(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5048972号
(P5048972)
(45)発行日　平成24年10月17日(2012.10.17)　(24)登録日　平成24年7月27日(2012.7.27)

(51)Int.Cl.　F I
G06F 12/16 (2006.01)　G06F 12/16 320M
G06F 12/16 310C
G06F 12/16 310D
G06F 12/16 320D

請求項の数 38　(全 16 頁)

(21)出願番号　特願2006-159772 (P2006-159772)
(22)出願日　平成18年6月8日 (2006.6.8)
(65)公開番号　特開2006-344223 (P2006-344223A)
(43)公開日　平成18年12月21日 (2006.12.21)
審査請求日　平成21年5月29日 (2009.5.29)
(31)優先権主張番号　60/688,980
(32)優先日　平成17年6月8日 (2005.6.8)
(33)優先権主張国　米国 (US)
(31)優先権主張番号　11/407,519
(32)優先日　平成18年4月19日 (2006.4.19)
(33)優先権主張国　米国 (US)

(73)特許権者　597154922
アルテラ　コーポレイション
Ａｌｔｅｒａ　Ｃｏｒｐｏｒａｔｉｏｎ
アメリカ合衆国　９５１３４　カリフォルニア州　サン　ホセ　イノベーション　ドライヴ　１０１
(74)代理人　100064621
弁理士　山川　政樹
(74)代理人　100098394
弁理士　山川　茂樹
(72)発明者　デイビッド・ルイス
カナダ国・エム５４　３ビイ６・オンタリオ・トロント・ホーランド　アベニュ・１８８

最終頁に続く

(54)【発明の名称】プログラマブル・デバイスの構成エラー検出の偽陽性の低減

(57)【特許請求の範囲】
【請求項１】
メモリに格納されたデータの保全性をテストする方法であって、
前記メモリから第１データを取り出すステップと、
エラー検出計算から前記第１データの少なくとも一部分を除外するように適合されるマスク・データを前記第１データに適用し、第２データを生成するステップと、
エラー検出結果を判定するために、前記第２データに対してエラー検出計算を実行するステップと、
前記第１データに少なくとも１つの重要なエラーが存在するか否かを判定するために、前記エラー検出結果を評価するステップと、
前記第１データに少なくとも１つの重要なエラーが存在するという前記判定に応答して、エラー信号を出力するステップと
をから構成され、
前記マスク・データの１ビットは前記第１データの複数ビットに対応し、前記第１データの複数ビットのうち少なくとも１ビットが構成データである場合、前記第１データの複数ビットは前記エラー検出計算から除外されないことを特徴とする方法。
【請求項２】
前記エラー検出計算は、少なくとも１つのソフト・エラーを検出するように適合されることを特徴とする請求項１に記載の方法。
【請求項３】

前記マスク・データは、前記第１データの前記除外される部分を、前記エラー検出計算に影響しない値に設定するように適合されることを特徴とする請求項１に記載の方法。
【請求項４】
前記第１データの前記除外される部分は、前記メモリのうち有用なデータを格納するのに使用されない部分に対応することを特徴とする請求項１に記載の方法。
【請求項５】
前記メモリの前記部分は、プログラマブル・デバイスの未使用部分の構成データを格納するように適合されることを特徴とする請求項４に記載の方法。
【請求項６】
前記マスク・データは、複数のマスク・ビットを含み、マスク・ビットのそれぞれは、前記第１データの１ビットに対応することを特徴とする請求項１に記載の方法。
【請求項７】
前記マスク・データは複数のマスク・ビットを含み、少なくとも１つのマスク・ビットは、前記第１データの複数のビットに対応することを特徴とする請求項１に記載の方法。
【請求項８】
前記第１データの各複数のビットは同一のサイズを有することを特徴とする請求項７に記載の方法。
【請求項９】
マスク・ビットに関連する前記第１データの少なくとも１つの複数のビットは、プログラマブル・デバイスのリソースに関連する構成データに対応することを特徴とする請求項７に記載の方法。
【請求項１０】
マスク・データを適用するステップは、
マスク・データ・メモリからマスク・データを取り出すステップと、
前記マスク・データのビットと前記第１データの少なくとも１つの対応するビットとの間でブール演算を実行するステップと
を含むことを特徴とする請求項１に記載の方法。
【請求項１１】
前記ブール演算はＡＮＤ演算であることを特徴とする請求項１０に記載の方法。
【請求項１２】
前記マスク・データ・メモリは、前記メモリに含まれることを特徴とする請求項１０に記載の方法。
【請求項１３】
前記マスク・データ・メモリは、外部データ・バスを介してアクセス可能な別のメモリに含まれることを特徴とする請求項１０に記載の方法。
【請求項１４】
メモリに格納されたデータの保全性をテストする方法であって、
前記メモリから第１データを取り出すステップと、
エラー検出結果を判定するために、前記第１データに対してエラー検出計算を実行するステップと、
前記第１データに少なくとも１つのエラーが存在するかどうかを判定するために、前記エラー検出結果を評価するステップと、
前記第１データに少なくとも１つのエラーが存在するとの判定に応答して、
エラー検出計算から前記第１データの少なくとも一部分を除外するように適合されるマスク・データを前記第１データに適用し、第２データを生成するステップと、
エラー検出結果を判定するために、前記第２データに対してエラー検出計算を実行するステップと、
前記第１データに少なくとも１つの重要なエラーが存在するかどうかを判定するために、前記エラー検出結果を評価するステップと、
前記第１データに少なくとも１つの重要なエラーが存在する前記判定に応答して、エラ

ー信号を出力するステップと
を有することを特徴とする方法。
【請求項１５】
前記エラー検出計算は、少なくとも１つのソフト・エラーを検出するように適合されることを特徴とする請求項１４に記載の方法。
【請求項１６】
前記マスク・データは、前記第１データの前記除外される部分を、前記エラー検出計算に影響しない値に設定するように適合されることを特徴とする請求項１４に記載の方法。
【請求項１７】
前記第１データの前記除外される部分は、前記メモリのうちで有用なデータを格納するのに使用されない部分に対応することを特徴とする請求項１４に記載の方法。
【請求項１８】
前記メモリの前記部分は、プログラマブル・デバイスの未使用部分の構成データを格納するように適合されることを特徴とする請求項１７に記載の方法。
【請求項１９】
前記マスク・データは複数のマスク・ビットを含み、マスク・ビットのそれぞれは、前記第１データの１ビットに対応することを特徴とする請求項１４に記載の方法。
【請求項２０】
前記マスク・データは複数のマスク・ビットを含み、少なくとも１つのマスク・ビットは、前記第１データの複数のビットに対応することを特徴とする請求項１４に記載の方法。
【請求項２１】
前記第１データの各複数のビットは同一のサイズを有することを特徴とする請求項２０に記載の方法。
【請求項２２】
マスク・ビットに関連する前記第１データの少なくとも１つの複数のビットは、前記プログラマブル・デバイスのリソースに関連する構成データに対応することを特徴とする請求項２０に記載の方法。
【請求項２３】
マスク・データを適用するステップは、
マスク・データ・メモリからマスク・データを取り出すステップと、
前記マスク・データのビットと前記第１データの少なくとも１つの対応するビットとの間でブール演算を実行するステップと
を有することを特徴とする請求項１４に記載の方法。
【請求項２４】
前記ブール演算はＡＮＤ演算であることを特徴とする請求項２３に記載の方法。
【請求項２５】
前記マスク・データ・メモリは、前記メモリに含まれることを特徴とする請求項２３に記載の方法。
【請求項２６】
前記マスク・データ・メモリは、外部データ・バスを介してアクセス可能な別のメモリに含まれることを特徴とする請求項２３に記載の方法。
【請求項２７】
メモリに格納されたデータの保全性をテストするシステムであって、
制御ユニットと、
前記制御ユニットからの第１信号に応答してメモリから第１データを取り出すように適合された第１インターフェースと、
前記制御ユニットからの第２信号に応答して第２メモリからマスク・データを取り出すように適合された第２インターフェースと、
前記第１インターフェースと前記第２インターフェースに結合され、かつ前記マスク・

データを前記第１データに適用し、これによって第２データを作成するように適合された論理を含むマスキング・ユニットであって、エラー検出計算から前記第１データの少なくとも一部分を除外するように適合された論理を含む、マスキング・ユニットと、
エラー検出結果を判定するために前記第２データに対して前記エラー検出計算を実行するように適合された論理を含み、前記マスキング・ユニットに結合して巡回冗長検査を行うエラー検出計算ユニットと
を備え、
前記制御ユニットは、前記第１データに少なくとも１つの重要なエラーが存在するかどうかを判定するために前記エラー検出結果を評価するように適合された論理と、前記第１データに少なくとも１つの重要なエラーが存在する前記判定に応答してエラー信号を出力するように適合された論理とを含み、
前記マスク・データの１ビットは前記第１データの複数ビットに対応し、前記第１データの複数ビットのうち少なくとも１ビットが構成データである場合、前記第１データの複数ビットは前記エラー検出計算から除外されないことを特徴とするシステム。
【請求項２８】
前記マスキング・ユニットは、前記マスク・データに応答して、前記第１データの前記除外される部分を、前記エラー検出計算に影響しない値に設定するように適合されることを特徴とする請求項２７に記載のシステム。
【請求項２９】
前記第１データの前記除外される部分は、前記メモリのうちで有用なデータを格納するのに使用されない部分に対応することを特徴とする請求項２７に記載のシステム。
【請求項３０】
前記メモリの前記部分は、プログラマブル・デバイスの未使用部分の構成データを格納するように適合されることを特徴とする請求項２９に記載のシステム。
【請求項３１】
前記マスク・データは複数のマスク・ビットを含み、前記マスキング・ユニットは、前記第１データの対応するビットに各マスク・ビットを適用するように適合された論理を含むことを特徴とする請求項２７に記載のシステム。
【請求項３２】
前記マスク・データは複数のマスク・ビットを含み、前記マスキング・ユニットは、前記第１データの対応する複数のビットに前記複数のマスク・ビットのうちの少なくとも１つを適用するように適合された論理を含むことを特徴とする請求項２７に記載のシステム。
【請求項３３】
前記第１データの各対応する複数のビットは同一のサイズを有することを特徴とする請求項３２に記載のシステム。
【請求項３４】
前記第１データの少なくとも１つの対応する複数のビットは、プログラマブル・デバイスのリソースに関連する構成データに対応することを特徴とする請求項３２に記載のシステム。
【請求項３５】
前記マスキング・ユニットは、前記マスク・データのビットと前記第１データの少なくとも１つの対応するビットとの間でブール演算を実行するように適合された論理を含むことを特徴とする請求項１に記載の方法。
【請求項３６】
エラー検出計算を実行するステップは巡回冗長検査を実行するステップを有することを特徴とする請求項１記載の方法。
【請求項３７】
エラー検出計算を実行するステップは巡回冗長検査を実行するステップを有することを特徴とする請求項１４記載の方法。

【請求項３８】
前記マスク・データの１ビットは前記第１データの複数ビットに対応することを特徴とする請求項１４記載の方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、プログラマブル・デバイスの分野に関し、プログラマブル・デバイスの構成エラーを検出するシステムおよび方法に関する。
【背景技術】
【０００２】
ＦＰＧＡなどのプログラマブル・デバイスは、通常、論理動作を実行するのに論理ゲートおよび／またはルックアップ・テーブルのセット合せを使用する数千個のプログラマブル論理セルを含む。プログラマブル・デバイスはまた、加算器、積和演算(multiply and accumulate)回路、位相ロック・ループ、メモリなど、特定の論理演算に適合された特殊化された論理デバイスを有する複数の機能ブロックを含む。論理セルや機能ブロックは、構成可能スイッチング回路を用いて相互接続される。構成可能スイッチング回路は、論理セルと機能ブロックの間の接続を選択的にルーティングする。論理セル、機能ブロック、スイッチング回路のセット合せを構成させることによって、プログラマブル・デバイスを、事実上すべてのタイプの情報処理機能を実行するように適合させることができる。
【０００３】
ＦＰＧＡなどのプログラマブル・デバイスの機能は、通常、構成ＲＡＭセル（ＣＲＡＭ）または構成メモリのセットに格納された構成データによって制御される。ＣＲＡＭ内の構成データは、プログラマブル・デバイスの所期の機能を実現するようにプログラマブル・デバイスを構成するために使用される論理信号を供給する。通常、ＣＲＡＭ内のデータは、論理セルの機能を定義するルックアップ・テーブルの値と、入力、出力、論理セル、および機能ブロックの間で信号をルーティングするために構成可能スイッチング回路によって使用されるマルチプレクサや他のスイッチング・デバイスのための制御信号の値と、プログラマブル・デバイスやその類別された機能ブロック動作のモードと論理セルなど、プログラマブル・デバイスの構成の他の態様を指定する値とを含む。構成データのコピーは、通常、フラッシュ・メモリまたはＲＯＭなどの不揮発性メモリに格納され、この不揮発性メモリは、プログラマブル・デバイスと同一のチップ・パッケージ内にあるか、プログラマブル・メモリ・デバイスに接続された外部構成デバイス上にある。構成データのこのコピーは、所望の機能性を実現するようにプログラマブル・デバイスを構成するために、プログラマブル・デバイスのＣＲＡＭセルにロードされる。
【０００４】
製造プロセスの進歩によりＣＲＡＭセルの物理的寸法が小さくなるにつれて、ＣＲＡＭセルは、瞬間的「ソフト・エラー」をより受けやすくなる。ソフト・エラーは、アルファ粒子または宇宙線などのバックグラウンド放射線によって誘発され、ＣＲＡＭセルが「０」から「１」へまたはその逆に状態を瞬間的に変化する。プログラマブル・デバイスの機能が、ＣＲＡＭセルに格納されたデータによって決定されるので、ＣＲＡＭセルの状態の一つの変化であっても、プログラマブル・デバイスの機能を変更するかディスエーブルする可能性がある。さらに、プログラマブル・デバイスが、より複雑になり、その構成データを格納するのに追加のＣＲＡＭセルを必要とするようになるにつれて、ソフト・エラーの頻度も高まる。
【０００５】
ソフト・エラーに対する以前の解決策には、ＣＲＡＭから構成データを読み取り、対応するエラー訂正コードを生成する、プログラマブル・デバイス上のエラー検出回路が含まれる。エラーの検出時に、エラー検出回路は、通常、プログラマブル・デバイスにその構成データを再ロードさせ、正しい動作のために再構成させるエラー信号を送出する。もう１つの手法では、構成データに、エラー訂正コードを含めることもできる。エラー検出訂

正回路と共に、これらのエラー訂正コードを使用して、構成データのセット全体を再ロードせずにＣＲＡＭ内の構成データを訂正することができる。構成データの再ロード中または訂正中に、プログラマブル・デバイスは、その通常動作を一時停止する。

【０００６】

多くの応用例が、プログラマブル・デバイスのＣＲＡＭの大きい部分を効果的に使用しない。たとえば、通常の応用例は、その応用例によって使用される論理セル、機能ブロック、および／またはスイッチング回路の動作を構成するのに、プログラマブル・デバイスのＣＲＡＭの４０％しか使用しない。ＣＲＡＭの残りの部分は、ある論理値をセットされるが、プログラマブル・デバイスの機能に影響しない。ＣＲＡＭのこれらの未使用部分が、その応用例によって使用されない、論理セル、機能ブロック、および／またはスイッチング回路の動作を構成するからである。ＣＲＡＭのうちで応用例によって使用されない論理セル、機能ブロック、および／またはスイッチング回路を制御する部分を、ＣＲＡＭの未使用部分と呼ぶ。ＣＲＡＭの未使用部分でのソフト・エラーは、重要でなく、無視すべきである。

【０００７】

エラー検出回路は、ＣＲＡＭの使用されている部分と未使用部分を区別しないので、以前のプログラマブル・デバイスは、一般に、ＣＲＡＭ内のすべてのエラーの検出時に構成データを再ロードしていた。しかし、ＣＲＡＭの大きな部分が、プログラマブル・デバイスの応用例によって使用されない場合があるので、これらのソフト・エラーの多くが、プログラマブル・デバイスの機能性に影響しない「偽陽性(false positive)」である。したがって、プログラマブル・デバイスは、しばしば、偽陽性に起因して構成データを不必要に再ロードし、これが、構成データのロード中のダウン・タイムに起因してプログラマブル・デバイスの性能を低下させ、構成データの不必要なロードや格納によって電力消費を増やす。

【特許文献１】米国仮出願（１５１１４－０８０２００ＵＳ）

【特許文献２】米国特許出願（Ａｌｔｅｒａ　Ａ２０８５）

【発明の開示】

【発明が解決しようとする課題】

【０００８】

したがって、装置および方法が、プログラマブル・デバイスの不必要な再ロードと再構成を避けるために偽陽性ソフト・エラーを検出することが望ましい。

【課題を解決するための手段】

【０００９】

本発明の実施態様では、デバイスが、マスキング・ユニットと感度マスク・データを使用して、メモリの未使用部分をエラー検出計算から除外することによって、偽陽性メモリ・エラー検出を低減させる。デバイスは、メモリからデータを読み取り、データ保全性を検証するエラー検出ユニットを含む。感度マスク・データはメモリの未使用部分を示す。メモリの未使用部分は、プログラマブル・デバイスの未使用部分に関する構成データに対応するものである。感度マスク・データの各ビットは、メモリからのデータの１つまたは複数のビットの使用状態を示す。マスク・データに応答して、マスキング・ユニットが、メモリの未使用部分からのデータを、エラー検出計算の結果を変更しない値に設定する。これによって、メモリの未使用部分からのデータのすべてのエラーが、エラー信号を送出しなくなる。

【００１０】

一実施態様は、メモリに格納されたデータの保全性をテストする方法であり、メモリから第１データを取り出すステップと、第２データを生成するために前記第１データにマスク・データを適用するステップとを含む。前記マスク・データは、エラー検出計算から前記第１データの少なくとも一部分を除外するように適合される。本方法は、エラー検出結果を判定するために、前記第２データに対してエラー検出計算を実行するステップと、前記第１データに少なくとも１つの重要なエラーが存在するかどうかを判定するために、前

記エラー検出結果を評価するステップと、前記第１データに少なくとも１つの重要なエラーが存在する前記判定に応答して、エラー信号を出力するステップをも含む。

【００１１】

他の実施態様では、前記エラー検出計算が、少なくとも１つのソフト・エラーを検出するように適合される。さらなる実施態様では、前記マスク・データが、前記第１データの前記除外される部分を、前記エラー検出計算に影響しない値に設定するように適合される。もう１つの実施態様では、前記第１データの前記除外される部分が、前記メモリのうちで有用なデータを格納するのに使用されない部分に対応する。もう１つの実施態様では、前記メモリの前記部分が、プログラマブル・デバイスの未使用部分の構成データを格納するように適合される。

【００１２】

一実施態様では、前記マスク・データが複数のマスク・ビットを含み、マスク・ビットのそれぞれが前記第１データの１ビットに対応する。もう１つの実施態様では、前記マスク・データが複数のマスク・ビットを含み、マスク・ビットのそれぞれが前記第１データの複数のビットに対応する。諸実施態様では、前記第１データの各複数のビットが、同一のサイズを有し、かつ／または、前記第１データの複数のビットが、前記プログラマブル・デバイスのリソースに関連する構成データに対応する。

【発明を実施するための最良の形態】

【００１３】

本発明を、図面を参照して説明する。

【００１４】

図１に、本発明の実施形態によるプログラマブル・デバイスでソフト・エラーを検出するシステム１００を示す。システム１００は、構成デバイス１１０に接続されたプログラマブル・デバイス１０５を含む。プログラマブル・デバイス１０５は、このプログラマブル・デバイスの機能を定義する構成データを格納するのに使用される構成メモリ１２０を含む。構成デバイス１１０は、構成データのコピー１１３を不揮発性メモリに格納する。プログラマブル・デバイス１０５のアクティブ化またはリセットに続いて、構成デバイス１１０は、構成データのコピー１１３を、構成メモリ１２０への格納のためにプログラマブル・デバイス１０５に供給するようになっている。

【００１５】

ソフト・エラーの発生を検出するために、プログラマブル・デバイス１０５は、エラー検出ユニット１１５を含む。エラー検出ユニット１１５は、チェックサム値またはデータ保全性を検証するのに使用される他のタイプのコードを用いて初期化される。エラー検出ユニット１１５は、構成メモリ１２０から構成データのすべてを読み取り、巡回冗長検査（ＣＲＣ）または当業者に既知の他のエラー検出アルゴリズムなどで、エラー検出計算を実行する。エラー検出ユニット１１５が、構成メモリ１２０内の構成データのすべてに対するエラー検出計算を完了した後に、エラー検出ユニットは、この計算の結果の値をチェックサム値と比較する。これらの値が一致しない場合は、ソフト・エラーが発生しており、エラー検出ユニット１１５は、出力１２５にエラー・フラグを生成する。このエラー・フラグに応答して、プログラマブル・デバイスの実施形態は、構成デバイスから構成データを再ロードする。一実施形態では、エラー検出ユニット１１５は、ソフト・エラーについて連続的にスクリーニングするために、繰り返してエラー検出計算を繰り返す。

【００１６】

上述のように、プログラマブル・デバイス１０５の多くの構成が、構成メモリ１２０の大きい部分を使用しない。不必要にエラー・フラグを送出する、構成メモリ１２０内の未使用部分でのソフト・エラーである偽陽性の発生を防ぐために、エラー検出ユニット１１５の実施形態は、そのエラー検出計算に感度マスク・データを組み込む。感度マスク・データは、構成メモリ１２０のどの部分がプログラマブル・デバイスの現在の構成を指定するのに使用されているかを指定する。

【００１７】

一実施形態では、感度マスク・データ１１２が、構成デバイス１１０の一部に格納される。エラー検出ユニット１１５は、必要に応じて構成デバイス１１０から感度マスク・データ１１２を取り出す。エラー検出ユニット１１５は、感度マスク・データを使用して、構成メモリ１２０の未使用部分をそのエラー検出計算から除外する。その結果、構成メモリの未使用部分で発生するソフト・エラーは、エラー検出計算に影響せず、したがって、エラー・フラグを立てることはない。これによって、感度マスク・データの大きさに応じて、構成メモリ１２０の未使用部分でのソフト・エラーから生じる偽陽性の発生が実質的に減少するか除去される。

【００１８】

図２に、本発明の実施形態による、プログラマブル・デバイスでソフト・エラーを検出するシステムの詳細２００を示す。エラー検出ユニット２１５は、制御ユニット２２０、エラー検出コード・ジェネレータ２２５、マスカ(masker)ユニット２３０を含む。この実施形態では、エラー検出ユニットが、プログラマブル・デバイスの構成メモリ２３５に格納された構成データを継続的に読み取り、テストする。これを行うために、制御ユニット２２０は、構成メモリ２３５に格納された構成データの一部分を取り出すコマンドを発行する。制御ユニット２２０は、構成デバイス２１０に格納された感度マスク・データ２１２の一部も取り出す。代替実施形態では、感度マスク・データ２１２を構成メモリ２３５の部分に格納する。その実施形態では、感度マスク・データ２１２自体をも、他の構成データをマスクするためのエラー検出計算に含めなければならない。感度マスク・データ２１２のその部分は、プログラマブル・デバイスの構成メモリ２３５に格納された構成データのうちで取り出される部分に対応する。もう１つの実施形態では、感度マスク・データ２１２を、構成データ２１３と異なるメモリに格納することができる。この実施形態では、感度マスク・データ２１２を、構成データ２１３に使用されるものと異なる入出力ピンおよび／または異なるメモリ・インターフェースを使用してプログラマブル・デバイスに通信することができる。

【００１９】

構成データの取り出された部分と感度マスク・データ２１２の対応する部分は、マスカ・ユニット２３０に向けられる。マスカ・ユニット２３０は、感度マスク・データを使用して、エラー検出計算への構成メモリの未使用部分の寄与をマスク・オフするか除去する。マスカ・ユニット２３０の出力は、エラー検出コード・ジェネレータ２２５に向けられ、エラー検出コード・ジェネレータ２２５は、チェックサム・コードまたはＣＲＣコードなどのエラー検出計算を実行する。エラー検出コード・ジェネレータ２２５が、構成メモリ２３５内の構成データのすべてに関するエラー検出計算を完了したとき、得られるエラー訂正コードを制御ユニット２２０によって評価して、エラーが発生したかどうかを判定することができる。

【００２０】

一実施形態では、構成メモリ２３５の未使用部分に「０」の値がセットされる。構成メモリ２３５の使用部分には、プログラマブル・デバイスの所望の機能に応じて、「０」または「１」を設定される。感度マスク・データのうちで感度マスクの使用されている部分に対応する部分には、「１」がセットされ、感度マスク・データのうちで感度マスクの未使用部分に対応する部分には、「０」がセットされる。下で詳細に述べるように、感度マスク・データは、プログラマブル・デバイスの構成データの生成と共に生成することができる。

【００２１】

一実施形態では、マスカ・ユニット２３０は、取り出された構成データと対応する感度マスク・データとの間で、ＡＮＤなどのブール演算を適用する。取り出された構成データが、構成メモリ２３５の使用されている部分からのものである場合、対応する感度マスク・データは「１」になる。感度マスクからと、取り出された構成データとの「１」のブールＡＮＤは、構成データを未変更のままにする。マスカ・ユニット２３０は、得られるデータをエラー検出コード・ジェネレータ２２５に出力し、この場合、このデータは、未変

更の構成データである。

【００２２】

取り出された構成データが、構成メモリ２３５の未使用部分からのものである場合、対応する感度マスク・データは「０」になる。感度マスクからの「０」と取り出された構成データとのブールＡＮＤは、常に「０」となる。通常、構成メモリ２３５の未使用部分には、「０」がセットされる。したがって、構成メモリ２３５の未使用部分に関するマスカ・ユニット２３０からの結果の「０」データは、やはり「０」である。構成メモリ２３５の未使用部分でソフト・エラーが発生した場合、「０」が「１」に変化する。しかし、対応する感度マスク・データは、まだ「０」の値を有する。したがって、マスカ・ユニットの出力は、依然として「０」になる。

【００２３】

マスカ・ユニット２３０は、構成メモリ２３５の未使用部分からの構成データについて必ず「０」を出力するので、エラー検出計算の結果は、構成メモリ２３５の未使用部分でのソフト・エラーに関わりなく変更されない。したがって、構成メモリ２３５の未使用部分でのソフト・エラーからの偽陽性は実質的に減らされるか除去される。

【００２４】

図３Ａ～３Ｃに、本発明の実施形態による、プログラマブル・デバイスでソフト・エラーを検出する際に偽陽性の発生を低減させるのに使用されるマスク・データの例示の配置を示す。図３Ａに、構成メモリ３０５の一部と感度マスク３１０の対応する部分を示す。この例では、感度マスクの各ビットが、構成メモリの一つのビットに対応する。構成メモリ３０５のビット３１５が未使用の場合、感度マスク３１０の対応するビット３１７に「０」がセットされる。同様に、構成メモリ３０５のビット３２０が、プログラマブル・デバイスの構成を指定するのに使用されている場合、感度マスク３１０の対応するビット３２２に「１」がセットされる。

【００２５】

図３Ｂに、構成メモリ３２５の一部と感度マスク３３０の対応する部分を示す。この例では、感度マスクの各ビットが、構成メモリ３２５のビットの対に対応する。構成メモリ３２５のビット３３１、３３２が未使用の場合、感度マスク３３０の対応するビット３３３に「０」がセットされる。構成メモリ３２５のビット３３４、３３５が未使用の場合、感度マスク３３０の対応するビット３３６に「０」がセットされる。逆に、構成メモリ３２５のビット３３７、３３８がプログラマブル・デバイスの構成を指定するのに使用されている場合、感度マスク３３０の対応するビット３３９に「１」がセットされる。さらに、構成メモリのビットの対のうちの１つだけが使用されている場合、感度マスクの対応するビットに「１」をセットしなければならない。したがって、ビット３４０は未使用であるが、ビット３４１が使用されているので、感度マスク３３０のビット３４２には「１」がセットされる。図３Ｂの例のさらなる実施形態では、構成メモリ・ビットの１つまたは複数の行や構成メモリ・ビットの１つまたは複数の列の長方形など、構成メモリの２次元領域のビットに感度ビットが対応する。

【００２６】

図３Ｃに、構成メモリ３５０の一部と感度マスク３５５の対応する部分を示す。この実施形態では、感度マスクの各ビットが、特定の論理セル、機能ブロック、入出力ユニット、メモリ・ユニット、ＭＡＣユニット、またはマルチプレクサなど、プログラマブル・デバイスのリソースに対応する。プログラマブル・デバイス構成が、プログラマブル・デバイスの所与のリソースを使用しない場合、このリソースに対応する感度マスク・ビットに「０」がセットされる。この実施形態では、各感度マスク・ビットが、基礎になるプログラマブル・デバイス・リソースに関連するビット数に応じて、構成メモリ３５０の可変個数のビットに対応することができる。たとえば、論理セル３６５の構成を指定する、構成メモリ３５０の１つまたは複数のビット３６０が使用されている場合、感度マスク３５５のビット３６７に「１」がセットされる。逆に、マルチプレクサ３７５の構成を指定する１つまたは複数のビット３７０と、入出力ユニット３８５の構成を指定する１つまたは複

数のビット３８０は、未使用であり、したがって、感度マスク３５５のビット３７７、３８７には「０」がセットされる。

【００２７】

図４に、本発明の実施形態による、プログラマブル・デバイスの構成データ内でソフト・エラーを検出する方法４００を示す。ステップ４０５で、エラー検出ユニットを初期化する。ステップ４０５は、エラー検出コード値をリセットするステップと、正しいチェックサムまたはＣＲＣ値など、所望のエラー検出結果をエラー検出ユニットにロードするステップを含むことができる。

【００２８】

ステップ４１０で、感度マスク・データの一部分を取り出す。一実施形態では、ステップ４１０で、構成デバイスから感度マスク・データを取り出す。もう１つの実施形態では、感度マスク・データが、プログラマブル・デバイス自体の中で、構成メモリの一部またはプログラマブル・デバイスの別のメモリ・ユニットのいずれかに格納されている。もう１つの実施形態では、感度マスク・データが、プログラマブル・デバイスの外部の、ある他のメモリに格納されている。

【００２９】

ステップ４１５で、構成データのうちで、ステップ４１０で取り出された感度マスク・データに対応する部分を取り出す。ステップ４２０で、構成データのエラー評価を実行する。ステップ４２０の一実施形態では、上で説明したように、偽陽性の可能性を低減させるために、感度マスク・データを用いて、取り出された構成データをマスクする。ステップ４２０では、得られるマスクされた構成データに対してエラー検出計算を実行する。エラー検出計算の結果を、前に計算されたエラー検出動作と共に累積して、構成メモリ全体のエラー検出コードを判定する。

【００３０】

ステップ４２５で、構成メモリ内の構成データのすべてが評価されたかどうかを判定する。そうでない場合は、方法４００は、ステップ４１０に戻り、構成メモリの残りの部分についてステップ４１０から４２５を繰り返す。逆に、構成メモリ内の構成データのすべてを評価し終えた場合は、方法４００はステップ４３０に進む。ステップ４３０では、エラー検出計算の結果を評価する。一実施形態では、ステップ４３０で、エラー検出計算の結果を所望のエラー検出結果と比較する。これらの２つの値が一致しない場合は、ソフト・エラーが発生しており、エラー・フラグ信号を送る。ステップ４３０に続いて、方法４００を繰り返して、ソフト・エラーについて構成データを継続的に監視し、ソフト・エラーが検出された場合エラー・フラグを生成することができる。

【００３１】

方法４００の実施形態は、感度マスク・データに頻繁にアクセスする。感度マスク・データが、構成デバイス内など、プログラマブル・デバイスの外部に格納されている場合は、外部データ・バス・トラフィックの量に起因して、電力消費が増える場合がある。電力消費を低減させるために、本発明のもう１つの実施形態は、感度マスク・データ・アクセスの頻度を低減させる。

【００３２】

図５に、本発明のさらなる実施形態による、プログラマブル・デバイスの構成データ内でソフト・エラーを検出する方法５００を示す。ステップ５０５で、ステップ４０５に類似した形でエラー検出ユニットを初期化する。ステップ５１０で、プログラマブル・デバイスの構成メモリ内の構成データのすべてに対するエラー検出計算を実行する。ステップ５１０では、この計算を実行するのに感度マスク・データを全く使用しない。

【００３３】

ステップ５１５で、エラー検出計算の結果を評価する。構成データ内でエラーが検出されない場合は、実施形態の方法５００は、ステップ５０５に戻り、ソフト・エラーに関して構成メモリを監視する必要に応じて、ステップ５０５、５１０、５１５を繰り返す。

【００３４】

逆に、ステップ５１５で、ソフト・エラーが発生したと判定される場合は、方法５００はステップ５２０に進む。ステップ５２０は、前にステップ５１５で検出されたソフト・エラーが、構成メモリの使用されている部分または未使用部分のどちらで発生したかを判定するのに使用される。ステップ５２０では、感度マスク・データを使用して、プログラマブル・デバイスの構成メモリ内の構成データのすべてに対するエラー検出計算を実行する。一実施形態では、ステップ５２０は、方法４００に似た形で実行される。

【００３５】

ステップ５２０に続いて、ステップ５２５で、ステップ５２０の結果を評価して、ソフト・エラーが発生したかどうかを判定する。ステップ５２０の結果が、ソフト・エラーが発生したことを示す場合は、方法５００は、ステップ５３０に進んでエラー・フラグを発行し、プログラマブル・デバイスの構成データを再ロードしなければならないことを示す。ステップ５２０の結果がソフト・エラーを検出しない場合は、ステップ５１５によって検出されたソフト・エラーは、構成メモリの未使用部分で発生した。したがって、このエラーを無視することができ、方法５００は、構成データのさらなる監視のために、ステップ５２５からステップ５０５に進む。さらに、感度マスク・データが、潜在的なソフト・エラーの検出の後に限ってアクセスされるので、電力消費が減少する。

【００３６】

図６に、本発明の実施形態を実装するのに適する例示のプログラマブル・デバイスを示す。プログラマブル・デバイス６００は、論理アレイ・ブロック（ＬＡＢ）６０５、６１０、６１５などの複数のＬＡＢを含む。デバイス６００のＬＡＢは、行６３０、６３５、６４０、６４５に配置されるが、図６では原寸通りに図示されてはいない。各ＬＡＢは、論理動作を実行するために、データを格納し、取り出すのにレジスタだけでなく、論理ゲートを使用する複数のプログラマブル論理セルとルックアップ・テーブルのいずれかまたは双方を使用する。ＬＡＢ　６０５に、論理セル６２０、６２１、６２２、６２３、６２４、６２５、６２６、２７が詳細に示されている。論理セルは、図を明瞭にするために、図６の他のＬＡＢからは省略されている。一実施形態では、１つのＬＡＢ内の論理セルの配置と行内のＬＡＢの配置が、プログラマブル・スイッチング回路の構成可能接続の階層システムを提供する。ＬＡＢ内の論理セルの間の接続、同一行内の異なるＬＡＢ内のセルの間の接続、異なる行内のＬＡＢ内のセルの間の接続が必要とするリソースが徐々に多くなる。

【００３７】

ＬＡＢ内に配置された論理セルの他に、プログラマブル・デバイス６００は、積和演算ブロック（ＭＡＣ）６５５やランダム・アクセス・メモリ・ブロック（ＲＡＭ）６６０などの特殊化された機能ブロックをも含む。プログラマブル・デバイスの構成は、少なくとも一部には、構成メモリ６７５に格納された構成データによって指定される。構成データは、論理セルの機能を定義するルックアップ・テーブルの値と、入力、出力、論理セル、機能ブロックの間で信号をルーティングするために構成可能スイッチング回路によって使用されるマルチプレクサや他のスイッチング・デバイスのための制御信号の値と、プログラマブル・デバイスやその類別された機能ブロックの動作モードと論理セルなどのプログラマブル・デバイスの構成の他の態様を指定する値とを含むことができる。構成メモリ６７５は、図６ではモノリシック・ユニットとして図示されているが、一部のプログラマブル・デバイスでは、構成メモリ６７５が、プログラマブル・デバイス上のある範囲の位置に物理的に分散される。これらのタイプのプログラマブル・デバイスでは、構成メモリの諸部分が、プログラマブル・デバイスの、論理セル内、機能ブロック内、構成可能スイッチング回路内にあることが可能である。

【００３８】

図を明瞭にするために、プログラマブル・デバイス６００のうちで図６に示された部分には、少数の論理セル、ＬＡＢ、機能ブロックだけが含まれている。通常のプログラマブル・デバイスには、数千個または数万個のこれらの要素が含まれる。一部の実装では、上述のエラー検出ユニットを、専用の機能ブロックとして実装することができる。他の実施

形態では、エラー検出ユニットのすべてまたは一部分を、プログラマブル・デバイス６００の論理セルや他のプログラマブル・リソースを使用して実装することもできる。

【００３９】

図７に、本発明の実施形態と共に使用される構成データを作成するのに適する例示のコンパイル処理８００を示す。コンパイル処理８００は、ユーザ設計を、そのユーザ設計を実装するためにプログラマブル・デバイスを構成するように適合されたプログラマブル・デバイス構成に変換する。抽出フェーズ８０５では、たとえばハードウェア記述言語で表されたユーザ設計の記述を、レジスタ・トランスファ層記述に変換する。

【００４０】

合成フェーズ８１０では、ユーザ設計のレジスタ・トランスファ層記述を一組の論理ゲートに変換する。合成フェーズ８１０の実施形態では、同等な論理ゲートの間で選択して、上で説明したようにソフト・エラーに対する抵抗を改善することができる。テクノロジ・マッピング・フェーズ８１５では、一組の論理ゲートをアトムのセットに副分割する。このアトムは、プログラマブル・デバイスの論理セルまたは他の機能ブロックの機能と一致する論理ゲートのグループである。所与のユーザ設計を、そのユーザ設計を実装するのに使用されるプログラマブル・デバイスの基礎になるハードウェアに応じて、アトムの、任意の個数の異なるセットに変換することができる。

【００４１】

テクノロジ・マッピング・フェーズ８１５に続いて、クラスタ・フェーズ８２０では、関連するアトムを一緒にクラスタにグループ化する。配置フェーズ８２５では、アトムのクラスタをプログラマブル・デバイス上の位置に割り当てる。ルート・フェーズ８３０では、ユーザ設計を実装するアトムを接続するのに使用される、プログラマブル・デバイスの構成可能スイッチング回路の構成を決定する。

【００４２】

遅延アノテータ(delay annotator)フェーズ８３５では、プログラマブル・デバイスのタイミング・モデルを使用して、アトムのセットとそれらに関連する構成可能スイッチング回路内の接続に関する信号遅延を判定する。タイミング分析フェーズ８４０では、たとえばユーザ設計の一部が最大の信号遅延を有することを判定することによって、ユーザ設計を実装するときのプログラマブル・デバイスの最大動作速度を判定する。

【００４３】

アセンブラ・フェーズ８４５では、ユーザ設計を実装するのに使用される論理セルのそれぞれの構成と論理セルを接続するのに使用される構成可能スイッチング回路の構成を含む、ユーザ設計を実装するプログラマブル・デバイスの構成を指定する構成情報のセットを生成する。一実施形態では、アセンブラ・フェーズ８４５で、構成データに対応する感度マスク・データも生成する。この実施形態では、アセンブラ・フェーズで、構成メモリのうちでユーザ設計によって使用される部分を識別し、それ相応に感度マスク・データを設定する。アセンブラ・フェーズ８４５では、構成情報を構成ファイルに書き込むことができ、その後、この構成ファイルを使用して、ユーザ設計のインスタンスを実装するために１つまたは複数のプログラマブル・デバイスを構成することができる。

【００４４】

添付文書を読んだ後に、当業者は、さらなる実施形態を思い浮かべることができる。たとえば、本発明をプログラマブル・デバイスを参照して説明したが、本発明は、標準ＡＳＩＣ、ストラクチャードＡＳＩＣ、ゲート・アレイ、一般的なディジタル論理デバイスを含む、ソフト・エラーに対してデータ保全性を保つ必要があるすべてのタイプのディジタル・デバイスに同等に適用可能である。他の実施形態では、上で開示した発明のセット合せまたは副セット合せを有利に生成することができる。本アーキテクチャのブロック図と流れ図は、理解を容易にするためにグループ化されている。しかし、ブロックのセット合せ、新たなブロックの追加、ブロックの再配置などが、本発明の代替実施形態で企図されていることを理解されたい。

【００４５】

したがって、本明細書と図面は、制限的な意味ではなく例示的な意味で顧慮されなければならない。しかし、請求項に示された本発明の広義の趣旨および範囲から逸脱せずに、これに対してさまざまな修正および変更を加えることができることは明白であろう。

【図面の簡単な説明】

【００４６】

【図１】本発明の実施形態による、プログラマブル・デバイスでソフト・エラーを検出するシステムを示す図である。

【図２】本発明の実施形態による、プログラマブル・デバイスでソフト・エラーを検出するシステムの詳細を示す図である。

【図３Ａ】本発明の実施形態による、プログラマブル・デバイスでソフト・エラーを検出する際に偽陽性の発生を低減させるのに使用される感度マスク・データの例示の配置を示す図である。

【図３Ｂ】本発明の実施形態による、プログラマブル・デバイスでソフト・エラーを検出する際に偽陽性の発生を低減させるのに使用される感度マスク・データの例示の配置を示す図である。

【図３Ｃ】本発明の実施形態による、プログラマブル・デバイスでソフト・エラーを検出する際に偽陽性の発生を低減させるのに使用される感度マスク・データの例示の配置を示す図である。

【図４】本発明の実施形態による、プログラマブル・デバイスの構成データ内でソフト・エラーを検出する方法を示す図である。

【図５】本発明のもう１つの実施形態による、プログラマブル・デバイスの構成データ内でソフト・エラーを検出する方法を示す図である。

【図６】本発明の実施形態を実装するのに適する例示のプログラマブル・デバイスを示す図である。

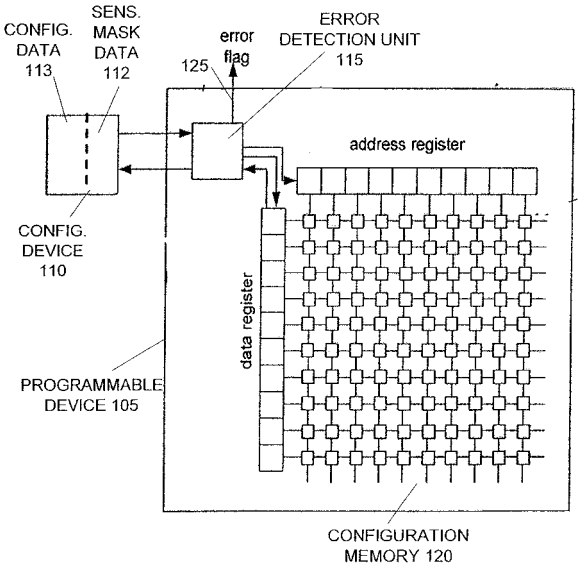
【図７】本発明の実施形態と共に使用される構成データおよびマスク・データを作成するのに使用される例示のコンパイル処理を示す図である。

【符号の説明】

【００４７】

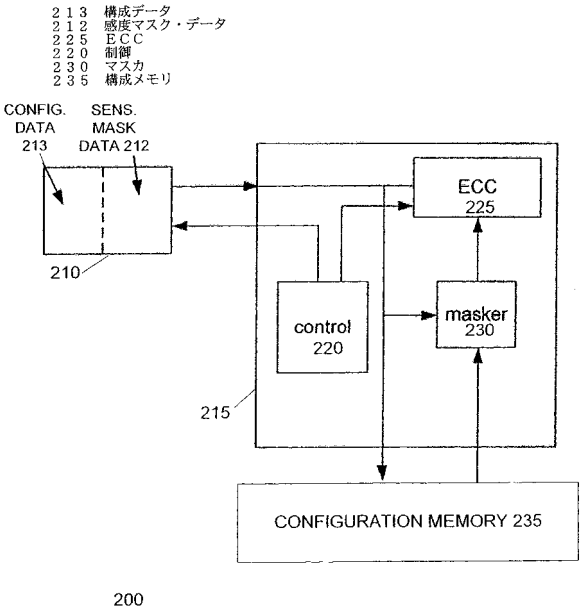
２００　システムの詳細、２１０　構成デバイス、２１２　感度マスク・データ、２１３　構成データ、２１５　エラー検出ユニット、２２０　制御ユニット、２２５　エラー検出コード・ジェネレータ、２３０　マスカ・ユニット、２３５　構成メモリ

【図１】

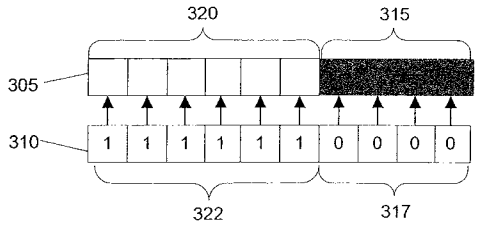


１１３　構成データ
１１２　感度マスク・データ
１２５　エラー・フラグ
１１５　エラー検出ユニット
address register:　アドレス・レジスタ
data register:　データ・レジスタ
１１０　構成デバイス
１０５　プログラマブル・デバイス
１２０　構成メモリ

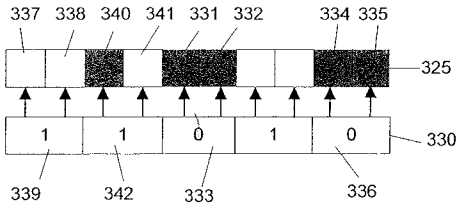
【図２】



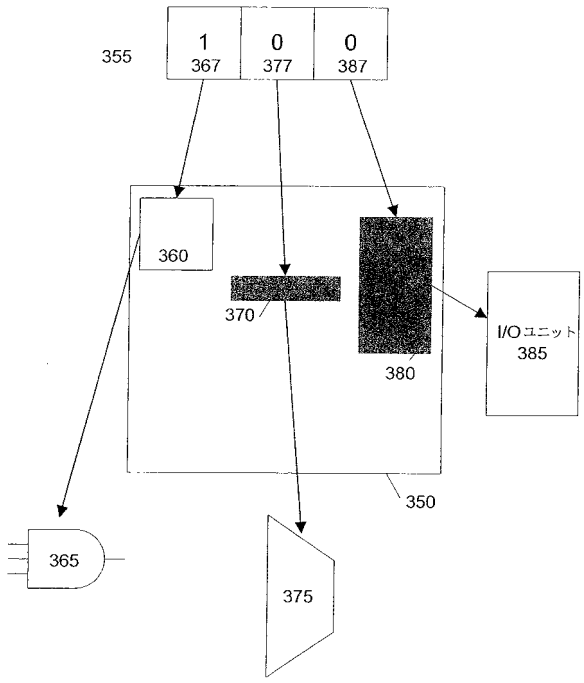
【図３Ａ】



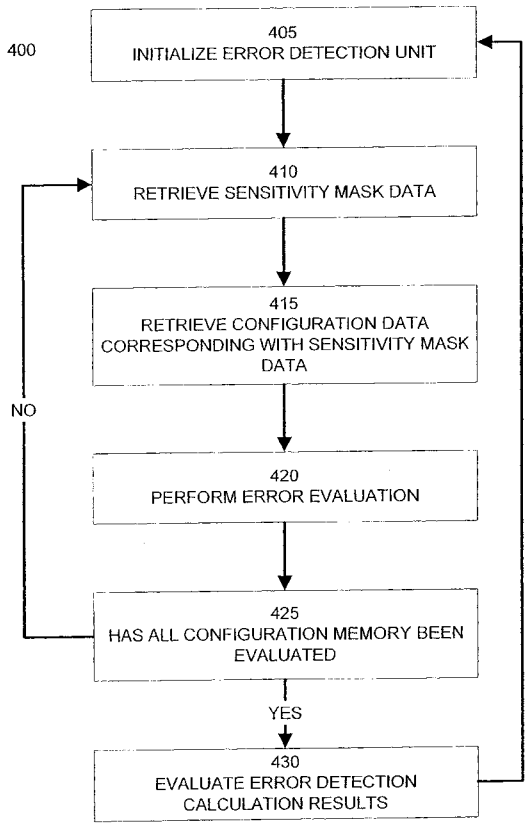
【図３Ｂ】



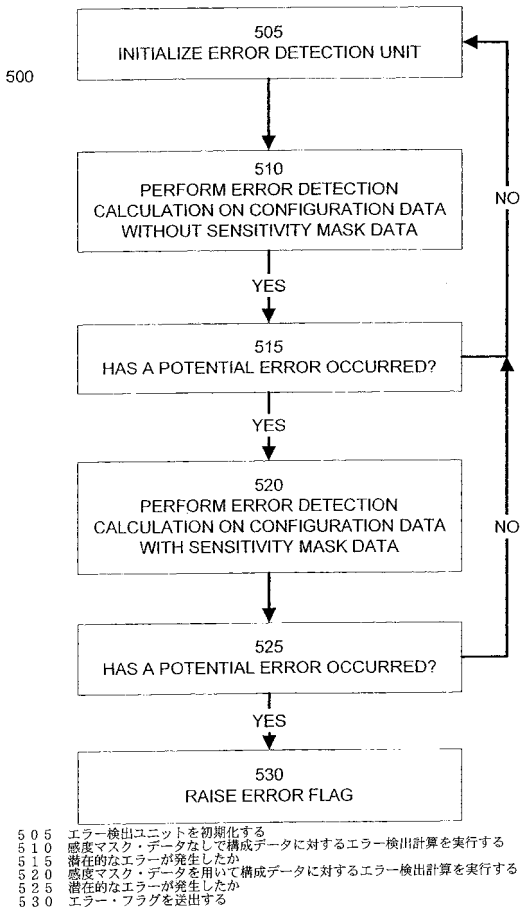
【図３Ｃ】



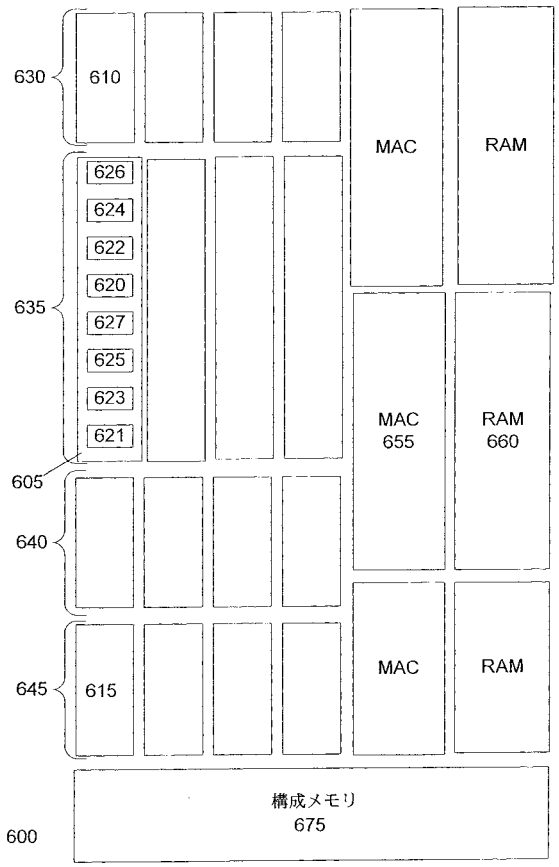
【図４】


400
405
INITIALIZE ERROR DETECTION UNIT
410
RETRIEVE SENSITIVITY MASK DATA
415
RETRIEVE CONFIGURATION DATA CORRESPONDING WITH SENSITIVITY MASK DATA
420
PERFORM ERROR EVALUATION
425
HAS ALL CONFIGURATION MEMORY BEEN EVALUATED
NO
YES
430
EVALUATE ERROR DETECTION CALCULATION RESULTS
４０５　エラー検出ユニットを初期化する
４１０　感度マスク・データを取り出す
４１５　感度マスク・データに対応する構成データを取り出す
４２０　エラー評価を実行する
４２５　構成メモリのすべてを評価したか
４３０　エラー検出計算結果を評価する

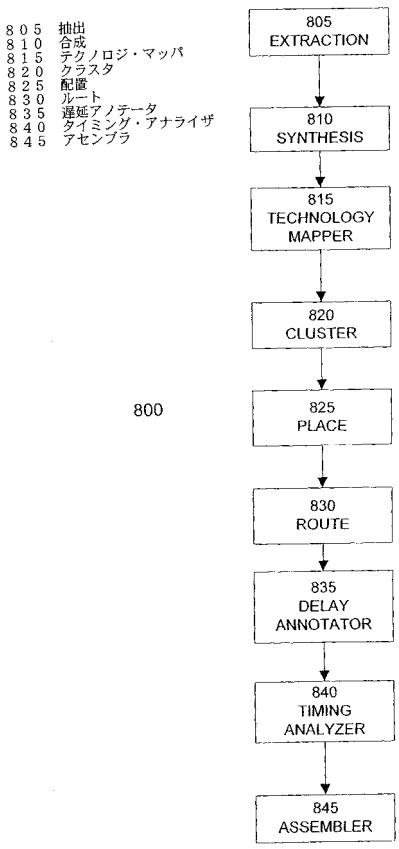
【図５】


500
505
INITIALIZE ERROR DETECTION UNIT
510
PERFORM ERROR DETECTION CALCULATION ON CONFIGURATION DATA WITHOUT SENSITIVITY MASK DATA
NO
YES
515
HAS A POTENTIAL ERROR OCCURRED?
YES
520
PERFORM ERROR DETECTION CALCULATION ON CONFIGURATION DATA WITH SENSITIVITY MASK DATA
NO
525
HAS A POTENTIAL ERROR OCCURRED?
YES
530
RAISE ERROR FLAG
５０５　エラー検出ユニットを初期化する
５１０　感度マスク・データなしで構成データに対するエラー検出計算を実行する
５１５　潜在的なエラーが発生したか
５２０　感度マスク・データを用いて構成データに対するエラー検出計算を実行する
５２５　潜在的なエラーが発生したか
５３０　エラー・フラグを送出する

【図６】


630
610
635
626
624
622
620
627
625
623
621
605
640
645
615
MAC
RAM
MAC
655
RAM
660
MAC
RAM
構成メモリ
675
600

【図７】


８０５　抽出
８１０　合成
８１５　テクノロジ・マッパ
８２０　クラスタ
８２５　配置
８３０　ルート
８３５　遅延アノテータ
８４０　タイミング・アナライザ
８４５　アセンブラ
800
805
EXTRACTION
810
SYNTHESIS
815
TECHNOLOGY MAPPER
820
CLUSTER
825
PLACE
830
ROUTE
835
DELAY ANNOTATOR
840
TIMING ANALYZER
845
ASSEMBLER

フロントページの続き

(72)発明者　ロバート・ブレーク
アメリカ合衆国・９５０７０・カリフォルニア州・サラトガ・オールド　ウッド　ロード・１４３９４
(72)発明者　リチャード・ジイ・クリフ
アメリカ合衆国・９４０２２・カリフォルニア州・ロスアルトス・ミドルバリー　レーン・６
(72)発明者　スリニバス・ティ・レディ
アメリカ合衆国・９４５３９・カリフォルニア州・フレモント・カメリア　コート・２２８９

審査官　中野　裕二

(56)参考文献　米国特許第０６５５３５２３（ＵＳ，Ｂ１）
特開２００６－２２１３３４（ＪＰ，Ａ）
特開平１１－１６１５５９（ＪＰ，Ａ）
特開平１０－１４６４３７（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１２／１６