはＯＳ６０の拡張機能または動的ライブラリ毎に、個々のコードをアプリケーションサーバ５１に上に備えている。

【００１４】

拡張ハードウェア５６を構成するプログラマブル論理回路の構成が、クライアントコンピュータ間で異なる場合は、拡張コード５８を、適当なゲート数と入出力端子数の論理回路の機能をブール式等で記述した基本モジュールと、それらの接続関係を表現したコードから構成される。この基本モジュールをそれぞれプログラマブル論理回路の基本プログラムとして割り付ける機能と、複数のプログラマブルロジックチップにまたがる大きな拡張コードの場合には、基本モジュールを接続の度合いに応じて分割し、各プログラマブルロジックチップに配置、配線する機能を、アプリケーションサーバ５１またはクライアントコンピュータ上に持つ。

【００１５】

拡張ハードウェア５６を利用する複数のアプリケーションを同時に実行できるように、必要のなくなったハードウェア資源を別のアプリケーションプログラムのために再利用するハードウェア資源の管理機能と、拡張ハードウェア５６に入りきらない拡張コードを時分割で入れ替えるコード入れ替え機能を持つ。クライアントコンピュータ上で実行されるアプリケーションプログラム毎に適宜設定されるプライオリティ値、メインプロセッサ５５の処理能力値、拡張ハードウェア５６の処理能力値、ハードウェア資源量、コードを入れ替えるために必要な処理能力値を基に、ハードウェア資源に入りきらない複数のアプリケーションプログラムに対して選択する拡張ハードウェア管理機能を持つ。複数のアプリケーションが同時に同じ拡張コードを拡張ハードウェア５６で利用する場合には、内部状態のみを時分割で切り替えて機能を共有する。

【００１６】

以上のように、ネットワークで接続されたコンピュータ上で、アプリケーションサーバから配布されたアプリケーションプログラムをクライアントコンピュータ側で実行する際、プログラムにより機能を随時変更し、再構成可能な拡張ハードウェアをクライアントコンピュータに搭載する。そして、アプリケーションサーバに格納されたアプリケーションプログラムには、メインプロセッサコードと拡張コードを含ませておく。実行時には、拡張ハードウェアの有無、種類を判断するＯＳのコード選択機能によって、クライアントコンピュータ側で拡張ハードウェアの構成を変え、処理に適した構成にする。これによって、クライアントコンピュータにおいて、アプリケーションプログラムを高速に処理することができる。

【００１７】

また、従来、ネットワーク上で、クライアントコンピュータ側に特殊なハードウェアを必要とする新しいサービスを開始しようとする場合、クライアントコンピュータ側のユーザは、そのために新しいハードウェアを導入する必要があった。またサービスの提供者は、新しいハードウェアをもつ一部のユーザに対してのみ、新しいサービスを提供していた。

しかし上述のように、プログラマブル論理回路を搭載した拡張ハードウェアを有している構成では、プログラマブル論理回路を再構成することによって、新しいハードウェアを導入することなく、新しいサービスを開始することが可能となる。

【００１８】

このように拡張ハードウェアにプログラマブル論理回路を用いて、回路を再構成しながら処理を実行していく場合には、ＣＰＵによる制御が必要であった。例えばアプリケーションプログラムにおいて、処理のフローが決定されていて、そのために用いる機能回路があらかじめ選択指定可能な場合にも、常にメインプロセッサを動作させた処理方法が用いられている。特に、上述のシステム例のように、クライアントコンピュータに拡張ハードウェアが搭載されている場合、拡張コードによって再構成する制御は、クライアントコンピュータのメインプロセッサにより行われる。そのため、アプリケーションプログラムのメインプログラムにはあらかじめ必要な拡張コードが関連づけされていて、拡張ハードウェアの再構成の都度、メインプロセッサの制御によって拡張コードを用いて行われている。このため、多種多様な拡張ハードウェアを頻繁に再構成しながらアプリケーションプログラムを実行していく場合には、常に、メインプロセッサが処理の進行状態を監視するとともに、再構成のための制御を行う必要があり、メインプロセッサの負荷が大きくなる欠点がある。また、拡張ハードウェアの動作とともにメインプロセッサも動作させる必要があり、クライアント全体の消費電力も増大する欠点がある。

【００１９】

このように、回路の再構成のための制御によって、メインプロセッサの処理負荷が常時発生することにより、消費電力として、メインプロセッサと拡張ハードウェアとを合わせた増加となってしまう問題がある。また、回路の再構成のためのメインプロセッサの負荷のため、システム全体のパフォーマンスが制約を受けて低下してしまう欠点があった。

【００２０】

次に、プログラマブル論理回路の新しいデバイス技術について述べる。アプリケーションの処理に合わせた処理回路をプログラマブル論理回路上に構成し、この専用の処理回路を用いて高速処理を実現するというリコンフィギュラブルコンピューティングにプログラマブル論理回路が活用されはじめている。リコンフィギュラブルコンピューティングでは、アプリケーション処理で必要となる複数の処理回路の回路情報を記憶装置へ事前に格納しておき、必要に応じて記憶装置から読み出した回路情報をプログラマブル論理回路に書き込むことで、その時点で必要となる回路を生成する。この技術はキャッシュロジック技術とかバーチャルロジック技術と呼ばれる。

【００２１】

キャッシュロジック技術は、同じプログラマブル論理回路上に必要に応じて異なる回路を構成するという時分割駆動技術である。その結果、回路規模の小さなプログラマブル論理回路を用いて、その回路規模以上の回路を実現でき、回路装置の小型化と低コスト化が可能となる。しかしながら、プログラマブル論理回路に書きこむ回路情報の規模によっては、回路の再構成時間が長くなり、専用の処理回路を用いて高速処理を実現するというリコンフィギュラブルコンピューティングの効果を損なうという問題がある。

【００２２】

この問題のひとつの解決方法が、マルチコンテキスト技術と呼ばれるデバイス技術である。すなわち、プログラマブル論理回路内に複数の回路情報を格納するメモリを備え、必要に応じてメモリを切り替えて回路を再構成することにより、回路の再構成時間を大幅に短縮できる。

【００２３】

マルチコンテキスト技術の従来例のひとつが、ＦＰＤ’９５の“Ａ　Ｆｉｒｓｔ　Ｇｅｎｅｒａｔｉｏｎ　ＤＰＧＡ　Ｉｍｐｌｅｍｅｎｔａｔｉｏｎ”で示されたＤＰＧＡである。図１８は、ＤＰＧＡの論理セル構造の一例の説明図である。図中、７１はＤＲＡＭ、７２，７５はマルチプレクサ、７３はルックアップテーブル、７４はフリップフロップである。図１８に示すように、ＤＰＧＡの論理セルは、４組の構成を格納する４×３２ビット

のＤＲＡＭ７１、４つの８入力マルチプレクサ７２、４入力ルックアップテーブル７３、フリップフロップ７４、出力を切り替えるマルチプレクサ７５で構成されている。３２ビットのＤＲＡＭ７１の出力のうち、１２ビットが４つの８入力マルチプレクサ７２の状態を、１６ビットが４入力ルックアップテーブル７３の状態を、１ビットがマルチプレクサ７５の状態を決定し、残り３ビットは保留されている。４組のＤＲＡＭ７１には、それぞれ異なるデータが格納されており、メモリ出力を切り替えることにより、異なる回路を構成することができる。

【００２４】

マルチコンテキスト技術は、回路の再構成時間を大幅に短縮することができる。また、回路を再構成するための回路情報をプログラマブル論理回路に転送するときの入出力バスラインの負荷を小さくできるため、高速化と低消費電力化に有利な構成である。しかし、回路情報を格納するためのメモリ回路領域をプログラマブル論理回路と一体構成する必要があり、集積回路化した場合のプログラマブル論理回路全体の回路規模が大きくなるという欠点を有している。

【００２５】

【発明が解決しようとする課題】

本発明は、上述した事情に鑑みてなされたもので、プログラマブル論理回路を再構成するためのメインプロセッサの負荷を大幅に軽減させることができ、これにより低消費電力化を図るとともに、メインプロセッサを有効に活用できるようにしてシステム全体のパフォーマンスを向上させた情報処理装置を提供することを目的とするものである。

【００２６】

【課題を解決するための手段】

一般にアプリケーションプログラムにおいて、処理のフローが決定されていて、そのために用いる処理回路が複数の種類の組み合わせであらかじめ選択指定可能な場合、同じ順番を単位として一定の繰り返される処理の場合などには、あらかじめ使用する回路情報と順番を決定できる。本発明はこれを利用し、プログラマブル論理回路に処理回路を構成するための回路情報を、予め順番に回路情報保持手段に保持させておく。そしてＣＰＵなどの制御手段は、プログラマブル論理回路で処理を行う処理データのヘッダ部などに、使用する回路情報を特定するための情報を含めて、処理を依頼する。インタフェース手段は、与えられた処理データの例えばヘッダ部などから、プログラマブル論理回路の再構成に使用する最初の回路情報を特定するとともに、特定された回路情報から回路情報保持手段に保持されている順番に回路情報を取り出し、プログラマブル論理回路を再構成して処理回路を構成し、構成した処理回路に処理データの処理を行わせる。

【００２７】

このようにして、プログラマブル論理回路では、予め設定されている順番で、処理回路の再構成と実行が自動的に行われる。そのため、ＣＰＵなどの制御手段では、最初に実行指示を行うだけで、以後順番に行われるプログラマブル論理回路の再構成や実行指示を行う必要がない。これによってＣＰＵなどの制御手段によるプログラマブル論理回路の制御を行うための処理ステップが大幅に削減され、高速化を図ることができるとともに、消費電力を大幅に低減することができる。これとともに、プログラマブル論理回路を含めた情報処理装置における全体のパフォーマンスを向上させることができる。

【００２８】

【発明の実施の形態】

図１は、本発明の情報処理システムの実施の一形態を示す構成図である。図中、１は情報処理システム、２はネットワーク、３は外部機器、１１はＣＰＵ、１２はホストバス、１３はチップセット、１４はメインメモリ、１５はシステムバス、１６はハードディスクインタフェース、１７はハードディスクドライブ、１８は通信インタフェース、１９は拡張ハードウェア部、２１はプログラマブル論理回路、２２はローカルメモリ、２３はプログラマブル論理回路インタフェース、２４は外部入出力インタフェースである。

【００２９】

情報処理システム１において、ＣＰＵ１１のホストバス１２に、チップセット１３に含まれる図示しないメモリコントローラを介して、ＤＲＡＭで構成されるメインメモリ１４が接続されている。ホストバス１２は、チップセット１３に含まれる図示しないバスブリッジを介してシステムバス１５に接続されている。システムバス１５は、例えばＰＣＩバスやＩＳＡバス、その他各種のバスを使用することができる。例えばシステムバス１５としてＰＣＩバスを用いた場合、チップセット１３内にはホスト－ＰＣＩブリッジを備えていればよい。

【００３０】

システムバス１５には、この例では、ハードディスクインタフェース１６を介してハードディスクドライブ１７が接続され、また通信インタフェース１８を介してネットワーク２と接続されている。さらに拡張ハードウェア部１９が接続され、この拡張ハードウェア部１９を介して外部機器３が接続されている。もちろん他の種々の機器が直接あるいはインタフェースを介してシステムバス１５に接続されていてもよく、また、ハードディスクおよびネットワークについても接続は任意である。

【００３１】

この例では、ハードディスクドライブ１７にはアプリケーションプログラムが格納されている。アプリケーションプログラムは、ハードディスクインタフェース１６、システムバス１５、および、チップセット１３に含まれる図示しないバスブリッジを介して、ハードディスクドライブ１７からメインメモリ１４にロードされてＣＰＵ１１によって実行される。

【００３２】

また、通信インタフェース１８は、ＬＡＮやインターネットなどのネットワーク２を介して、様々な機器との間でデータの転送を行うことができる。ＣＰＵ１１によって実行されるアプリケーションプログラムは、この通信インタフェース１８を介して通信を行うことにより、ネットワーク２に接続される例えば記憶装置に格納されている情報へのアクセスを行うことができ、様々なアプリケーションプログラムやデータなどを入手できる。この場合、ネットワーク２に接続される通信インタフェース１８を介して転送したアプリケーションプログラムをメインメモリ１４に格納して実行したり、あるいはシステムバス１５から直接プログラマブル論理回路インタフェース２３を介してプログラマブル論理回路２１へ転送することもできる。

【００３３】

拡張ハードウェア部１９は、プログラマブル論理回路２１、ローカルメモリ２２、プログラマブル論理回路インタフェース２３を有している。またこの例では、外部入出力インタフェース２４もこの拡張ハードウェア部１９に設けられている。プログラマブル論理回路２１は、回路情報を変更することによって機能を随時変更し再構成することが可能であり、プログラマブル論理回路インタフェース２３を介してシステムバス１５に接続されている。また、この例では外部入出力インタフェース２４を介して外部機器３とも接続されており、プログラマブル論理回路２１における処理により、外部機器３を制御可能に構成した例を示している。

【００３４】

ローカルメモリ２２は、プログラマブル論理回路２１で処理を行う処理データや、処理後の中間処理データなどを保持することができる。また、予め使用する回路情報と順番を保持する回路情報保持手段としても機能する。

【００３５】

プログラマブル論理回路インタフェース２３は、システムバス１５によってＣＰＵ１１やメインメモリ１４、ローカルメモリ２２、プログラマブル論理回路２１との間でデータ転送や制御を行うためのものである。またプログラマブル論理回路インタフェース２３は、転送されてきた処理データからプログラマブル論理回路の再構成に使用する最初の回路情報を特定する機能を有している。この最初の回路情報が例えば処理データのヘッダ情報として付加されている場合、処理データのヘッダ情報を解釈することによって実現できる。

また、回路再構成のための回路情報や処理データ、中間処理データの最後に付加されるＥＯＦのマーカ検出機能なども含まれている。さらに、特定した最初の回路情報をもとにローカルメモリ２２から回路情報を順番に取り出し、順次プログラマブル論理回路２１を再構成して処理回路を構成し、構成した処理回路に処理データの処理を行わせる。また、ローカルメモリ２２とプログラマブル論理回路２１との間での処理データや中間処理データの転送も行う。

【００３６】

なお、予め使用する回路情報等は、例えばメインメモリ１４やハードディスクドライブ１７に格納しておいてもよい。この場合、プログラマブル論理回路インタフェース２３は、メインメモリ１４やハードディスクドライブ１７に対して、ＣＰＵ１１を動作させずにデータ転送を行う機能を有していればよい。例えばメインメモリ１４とローカルメモリ２２を同じメモリ空間においてアクセス可能に構成し、いずれのメモリを利用しているかを意識しないでアクセスできるように構成してもよい。

【００３７】

また、拡張ハードウェア部１９は、集積化により一体化することで、入出力部のライン負荷を低減させたり、専用バス化の構成により、高速化及び低消費電力化を図ることができる。

【００３８】

図２は、プログラマブル論理回路の一例を示す平面構造図、図３は、同じく内部構造の一例を示すブロック図である。図中、３１は論理セル、３２は配線領域、３３は入出力端子、４１はコンフィギュレーションメモリ、４２は回路素子である。プログラマブル論理回路２１は、回路情報を格納するためのコンフィギュレーションメモリ４１と、論理セル３１や配線領域３２からなる回路素子４２と、入出力端子３３とで構成されている。

【００３９】

コンフィギュレーションメモリ４１は、ＥＥＰＲＯＭ、ＳＲＡＭなどの書き換え可能なメモリ素子で構成されている。回路情報はアドレスとデータの対で構成される。コンフィギュレーションメモリ４１にアドレスを与えて、そのアドレスに対応するメモリセルにアドレスと対になったデータを格納すると、このデータに従って、論理セル３１内の回路構成や、論理セル３１と入出力端子３３を相互に接続する配線領域３２の接続状態が再構成される。コンフィギュレーションメモリ４１の一部分を書き換えることにより、プログラマブル論理回路２１が動作中であっても、回路を部分的に再構成することができる。

【００４０】

プログラマブル論理回路２１に再構成された回路素子４２に、入出力端子３３を介して処理すべきデータ（処理データや中間処理データ）が入力され、また、その処理結果（中間処理データ）が出力される。データ入力先の論理セルと、データ出力元の論理セルは、論理セルの位置に対応するセル座標を示した制御コードによってアプリケーションプログラムが指定する。

【００４１】

以上のシステム構成によって、プログラマブル論理回路２１による処理データの入出力方法から、以下のような処理形態があげられる。データとしては、マルチメディアのような画像、音声などのストリーミングデータなどがある。

▲１▼　ネットワーク２を経由して情報システム１に入力したデータを、システムバス１５を経由して、プログラマブル論理回路２１に再構成した処理回路で処理する。

▲２▼　ハードディスクドライブ１７などの記憶装置に格納されたデータを、システムバス１５を経由して、プログラマブル論理回路２１に再構成した処理回路で処理する。

▲３▼　外部の記憶装置からの転送やアプリケーションの処理結果などのデータで、メインメモリ１４に一時的に格納されたデータを、システムバス１５を経由して、プログラマブル論理回路２１に再構成した処理回路で処理する。

▲４▼　ローカルメモリに格納されたデータを、プログラマブル論理回路インタフェース２３を経由して、プログラマブル論理回路２１に再構成した処理回路で処理する。

【００４２】
次に、本発明の情報処理システムの実施の一形態における動作について説明する。アプリケーションプログラムにおいて、処理のフローが決定されていて、そのために用いる機能回路が複数の種類の組み合わせであらかじめ選択指定可能な場合や、同じ順番を単位として繰り返される処理の場合などには、あらかじめ使用する回路情報と順番を決定できる。そこで、アプリケーションの処理が開始される前に、予め、使用する回路情報と順番等の情報を参照テーブルとして例えばローカルメモリ２２などに登録しておく。なお、全く不定な順番で、任意の処理の次にどのような処理を行うことになるのか特定できない場合には、メインプロセッサによって、その都度、制御を行えばよいので、ここでは対象外とする。

【００４３】
図４は、参照テーブルの一例の説明図である。図４に示した例では、処理回路の順番を示す処理ステップＮｏ．に対応して、再構成のための回路情報が格納されているメモリ中の開始アドレスを示すポインタ、中間処理を行うことになる場合に対応してアロケーションされた中間処理データを格納するメモリ中の開始アドレスを示すポインタ、プログラマブル論理回路２１に同時に再構成可能な回路数の情報によって参照テーブルが構成されている。

【００４４】
アプリケーションの処理でｉ番目の処理回路で処理した結果を次のｉ＋１番目で処理する場合には、ｉ番目の処理回路から出力されるデータは中間処理データとなる。また最後のＭ番目で処理した結果は、最終出力データとなる。この中間処理データを格納するメモリの開始アドレスポインタが図４におけるポインタＰｄｉである。なお、最終データもさらに別のアプリケーションで用いられることも考えられるので、最終データも中間処理データ用メモリへ格納しておいてもよい。

【００４５】
また、再構成する処理回路について、プログラマブル論理回路の回路規模に応じて、同時に再構成できる回路数Ｎｉは、ｉ番目の回路からの同時再構成可能最大数である。このＮｉに対応する回路の再構成のモードとして、１個の場合には自動的に単独コンフィギュレーションモードになり、複数個の場合には単独コンフィギュレーションモードまたは連続コンフィギュレーションモードの選択ができる。これは、アプリケーション開始時にＣＰＵによって選択指定できる。

【００４６】
この図４に示すような参照テーブルは、アプリケーションが実行されるときに情報処理システムで用いられるプログラマブル論理回路２１の品種や回路規模サイズ、メインメモリ１４やローカルメモリ２２などのメモリサイズなどを考慮して、ＣＰＵ１１によって作成する。あるいは、当然ながら、決まった処理を同じシステムで行う場合には、処理の実行の都度にこの参照テーブルを作成する必要はなく、予め用意しておくこともできる。

【００４７】
図５は、処理データおよび回路情報と、これらを格納するメモリアドレス空間の説明図である。図４に示すような参照テーブルの情報は、図５に示すように、例えば処理データのヘッダ部に付加しておくことができる。プログラマブル論理回路インタフェース２３において、処理データを受け取った際にヘッダ部を解析し、これらの情報を取り出して、参照テーブルを作成することができる。また、このような参照テーブルをもとに、ある処理ステップｉを実行する際には、処理ステップＮｏ．ｉに対応付けられている回路情報用のポインタＰｃｉと中間処理データ用のポインタＰｄｉが割り付けられる。処理の順番によっては、同じポインタを用いて上書きでアロケートすることによって、メモリの使用効率を向上させることもできる。

【００４８】
なお、図５に示すメモリアドレス空間は、ローカルメモリ２２のメモリアドレス空間、あるいは、メインメモリ１４とローカルメモリ２２を一体としたメモリアドレス空間であっ

てよい。後者の場合、例えば回路情報や中間処理データを格納する領域などがメインメモリ１４上に確保される場合もある。

【００４９】

前述の回路情報と順番において、図４に示すように、処理ステップＮｏ．として最初の０番からＭ番までのＭ＋１ステップの場合で、ｉ番目の処理を行う際の動作について述べる。再構成する回路情報を格納するメモリの開始アドレスポインタをＰｃｉとする。また、中間処理データを格納するメモリの開始アドレスポインタをＰｄｉとする。さらに、再構成する処理回路について、プログラマブル論理回路の回路規模に応じて、同時に再構成できる回路数をＮｉとする。

【００５０】

アプリケーションの処理データファイル及び回路情報を読み出し、あらかじめアプリケーションプログラムの実行前に、回路情報をメモリ空間に割り付けておく。また、ＣＰＵ１１によって前述の４種の情報をアプリケーションの処理データファイルのヘッダ部に付加する。なお、処理データファイルや回路情報は、例えば、ネットワーク２に接続された記憶装置、情報処理システム１内部のハードディスクドライブ１７などの記憶装置、メインメモリ１４、あるいは、ローカルメモリ２２など、いずれの記憶装置に記憶されていてもよい。

【００５１】

アプリケーションプログラムの実行がＣＰＵ１１によって開始されると、処理データファイルがプログラマブル論理回路インタフェース２３に転送されてヘッダ部に付加されている情報が解釈される。最初の回路情報によってプログラマブル論理回路２１に対する処理回路の再構成が行われ、回路情報の最後を示すＥＯＦのマーカにより、再構成が終了する。このＥＯＦマーカをプログラマブル論理回路インタフェース２３で検出し、ヘッダ部に続くデータ部の転送が行われて、再構成した処理回路におけるデータ処理が進められていく。

【００５２】

処理されたデータを次に再構成する処理回路で用いる場合には、処理されたデータを中間処理データとして、さきにメモリ空間にアロケートした領域に一旦格納する。そして、次の処理回路を再構成した後、中間処理データを新たに再構成した処理回路に入力し、データ処理を進める。最終データは、例えば外部入出力インタフェース２４を介して外部機器３に転送されたり、あるいはメインメモリ１４，ハードディスクドライブ１７，ネットワーク２に接続された記憶装置などに転送される。または、中間処理データと同様に格納されていてもよい。

【００５３】

以上のように、ＣＰＵ１１が最初の開始制御を行うことによって、回路の再構成とデータ処理が順次実行されていくので、ＣＰＵ１１はプログラマブル論理回路２１の再構成のための処理を行う必要がない。そのため、ＣＰＵ１１の負荷を軽減して消費電力を大幅に低減し、またＣＰＵ１１とプログラマブル論理回路２１との並行動作を可能として情報処理システムにおける全体のパフォーマンスを向上させることができる。

【００５４】

次に、プログラマブル論理回路２１への処理回路の再構成領域とそれぞれの処理回路によって処理された中間処理データのメモリへの格納あるいは連続処理について、以下、３つの形態をあげて説明する。

【００５５】

図６は、プログラマブル論理回路への処理回路の第１の再構成例の説明図、図７は、同じく動作時の一例を示すタイミングチャートである。図６に示す例では、シングルタスクモードで、処理回路の再構成とその処理回路での処理の実行を順次行いながら、中間処理データの入出力をその間に入れていく方式を示している。この例は、プログラマブル論理回路２１の構成可能な最大回路規模と再構成する回路規模との関係において、連続する処理回路が同時に再構成できる領域が確保できない場合に対応する形態である。

【００５６】
最初の処理回路の再構成が完了し、データ処理が行われて得られた結果は、一時的にプログラマブル論理回路２１の内部メモリまたはローカルメモリ２２やメインメモリ１４など外部メモリを利用して中間処理データのまま格納する。そして、最初に再構成された領域に上書きして次の処理回路の再構成を行う。次の処理回路の再構成が完了すると、続けて中間処理データに対してデータ処理を行う。このようにしてデータ処理が継続されていく。処理モードとしては、回路の同時再構成可能な数Ｎｉは、１個である単独コンフィギュレーションモードの場合である。
【００５７】
一例として、図６に示す例について、図７に示すタイミングチャートを用いて説明する。まず、ＣＰＵ１１によって、例えば図４に示すような参照テーブルが用意され、図５に示すようにメモリアドレス空間が割り当てられる。プログラマブル論理回路インタフェース２３は、ＣＰＵ１１から処理データが転送されてくると、その処理データのヘッダ部に付加された情報を解釈して、プログラマブル論理回路２１に対する回路Ａの再構成を開始する（図７－▲１▼）。
【００５８】
回路Ａの再構成が完了すると、処理データを回路Ａに入力し、回路Ａを用いた処理が開始される（図７－▲２▼）。回路Ａによって得られた結果は、中間処理データとしてメモリに格納されていく（図７－▲３▼）。
【００５９】
処理データのＥＯＦが検出されて処理が終了すると、次の回路Ｂの再構成を行う（図７－▲４▼）。最初に再構成された回路Ａの領域に上書きして次の処理回路である回路Ｂの再構成を行うことができる。図６は、この回路Ｂを再構成したときの状態を示している。回路情報のＥＯＦが検出されると、回路Ｂの再構成が完了する（図７－▲５▼）。回路Ｂの再構成が完了すると、回路Ｂは中間処理データに対してデータ処理を開始する（図７－▲６▼）。
【００６０】
このステップが繰り返されて、処理ステップＭで終了する。このように、最初にＣＰＵ１１による制御を施すだけで、あとは、処理回路の再構成と、再構成された処理回路における処理の実行が、あらかじめ設定された規定通りに連続して行われる。これによって、ＣＰＵ１１の制御負荷が大幅に軽減され、消費電力が大幅に低減できる。
【００６１】
図８は、プログラマブル論理回路への処理回路の第１の再構成例における動作の一例を示すフローチャートである。上述の例について処理動作をまとめると図８に示すようになる。Ｓ１０１において処理ステップを示す変数ｉを０に初期化し、Ｓ１０２において、処理データを入力し、ヘッダ部の情報を解釈してポインタ等の設定を行う。
【００６２】
Ｓ１０３において、回路情報Ｐｃｉにアクセスしてプログラマブル論理回路２１に対して処理回路の再構成を行う。Ｓ１０４において回路情報の終了を示すＥＯＦを検出すると、処理回路の再構成を終了する。そしてＳ１０５において、中間処理データＰｄ（ｉ－１）へアクセスして再構成した処理回路に入力し、あるいは、処理データを再構成した処理回路に入力して、Ｓ１０６においてデータ処理を開始する。
【００６３】
Ｓ１０７において、処理データあるいは中間処理データの終了を示すＥＯＦを検出すると、処理回路における処理を終了する。Ｓ１０８において、処理ステップを示す変数ｉに１を加え、Ｓ１０９において、変数ｉの値がＭ以下か否かを判定する。変数ｉの値がＭ以下であれば、Ｓ１１０において、処理結果を中間処理データとして中間処理データ用のメモリ領域Ｐｄ（ｉ－１）に格納する。そしてＳ１０３へ戻り、次の処理回路の再構成および処理を繰り返す。
【００６４】

処理ステップＭまでの処理を行い、処理ステップを示す変数ｉの値がＭを超える場合には、Ｓ１１１において、処理後のデータを最終データとして出力し、プログラマブル論理回路２１における処理を終える。

【００６５】

なお、上述の説明では中間処理データを生成して処理する形態で記述したが、図２に示すように回路Ａの結果を内部メモリで受けて、引き続き回路Ｂで処理する場合には、内部メモリの代わりに、データラッチ回路を用いた一時的なデータ保持方法の利用も可能である。

【００６６】

図９は、プログラマブル論理回路への処理回路の第２の再構成例の説明図、図１０は、同じく動作時の一例を示すタイミングチャートである。図９に示す例では、シングルタスクモードではあるが、複数の処理が同時に形成できる場合に、複数の処理回路の再構成を行ってから、処理を連続して実行する形態である。処理モードとして、回路の同時再構成可能な数Ｎｉまでの再構成を連続的に行う、連続コンフィギュレーションモードの選択である。

【００６７】

プログラマブル論理回路２１の構成可能な最大回路規模と再構成する処理回路の回路規模との関係において、連続する処理回路が同時に再構成できる領域が確保できるが、データ処理が行われて得られた結果は、一時的にプログラマブル論理回路の内部メモリまたはローカルメモリやメインメモリの外部メモリを利用して中間処理データのまま格納して行っていく場合である。ただし、中間処理データを介しながら分割して処理を進めていく場合も含めることができる。当然ながら、処理モードとして、処理回路を一つずつ再構成してゆく単独コンフィギュレーションモードも選択できる。この場合には、図６に示した形態として扱うこともできる。

【００６８】

一例として、図９に示す例について、図１０に示すタイミングチャートを用いて説明する。まず、ＣＰＵ１１によって、図４に示すような参照テーブルが用意され、図５に示すようにメモリアドレス空間が割り当てられる。プログラマブル論理回路インタフェース２３は、ＣＰＵ１１から処理データが転送されてくると、その処理データのヘッダ部に付加された情報を解釈する。そして、プログラマブル論理回路２１に再構成可能な複数個の回路、ここでは回路Ａと回路Ｂの再構成を開始する（図１０－▲１▼、▲２▼）。

【００６９】

２つの回路情報のＥＯＦが検出されると再構成を完了し、これらの処理回路を用いた処理の実行が開始される（図１０－▲３▼、▲４▼）。ここでは、回路Ａによって処理された結果が回路Ｂに入力され、回路Ｂによる処理結果が中間処理データとしてメモリに格納されていく。ここでは、最後の処理回路による処理結果のみを示している（図１０－▲５▼、▲６▼）。

【００７０】

このような処理が繰り返されて、処理ステップＭで終了する。このように連続処理が可能になるため、中間処理データのすべてに対してメモリへ格納せずに処理を進めることができる。そのため、中間処理データを格納する回数が低減され、高速処理、消費電力の低減が図れる。また、最初にＣＰＵ１１による制御を施すだけで、あとは、処理回路の再構成と、再構成された処理回路における処理の実行が、あらかじめ設定された規定通りに連続して行われる。これによって、ＣＰＵ１１の制御負荷が大幅に軽減され、消費電力が大幅に低減できる。

【００７１】

図１１は、プログラマブル論理回路への処理回路の第２の再構成例における動作の一例を示すフローチャートである。上述の第２の構成例について処理動作をまとめると図１１に示すようになる。Ｓ１２１において処理ステップを示す変数ｉを０に初期化し、Ｓ１２２において、処理データを入力し、ヘッダ部の情報を解釈してポインタ等の設定を行う。

【００７２】
Ｓ１２３において、再構成した回路数を示す変数ｊを０に初期化する。Ｓ１２４において、回路情報Ｐｃ（ｉ＋ｊ）にアクセスしてプログラマブル論理回路２１に対して処理回路の再構成を行う。Ｓ１２５において回路情報の終了を示すＥＯＦを検出すると、処理回路の再構成を終了する。Ｓ１２６において変数ｊの値に１を加算し、Ｓ１２７において、新たな変数ｊの値が処理ステップｉにおける同時構成可能な回路数Ｎｉより小さいか否かを判定する。新たな変数ｊの値がＮｉより小さければ、Ｓ１２４へ戻って、同時に再構成可能な次の処理回路の再構成を行う。変数ｊの値がＮｉに達すると、Ｓ１２８において、同時に再構成可能な回路数までの処理回路の再構成を終了する。

【００７３】
Ｓ１２９において、変数ｉが０の場合には処理データを、また変数ｉが０でない場合には中間処理データＰｄ（ｉ＋Ｎｉ－１）へアクセスして、再構成した処理回路に入力し、Ｓ１３０においてデータ処理を開始する。

【００７４】
Ｓ１３１において、処理データあるいは中間処理データの終了を示すＥＯＦを検出すると、処理回路における処理を終了する。Ｓ１３２において、処理ステップを示す変数ｉにＮｉを加え、Ｓ１３３において、新たな変数ｉの値がＭ以下か否かを判定する。変数ｉの値がＭ以下であれば、Ｓ１３４において、処理結果を中間処理データとして中間処理データ用のメモリ領域Ｐｄ（ｉ＋Ｎｉ－１）に格納する。そしてＳ１２３へ戻り、次の複数の処理回路の再構成および再構成した複数の処理回路による連続処理を繰り返す。

【００７５】
処理ステップＭまでの処理を行い、処理ステップを示す変数ｉの値がＭを超える場合には、Ｓ１３５において、処理後のデータを最終データとして出力し、プログラマブル論理回路２１における処理を終える。

【００７６】
なお、上述の第１の再構成例と同様に、シングルタスクモードで単独コンフィギュレーションを行う場合には、それぞれの処理ステップごとに中間処理データをメモリへ格納していけばよい。連続した回路の同時構成が可能なため、連続した処理が可能になる。当然、必要に応じて中間処理データはローカルメモリ２２などへ格納されてもよい。

【００７７】
図１２は、プログラマブル論理回路への処理回路の第３の再構成例の説明図、図１３は、同じく動作時の一例を示すタイミングチャートである。図１２に示す例では、マルチタスクモードとして、処理回路の再構成が完了すると処理が実行されるとともに、その処理の実行中に並行して次の処理回路の再構成がプログラマブル論理回路の別の領域に対して行われる。そして、先の処理回路による処理結果が連続して処理されたり、別のデータによる並列処理などが行われていく。例えば回路Ａによるプロセス中に回路Ｂのコンフィギュレーションを行って処理を連続的に行っていくものである。しかも、処理回路の再構成は、プログラマブル論理回路２１に同時再構成可能な数Ｎｉまでの処理回路の再構成を連続的に行う、連続コンフィギュレーションモードが選択される。

【００７８】
なおこの例では、さきの処理回路の再構成が完了してデータ処理が行われているときに、次の処理回路の再構成が並列して行われていく。さきの処理回路によるデータ処理の結果は、一時的にプログラマブル論理回路２１の内部メモリまたはローカルメモリ２２やメインメモリ１４等の外部メモリを利用して中間処理データのまま格納して、次の回路の再構成が完了したときに続けてデータ処理が継続されていく。しかし、次の処理回路の再構成が、さきの処理回路の出力までに完了する場合には、図９に示したように回路Ａの出力を直接、回路Ｂに入力するように構成してもよい。

【００７９】
一例として、図１２に示す例について、図１３に示すタイミングチャートを用いて説明する。まず、ＣＰＵ１１によって、図４に示すような参照テーブルが用意され、図５に示す

ようにメモリアドレス空間が割り当てられる。プログラマブル論理回路インタフェース２３は、ＣＰＵ１１から処理データが転送されてくると、その処理データのヘッダ部に付加された情報を解釈する。プログラマブル論理回路インタフェース２３は、ＣＰＵ１１から処理データが転送されてくると、その処理データのヘッダ部に付加された情報を解釈して、プログラマブル論理回路２１に対する回路Ａの再構成を開始する（図１３－▲１▼）。

【００８０】

回路Ａに関する回路情報のＥＯＦを検出し、回路Ａの再構成が完了すると、処理データを回路Ａに入力し、回路Ａを用いた処理が開始される（図１３－▲２▼）。回路Ａによって得られた結果は、中間処理データとしてメモリに格納されていく（図１３－▲３▼）。

【００８１】

この回路Ａにおける処理と並行して、次の回路Ｂの再構成を開始する（図１３－▲４▼）。このような処理回路において処理を行っている間に、プログラマブル論理回路に回路の同時再構成可能な数Ｎｉまでの再構成を連続的に行うことができる。

【００８２】

回路Ａにおける処理において処理データのＥＯＦを検出して処理の終了を検出するとともに、次の回路Ｂの再構成の終了を検出すると、回路Ｂにおける中間処理データを用いた処理の実行が開始される（図１３－▲５▼、▲６▼）。

【００８３】

このように、処理回路の再構成と、再構成した処理回路の実行の並列処理によって、処理回路の再構成による処理の待ち時間を短くして高速化を図ることができるとともに、処理を連続して実行することによる処理の高速化を図ることができる。また、最初にＣＰＵ１１による制御を施すだけで、あとは、処理回路の再構成とその実行を、あらかじめ設定された規定通りに連続して行うことができる。これによって、ＣＰＵ１１の制御負荷が大幅に軽減され、消費電力が大幅に低減できる。

【００８４】

図１４は、プログラマブル論理回路への処理回路の第３の再構成例における動作の一例を示すフローチャートである。上述の第３の構成例について処理動作をまとめると図１４に示すようになる。Ｓ１４１において処理ステップを示す変数ｉを０に初期化し、Ｓ１４２において、処理データを入力し、ヘッダ部の情報を解釈してポインタ等の設定を行う。

【００８５】

Ｓ１４３において、回路情報Ｐｃｉにアクセスしてプログラマブル論理回路２１に対して処理回路の再構成を行う。Ｓ１４４において回路情報の終了を示すＥＯＦを検出すると、処理回路の再構成を終了する。そしてＳ１４５において処理ステップｉの処理回路の再構成が終了していることを確認後、Ｓ１４６において、中間処理データＰｄ（ｉ－１）へアクセスして再構成した処理回路に入力し、あるいは、処理データを再構成した処理回路に入力して、Ｓ１４７においてデータ処理を開始する。

【００８６】

このような再構成された処理回路における処理の実行と並行して、Ｓ１６１以降の処理回路の再構成を行う。まずＳ１６１において変数ｉの値を変数ｋに待避し、Ｓ１６２において、再構成した回路の処理ステップを示す変数ｊの値をｋ＋１に初期化する。そして、Ｓ１６３において、回路情報Ｐｃ（ｊ）にアクセスしてプログラマブル論理回路２１に対して、空いている領域に処理回路の再構成を行う。Ｓ１６４において回路情報の終了を示すＥＯＦを検出すると、処理回路の再構成を終了する。Ｓ１６５において変数ｊの値に１を加算し、Ｓ１６６において、同時に再構成した回路数が、処理ステップｋにおいて同時構成可能な回路数Ｎｋより小さいか否かを判定する。この判定のために、いままで再構成した回路数＋１を示す新たな変数ｊの値と、ｋ＋Ｎｋの値とを比較する。新たな変数ｊの値がｋ＋Ｎｋより小さければ、Ｓ１６３へ戻って、同時に再構成可能な次の処理回路の再構成を行う。変数ｊの値がｋ＋Ｎｋに達すると、同時に構成可能な回路数に達したものとして、処理回路の再構成を一旦停止する。

【００８７】

Ｓ１６７において、変数ｊの値がＭに達していれば、すべての処理回路の再構成が終了したものとして、再構成の処理を終了する。変数ｊの値がＭ以下であれば処理を継続し、Ｓ１６８において、再構成が終了している処理回路のすべてにおいて処理が終了するまで待ち、その後、Ｓ１６１へ戻って、次の同時に構成可能な回路数だけの処理回路の再構成処理を行う。

【００８８】

このような処理回路の再構成処理と並行して行われていた処理回路における処理の実行は、Ｓ１４８において処理データあるいは中間処理データの終了を示すＥＯＦを検出して終了する。Ｓ１４９において、処理ステップを示す変数ｉに１を加え、Ｓ１５０において、新たな変数ｉの値がＭ以下か否かを判定する。変数ｉの値がＭ以下であれば、Ｓ１５１において、処理結果を中間処理データとして中間処理データ用のメモリ領域Ｐｄ（ｉ－１）に格納する。そしてＳ１４５へ戻り、次の処理回路の再構成終了を確認してから、次の処理回路による連続処理を繰り返す。

【００８９】

処理ステップＭまでの処理を行い、処理ステップを示す変数ｉの値がＭを超える場合には、Ｓ１５２において、処理後のデータを最終データとして出力し、プログラマブル論理回路２１における処理を終える。

【００９０】

なお、上述の説明では、処理回路の再構成を、同時に構成可能な回路数ごとに行っている。しかしこれに限らず、例えば処理が終了した処理回路の領域を順次開放してゆき、次の処理回路が構成可能な領域が確保できた時点ですぐに次の処理回路を再構成するように構成してもよい。

【００９１】

上述の３つの例は、適宜組み合わせることが可能である。図１５は、プログラマブル論理回路への処理回路の第４の再構成例の説明図、図１６は、同じく動作時の一例を示すタイミングチャートである。この例では、最初に上述の第２の再構成例で示したように、同時に構成可能な回路数だけの処理回路の再構成を先に行っておき、その後、第３の再構成例で示したように、処理が終了した処理回路の領域を開放して新たな処理回路の再構成を行うことができる。

【００９２】

一例として、図１５に示す例について、図１６に示すタイミングチャートを用いて説明する。まず、図１５（Ａ）に示すように、プログラマブル論理回路２１に再構成可能な複数個の回路、ここでは回路Ａと回路Ｂの再構成を開始する（図１６－▲１▼、▲２▼）。

【００９３】

２つの回路情報のＥＯＦが検出されると再構成を完了し、これらの処理回路を用いた処理の実行を開始する（図１６－▲３▼、▲４▼）。ここでは、回路Ａによって処理された結果が回路Ｂに入力され、回路Ｂによる処理結果が中間処理データとしてメモリに格納されていく（図１６－▲５▼、▲６▼）。

【００９４】

回路Ａの処理が終了した時点で、回路Ａが配置されていたプログラマブル論理回路２１内の領域は不要となる。そのため、回路Ｂの処理実行中に並行して、回路Ａが構成されていた領域も用いて、回路Ｃの再構成を開始する（図１６－▲７▼）。このような処理回路の再構成は、処理回路を配置可能であればいくつでも行ってよい。このようにして、図１５（Ｂ）に示すように、回路Ａが配置されていた領域も用いて回路Ｃの再構成が行われる。

【００９５】

回路Ｂにおける処理において処理データのＥＯＦを検出して処理の終了を検出するとともに、次の回路Ｃの再構成の終了を検出すると、回路Ｂから出力された中間処理データを用いて、回路Ｃにおける処理の実行が開始される（図１６－▲８▼、▲９▼）。このような処理が繰り返されて、処理ステップＭまでの処理を連続して行うことができる。

【００９６】

なお、上述の説明では、図１に示した構成をもとにして説明してきた。本発明の情報処理システムは図１に示す構成に限られることはなく、種々の変形が可能である。そのうちのいくつかの変形例については既に述べた。さらに極端な例として、例えば拡張ハードウェア部１９の部分のみの構成や、さらには外部入出力インタフェース２４も設けず、プログラマブル論理回路インタフェース２３を介してのみデータの入出力を行う構成などであってもよい。

【００９７】

【発明の効果】

以上の説明から明らかなように、本発明によれば、あらかじめ使用する回路と順番を規定できる場合には、メインプロセッサによる最初のデータ処理開始の制御だけで、あとは処理データ（例えば処理データのヘッダ情報）を解釈するだけで、プログラマブル論理回路の複数の機能回路によるデータ処理を順次実行してゆくことができる。このため、メインプロセッサによる制御の処理ステップを削減して高速化を図ることができるとともに、消費電力も低減することができる。

【００９８】

また、回路の再構成のための制御に対するメインプロセッサの負荷が軽減されることによって、その処理パワーを他の処理へ使用できるため、システム全体のパフォーマンスの向上を図ることができるという効果がある。

【図面の簡単な説明】

【図１】本発明の情報処理システムの実施の一形態を示す構成図である。

【図２】プログラマブル論理回路の一例を示す平面構造図である。

【図３】プログラマブル論理回路の内部構造の一例を示すブロック図である。

【図４】参照テーブルの一例の説明図である。

【図５】処理データおよび回路情報と、これらを格納するメモリアドレス空間の説明図である。

【図６】プログラマブル論理回路への処理回路の第１の再構成例の説明図である。

【図７】プログラマブル論理回路への処理回路の第１の再構成例における動作時の一例を示すタイミングチャートである。

【図８】プログラマブル論理回路への処理回路の第１の再構成例における動作の一例を示すフローチャートである。

【図９】プログラマブル論理回路への処理回路の第２の再構成例の説明図である。

【図１０】プログラマブル論理回路への処理回路の第２の再構成例における動作時の一例を示すタイミングチャートである。

【図１１】プログラマブル論理回路への処理回路の第２の再構成例における動作の一例を示すフローチャートである。

【図１２】プログラマブル論理回路への処理回路の第３の再構成例の説明図である。

【図１３】プログラマブル論理回路への処理回路の第３の再構成例における動作時の一例を示すタイミングチャートである。

【図１４】プログラマブル論理回路への処理回路の第３の再構成例における動作の一例を示すフローチャートである。

【図１５】プログラマブル論理回路への処理回路の第４の再構成例の説明図である。

【図１６】プログラマブル論理回路への処理回路の第４の再構成例における動作時の一例を示すタイミングチャートである。

【図１７】従来の情報処理システムの一例を示すブロック図である。

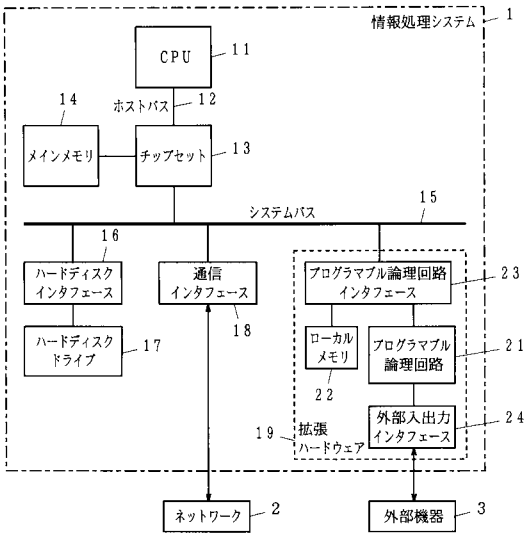
【図１８】ＤＰＧＡの論理セル構造の一例の説明図である。

【符号の説明】

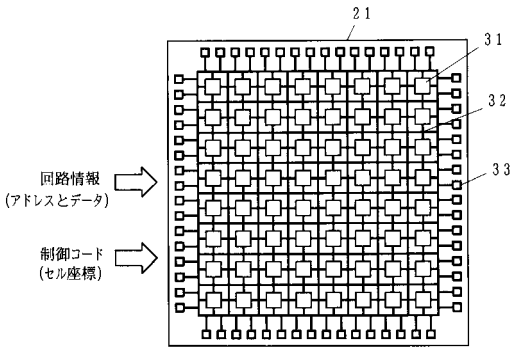
１…情報処理システム、２…ネットワーク、３…外部機器、１１…ＣＰＵ、１２…ホストバス、１３…チップセット、１４…メインメモリ、１５…システムバス、１６…ハードディスクインタフェース、１７…ハードディスクドライブ、１８…通信インタフェース、１９…拡張ハードウェア部、２１…プログラマブル論理回路、２２…ローカルメモリ、２３

…プログラマブル論理回路インタフェース、２４…外部入出力インタフェース、３１…論理セル、３２…配線領域、３３…入出力端子、４１…コンフィギュレーションメモリ、４２…回路素子、５１…アプリケーションサーバ、５２，５３…クライアントコンピュータ、５４…通信ネットワーク、５５…メインプロセッサ、５６…拡張ハードウェア、５７…アプリケーションプログラム、５８…拡張コード、５９…メインプロセッサコード、６０…ＯＳ、６１…コード選択機能、７１…ＤＲＡＭ、７２，７５…マルチプレクサ、７３…ルックアップテーブル、７４…フリップフロップ。

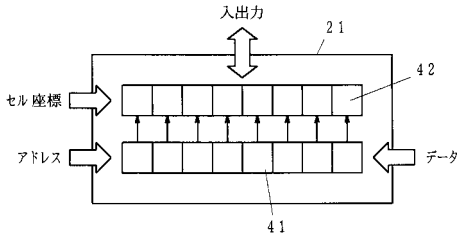
【図１】



【図２】



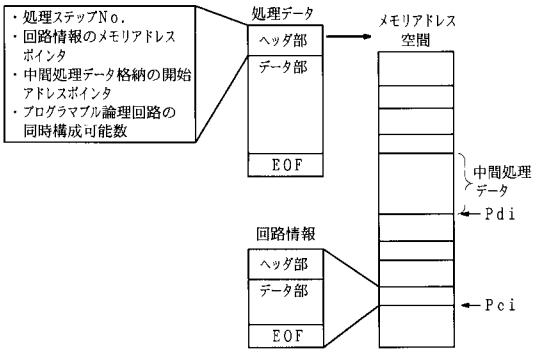
【図３】



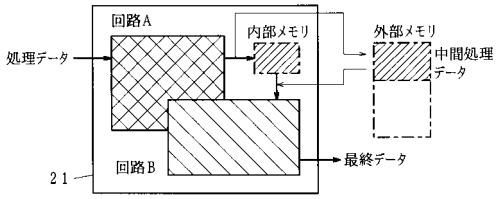
【図４】

処理ステップNo.	再構成する回路情報を格納するメモリの開始アドレスポインタ	中間処理データを格納するメモリの開始アドレスポインタ	プログラマブル論理回路への同時構成可能回路数
0	Pc0	Pd0	N0
1	Pc1	Pd1	N1
2	Pc2	Pd2	N2
…	…	…	…
i	Pci	Pdi	Ni
…	…	…	…
M	PcM	PdM	NM

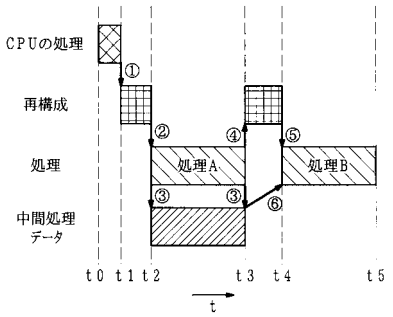
【図５】



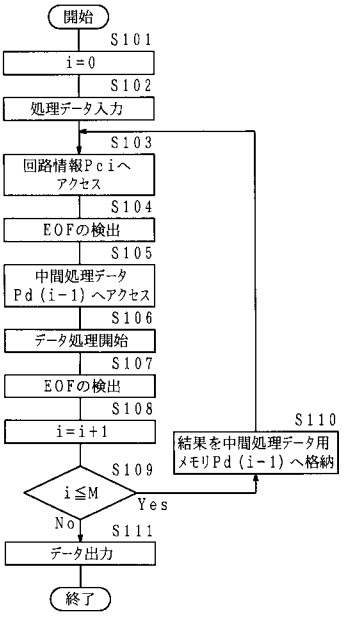
【図６】



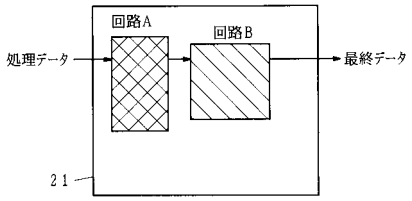
【図７】



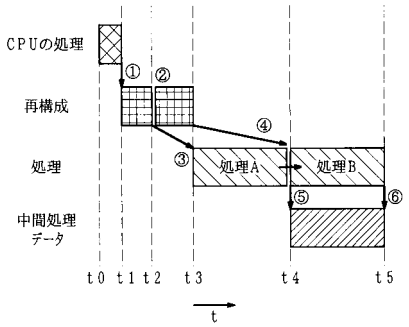
【図８】



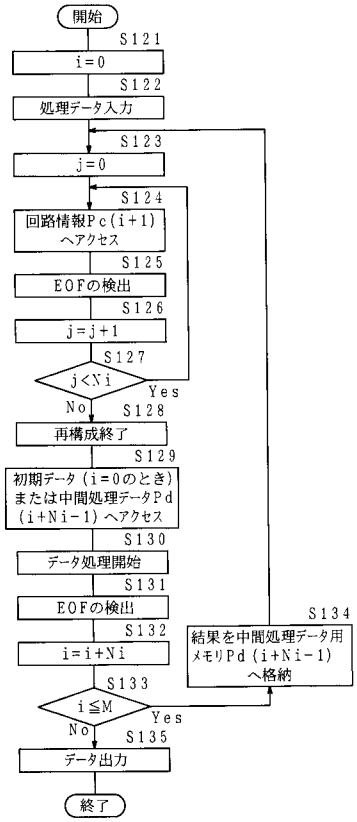
【図９】



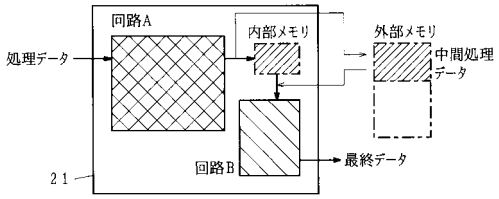
【図１０】



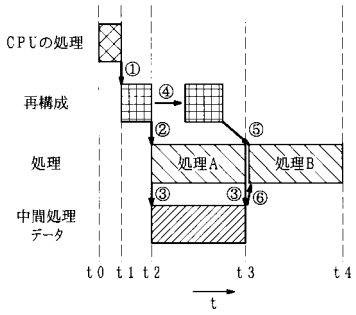
【図１１】



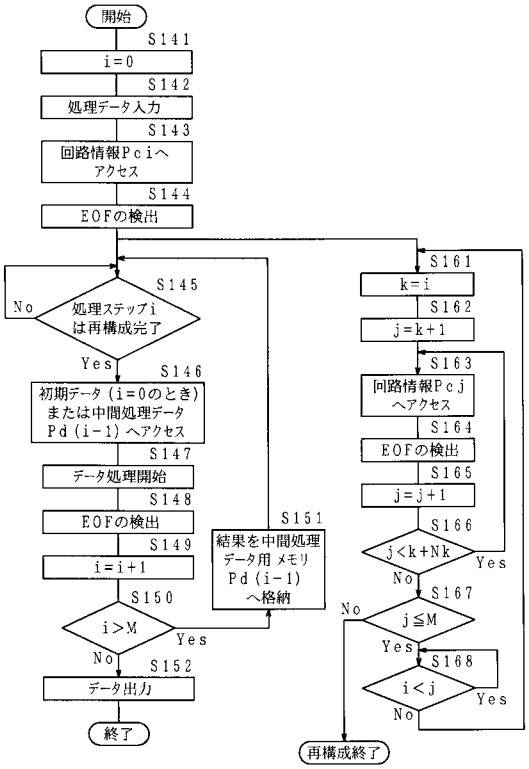
【図１２】



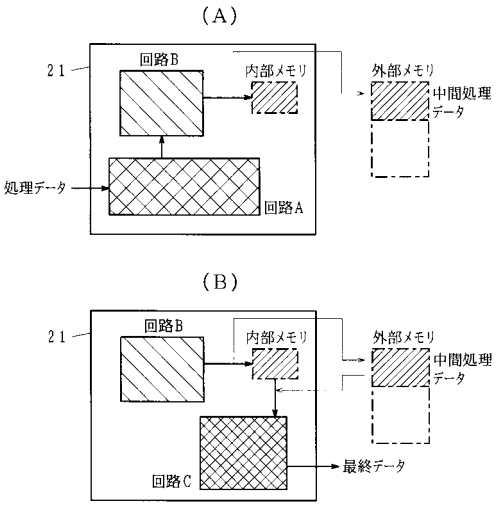
【図１３】



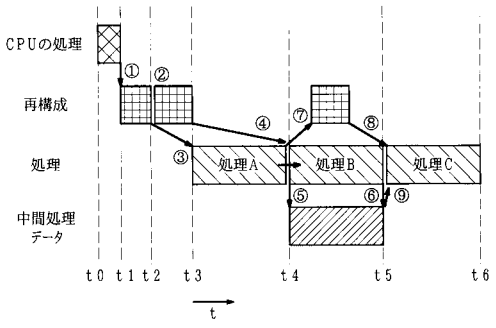
【図１４】



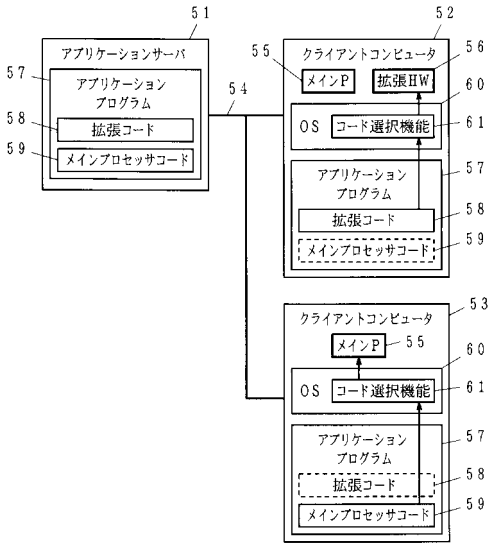
【図１５】



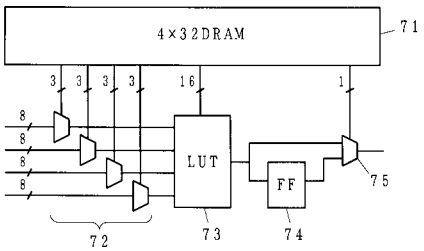
【図１６】


CPUの処理
再構成
処理
中間処理データ
①
②
③
④
⑤
⑥
⑦
⑧
⑨
処理A
処理B
処理C
t0 t1 t2 t3 t4 t5 t6
t

【図１７】


51
52
アプリケーションサーバ
アプリケーション
プログラム
拡張コード
メインプロセッサコード
57
58
59
54
55
クライアントコンピュータ
メインP
拡張HW
56
60
OS
コード選択機能
61
53

【図１８】


4×32DRAM
71
3
16
1
8
LUT
FF
75
72
73
74

フロントページの続き

(58)調査した分野(Int.Cl.[7], ＤＢ名)
H03K 19/173 101
H01L 21/82
H01L 21/82