(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4097436号
(P4097436)
(45)発行日　平成20年6月11日(2008.6.11)　(24)登録日　平成20年3月21日(2008.3.21)

(51)Int.Cl.　F I
G06F 11/20 (2006.01)　G06F 11/20 310K

請求項の数 8 (全 23 頁)

(21)出願番号　特願2002-45800(P2002-45800)
(22)出願日　平成14年2月22日(2002.2.22)
(65)公開番号　特開2002-328815(P2002-328815A)
(43)公開日　平成14年11月15日(2002.11.15)
審査請求日　平成17年1月19日(2005.1.19)
(31)優先権主張番号　09/810,103
(32)優先日　平成13年3月15日(2001.3.15)
(33)優先権主張国　米国(US)

(73)特許権者　398038580
ヒューレット・パッカード・カンパニー
HEWLETT-PACKARD COMPANY
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000
(74)代理人　100081721
弁理士　岡田　次生
(74)代理人　100105393
弁理士　伏見　直哉
(74)代理人　100111969
弁理士　平野　ゆかり

最終頁に続く

(54)【発明の名称】冗長コントローラシステムからコントローラをオンライン除去する方法

(57)【特許請求の範囲】
【請求項１】
第１のメモリを有する第１のコントローラおよび第２のメモリを有する第２のコントローラを有する冗長コントローラシステムから、オンラインであるコントローラを除去するための方法であって、
早期検出信号に応じて、前記第１のコントローラが、前記冗長コントローラシステムから該第１のコントローラの除去が行われていることを検出するステップであって、該早期検出信号は、前記第１のコントローラの前記システムからの除去が行われている間にわたって発せられる、ステップと、
前記第１のコントローラおよび前記第２のコントローラの両方が、それぞれの前記第１および第２のメモリに対する未処理のメモリアクセスを完了させることを含むシャットダウンシーケンスを実行するステップと、
前記第１のコントローラが、前記第１のメモリを自己リフレッシュモードにすると共に、前記第２のコントローラが、前記第２のメモリを自己リフレッシュモードにするステップと、
前記第１のコントローラが、オフラインに留まって前記冗長コントローラシステムからの前記第１のコントローラの除去が完了するのを待つと共に、前記第２のコントローラが、オンラインになる処理を開始するステップと、
を含む方法。
【請求項２】

前記早期検出信号に応じて前記第１のコントローラの第１のプロセッサに割り込みがかけられると共に、前記第２のコントローラの第２のプロセッサに割り込みがかけられることによって前記シャットダウンシーケンスは実行される、
請求項１に記載の方法。
【請求項３】
前記第１のプロセッサに割り込みがかけられることにより、前記第１のプロセッサによる未処理のプロセッサタスクが終了されると共に、前記第２のプロセッサに割り込みがかけられることにより、前記第２のプロセッサによる未処理のプロセッサタスクが終了される、
請求項２に記載の方法。
【請求項４】
前記第１のメモリを前記自己リフレッシュモードにすることは、前記第１のメモリを、メモリの内容を維持するためのメモリコントローラからの外部リフレッシュサイクルを必要としないモードにすることを含む、
請求項１に記載の方法。
【請求項５】
前記第２のメモリを自己リフレッシュモードにすることは、前記第２のメモリを、メモリの内容を維持するためのメモリコントローラからの外部リフレッシュサイクルを必要としないモードにすることを含む、請求項１に記載の方法。
【請求項６】
前記第２のメモリの自己リフレッシュモードを終了させた後で、前記第２のコントローラが、オンラインになるプロセスを開始する、請求項１に記載の方法。
【請求項７】
前記オンラインになるプロセスは、前記第２のメモリの自己リフレッシュモードを終了させた後で、前記冗長コントローラシステムからの前記第１のコントローラの除去が完了する前に、直ちに開始されるように構成される、
請求項６に記載の方法。
【請求項８】
請求項１から７のいずれかに記載の方法を実行するように構成された冗長コントローラシステム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は一般に、冗長コントローラを使用した冗長コントローラシステムおよびデータ格納システムに関し、さらに詳細には、オンラインコントローラ除去システムを有する冗長コントローラデータ格納システムおよび方法に関する。
【０００２】
【従来の技術】
極めて信頼性の高い冗長データ格納システムを提供するために、多重コントローラシステムが使用されている。例えば、ハードディスクドライブ業界においては、多重コントローラシステムは、ディスクドライブの耐故障性およびディスクドライブ性能を向上させるために複数のディスクドライブを組み合わせて使用する、ＲＡＩＤ（ｒｅｄｕｎｄａｎｔ ａｒｒａｙ ｏｆ ｉｎｄｅｐｅｎｄｅｎｔ ｄｉｓｋｓの略）システムの一部として使用されている。動作時には、ＲＡＩＤシステムは、多重コントローラを使用することによって冗長性を持たせている。多重コントローラは、ユーザのデータを複数のハードディスク間でストライプしている。上記冗長アレイは、任意のコントローラから動作させることができる。多重コントローラが存在する場合、それらのコントローラを使用することにより性能が向上し、および／またはホストコンピュータシステムの接続ポート数が増加する。データにアクセスする場合、多重コントローラＲＡＩＤシステムにより、すべてのハードディスクが同時に作動し、速度および信頼性が大幅に向上する。

【０００３】
ＲＡＩＤシステムの構成は、様々なＲＡＩＤレベルによって定義される。その様々なＲＡＩＤレベルの範囲は、データのストライピング（各ファイルのデータブロックを複数のハードディスクに分散して格納する）を提供し、冗長性を持たせることなくディスクドライブの速度および性能を向上させるＬＥＶＥＬ０から始まっている。ＲＡＩＤ　ＬＥＶＥＬ１は、ディスクのミラリングを提供し、ミラーハードディスク対を介して、データの１００％の冗長度をもたらしている（すなわち、データの同一ブロックが２つのハードディスクに書き込まれる）。他のディスクドライブにおけるＲＡＩＤ　ＬＥＶＥＬでは、様々なデータストライピングおよびディスクミラリングを提供し、性能を向上させるためのエラー修正、耐故障性、効率および／またはコストを改善している。

【０００４】
ＲＡＩＤ　ＬＥＶＥＬ５は、データをブロックに区切り、それらをディスクドライブに渡ってストライプしている。データブロックからパリティブロックが計算され、ディスクに格納される。すべてのデータブロックおよびパリティブロックは、異なるディスク上に格納される（ストライプされる）。ディスクドライブのいずれの１つが故障しても、データブロックまたはパリティブロックの１つが失われるだけである。その場合、冗長アレイは、失われたブロックを数学的に再生成することができる。また、ＲＡＩＤ５は、データブロックおよびパリティブロックが格納されるディスクを循環させる（つまり、すべてのディスクは、そのディスク上にいくつかのパリティブロックを格納する）。ＲＡＩＤ　ＬＥＶＥＬ６は、上記ステップをさらに進め、異なる数学式を用いて２つの「パリティ」ブロックを計算している。これにより上記冗長アレイは、２つの故障ディスクドライブを許容することができ、すべてのデータを再生成することができる。

【０００５】
知られている多重コントローラシステムは、ミラー二重コントローラデータ格納システムを備えている。各コントローラはそれぞれ独自のメモリを含んでおり、そのほとんどは、「ミラーイメージ」すなわち他のメモリと同じ「メモリイメージ」である。ミラーメモリを二重コントローラで使用することにより、コントローラの１つまたはそのメモリが破損または失われた場合、速やかに復旧することができ、かつ、データの損失を防止することができる。メモリのミラーコピーがなければ、コントローラが突然故障した場合に、そのコントローラ上の重要なデータが失われてしまうであろう。

【０００６】
例えば、コントローラＡおよびコントローラＢを有するミラーメモリ二重コントローラシステムでは、ミラー読出しおよび書込みにより、コントローラＡのメモリが、コントローラＢのメモリの「ミラーイメージ」になる。損失すなわちコントローラＢが故障すると、すべてのシステムの動作は自動的にコントローラＡに切り換わり、コントローラＡがシステム全体をラン、すなわち動作させる。

【０００７】
コンピュータシステムのアプリケーション数の増加により、プロセッサ故障時間の極めて大きな制限を始めとする、非常に高度の信頼性が要求されている。例えば、知られているシステムでは、コントローラの総故障時間は、１年当たり５分未満であることが要求されている。通常、損失すなわちコントローラの１つが故障すると、関連するデータ格納システムの冗長性および信頼性を維持するために、直ちにコントローラを交換しなければならない。上記の要求により、通常、高度の信頼性および「動作可能時間」が要求されるシステムの場合、交換するコントローラを、オンライン挿入すなわち「ホット」挿入しなければならず、その間、他のコントローラ（例えば、コントローラＡ）は、動作状態を維持していなければならない。オペレーティングシステムは、交換コントローラが挿入されたことを自動的に認識する。

【０００８】
通常、多重コントローラシステムはホストに接続される。従ってホストシステムは、しばしばコントローラボードを交換することでかなりの長時間の間データ格納システムを停止

させたり、ホストシステムのタイムアウトを引き起こしたりしないよう要求する。
【０００９】
動作中のシステムに交換コントローラを挿入すると、交換コントローラが、動作中のシステムにおいて試験され加えられるまでの間、システム利用上の損失をしばしば引き起こす。ミラーメモリシステムの一部として交換コントローラが追加される場合、動作中のシステムに交換コントローラを追加することに関連して問題が増加する。
【００１０】
知られているミラーメモリ二重コントローラシステムでは、コントローラＡをシステム内で動作中に交換コントローラＢをホット挿入する場合、コントローラＡおよび交換コントローラＢの両方をリセットし、コントローラの各々に、プロセッサのサブシステムを自己診断させなければならない。
【００１１】
各コントローラは、自己の共用メモリシステムを試験し、ハードウェアが正常に動作していることを確認する。各コントローラは、その共用メモリの内容をチェックし、システムに対するメモリイメージの「有効」性を確認する。この例では、コントローラＡのみが、システムの有効メモリイメージを持つことになる。
【００１２】
次に、各コントローラは、互いのレビジョン、システムの最新ビュー、およびシステムが最後にアクティブであったときのシステム状態、に関する情報を交換する。これらの情報を共有した後、ファームウェアは、いずれのコントローラが有効メモリイメージを有しているかを決定する。この例の場合、コントローラＡが有効メモリイメージを有している。コントローラＡの共用メモリイメージは、コントローラＢにコピーされ検証されるが、そのためには、コントローラＡのプロセッサは、コントローラＡのすべての共用メモリを読み出し、コントローラＢのすべての共用メモリロケーションに書き込まなければならない。
【００１３】
次に、コピーオペレーションが成功したことを検証するために、両コントローラのメモリが読み出され、比較される。メモリシステムが大きい場合、上記のプロセスには数分の時間が必要である。最終構成ステップが実行されると、コントローラはオンラインになり、全動作状態になる。
【００１４】
上記プロセスの多くのステップは、実行するにあたり数十秒の時間が必要なことがある。コントローラＡの共用メモリイメージをコントローラＢにコピーし、検証するプロセスには、数分の時間が必要なことがある。ホット挿入に必要なこの延長期間の間に、ほとんどのホストコンピュータのオペレーティングシステムがタイムアウトすることになる。
【００１５】
【発明が解決しようとする課題】
システムの故障時間を短縮し、かつ、ホストコンピュータのオペレーティングシステムタイムアウトの原因になることのない冗長ミラーメモリ多重コントローラシステムに使用するためのホット挿入および／またはシステムおよび方法を持つことが望ましい。また、システムの故障時間またはホストのタイムアウトを最小にする、有効なコントローラリセット処理方法を持つことが望ましい。
【００１６】
【課題を解決するための手段】
本発明は、冗長コントローラを使用した多重コントローラシステムおよびデータ格納システムに関し、さらに詳細には、オンラインコントローラ除去システムを有する冗長コントローラデータ格納システムおよび方法に関する。
【００１７】
本発明の一実施形態によれば、冗長コントローラシステムからコントローラをオンライン除去する方法が提供される。上記冗長コントローラシステムは、第１のコントローラおよ

び第２のコントローラを備えており、上記方法には、冗長コントローラシステムからの、上記第１のコントローラの部分除去を検出するステップが含まれている。未処理のメモリアクセスの完了を含むシャットダウンシーケンスが、第１のコントローラおよび第２のコントローラに対して実行される。第１のコントローラは、自己リフレッシュモードに置かれる第１のメモリを有するように定義されている。冗長コントローラシステムからの第１のコントローラの除去は、上記第１のメモリによってモード化される自己リフレッシュが完了した後、終了する。

【００１８】

本発明の他の実施形態によれば、冗長コントローラシステムからコントローラをオンライン除去する方法が提供される。上記冗長コントローラシステムは、第１のコントローラおよび第２のコントローラを備えている。第１のコントローラは、第１のプロセッサを備えるように定義され、第２のコントローラは、第２のプロセッサを備えるように定義されている。冗長コントローラシステムからの上記第１のコントローラの部分除去が検出される。上記第１のコントローラおよび第２のコントローラの未処理のメモリアクセスを完了させるための上記第１のプロセッサへの割込みおよび第２のプロセッサへの割込みを含むシャットダウンシーケンスが、第１のコントローラおよび第２のコントローラに対して実行される。第１のコントローラは、第１のメモリを有するように定義され、第１のメモリを自己リフレッシュモードにしている。冗長コントローラシステムからの第１のコントローラの除去は、上記第１のメモリによる自己リフレッシュモードが完了した後、終了する。

【００１９】

本発明の他の実施形態によれば、冗長コントローラをオンライン除去するために構成された冗長コントローラシステムが提供される。システムは、第１のメモリを含む第１のコントローラ、第１のプロセッサ、および上記第１のコントローラの部分除去を早期検出するためのシステムを備えており、第１のコントローラの部分除去が検出されると、第１のコントローラは、上記第１のメモリに対して未処理になっているメモリアクセスの完了を含むシャットダウンシーケンスを実行する。第２のコントローラは、第２のメモリおよび第２のプロセッサを備えており、上記第１のコントローラの部分除去が検出されると、上記第１のメモリに対して未処理になっているメモリアクセスの完了を含むシャットダウンモードを実行する。第１のコントローラおよび第２のコントローラによるシャットダウンシーケンスが完了すると、上記第２のメモリが自己リフレッシュモードに置かれ、冗長コントローラシステムからの第１のコントローラの除去が完了する。

【００２０】

【発明の実施の形態】

本発明の好ましい実施形態について、本発明を実践することができる特定の実施形態の説明用として示す、本明細書の一部を形成する添付の図面に照らして、以下に詳細に説明する。本発明の範囲を逸脱することなく他の実施形態を利用し、また、構造あるいはロジックを変更することができることを理解しなければならない。したがって以下の詳細説明を、本発明を制限するものとして捕えてはならない。本発明の範囲については、特許請求の範囲の各クレームによって定義されている。

【００２１】

図１は、本発明による冗長コントローラデータ格納システムの一例示的実施形態を、符号３０で総括して示したものである。冗長コントローラデータ格納システムは、冗長ミラーメモリ、オンラインすなわち「ホット」挿入システムを有する多重コントローラシステム、及びシステム故障時間を短縮しかつコントローラの交換中にホストコンピュータのオペレーティングシステムがタイムアウトしないための方法を提供している。本発明の一態様では、冗長コントローラデータ格納システム３０は、二重コントローラシステムである。本明細書において説明する例示的実施形態は、二重コントローラシステムを採用しているが、これらの実施形態は、他の多重コントローラ環境（つまり、３つ以上のコントローラを有するシステム）にも適用することができる。

【００２２】

本発明によるコンポーネントは、マイクロプロセッサ、プログラマブルロジックすなわち状態マシン、およびファームウェアを介してハードウェアで実施することができ、あるいは所定の装置内のソフトウェアで実施することもできる。好ましい一実施形態では、ソフトウェア中に本発明による１つまたは複数のコンポーネントが存在し、ハードウェアを介して使用されている。また、本発明によるコンポーネントは、１つまたは複数のコンピュータ可読媒体のソフトウェア中に存在させることもできる。本明細書で使用されているコンピュータ可読媒体という用語は、フロッピーディスク、ハードディスク、ＣＤ－ＲＯＭ、フラッシュメモリ、読出し専用メモリ（ＲＯＭ）、およびランダムアクセスメモリ（ＲＡＭ）など、任意の種類の揮発性メモリまたは不揮発性メモリを含むものとして定義されている。また、本発明によるシステムは、注文製作装置ハードウェアおよび／または専用単一目的ハードウェアが組み込まれた、マイクロプロセッサ埋込みシステム／装置を使用することができる。

【００２３】

一例示的実施形態では、システム３０は、第１のコントローラ３２および第２のコントローラ３４を有する冗長ミラーコントローラデータ格納システムである。第１のコントローラ３２および第２のコントローラ３４は、（例えば）ディスクアレイなど、データ格納システム３６へのデータの冗長又はミラーによる読出し及び書込み用に構成されているが、このデータ格納システム３６は通信バス３８を介しており、（例えば）ＲＡＩＤ　ＬＥＶＥＬ１においてはここで、ミラー書込みや、アレイがコピーからの読み出しを行うモードを含み、ＲＡＩＤ　ＬＥＶＥＬ５または６の場合、ユーザのアクセスは、上記ディスクアレイ間でストライプされる。さらに、第１のコントローラ３２および第２のコントローラ３４は、通信バス４０を介して互いに通信している。第１のコントローラ３２および第２のコントローラ３４は、通信バスプロトコルを使用してデータ格納システム３６と通信し、かつ、相互に通信している。一態様では、上記通信バスプロトコルは標準プロトコルである。他の適切な通信バスプロトコルは、本出願から当分野の技術者には明らかになるであろう。データ格納システム３６は、磁気ハードディスクデータ格納システムを備えている。他の態様では、データ格納システム３６は、フラッシュメモリ、ランダムアクセスメモリ（ＲＡＭ）、ＣＤ書込み可能媒体、光磁気媒体等、他の読出し／書込み可能データ格納媒体を備えている。

【００２４】

冗長コントローラデータ格納システムは、ホストまたはコントロールシステムインタフェース４２を介して、ホストまたはコントロールシステムと通信するように構成されている。ホストまたはコントロールシステムは、サーバであり、コンピュータネットワークであり、中央計算機であり、あるいは他のコントロールシステムである。一態様では、冗長コントローラデータ格納システム３０は、ホストとインタフェースし、ＲＡＩＤシステム（例えばＲＡＩＤ　ＬＥＶＥＬ０、ＲＡＩＤ　ＬＥＶＥＬ１、ＲＡＩＤ　ＬＥＶＥＬ２、ＲＡＩＤ　ＬＥＶＥＬ３、ＲＡＩＤ　ＬＥＶＥＬ４、ＲＡＩＤ　ＬＥＶＥＬ５、またはＲＡＩＤ　ＬＥＶＥＬ６システム）として動作するように構成される。

【００２５】

一実施形態では、第１のコントローラ３２は、「ミラーされた」メモリ５０、タスクプロセッサ５２、およびシステムオペレーションプロセッサ５４を備えている。同様に、第２のコントローラ３４も、「ミラーされた」メモリ５６、タスクプロセッサ５８、およびシステムオペレーションプロセッサ６０を備えている。第１のコントローラ３２および第２のコントローラ３４は、メモリ５０およびメモリ５６をミラーメモリシステムの一部として動作させる「メモリコントローラ」を備えている。本明細書で使用される「ミラーメモリ」という用語は、１つのメモリのメモリイメージが他のメモリに複製すなわち「ミラーされている」システムを含むものとして定義されている。本発明においては、第１のコントローラ３２のメモリ５０は、第２のコントローラ３４のメモリ５６に複製すなわち「ミラーされている」。二重コントローラミラーメモリシステムは、冗長コントローラシステム３０に故障許容環境を提供しており、コントローラの一方、あるいはコントローラメモ

リシステムの一方が故障した場合に、もう一方のコントローラおよびそのミラーメモリが存在することにより、中断することなく故障が復旧され、システムコマンドの処理が継続される。また、コントローラの一方を除去し、かつ、挿入する場合、本発明により、もう一方のコントローラを介してオペレーティングシステムが維持され、ホストがタイムアウトする期間以内にシステム故障時間が短縮される。ミラーメモリ二重コントローラディスク格納システムの一例示的実施形態が、本出願人に譲渡された、１９９７年１２月１６日発行の米国特許第５，６９９，５１０号に開示されている。また、同じくこの出願の出願人に譲渡された、１９９９年７月２７日発行の米国特許出願第５，９２８，３６７号に、他のミラーメモリ二重コントローラディスク格納システムが開示されている。

【００２６】

本発明による冗長コントローラデータ格納システム３０では、各コントローラ３２および３４は、それぞれ独自のメモリ５０および５６を備えており、メモリは「ミラーイメージ」、すなわち上で示したように、他のメモリと同じ「ミラーイメージ」を有している。ミラーメモリは、一方のコントローラまたはそのメモリが故障すなわち損失した場合における迅速な復旧を可能にしている。一態様では、ミラー読出しおよびミラー書込みにより、第１のコントローラ３２のメモリ５０が、第２のコントローラ３４のメモリ５６の「ミラーイメージ」になる。第２のコントローラ３４が損失すなわち故障すると、すべてのシステムオペレーションは、第１のコントローラ３２に自動的に切り換えられ、システムに他のコントローラが挿入されるまでの間、単一コントローラシステムとして第１のコントローラ３２がシステム全体をラン、すなわち動作させる。

【００２７】

本発明による冗長コントローラデータ格納システム３０により、一方のコントローラのホット挿入時、すなわちオンライン挿入時における冗長コントローラシステムの継続動作が提供される。例えば、第２のコントローラ３４が損失すなわち故障すると、冗長コントローラシステムは、第１のコントローラ３２を介して動作する。第２のコントローラ３４は、システム３０にオンライン挿入すなわちホット挿入することができる。詳細には、システムに第２のコントローラ３４をもたらすためのホット挿入プロセスの間、システムオペレーションプロセッサ５４が、データ格納システム３６に対する、メモリ５０を介したデータ読出しおよび書込みなどのシステムオペレーションコマンドの処理を継続する。タスクプロセッサ５２は、システムオペレーションプロセッサ５４がシステムオペレーションコマンドを処理している間、冗長コントローラデータ格納システム３０を遅延させることなく、バックグラウンドタスクを処理する。

【００２８】

好ましい一実施形態では、タスクプロセッサ５２は、システムオペレーションプロセッサ５４がシステムオペレーションコマンドの処理を継続している間、第１のミラーメモリ５０のメモリイメージを第２のメモリ５６にコピーするように動作している。したがって、第２のコントローラ３４を冗長コントローラシステム３０にホット挿入することにより、システムオペレーションコマンドの処理が遅延することはなく、および／または、ホストシステムインタフェース４２を介したホストシステムがタイムアウトすることもない。一例示的実施形態では、タスクプロセッサ５２は、システムオペレーションプロセッサ５４またはた他のシステムプロセッサを直接煩わせることなく、専用データ処理ハードウェアを介してバックグラウンドタスクを実行している。一態様では、上記データ処理ハードウェアは、専用集積回路（ＡＳＩＣ）の一部として、インテリジェントＤＭＡエンジンに結合されている。タスクプロセッサ５２は、第１のコントローラ３２がシステムオペレーションプロセッサ５４を介して動作を継続している間、特定のバックグラウンドタスクを処理する能力を有している。一態様では、タスクプロセッサ５２は、メモリ間コピータスク、メモリ自己診断、およびその他のタスクを実行している。

【００２９】

図２は、一括して８０で示される、本発明による冗長コントローラデータ格納システムにコントローラをホット挿入する方法の一例示的実施形態を示したものである。上記方法に

は、第１のメモリ、タスクプロセッサ、およびシステムオペレーションプロセッサを備えるように第１のコントローラを構成するステップが含まれている。上記第１のメモリは、第１のメモリイメージを含んでいる。図に示す一例示的実施形態では、第１のコントローラ３２は、メモリ５０、タスクプロセッサ５２、およびシステムオペレーションプロセッサ５４を備えるように構成されている。冗長コントローラシステム３０は、８４に示すように、単一コントローラシステムとして、第１のコントローラ３２を介して動作する。８５で、システムオペレーションコマンドが、システムオペレーションプロセッサ５４を介して処理される。８６で、冗長コントローラシステム３０に第２のコントローラ３４が挿入される。第２のコントローラ３４は、第２のメモリ５６を備えている。８８で、第１のコントローラを介してシステムオペレーションコマンドが処理されている間に、タスクプロセッサ５２を用いてバックグラウンドタスクが処理される。上記バックグラウンドタスクには、メモリ５０の第１のイメージの、第２のメモリ５６へのコピーが含まれている。

【００３０】

図３は、一括して９０で示される、本発明による冗長コントローラデータ格納システムにコントローラをホット挿入する方法の他の例示的実施形態を示したものである。上記方法には、一括して９２で示される、第１のメモリ５０、タスクプロセッサ５２、およびシステムオペレーションプロセッサ５４を備えるように第１のコントローラ３２を構成するステップが含まれている。第１のメモリ５０は、第１のメモリイメージを含んでいる。９４で、冗長コントローラシステム３０は、第１のコントローラ３２を介して動作する。９６で、システムオペレーションコマンドが、システムオペレーションプロセッサ５４を介して処理される。９８で、冗長コントローラシステム３０に第２のコントローラ３４が挿入される。第２のコントローラ３４は、第２のメモリ５６を備えている。１００で、第１のメモリ５０は、第２のメモリ５６へのミラー書込み用、及び共用すなわちミラーメモリ５０に対するローカル読取り専用として構成される。従って第１のコントローラ３２は、その独自のメモリイメージを読み出すために動作することができるが、第２のコントローラ３４が、冗長コントローラデータ格納システム３０内で完全に動作状態になるまで（すなわち自己診断が終了し、オンラインになるまで）、ミラー書込みおよびミラー読出しとして動作することはない。１０２で、第１のコントローラ３２を介してシステムオペレーションコマンドが処理されている間に、バックグラウンドタスクが処理される。バックグラウンドタスクは、タスクプロセッサ５２を使用して処理される。バックグラウンドタスクには、メモリ５０の第１のイメージの、第２のメモリ５６へのコピーが含まれている。

【００３１】

図４は、本発明による冗長コントローラデータ格納システムの他の例示的実施形態を、一括して１１０で示したものである。冗長コントローラデータ格納システム１１０は、本明細書において既に説明した冗長コントローラデータ格納システム３０と類似している。冗長コントローラデータ格納システム１１０は、ホストシステムのタイムアウトの原因になり得る、システムオペレーティングコマンドの処理に対するあらゆる割込みを最小化する、冗長コントローラデータ格納システムへのコントローラのホット挿入システムおよび方法を備えている。

【００３２】

冗長コントローラデータ格納システム１１０は、第１の冗長コントローラ１１２および第２の冗長コントローラ１１４を備えている。第１のコントローラ１１２は、第１のミラーメモリ１２０、第１のメモリコントローラ１２２、および第１のシステムオペレーションプロセッサ１２４を備えている。一態様では、第１のコントローラ１１２は、ディスクインタフェース１２６およびディスクインタフェース１２８を介してデータ格納システムと通信し、また、ホストインタフェース１３０を介して、ホストまたはコントロールシステムと通信している。一態様では、第１のコントローラ１１２は、通信バス１３２を介してディスクインタフェース１２６、ディスクインタフェース１２８、およびホストインタフェース１３０と通信している。一実施形態では、通信バス１３２は、当分野の技術者に知られているＰＣＩバスとして構成されている。一実施形態では、１３０、１２６および１

２８で示すホストインタフェースおよびディスクインタフェースは、「ＦＣループ」として動作することができるファイバチャネルバスである。他の適切なバス構成については、本出願を読めば、当分野の技術者には明らかになるであろう。

【００３３】

一態様では、メモリコントローラ１２２は、タスクプロセッサ１３４、割込みロジック１３６、およびメモリバッファ／通信モジュール１３８を備えている。一態様では、タスクプロセッサ１３４は、専用ファームウェアおよび／またはメモリバッファコンポーネントを備えており、システムオペレーションプロセッサ１３６を介したシステムオペレーションコマンドの処理を中断することなく、定義済みバックグラウンドタスクを処理している。メモリコントローラ１２２のためのホットプラグ警告／早期検出システムが、１４２で示されている。同様に、メモリコントローラ１２２のためのリセットロジックが、１４０で示されている。

【００３４】

同様に、第２の冗長コントローラ１１４は、第２の共用すなわちミラーメモリ１６０、第２のメモリコントローラ１６２、および第２のシステムオペレーションプロセッサ１６４を備えている。第２のコントローラ１１４は、ディスクインタフェース１６６およびディスクインタフェース１６８を介してデータ格納システムと通信し、また、ホストインタフェース１７０を介して、ホスト／コントロールシステムと通信している。第２のコントローラ１１４は、通信バス１７２を介してディスクインタフェース１６６、ディスクインタフェース１６８、およびホストインタフェース１７０と通信している。

【００３５】

第２のメモリコントローラ１６２は、タスクプロセッサ１７４、割込みロジック１７６、およびメモリ／通信モジュール１７８を備えている。第２のコントローラ１１４のためのリセットロジックは、１８０で示されている。ホットプラグ警告／早期検出システムが第２のメモリコントローラ１６２に設けられており、１８２で示されている。第１のコントローラ１１２および第２のコントローラ１１４は、両コントローラ間の通信バスを介して通信している。一態様では、ミラーバス２００が、第１のコントローラ１１２の第１のメモリコントローラ１２２と、第２のコントローラ１１４の第２のメモリコントローラ１６２とをリンクしている。また、交互通信経路が、第１のメモリコントローラ１２２と第２のメモリコントローラ１６２の間に設けられており、２０２で示されている。交互通信経路２０２は、第１のメモリコントローラ１２２のメモリ／通信モジュール１３８にリンクされ、かつ、第２のメモリコントローラ１６２の第２のメモリ／通信モジュール１７８にリンクされている。存在検出ライン２０４は、コントローラの存在について第１のコントローラ１１２と第２のコントローラ１１４の間の通信を提供している（例えば、ホット挿入プロセスの一部として）。

【００３６】

一実施形態では、メモリ１２０およびメモリ１６０は、ランダムアクセスメモリ（ＲＡＭ）である。一例示的実施形態では、上記ランダムアクセスメモリは、同期ダイナミックＲＡＭ（ＳＤＲＡＭ）である。一態様では、メモリ１２０およびメモリ１６０のサイズは、５１２バイトから数ギガバイトの範囲に及んでいる。好ましい一実施形態では、メモリ１２０およびメモリ１６０は、バッテリバックアップ式ＲＡＭなどの不揮発性メモリであり、そのため、電源がシャットダウン（例えばコントローラリセット）された場合においても、メモリはその記憶内容（すなわち、記憶状態）を保持している。

【００３７】

一態様では、メモリコントローラ１２２およびメモリコントローラ１６２は、特定用途向け専用集積回路（ＡＳＩＣ）チップまたはモジュールの一部である。タスクプロセッサ１３４およびタスクプロセッサ１７４は、システムオペレーションプロセッサを介してシステムコマンドが処理されている間に、定義済み専用バックグラウンドタスクを処理するように動作する。

【００３８】

一実施形態では、タスクプロセッサ１３４によって実行されるすべてのバックグラウンドタスクすなわち機能は、メモリ１２０およびメモリ１６０に格納されているデータに対して働き、これらのタスクの結果は、適当なメモリ１２０またはメモリ１６０に置き戻される。タスクプロセッサ１３４およびタスクプロセッサ１７４は、専用データ処理ハードウェアを使用して、プロセッサ１２４あるいはプロセッサ１６４などの他のシステムプロセッサを直接煩わせることなく、バックグラウンドタスク機能を実行する。一態様では、タスクプロセッサ１３４および／またはタスクプロセッサ１７４は、インテリジェントＤＭＡエンジンに結合された、ＡＳＩＣチップまたはモジュールの一部であるデータ処理ハードウェアを利用している。タスクプロセッサ１３４およびタスクプロセッサ１７４の例示的実施形態のさらに詳細については、本明細書の中で後述する。

【００３９】

一態様では、第１のメモリ１２０と第２のメモリ１６０の間のミラー読出しおよびミラー書込みは、ミラーバス２００を介して達成される。さらに、交互通信経路すなわちバス２０２が、メモリコントローラ１２２とメモリコントローラ１６２の間に設けられている。したがって、冗長コントローラデータ格納システムにコントローラが挿入され、冗長コントローラシステムの一部として未だオンラインになっていない状態において、第１のメモリコントローラ１２２は、交互通信経路２０２を介して第２のメモリコントローラ１６２と通信することができる。このような通信には、ハードウェアおよびファームウェアのレビジョン情報の交換、システム内でコントローラが交換された場合に検出するシリアル番号の交換、互いの動作状態に関する情報の交換、およびホット挿入シーケンスにおける次のステップへの移行のタイミングの相互通知が含まれている。また、故障によりミラーバス２００を介した通信が妨害された場合に、通信バスを用いて、いずれのコントローラが動作状態を維持すべきかが交渉される。ファームウェアの他の領域は、他の目的のためにこの通信バスを使用している。ホットプラグ警告１４２およびホットプラグ警告１８２は、コントローラがホット挿入されていることを冗長コントローラシステムに知らせる早期検出信号を、対応するメモリコントローラ１２２およびメモリコントローラ１６２に提供するように動作する。早期警告ロジックは、リセットロジックと共同して、ホット挿入されるコントローラがオンラインになるまで、コントローラをリセット状態に保持する。コントローラがホット除去されている間、早期検出信号は、コントローラが除去されていることについての早期警告を提供する。ホットプラグ警告１４２およびホットプラグ警告１８２早期検出システムは、機械的手段あるいは電気的手段、例えばコネクタピン、押しボタン警告、センサ検出（例えば光センサ）、またはその他の検出システムの使用を介して、早期検出信号を受け取ることができる。存在検出ライン２０４は、コントローラがシステムから除去されたこと、あるいはシステムに挿入されたことを他のコントローラに知らせるように動作する。

【００４０】

プロセッサ１２４およびプロセッサ１６４は、メモリコントローラ１２２およびメモリコントローラ１６４を介して、対応するメモリ１２０およびメモリ１６０と通信する、システムコマンドを操作するためのシステムオペレーションプロセッサである。このようなシステムコマンドには、ディスクインタフェース１２６、１３２、１６６および１６８を介して、対応するデータ格納システムからデータを読出し、かつ、システムにデータを書込むための、ホストインタフェース１３０およびホストインタフェース１７０を介して受け取るシステムコマンドが含まれている。プロセッサ１２４およびプロセッサ１６４は、システム割込みオペレーション、リセットオペレーション、または他のシステムプロセスの処理および管理など、他のシステムオペレーションを実行するように動作する。

【００４１】

図５は、本発明による冗長コントローラシステムに使用されるタスクプロセッサの一例示的実施形態を示したものである。タスクプロセッサ１３４は、一例として示したものであるが、タスクプロセッサ１７４についても、タスクプロセッサ１３４と同様である。タスクプロセッサ１３４は、データ処理ハードウェアを介して、定義済み機能を実行すること

が好ましい。これらのタスクは、「バックグラウンドタスク」として処理されるため、システムオペレーションプロセッサ１２４を介してシステムコマンドが動作している間に完了させることができる。一例示的実施形態では、タスクプロセッサ１３４は、メモリイメージをメモリ１２０とメモリ１６０の間でコピーするためのメモリ間コピータスク２０６を含んでいる。また、タスクプロセッサ１３４は、関連するメモリ１２０の自己診断を実行するための、１つまたは複数のメモリ自己診断タスク２０８を含んでいる。メモリ自己診断２０８は、冗長コントローラシステムへのコントローラの挿入時、あるいは冗長コントローラシステム１１０が動作中の任意のタイミングで実行させることができる。典型的なメモリ自己診断には、メモリイメージ、メモリチャンクすなわちデータブロックの読出し、および内部バッファ（例えば、メモリコントローラ１２２の内部バッファ）への保管が含まれている。テストパターンがメモリブロックに書き込まれ、それがリードバックされて正当性が検証される。このステップは、多くのテストパターンを用いて繰り返される。一態様では、タスクプロセッサは、１回のテストで１ないし３０のパターンを実行することができる。内部バッファに格納されていた元のデータブロックが、外部メモリブロックにライトバックされる。このプロセスは、メモリのすべてのブロックのテストが終了するまで繰り返される。タスクプロセッサ１３４の他のタスクには、二重ブロックパリティ生成２１０、単一ブロックパリティ生成２１２、ブロックパターン認識２１４、およびチェックサム生成２１６が含まれている。

【００４２】

図６は、タスクオペレーションを処理するために、タスクプロセッサ１３４およびタスクプロセッサ１７４によって使用されるデータ構造の一例示的実施形態を示したものである。他の適切なデータ構造については、本出願を読めば、当分野の技術者には明らかになるであろう。一例示的実施形態では、要求元プロセッサが、タスク記述ブロック（ＴＤＢ）をメモリ１２０に書き込んでいる。タスク記述ブロックには、コマンドコードおよび要求を処理するために必要なコマンド専用情報（ブロックアドレス、ブロックサイズ、データパターン、パリティ係数に対するポインタ等）が含まれている。要求元プロセッサは、次に、タスクプロセッサ１３４に対して局部的に、要求エントリを要求キュー（例えば、２２０で示されるキュー「０」）に挿入する。このエントリには、コマンドコード要求ヘッダ２２２、関連するタスク記述ブロックに対するＴＤＢポインタ２２４、およびキューポインタ２２６で示される、応答のためのキュー番号が含まれている。

【００４３】

キュー「０」２２０信号が、空ではない信号を送ると、タスクプロセッサは、キュー２２０から要求エントリを読み出す。タスクプロセッサは、要求情報を用いてタスクデータブロック２２８を読み出し、読み出したタスクデータブロック２２８が要求と一致していることを確認する。次に、タスクプロセッサは、所望の機能を実行する。タスクプロセッサは、完了応答エントリを、２３０で示される指定応答キューに置き、応答キュー２３０を通して要求元プロセッサ１２４に完了通知される。

【００４４】

図７は、タスクプロセッサ１３４による処理に適したメモリブロックに分割された第１のメモリ１２０に含まれるメモリイメージの一例示的実施形態を示したものである。詳細には、タスクプロセッサ１３４によって処理されるバックグラウンドタスクは、特定のタスクプロセッサ１３４を備えるメモリコントローラ１２２の内部にバッファリングするには大き過ぎる、メモリ１２０に格納されているデータブロックに対して働くことができる。したがってタスクプロセッサ１３４は、メモリイメージすなわちブロックを、タスクプロセッサによって処理することができるサイズに対応するメモリブロックすなわちチャンク中に構成するように動作する。図に示す例示的実施形態では、第１のメモリ１２０に格納されているメモリイメージは、メモリブロック「１」２３２、メモリブロック「２」２３４、メモリブロック「３」２３６、メモリブロック「４」２３８ないしメモリブロック「Ｎ」２４０中に構成される。一態様では、各チャンクは最大５１２バイトであり、メモリコントローラ１２２の内部に内部バッファリングするには十分に小さく、かつ、タスク処

理システムを有効に使用するには十分な大きさである。一態様では、タスクプロセッサ１３４は、メモリコントローラ１２２の制限内で動作しつつ、メモリイメージ当たりのメモリブロック数が最小になるメモリブロックサイズを構成するように動作している。最大使用可能メモリブロックが５１２バイトである一態様では、タスクプロセッサ１３４は、ブロック中にメモリイメージを構成し、第１のメモリブロックおよび最後のメモリブロックのみ、最大すなわち５１２バイト未満にすることができる。図に示す例示的実施形態では、メモリブロック「１」２３２およびメモリブロック「Ｎ」２４０を、最大メモリブロックサイズ未満にすることができる。メモリブロック「２」２３４、メモリブロック「３」２３６、およびメモリブロック「４」２３８等は、最大メモリブロックサイズ（例えば５１２バイト）にすることができる。

【００４５】

タスクプロセッサ１３４およびシステムオペレーションプロセッサ１２４は、いずれもメモリ１２０に格納されているデータに対して動作する。冗長コントローラシステム１１０は、冗長コントローラシステムへの第２のコントローラの追加を含め、タスクプロセッサ１３４を介してタスクが処理されている間、システムコマンドの処理を継続することができるように構成されることが望ましい。したがって優先順位は、タスクプロセッサ１３４と他のシステムオペレーションとの間で割り振られるが、これにはメモリ１２０にアクセスするためのプロセッサ１２４を介して達成されるものなどが挙げられる。好ましい一実施形態では、タスクプロセッサ１３４に割り当てられている優先順位は、オペレーティングシステムの性能が、タスクプロセッサ１３４を介したバックグラウンドタスクの動作によって極端に低下しないように、プロセッサ１２４よりも低い優先順位（例えば、最下位の優先順位）になっている。別法としては、タスクプロセッサ１３４のメモリアクセス優先順位を、他のシステムオペレーションと同じか、あるいはそれより高くすることができる。あるいは、ファームウェアを利用して、タスクプロセッサ１３４を介して達成される個々のタスクのメモリアクセス優先順位を高くすることができる。

【００４６】

図８ないし図１０は、システムの中断を最小にする、本発明による冗長コントローラシステムへのコントローラのオンライン「ホット」挿入の一例示的実施形態を示したもので、ここでも、本明細書において既に説明した図１ないし図７が参照されている。

【００４７】

図８に、本発明による冗長コントローラシステムにコントローラをホット挿入する方法の一例示的実施形態を示す図が、一括して２５０で示されている。この例示的実施形態では、冗長コントローラシステムは、第１のコントローラ１１２を介して動作しており、第２のコントローラは、冗長コントローラシステムから除去されている。２５２で、冗長コントローラシステム１１０は、単一コントローラシステムとして、第１のコントローラ１１２を介して動作する。２５４で、第１のコントローラ１１２が、冗長コントローラシステム１１０に第２のコントローラ１１４が追加されたことを検出する。冗長コントローラシステム１１０に第２のコントローラ１１４が追加されたことを検出した後、第１のコントローラ１１２は、単一コントローラシステムとしての動作を継続する。第１のコントローラは、冗長コントローラシステム１１０に第２のコントローラ１１４が追加されたことを示す検出信号を受け取る。一態様では、冗長コントローラシステム１１０に第２のコントローラ１１４がホット挿入される場合、ホット挿入されるコントローラは、コントローラが完全に挿入され、所定の位置にラッチされるまで、リセット状態に保持される。存在検出ラインが、新しいコントローラの装着を検出し、挿入されたコントローラ１１４の存在が、存在検出ライン２０４を介して第１のコントローラ１１２に通信される。

【００４８】

２５６で、第２のコントローラ１１４に電源が投入され、所定の位置へのラッチ待ち状態になり、次に、プロセッササブシステムの自己診断が実行される。プロセッササブシステムの自己診断には、フラッシュＲＯＭに存在しているプロセッササブシステムのファームウェアイメージのテスト、およびマイクロプロセッサローカルメモリのテストが含まれて

おり、周辺チップレジスタおよびデータ経路のテストが実行される。２５８で、第１のコントローラ１１２および第２のコントローラ１１４が、交互通信経路２０２を介して通信する。第２のコントローラ１１４が冗長コントローラシステム１１０の一部として未だ「オンライン」になっていない状態であっても、第１のコントローラ１１２および第２のコントローラ１１４は、交互通信経路２０２、メモリ／通信モジュール１３８、およびメモリ／通信モジュール１７８を介して、相互に通信する。交互通信経路２０２を介した、第１のコントローラ１１２と第２のコントローラ１１４の間のサンプル通信には、コントローラ間の互換性を確認するためのハードウェアおよびファームウェアのレビジョン情報の交換、テストの結果が出てテストが完了した時の通知、及びホット挿入プロセス中におけるコントローラ間の同期点の通信が含まれている。

【００４９】

２６０で、第２のコントローラ１１４が、その共用メモリに対する自己診断の実行を含む自己診断を継続する。本明細書において既に説明したように、これらのテストを、バックグラウンドタスクとしてタスクプロセッサ１７４を介して、システムを中断させることなく実行することができる。２６２で、すべてのテストに合格しない場合、リカバリモード２６４に入る。リカバリモード２６４には、冗長コントローラシステムおよび／またはホストへのエラー状態の提供が含まれている。一実施形態では、コントローラに不良のマークが施され、オフラインに維持される。もう一方のコントローラに対して、上記プロセスが繰り返される。すべてのテストに合格した場合は、２６６で、第２のコントローラ１１４が、第１のコントローラ１１２にメッセージを送り、冗長コントローラシステム１１０に追加されるべく準備が整ったことを知らせる。

【００５０】

図９に、本発明による冗長コントローラシステムにコントローラをホット挿入する方法をさらに示す図が、一括して２２０で示されている。２７２で、第１のコントローラ１１２は、メモリ１２０を共用書込みおよびローカル読出し専用として位置づける。したがって、その時点以降、メモリ１２０に書き込まれるデータはすべて、同時に第２のコントローラ１１４のメモリ１６０にミラーすなわち書き込まれる。この時点では、メモリ１６０のメモリイメージは、メモリ１２０のメモリイメージの「ミラー」コピーではないため、冗長コントローラシステム１１０の一部として、メモリ１２０から読み出すことができるのはデータのみである。２７４で、第１のコントローラ１１２の許可が出るまで（例えば、交互通信経路２０２を介して）、第２のコントローラ１１４による共用メモリ１２０および独自のメモリ１６０への書込みが禁止される。

【００５１】

２７６で、第１のコントローラ１１２が、バックグラウンドタスクを介して、その共用メモリ１２０のすべてを、共用メモリ１２０の同一ロケーションにコピーバックする。詳細にはタスクプロセッサ１３４はバックグラウンドタスクを含んでいるが、ここでタスクプロセッサ１３４は、メモリ１２０からメモリブロックを読み出し、読み出したメモリブロックをバッファに格納し、そしてメモリ１２０の同一ロケーションに書き込んでいる。第１のコントローラ１１２は、共用書込みモードに構成されているため、上記オペレーションの結果、第１のコントローラ１１２は、メモリブロック１２０からメモリブロックをローカルに読み出すことになるが、第１のコントローラ１１２が、メモリ１２０の同一ロケーションにライトバックする場合、同時に第２のコントローラ１１４のメモリ１６０の同一ロケーションに書き込むことになる。２７８で、上記バックグラウンドタスクの間、第１のコントローラ１１２は、プロセッサ１２４を介してシステムオペレーションコマンドを実行する動作状態を継続する。一態様では、メモリイメージは、一度に１つのメモリブロックがコピーされる。バックグラウンドタスク完了後、ここで第１のメモリ１２０のメモリイメージは第２のメモリ１６０のメモリイメージのミラーであり、冗長コントローラシステム１１０への第２のコントローラ１１４の追加プロセスが、２８０で示す通り継続される。

【００５２】

図１０は、本発明による冗長コントローラシステムへのコントローラの追加方法の一例示的実施形態を示したものであり、２９０で示す通り、第１のコントローラ１１２のメモリイメージが第２のコントローラメモリ１６０のメモリにミラーされた後又はコピーされた後のものである。２９２で、第１のコントローラ１１２が再構成され、ミラー書込みおよびミラー読出しモードに構成される。ここで第１のメモリ１２０および第２のメモリ１６０の両メモリに対して、データロケーションの読み出し及び書き込みが可能となる。２９４で、第１のコントローラ１１２がすべてのメモリロケーションを読み出し、タスクプロセッサ１３４を介したバックグラウンドタスクを用いて、第１のコントローラの共用メモリ１２０と第２のコントローラの共用メモリ１６０の一致性が比較される。したがって、この時点でシステムオペレーションコマンドが中断されることはない。

【００５３】

２９６で、メモリが一致しない場合、リカバリモード２９８に入る。メモリが一致した場合は、３００で、第１のコントローラ１１２および第２のコントローラ１１４が再構成され、第２のコントローラが冗長コントローラシステム１１０に追加される。この時点で冗長コントローラシステム１１０は、ミラーメモリ冗長コントローラシステムとして完全な動作状態になり、また、第２のコントローラは、ホストをタイムアウトさせることなく、システムオペレーションの処理を最短時間だけ中断させるだけで、冗長コントローラシステムにホット挿入される。

【００５４】

コントローラのリセット

知られている二重コントローラシステムでは、一方のコントローラがリセットされると、もう一方のコントローラのマイクロプロセッサへの割込みが生じる。その場合、割り込まれた側のコントローラのプロセッサは、割込みおよびリセットの原因を処理しなければならない。この過去の方法には多くの欠点が知られている。第１のコントローラと第２のコントローラの間のリセットのタイミングは様々である。一方のコントローラが「リセットループ」に入ると、そのリセットは他のコントローラのプロセッサに割込みをかけるが、ここでコントローラシステムの性能に影響を及ぼす拡散動作を備え、洗練度の劣る、ミラリングインタフェースの状態変化に連動する。第２のコントローラが、「スタック」すなわちエラティックプロセッサを有している場合、第２のコントローラのプロセッサは、割込みを処理するためには役に立たないため、第１のコントローラにより、第２のコントローラをリセットさせることはできない。したがって第１のコントローラは、システムを復旧させるためには、リセットを発生する「ウォッチドッグ」モードに頼らざるを得ない。第２のコントローラが、システムを使用して格納されているデータを損傷させる原因となるためには、はるかに長い適期が存在している。

【００５５】

図１１および図１２は、本発明による冗長コントローラシステムを使用してコントローラリセットを処理するシステムおよび方法の一例示的実施形態を示したものである。本発明による冗長コントローラシステムを使用したコントローラリセット処理方法により、ミラリングバスが使用可能状態になった場合に、リセットのみが第２のコントローラに伝播することができるように、一方のコントローラに対するリセットがローカライズされる。このローカライズにより、故障したコントローラが他のすべてのコントローラをリセット状態に保留することを回避している。図１１および図１２も、本明細書において既に説明した図１ないし図１０が参照されている。

【００５６】

図１１に、本発明による多重冗長コントローラシステムにおけるコントローラリセット処理方法の一例示的実施形態が４００で示されている。冗長コントローラシステムは、アクティブに接続されミラー対として動作する、第１のコントローラ１１２および第２のコントローラ１１４を備えており、４０２で、第２のコントローラ１１４に対するリセット状態が検出される。第２のコントローラ１１４はリセットされ、シャットダウンプロセスが開始される。４０４で、シャットダウンプロセスの一環として、第１のコントローラ１１

２と第２のコントローラ１１４の間の通信リンクを介して、第１のコントローラ１１２に、コントローラがリセットされたことが通知される。好ましい一実施形態では、ミラーバス２００を介して、第１のコントローラ１１２に、第２のコントローラがリセットされていることが通知される。４０６で、第１のコントローラ１１２および第２のコントローラ１１４の両コントローラに対して、シャットダウンプロセスが実行される。したがって第１のコントローラ１１２および第２のコントローラ１１４は、同時にシャットダウンプロセスを通過する。

【００５７】

図１２に、本発明による二重コントローラシステムにおけるコントローラリセット方法の一例示的実施形態をさらに示す図が４１０で示されている。４１２で、両コントローラに対するシャットダウンプロセスが完了すると、第１のコントローラ１１２および第２のコントローラ１１４に電源が投入される。シャットダウンプロセスにより、すべての内部バッファがフラッシュされ、メモリが置かれる。４１４で、シャットダウンプロセスの一部として、第１のコントローラおよび第２のコントローラがリセットされる。４１６で、第１のコントローラ１１２と第２のコントローラ１１４の間のミラーバス２００インタフェースの使用が禁止される。一態様では、リセットの作用により、第１のコントローラ１１２および第２のコントローラ１１４が、ミラーバス２００の使用を禁止している。ミラーバスの使用を禁止する動作により、ミラーバスが再び使用可能になるまで、ボード間でのさらなるリセットの伝播を防止している。

【００５８】

４１８で、第１のコントローラ１１２および第２のコントローラ１１４の各々が、自己診断を実行する。通常、自己診断には、内部メモリコントローラＡＳＩＣ１２２および１６２のテスト、およびすべてのデータ経路バスのテスト以外に、本出願において既に説明したように、マイクロプロセッササブシステムおよびＳＤＲＡＭメモリのテストが含まれている。

【００５９】

この時点では、第１のコントローラ１１２と第２のコントローラ１１４の間のミラーバス２００インタフェースは、使用禁止の状態が維持されている。したがってコントローラの一方によって生じる、あるいは自己診断の結果として生じるあらゆるリセットまたは割込みによって、もう一方のコントローラが影響されることはない。４２０で、自己診断に合格しなかった場合、自己診断が不合格となったコントローラは、４２２のリカバリモードに入る。通常リカバリモードには、ホストコンピュータシステムに対するエラーの発生、故障の通知、及びアレイ中の「不良」コントローラが使用禁止されないよう除去することが含まれている。残った「正常」コントローラは、データのミラーが必要ない単一コントローラモードで動作を開始する。

【００６０】

第１のコントローラ１１２および第２のコントローラ１１４の自己診断に合格すると、４２４で、第１のコントローラ１１２と第２のコントローラ１１４の間のミラーバス２００インタフェースが使用可能になる。したがって第１のコントローラ１１２および第２のコントローラ１１４は、リセットまたはリセットの原因が除去されたことを確認し、ミラー対としての動作を、再び継続することができる。

【００６１】

本発明による冗長コントローラシステムを使用した上述のコントローラリセット処理方法によると、ミラーバスと共に使用可能状態にされない限りリセットが第２のコントローラに伝播できないように、１つのコントローラに対するリセットをローカライズする。このローカライズにより、１つのコントローラに、システム内の他のすべてのコントローラをリセットさせることのないシステムファームウェアのための方法が可能となる一方で、多重コントローラボード間のリセットの同期化を、ハードウェアで管理する利点がもたらされる。

【００６２】

コントローラの除去
冗長コントローラデータ格納システムからオンラインコントローラを除去する場合、コントローラが「部分除去」されるかあるいは正しく除去されない場合、故障時間が発生することになる。これは、通常、コントローラがオンライン除去されている間、冗長コントローラが静止状態になるか、あるいはリセット状態に保持されることによって生じる。
【００６３】
冗長コントローラシステムからコントローラをオンライン除去するための知られているプロセスには早期警告スイッチすなわち短コネクタピンが含まれており、これはコントローラが除去されていることを冗長コントローラシステムに警告するものである。この警告により、コントローラはその時点のコントローラのメモリアクセスを終了させ、コントローラの不揮発メモリを自己リフレッシュモードにすることができる。
【００６４】
コントローラが完全に分離（例えば、長コントローラ検出ピンが開放）されると、「対になっている」コントローラのもう一方のコントローラが、冗長コントローラシステムを制御するための動作を再開することができる。検出ピンコネクタを利用しているシステムの場合、部分除去されるコントローラは検出ピンが完全に接点をシャットダウンするまでの間、システムを使用不可の状態に保持することができる。このセットアップでは、例えばコントローラが冗長コントローラシステムから「部分的にのみ除去」される等の誤った手順により、オンライン除去故障時間が延長し、ホストオペレーティングシステムのタイムアウト期間を超過するという場合がある。
【００６５】
図１３は、本発明による冗長コントローラシステムからコントローラをオンライン除去する方法の一例示的実施形態を示したものである。上記方法は４５０で示されている。オンライン除去方法４５０により、冗長コントローラシステムの故障時間を最短にする一方で、コントローラを冗長コントローラシステムから安全にオンライン除去することができる。
【００６６】
４５２で、冗長コントローラシステム１１０からコントローラが除去されていることが検出される。ここでも、本明細書において既に説明した図１から図１２を参照する。一態様では、コネクタ上の早期警告スイッチ又は短コネクタピンが、コントローラが除去されていることをシステム１１０に警告するが、検出は冗長コントローラシステム１１０からコントローラが完全に開放される前になされることが好ましい。上記警告は、ホットプラグ警告１４２またはホットプラグ警告１８２を介して受信される。本明細書で説明する一例示的実施形態では、第１のコントローラ１１２が冗長コントローラシステム１１０から除去される。
【００６７】
４５４で、冗長コントローラシステム１１０から第１のコントローラ１１２が除去されていることが検出されると、第１のコントローラ１１２および第２のコントローラ１１４に対するシャットダウンシーケンスが実行される。一態様では、４５６で、各コントローラに対するシャットダウンシーケンスでは、コントローラのプロセッサに割込みがかけられ、プロセッサによる稼動中の処理タスクの終了を可能にしている。４５８で、各コントローラに対するシャットダウンシーケンスではさらに、メモリ１２０およびメモリ１６０に対する未処理のメモリアクセスを完了し、内部バッファをフラッシングする。シャットダウンシーケンスの一部として、メモリコントローラ１２２によって状態語がメモリ１２０に書き込まれ、また、メモリコントローラ１６２によって状態語がメモリ１６０に書き込まれる。
【００６８】
好ましい一実施形態では、第１のメモリ１２０および第２のメモリ１６０は、自己リフレッシュモードを有しており、さらに好ましくは、バッテリバックアップを備えている。第１のコントローラ１１２および第２のコントローラ１１４に対するシャットダウンシーケ

ンスが完了すると、第１のメモリ１２０および第２のメモリ１６０が、それぞれ対応する第１のメモリコントローラ１２２および第２のメモリコントローラ１６２によって自己リフレッシュモードに置かれる。４６２で、除去を検出するコントローラーはオフラインに留まり、除去の終了を待つ。そのメモリは、自己リフレッシュモードに留まっている。４６４で、除去を検出しないコントローラによって、オンラインになるプロセスが直ちに開始される。

【００６９】

本明細書で説明する例示的実施形態では、メモリ１２０に対する自己リフレッシュプロセスが完了すると、除去を検出する第１のコントローラ１１２がオフラインに留まり、除去の終了を待ち、およびメモリ１２０が自己リフレッシュモードに留まる（一態様では、メモリは、バッテリでバックアップされたＤＲＡＭである）。除去を検出しない第２のコントローラ１１４は、直ちにオンラインになるプロセスを開始し、冗長コントローラシステムの故障時間を最短にする。メモリ１６０（例えばバッテリバックアップＤＲＡＭ）は、自己リフレッシュモードを解除され、メモリコントローラが使用する、予め書き込まれている状態語を保持している。

【００７０】

以上、好ましい実施形態について説明するために、本明細書において特定の実施形態を示し、かつ、記述したが、本発明の範囲を逸脱することなく、広範囲の様々な代替実施態様および／または等価実施態様を、上に示し、かつ、説明した特定の実施形態と置き換えることができることは、当分野の技術者には理解されよう。本発明を、極めて広範囲の様々な実施形態の中で実施することができることは、化学、機械、電気機械、電気、およびコンピュータ技術の分野の技術者には、容易に理解されよう。本出願は、本明細書において説明した好ましい実施形態のあらゆる応用、すなわち変形形態を包含することを意図している。したがって本発明は、特許請求の範囲の各クレームおよびその相当によってのみ制限されることを、明確に意図している。

【００７１】

（１）第１のコントローラ（３２、１１２）および第２のコントローラ（３４、１１４）を有する冗長コントローラシステム（３０、１１０）からコントローラをオンライン除去する方法であって、

早期警告検出を介して、前記冗長コントローラシステムからの前記第１のコントローラの部分除去を検出するステップ（４５２）と、

前記第１のコントローラおよび前記第２のコントローラに対して、未処理のメモリアクセスを完了させるステップ（４５８）を含むシャットダウンシーケンスを実行するステップ（４５４）と、

第１のメモリ（５０、１２０）を有するように前記第１のコントローラを定義し、かつ、前記第１のメモリを自己リフレッシュモードにするステップ（４６０）と、

前記冗長コントローラシステムからの前記第１のコントローラの除去を終了させるステップとを含む方法。

【００７２】

（２）第１のプロセッサ（５４、１２４）を含むように前記第１のコントローラを定義するステップと、第２のプロセッサを含むように前記第２のコントローラを定義するステップとを含み、前記第１のコントローラおよび前記第２のコントローラに対してシャットダウンシーケンスを実行する前記ステップ（４５４）が、前記第１のプロセッサに割り込むステップおよび前記第２のプロセッサに割り込むステップ（４５６）をさらに含む、（１）に記載の方法。

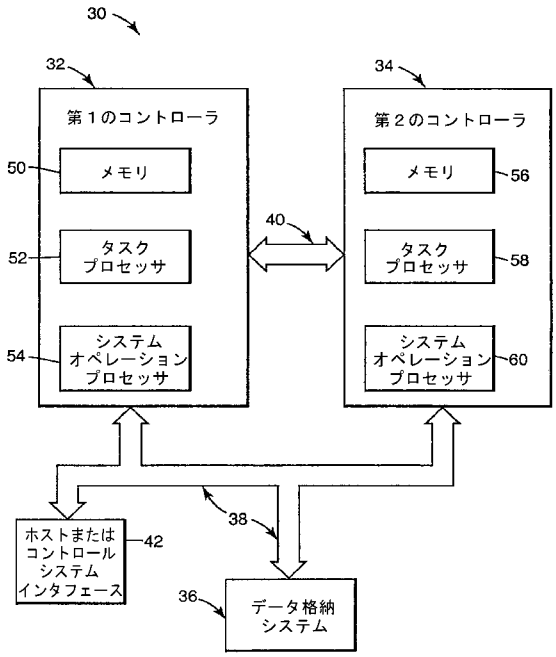
【００７３】

（３）前記第１のプロセッサに割り込む前記ステップが、前記第１のプロセッサによる未処理のプロセッサタスクの終了を可能にするステップを含み、かつ、前記第２のプロセッサに割り込む前記ステップが、前記第２のプロセッサによる未処理のプロセッサタスクの終了を可能にするステップ（４５６）を含む、（３）に記載の方法。

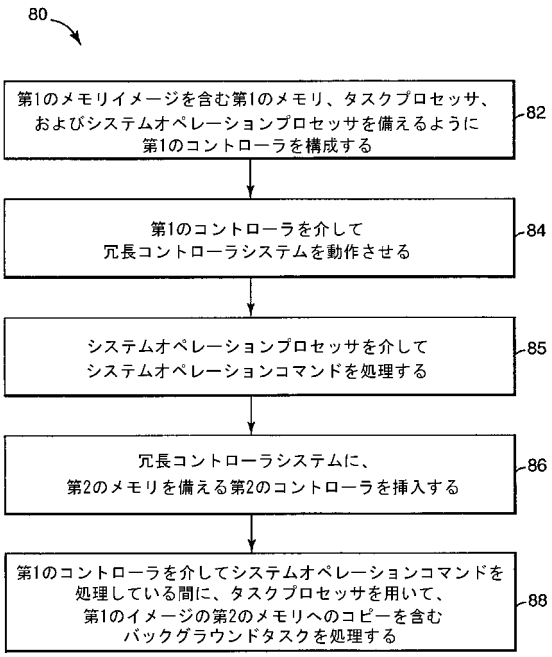
【００７４】
（４）前記第１のメモリを前記自己リフレッシュモードにする前記ステップ（４６０）が、前記第１のメモリを、メモリの内容を維持するためのメモリコントローラからの外部リフレッシュサイクルを必要としないモードにするステップを含む、（１）に記載の方法。
【００７５】
（５）第２のメモリを備えるように前記第２のコントローラを定義し、かつ、前記第２のメモリを自己リフレッシュモードにするステップ（４６０）を含む、（１）に記載の方法。
【００７６】
（６）前記第２のメモリを自己リフレッシュプロセスに置く前記ステップ（４６０）が、前記第２のメモリを、メモリの内容を維持するためのメモリコントローラからの外部リフレッシュサイクルを必要としないモードにするステップを含む、（５）に記載の方法。
【００７７】
（７）前記第２のメモリの自己リフレッシュモードを終了させた後で、前記第２のコントローラに、オンラインになるプロセスを開始させるステップ（４６４）を含む、（５）に記載の方法。
【００７８】
（８）前記オンラインになるプロセスが、前記自己リフレッシュモードを終了させた後で、かつ、前記冗長コントローラシステムからの前記第１のコントローラの除去を終了させる前に、直ちに開始されるように構成される（４６４）、（７）に記載の方法。
【００７９】
（９）第１のプロセッサ（５４、１２４）を含むように前記第１のコントローラを定義し、かつ、第２のプロセッサ（６０、１６４）を含むように前記第２のコントローラを定義するステップと、
前記第１のコントローラおよび前記第２のコントローラに対して、前記第１のプロセッサおよび前記第２のプロセッサに割り込み、前記第１のコントローラおよび前記第２のコントローラに対する未処理のメモリアクセスを完了させるステップ（４５６）を含むシャットダウンシーケンスを実行するステップ（４５４）と、
前記第１のメモリの自己リフレッシュモードを解除した後、前記冗長コントローラシステムからの前記第１のコントローラの除去を終了させるステップとを含む、（１）に記載の方法。
【００８０】
（１０）（１）から（９）に記載の方法を実行するように構成された冗長コントローラシステム（３０、１１０）。
【図面の簡単な説明】
【図１】冗長コントローラをホット挿入するために構成された、本発明による冗長コントローラデータ格納システムの一例示的実施形態を示す図である。
【図２】本発明による冗長コントローラデータ格納システムにコントローラをホット挿入するための方法の一例示的実施形態を示す図である。
【図３】本発明による冗長コントローラシステムにコントローラをホット挿入する方法の他の例示的実施形態を示す図である。
【図４】本発明による冗長コントローラをホット挿入するために構成された冗長コントローラデータ格納システムの他の例示的実施形態を示す構成図である。
【図５】本発明による冗長コントローラデータ格納システムに使用されるタスクプロセッサの一例示的実施形態を示す図である。
【図６】本発明による冗長コントローラシステムのタスクプロセッサによって利用されるデータ構造の一例示的実施形態を示す図である。
【図７】本発明による冗長コントローラデータ格納システムに使用される、メモリブロック中に構成されたメモリイメージを有するコントローラ共用メモリの一例示的実施形態を示す図である。

【図８】本発明による冗長コントローラシステムにコントローラをホット挿入する方法の一例示的実施形態を示す図である。
【図９】本発明による冗長コントローラシステムにコントローラをホット挿入する方法の一例示的実施形態をさらに示す図である。
【図１０】本発明による冗長コントローラシステムにコントローラをホット挿入する方法の一例示的実施形態をさらに示す図である。
【図１１】本発明による冗長コントローラシステムにおけるコントローラリセット処理方法の一例示的実施形態を示す図である。
【図１２】本発明による冗長コントローラシステムにおけるコントローラリセット処理方法の一例示的実施形態をさらに示す図である。
【図１３】本発明による冗長コントローラシステムからコントローラを除去する方法の一例示的実施形態を示す図である。
【符号の説明】
３２、１１２　第１のコントローラ
３０、１１０　冗長コントローラシステム
３４、１１４　第２のコントローラ
５０、１２０　第１のメモリ
５４、１２４　第１のプロセッサ
６０、１６４　第２のプロセッサ

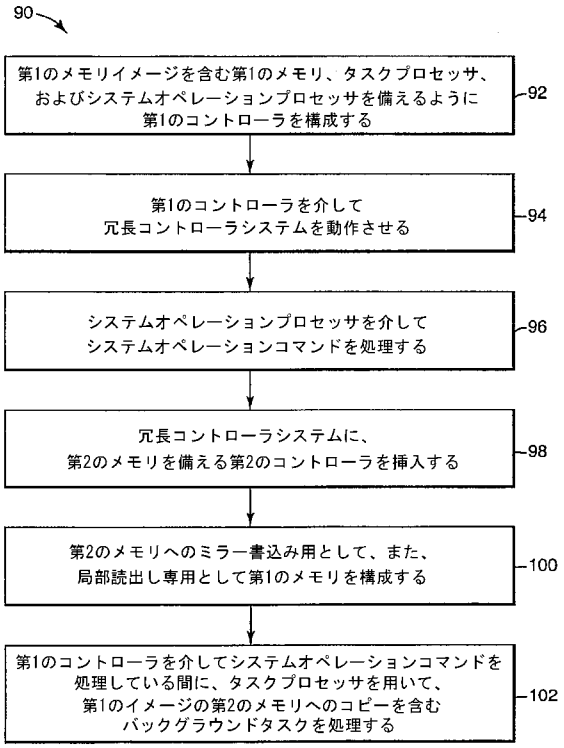
【図１】



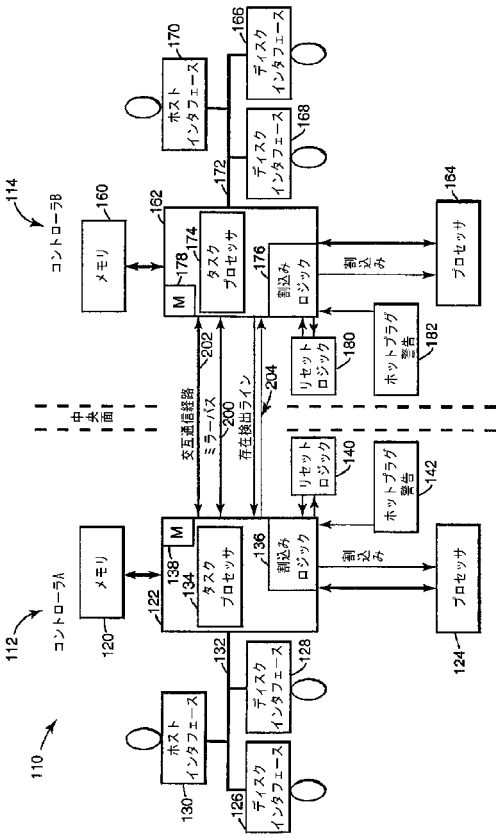
【図２】



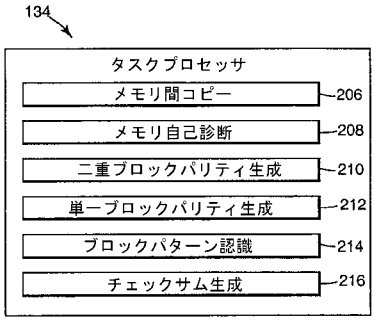
【図３】


90
第1のメモリイメージを含む第1のメモリ、タスクプロセッサ、およびシステムオペレーションプロセッサを備えるように第1のコントローラを構成する
92
第1のコントローラを介して冗長コントローラシステムを動作させる
94
システムオペレーションプロセッサを介してシステムオペレーションコマンドを処理する
96
冗長コントローラシステムに、第2のメモリを備える第2のコントローラを挿入する
98
第2のメモリへのミラー書込み用として、また、局部読出し専用として第1のメモリを構成する
100
第1のコントローラを介してシステムオペレーションコマンドを処理している間に、タスクプロセッサを用いて、第1のイメージの第2のメモリへのコピーを含むバックグラウンドタスクを処理する
102

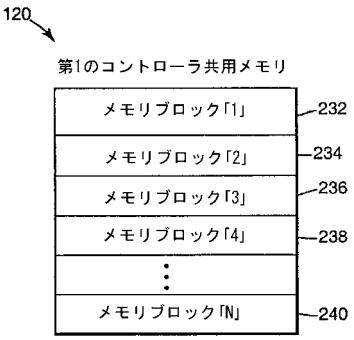
【図４】


ホストインタフェース
ディスクインタフェース
コントローラB
メモリ
タスクプロセッサ
割込みロジック
プロセッサ
リセットロジック
ホットプラグ警告
交互通信経路
ミラーバス
存在検出ライン
中央面
コントローラA
110
112
114
120
122
124
126
128
130
132
134
136
138
140
142
160
162
164
166
168
170
172
174
176
178
180
182
200
202
204

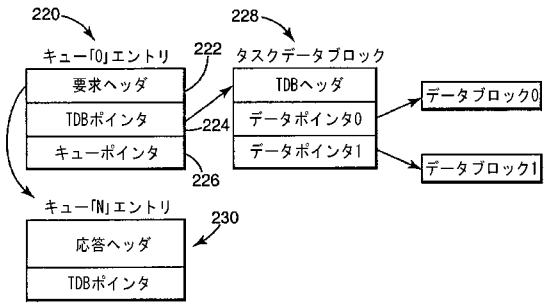
【図５】


134
タスクプロセッサ
メモリ間コピー
206
メモリ自己診断
208
二重ブロックパリティ生成
210
単一ブロックパリティ生成
212
ブロックパターン認識
214
チェックサム生成
216

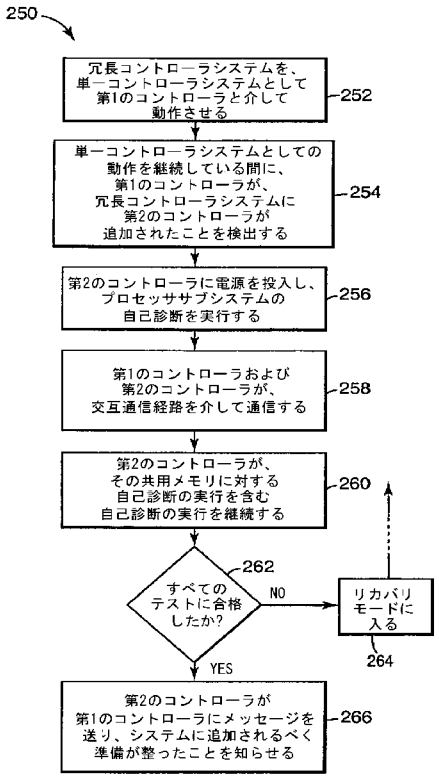
【図７】


120
第1のコントローラ共用メモリ
メモリブロック「1」
232
メモリブロック「2」
234
メモリブロック「3」
236
メモリブロック「4」
238
メモリブロック「N」
240

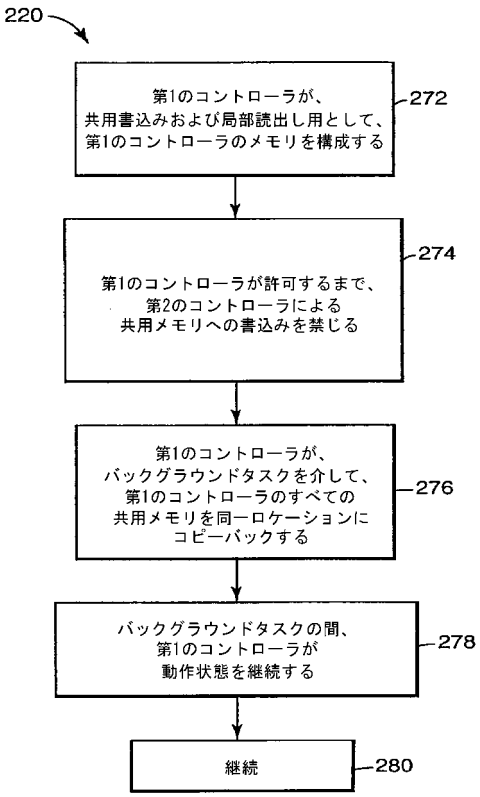
【図６】


220
キュー「0」エントリ
222
要求ヘッダ
TDBポインタ
224
キューポインタ
226
228
タスクデータブロック
TDBヘッダ
データポインタ0
データポインタ1
データブロック0
データブロック1
キュー「N」エントリ
230
応答ヘッダ
TDBポインタ

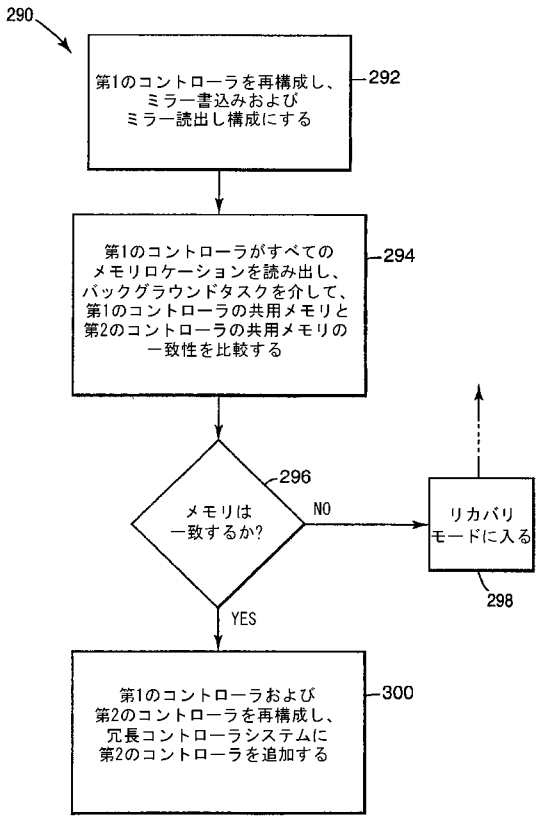
【図８】


250
冗長コントローラシステムを、単一コントローラシステムとして第1のコントローラと介して動作させる
252
単一コントローラシステムとしての動作を継続している間に、第1のコントローラが、冗長コントローラシステムに第2のコントローラが追加されたことを検出する
254
第2のコントローラに電源を投入し、プロセッササブシステムの自己診断を実行する
256
第1のコントローラおよび第2のコントローラが、交互通信経路を介して通信する
258
第2のコントローラが、その共用メモリに対する自己診断の実行を含む自己診断の実行を継続する
260
すべてのテストに合格したか?
262
NO
リカバリモードに入る
264
YES
第2のコントローラが第1のコントローラにメッセージを送り、システムに追加されるべく準備が整ったことを知らせる
266

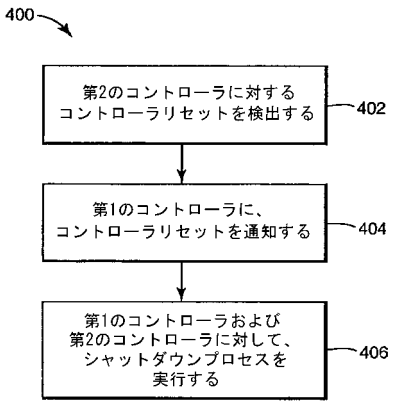
【図９】


220
第1のコントローラが、共用書込みおよび局部読出し用として、第1のコントローラのメモリを構成する
272
第1のコントローラが許可するまで、第2のコントローラによる共用メモリへの書込みを禁じる
274
第1のコントローラが、バックグラウンドタスクを介して、第1のコントローラのすべての共用メモリを同一ロケーションにコピーバックする
276
バックグラウンドタスクの間、第1のコントローラが動作状態を継続する
278
継続
280

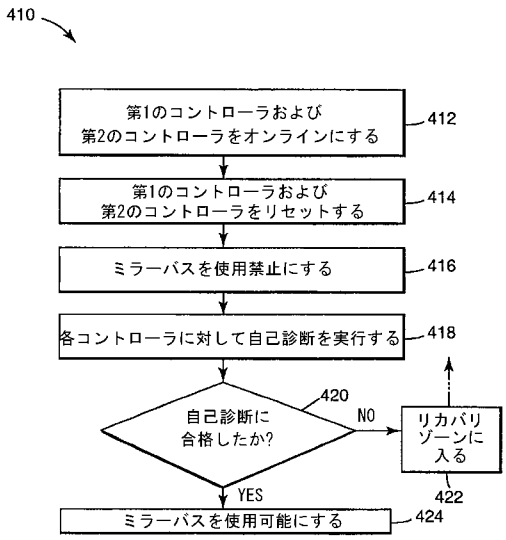
【図１０】


290
第1のコントローラを再構成し、ミラー書込みおよびミラー読出し構成にする
292
第1のコントローラがすべてのメモリロケーションを読み出し、バックグラウンドタスクを介して、第1のコントローラの共用メモリと第2のコントローラの共用メモリの一致性を比較する
294
メモリは一致するか?
296
NO
リカバリモードに入る
298
YES
第1のコントローラおよび第2のコントローラを再構成し、冗長コントローラシステムに第2のコントローラを追加する
300

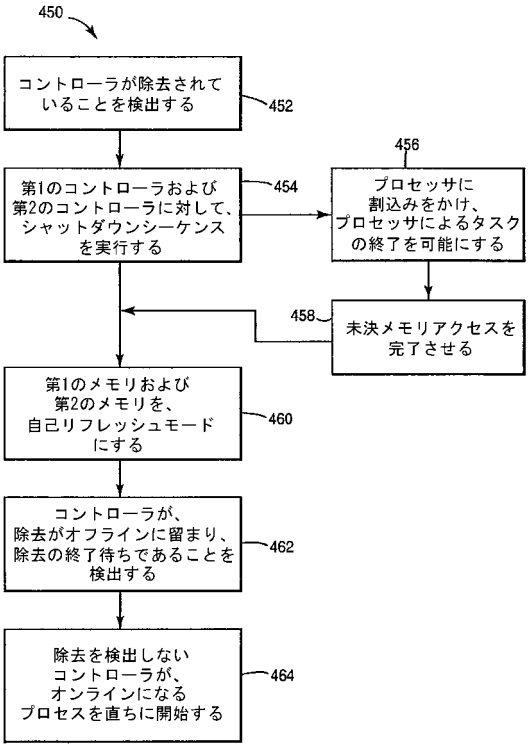
【図１１】


400
第2のコントローラに対するコントローラリセットを検出する
402
第1のコントローラに、コントローラリセットを通知する
404
第1のコントローラおよび第2のコントローラに対して、シャットダウンプロセスを実行する
406

【図１２】


410
第1のコントローラおよび
第2のコントローラをオンラインにする
412
第1のコントローラおよび
第2のコントローラをリセットする
414
ミラーパスを使用禁止にする
416
各コントローラに対して自己診断を実行する
418
自己診断に
合格したか?
420
NO
リカバリ
ゾーンに
入る
422
YES
ミラーパスを使用可能にする
424

【図１３】


450
コントローラが除去されて
いることを検出する
452
第1のコントローラおよび
第2のコントローラに対して、
シャットダウンシーケンス
を実行する
454
456
プロセッサに
割込みをかけ、
プロセッサによるタスク
の終了を可能にする
458
未決メモリアクセスを
完了させる
第1のメモリおよび
第2のメモリを、
自己リフレッシュモード
にする
460
コントローラが、
除去がオフラインに留まり、
除去の終了待ちであることを
検出する
462
除去を検出しない
コントローラが、
オンラインになる
プロセスを直ちに開始する
464

フロントページの続き

(72)発明者　バリー・ジェイ・オールドフィールド
アメリカ合衆国８３７１３アイダホ州ボイジー、ウェスト・ダニエル・コート　１１３０２
(72)発明者　クリストファー・ダブリュ・ヨハンソン
アメリカ合衆国８３６２９アイダホ州ホースシュー・ベンド、リバー・ブラフ・レーン　３２

審査官　▲高▼橋　正▲徳▼

(56)参考文献　特開平０８－２７２７５３（ＪＰ，Ａ）
特開平０８－１９０４９４（ＪＰ，Ａ）
特許第２６０８９０４（ＪＰ，Ｂ２）
国際公開第００／０６７１２６（WO，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F11/16-11/20