(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第5646644号
(P5646644)

(45)発行日　平成26年12月24日(2014.12.24)　(24)登録日　平成26年11月14日(2014.11.14)

(51)Int.Cl.　　F I
G06F 9/50 (2006.01)　　G06F 9/46 465E
G06F 12/00 (2006.01)　　G06F 12/00 550A
G06F 15/177 (2006.01)　　G06F 15/177 A

請求項の数 17　(全 16 頁)

(21)出願番号　特願2012-541131(P2012-541131)
(86)(22)出願日　平成22年11月22日(2010.11.22)
(65)公表番号　特表2013-512509(P2013-512509A)
(43)公表日　平成25年4月11日(2013.4.11)
(86)国際出願番号　PCT/US2010/057561
(87)国際公開番号　WO2011/066202
(87)国際公開日　平成23年6月3日(2011.6.3)
審査請求日　平成25年11月22日(2013.11.22)
(31)優先権主張番号　12/624,626
(32)優先日　平成21年11月24日(2009.11.24)
(33)優先権主張国　米国(US)

早期審査対象出願

(73)特許権者　591016172
アドバンスト・マイクロ・ディバイシズ・インコーポレイテッド
ＡＤＶＡＮＣＥＤ　ＭＩＣＲＯ　ＤＥＶＩＣＥＳ　ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国、９４０８８－３４５３　カリフォルニア州、サニィベイル、ピィ・オゥ・ボックス・３４５３、ワン・エイ・エム・ディ・プレイス、メイル・ストップ・６８（番地なし）
(74)代理人　100108833
弁理士　早川　裕司
(74)代理人　100111615
弁理士　佐野　良太

最終頁に続く

(54)【発明の名称】分散型多重コアメモリ初期化

(57)【特許請求の範囲】
【請求項１】
メモリ初期化タスクを含むブート処理タスクを複数の処理ノードに行わせる方法であって、
制御処理ノードでメモリ初期化タスクを複数のメモリ初期化サブタスクに分割することと、
対応する処理ノードを各メモリ初期化サブタスクが有するように、前記複数のメモリ初期化サブタスクを前記複数の処理ノードの間で分散させることと、
前記対応する処理ノードで各メモリ初期化サブタスクを実行してサブタスク結果を生成することと、
前記複数の処理ノードからのサブタスク結果を前記制御処理ノードで結合することと、
を備え、
前記複数のメモリ初期化サブタスクは、前記複数の処理ノードで並列に又はシーケンスに実行され得るものであって、
前記分散させることは、
シリアルプレゼンスディテクト（ＳＰＤ）値をデュアルインラインメモリモジュールから読み出すメモリ初期化サブタスクを、前記複数の処理ノードの１つに割り当てることと、
前記ＳＰＤ値に依存しないメモリ初期化サブタスクを、前記複数の処理ノードの他の１つに割り当てることとを含む、

方法。
【請求項２】
前記複数のメモリ初期化サブタスクを分散させるのに先立ち前記複数の処理ノードの間の通信リンクを初期化することを更に備える請求項１の方法。
【請求項３】
前記対応する処理ノードで各メモリ初期化サブタスクを実行することは前記対応する処理ノードでの実行のために各メモリ初期化サブタスクをスケジューリングすることを含む請求項１の方法。
【請求項４】
前記対応する処理ノードで各メモリ初期化サブタスクを実行することはソフトウエアルーチンを実行することを含む請求項１の方法。
【請求項５】
前記対応する処理ノードでの前記メモリ初期化サブタスクの実行に関する状況報告を前記制御処理ノードで受信することを更に備える請求項１の方法。
【請求項６】
前記複数の処理ノードの１つは、そのメモリ初期化サブタスクを、前記複数の処理ノードの他の１つであって当該他の１つに対応するメモリ初期化サブタスクを実行する他の１つと並列に実行する請求項１の方法。
【請求項７】
システムメモリと、
複数の処理ノードとを含むコンピュータシステムであって、
前記複数の処理ノードの各々は、
プロセッサコアと、前記複数の処理ノードの他の少なくとも１つに接続するための通信インタフェースと、前記システムメモリへの及び前記システムメモリからのデータフローを管理するためのメモリ制御器とを含み、
前記複数の処理ノードは、
マスター処理ノードと、複数の実行処理ノードとを含み、
前記マスター処理ノードは、
メモリ初期化タスクを複数のメモリ初期化サブタスクに分割することと、割り当てられたメモリ初期化サブタスクの各々が対応する実行処理ノードを有するように、前記複数のメモリ初期化サブタスクを前記複数の実行処理ノードに割り当てることとによって、前記メモリ初期化タスクを前記システムメモリに対して実行するように構成されており、
前記複数のメモリ初期化サブタスクは、
前記複数の実行処理ノードの１つに割り当てられたメモリ初期化サブタスクが、前記複数の実行処理ノードの他の１つに割り当てられた他のメモリ初期化サブタスクに依存することなく順序を問わずに実行可能となるように、前記複数の実行処理ノードで並列に又はシーケンスに実行され得るものであり、
前記割り当てることは、
シリアルプレゼンスディテクト（ＳＰＤ）値を前記システムメモリのデュアルインラインメモリモジュールから読み出すメモリ初期化サブタスクを、前記複数の処理ノードの１つに割り当てることと、
前記ＳＰＤ値に依存しないメモリ初期化サブタスクを、前記複数の処理ノードの他の１つに割り当てることとを含む、
コンピュータシステム。
【請求項８】
各実行処理ノードは、前記マスター処理ノードによって割り当てられた各メモリ初期化サブタスクを実行するとともに、サブタスク結果を生成するように構成されている請求項７のコンピュータシステム。
【請求項９】
前記マスター処理ノードは、前記複数の実行処理ノードからのサブタスク結果を結合す

るように構成されている請求項７のコンピュータシステム。
【請求項１０】
前記システムメモリは、複数のデュアルインラインメモリモジュールを含む請求項７のコンピュータシステム。
【請求項１１】
前記マスター処理ノードは、前記メモリ初期化サブタスクの状況を前記複数の実行処理ノードから受信するように構成されている請求項７のコンピュータシステム。
【請求項１２】
前記マスター処理ノードは、メモリ初期化サブタスクを実行する複数の実行処理ノードの１つである請求項７のコンピュータシステム。
【請求項１３】
前記複数の実行処理ノードの１つは、そのメモリ初期化サブタスクを、前記複数の処理ノードの他の１つであって当該他の１つに対応するメモリ初期化サブタスクを実行する他の１つと並列に実行する請求項７のコンピュータシステム。
【請求項１４】
前記マスター処理ノードは、前記複数の実行処理ノードで前記複数のメモリ初期化サブタスクを実行した結果に基づいて、前記マスター処理ノードの前記メモリ制御器内の１つ以上のレジスタをプログラムすることによって、前記メモリ初期化タスクを実行する請求項７のコンピュータシステム。
【請求項１５】
メモリアレイと、
複数の処理ノードとを含むコンピュータシステムであって、
前記複数の処理ノードの各々は、
プロセッサコアと、前記メモリアレイと通信するためのメモリ制御器と、前記複数の処理ノードの他の少なくとも１つに接続するための通信インタフェースとを含み、
前記複数の処理ノードのうち第１の処理ノードは、
メモリ初期化タスクを複数のサブタスクに分割し、前記複数のサブタスクを前記複数の処理ノードのうち他の処理ノードに割り当て、
前記複数の処理ノードの各々は、
割り当てられたサブタスクを実行しながら前記メモリアレイの任意の部分にアクセスするとともに、サブタスク実行結果を前記複数の処理ノードのうち前記第１の処理ノードに戻すことができ、これにより、前記複数の処理ノードが前記メモリアレイのメモリ初期化を実行することを可能にし、
前記複数のサブタスクを前記複数の処理ノードのうち他の処理ノードに割り当てることは、
シリアルプレゼンスディテクト（ＳＰＤ）値をデュアルインラインメモリモジュールから読み出すメモリ初期化サブタスクを、前記複数の処理ノードの１つに割り当てることと、
前記ＳＰＤ値に依存しないメモリ初期化サブタスクを、前記複数の処理ノードの他の１つに割り当てることとを含む、
コンピュータシステム。
【請求項１６】
前記メモリアレイは、ダブルデータレートバスを介して前記複数の処理ノードに接続された複数のデュアルインラインメモリモジュールを含む請求項１５のコンピュータシステム。
【請求項１７】
前記複数の処理ノードのうち前記第１の処理ノードは、前記複数の処理ノードで前記複数のサブタスクを実行した結果に基づいて、前記第１の処理ノードの前記メモリ制御器内の１つ以上のレジスタをプログラムすることによって、前記メモリ初期化タスクを実行し、これにより、前記複数の処理ノードの１つのみで前記メモリ初期化タスクを実行するこ

とによってブートを行うのに必要であろう時間と比べてブート時間要件を低減する請求項１５のコンピュータシステム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は概してデータ処理システムに関する。１つの態様においては、本発明はシステムブート(system boot)に際してのメモリ初期化のための方法及び装置に関する。

【背景技術】

【０００２】

データ処理システム又はコンピューティングシステムは、独立したコンピューティング能力を１人以上のユーザに与えるように設計されており、例えばメインフレーム、ミニコンピュータ、ワークステーション、サーバ、パーソナルコンピュータ、インターネット端末、ノートブック、及び埋め込み型システムを含む多くの形態において見出すことができる。一般的にコンピュータシステムアーキテクチャは、選択されたシステムコンポーネント、関連するメモリ及び制御論理（典型的にはシステムボード上の）並びに入力及び／又は出力（Ｉ／Ｏ）をシステムに提供する複数の周辺デバイスへの高速高帯域幅(high speed, high bandwidth)のアクセスを１つ以上のマイクロプロセッサコアにもたらすことを含むように設計される。例えば図１は、従来のコンピュータシステム１００のための例示的なアーキテクチャを示している。コンピュータシステム１００は、「ノース(north)」ブリッジ１０４を介してシステムメモリ１０８に接続される１つ以上のプロセッサ１０２を含む。典型的には、メモリアレイ１０８は１つ以上のメモリモジュールを含み、またメモリモジュールの追加又は交換のためのメモリスロットを含むこともある。ノースブリッジ回路１０４は種々のメモリモジュールとインタフェースするようにプログラム可能であり、また図示されるように、ノースブリッジ回路１０４は複数のメモリモジュール１０８の間で共有される。その結果、異なる複数のメモリモジュールにデータ投入される(populated)場合、ノースブリッジ回路１０４は、各メモリモジュールが正確に動作することを可能にするパラメータでプログラムされる必要がある。図示されるノースブリッジ回路１０４は、高速高帯域幅バス（例えばメモリバス１０７）を介してメモリ１０８に接続され、また高速高帯域幅バス（例えばエーリンク(Alink)又はＰＣＩバス）を介して「サウス(south)」ブリッジ１１２にも接続される。「サウス」ブリッジ１１２は、１つ以上のＩ／Ｏデバイス、例えば、周辺コンポーネント相互接続(Peripheral Component Interconnect)（ＰＣＩ）バス１１０（ＰＣＩバス１１０は次いでネットワークインタフェースカード（ＮＩＣ）１２０に接続される）、シリアルＡＴアタッチメント(serial AT Attachment)（ＳＡＴＡ）インタフェース１１４、ユニバーサルシリアルバス（ＵＳＢ）インタフェース１１６、及びローピンカウント(Low Pin Count)（ＬＰＣ）バス１１８（ＬＰＣバス１１８は次いでスーパー入力／出力制御器チップ（ＳｕｐｅｒＩ／Ｏ）１２２及びＢＩＯＳメモリ１２４に接続される）に接続される。理解されるであろうように、他のバス、デバイス、及び／又はサブシステム、例えば、キャッシュ、モデム、並列又はシリアルのインタフェース、ＳＣＳＩインタフェース等がコンピュータシステム１００内に含まれ得ることが分かるはずである。また、ノースブリッジ１０４及びサウスブリッジ１１２は、単一チップ又は複数のチップで実装することができ、総称して「チップセット」と称される。代替的には、ブリッジチップの機能の全て又は一部はプロセッサ１０２内にある場合がある。

【０００３】

コンピュータシステムは典型的には基本入力／出力システム（ＢＩＯＳ）と称される組み込み型ソフトウエアルーチンを含み、ＢＩＯＳは、プログラマ又はユーザがシステムハードウエアとやりとりすることができるように、システムハードウエアとオペレーティングシステムの間でのソフトウエアインタフェースを提供する。ＢＩＯＳ命令は、不揮発性メモリ１２４（例えばＲＯＭ（リードオンリメモリ）、ＰＲＯＭ（プログラム可能なＲＯＭ）、ＥＰＲＯＭ（消去可能なＲＯＭ）、ＥＥＰＲＯＭ（電気的に消去可能なＲＯＭ）、

フラッシュＲＡＭ（ランダムアクセスメモリ）等）内に記憶されており、そしてメモリを試験すること及び初期化すること、システムをインベントリすること(inventorying)及び初期化すること、並びにシステムを試験することを含む電源投入時の重要なコンピュータシステム機能を制御するために用いられる。電源投入時のこれらの機能は、「システムブート(system boot)」又は「システムをブートすること(booting the system)」と称され、システム電源投入のたびに又はシステムがリセットされるたびに生じ得る。図１に示される従来のコンピュータシステム１００においては、ＢＩＯＳは、コンピューティングシステム１００をブートするために、ブートストラッププロセッサ(boot strap processor)と称されるプロセッサ１０２の１つ上で実行する。動作においてブートストラッププロセッサ１０２は、ノースブリッジ１０４を介してメモリアレイ１０８と通信する。ノースブリッジ回路１０４は、ダブルデータレート（ＤＤＲ）バス等のメモリバス１０７とインタフェースする１つ以上のチャネル制御器に接続されるメモリ制御器（ＭＣ）１０６を含む。ノースブリッジ回路１０４はまた、標準的なバス１０９、例えば周辺コンポーネント相互接続（ＰＣＩ）バス上で１つ以上のサウスブリッジ１１２とも通信する。サウスブリッジ１１２は１つ以上の入力／出力（Ｉ／Ｏ）デバイス１２０，１２２，１２４と通信するが、追加的な又はより少数のデバイス（図示せず）がサウスブリッジ１１２に接続されているかもしれない。

【発明の概要】

【発明が解決しようとする課題】

【０００４】

システム初期化に際して、ブートストラッププロセッサ１０２は、相当な時間を必要とし得るシステムブートの間に、全てのメモリ試験及びクリーニングを行う。大型サーバシステムは、多くの場合にブートするために数分を必要とし、その間、他のプロセッサはアイドルである。図示される例では、ブートストラッププロセッサ上で実行中のＢＩＯＳは、ＤＤＲメモリバス１０７上でのＤＩＭＭメモリ１０８からの情報に基づいてストラッププロセッサ１０２からレジスタをプログラムすることによって、ノースブリッジ回路１０４内のメモリ制御器１０６を初期化する。メモリ初期化は、メモリモジュールのデータ投入状況(population)を検証することと、メモリの適切な動作（スタックビット無し）を検証することと、メモリを既知の値に初期化し又はクリーニングすることと、を含み得る。大きなメモリ１０８（例えば８、１６、又は３２ギガバイトのメモリ）を伴う従来のシステムは、メモリを初期化するのに数分を必要とすることがあり、特に「ＤＤＲトレーニング」処理は、ブートストラッププロセッサ１０２を用いて、メモリ初期化タスク間の依存を問わずタスクの順序付けられたシーケンスを連続的に実行することによって、メモリを初期化する。この遅延は、大量のメモリ１０８が各制御器１０６に接続される場合に深刻である。

【０００５】

そこで、本願発明者によって発見されてきた当該分野における種々の問題に対処する改良されたメモリ初期化デバイス、方法論、及びシステムに対する要求が存在し、ここで、従来の解決法及び技術の種々の限界及び不利益は、後述の図面及び詳細な説明を参照して本願の残りを精査した後に当業者にとって明らかになるはずであるが、背景技術欄のこの説明は、説明される主題が従来技術であるとの了解を意図するものではないことが理解されるべきである。

【課題を解決するための手段】

【０００６】

概して、本発明の実施形態は、先行するタスクに依存せずに順序不問で実行が達成され得るようにメモリ初期化タスクをコアに分散させることによって多重コアコンピュータシステム内でメモリを初期化するためのシステム、方法、及び装置を提供する。選択された実施形態においては、複数の処理ノード又はコア及び複数のシステムメモリコンポーネントを含むシステムが提供される。選択された実施形態においては、処理ノード又はコアの１つは、メモリ初期化タスクを多重ＣＰＵノード／コアによって処理される個々のサブタ

スクに分割するための制御ノード／コア機能を提供するように選択される。個々のサブタスクは、実行のための他の「スレーブ(slave)」コアに送られ又は分散させられる。各スレーブコアでは、単一又は複数のサブタスクを処理する資源をスレーブコアが有している場合に、サブタスクは実行のためにスケジューリングされる。結果として、所与のスレーブコアでの所与のサブタスクの実行の時間は、他のスレーブコアでの他のサブタスクの実行と重複してよいし又はしなくてもよい。その結果、スレーブコアは、サブタスクを並列に又はシリアルに実行することが可能である。サブタスクが処理された後に、それらは制御コアが元のタスクを完了するために再結合される。これにより、スレーブコアはサブタスクを並列に実行することができるので、同時タスク実行が可能になり、結果として全体の実行時間が減少すると共に最適な性能がもたらされる。

【０００７】

本発明の種々の実施形態によると、メモリ初期化タスク等のブート処理タスクを複数の処理ノードに行わせるための方法が提供される。方法の例示的な実施形態においては、制御処理ノードは、ブート処理タスク（例えばメモリ初期化タスク）を複数のブート処理サブタスク（例えばメモリ初期化サブタスク）に分割し、次いで対応する処理ノードを各サブタスクが有するように、制御処理ノードを含んでいてよい複数の処理ノードの間でサブタスクを分散させる。サブタスクを分散させるのに先立ち、制御処理ノードは、複数の処理ノードの間の通信リンクを初期化する。サブタスク分散の例を提供すると、処理ノードの１つは、シリアルプレゼンスディテクト(serial presence detect)（ＳＰＤ）値をＤＩＭＭメモリから読み出すメモリ初期化サブタスクを割り当てられる一方で、処理ノードの他の１つは、ＳＰＤ値に依存しない複雑な初期化タスクを実行するメモリ初期化サブタスクを割り当てられる。各メモリ初期化サブタスクは、実行のためにスケジューリングされ、次いで対応する処理ノードで実行されてサブタスク結果を生成する。選択された実施形態においては、対応する処理ノードでメモリ初期化サブタスクを実行するために、ソフトウエアルーチンが用いられる。制御処理ノードは、対応する処理ノードでのメモリ初期化サブタスクの実行に関する状況報告(status reports)を受信し、そして複数の処理ノードからのサブタスク結果が次いで制御処理ノードで結合され、ここで、複数のメモリ初期化サブタスクは、複数の処理ノードで並列に又はシーケンスに実行されてよい。例えば、処理ノードの１つは、それが割り当てられたメモリ初期化サブタスクを、処理ノードの他の１つであってその対応するメモリ初期化サブタスクを実行する他の１つと並列に実行することができる。

【０００８】

他の実施形態においては、システムメモリ（例えばＤＩＭＭベースのアレイ）と複数の処理ノードとを備えるコンピュータシステムが提供され、ここで、各処理ノードは、プロセッサコアと、他の単一又は複数のノードとの通信インタフェースと、システムメモリへの及びシステムメモリからのデータフローを管理するためのメモリ制御器と、を含む。処理ノードの１つはマスター処理ノードであり、マスター処理ノードは、メモリ初期化タスクを複数のメモリ初期化サブタスクに分割することと、対応する実行処理ノードを各割り当てられたメモリ初期化サブタスクが有するようにメモリ初期化サブタスクを１つ以上の実行処理ノード（マスター処理ノードを含んでいてよい）に割り当てることと、によってメモリ初期化タスクをシステムメモリ上で行うように構成される。各実行処理ノードは、マスター処理ノードによって割り当てられる各メモリ初期化サブタスクを実行すると共にサブタスク結果を生成するように構成される。マスター処理ノードは、実行処理ノードからメモリ初期化サブタスクの状況を受信するように、また、単一又は複数の実行処理ノードからのサブタスク結果を結合して、単一又は複数の実行処理ノードで複数のメモリ初期化サブタスクを実行した結果に基づいてマスター処理ノードのメモリ制御器内で１つ以上のレジスタをプログラムすることによってメモリ初期化タスクを行うように、構成される。選択された実施形態においては、マスター処理ノードは、実行処理ノードの１つに割り当てられたメモリ初期化サブタスクが、実行処理ノードの他の１つに割り当てられた他のメモリ初期化サブタスクに依存せずに順序不問で実行され得るように、メモリ初期化サブ

タスクを分割すると共に割り当てる。このようにして、複数のメモリ初期化サブタスクが実行処理ノードで並列に又はシーケンスに実行され得るように、第１の実行処理ノードは、そのメモリ初期化サブタスクを、単一又は複数の実行処理ノードの他の１つであってその対応するメモリ初期化サブタスクを実行している他の１つと並列に実行することができる。

【０００９】

更に他の実施形態においては、メモリアレイと複数の処理ノードとを備えるコンピュータシステムが提供される。メモリアレイは、ダブルデータレート（ＤＤＲ）バスを介して複数の処理ノードに接続される複数のデュアルインラインメモリモジュールとして実装されてよい。各処理ノードは、プロセッサ、メモリアレイと通信するためのメモリ制御器、及び複数の処理ノードの他の少なくとも１つとの通信インタフェース、として実装されてよい。動作において、第１の処理ノードは、メモリ初期化タスクを複数のサブタスクに分割すると共に複数のサブタスクを処理ノードに割り当てる。各処理ノードは、割り当てられたサブタスクを実行しながらメモリアレイの任意の部分からの情報を得ることができる。加えて、各処理ノードは、サブタスク実行結果を第１の処理ノードに戻し、それにより複数の処理ノードがメモリアレイのメモリ初期化を効率的に且つ迅速に行うことを可能にする。このようにして、第１の処理ノードは、そのメモリ制御器内で１つ以上のレジスタを、処理ノードで複数のサブタスクを実行した結果に基づいてプログラムすることによってメモリ初期化タスクを行い、それにより、処理ノードの１つのみでメモリ初期化タスクを実行することによってブートするのに必要であろう時間と比べてブート時間要求を低減する。

【図面の簡単な説明】

【００１０】

添付の図面を参照することによって、本発明はより良く理解されるであろうし、またその多くの目的、特徴、及び利益が当業者にとって明らかになるであろう。種々の図面を通して同じ参照番号の使用は同様の又は類似の要素を指定する。

【００１１】

【図１】図１は多重コアコンピュータシステムの単純化されたアーキテクチャのブロック図である。

【００１２】

【図２】図２は本発明の選択された実施形態に従う分散型多重コアメモリ初期化を伴うコンピューティングシステムアーキテクチャを示す図である。

【００１３】

【図３】図３は本発明の選択された実施形態に従う例示的な処理ノードを示す図である。

【００１４】

【図４】図４は例示的なＤＩＭＭ（デュアルインラインメモリモジュール）を示す図である。

【００１５】

【図５】図５は本発明の選択された実施形態に従う最適化されたメモリ初期化及び試験のフロー図である。

【発明を実施するための形態】

【００１６】

マスターコアの制御の下でメモリ初期化タスクをサブタスクに分割するために、分散型多重コアメモリ試験及び初期化の方法及び装置が提供される。マスターコアは次いで、多重コアによる別個の実行のためにサブタスクを送り又は分散し、完了した場合、結果は、元のタスクを完了させるための再結合のためにマスターコアに送り戻される。メモリ初期化タスクを別個のコアに分散させることによって、サブタスクの実行は、順序を問わず且つ先行するタスクに依存せずに達成され得る。この分散させられた実行は、先行するサブタスクが完了することを待たずにシーケンスを問わず幾つかのタスクが完了することを可能にし、それにより、メモリ初期化時間に対して要求される時間を短くしてデバイス性能

を向上させることができる。

【００１７】

本発明の種々の例示的な実施形態が添付図面を参照して以下に詳細に説明される。以下の説明において種々の詳細が記載されるが、これら特定の詳細なしに本発明が実施され得ること、及びデバイス設計者の特定の目標、例えば実装毎に変わるであろうプロセス技術又は設計関連の制約の遵守、を達成するために種々の実装特有の決定がここで説明される本発明に対してなされ得ること、が理解されるはずである。そのような開発努力は得てして複雑で且つ時間を要するものであろうが、この開示の利益を享受する当業者にとっては日常的な取り組みであるはずである。例えば、本発明を限定し又は曖昧にすることを避けるために、選択された態様が詳細によりはむしろブロック図形態で示される。ここに提供される詳細な説明の幾つかの部分は、コンピュータメモリ内に記憶されるデータに対して動作するアルゴリズム及び命令に関して提示される。そのような説明及び表現は、当業者が彼らの取り組みの内容を他の当業者に説明しまた伝えるために用いられる。概して、アルゴリズムは、所望の結果を結果としてもたらすステップの自己矛盾のないシーケンスを参照し、ここで「ステップ」は、必ずしもそうであるとは限らないが、記憶され、転送され、結合され、比較され、そして他の方法で操作されることが可能な電子的又は磁気的な信号の形態をとり得る物理量の操作を参照する。これらの信号をビット、値、要素、記号、文字、用語(terms)、数字、等として参照することは一般的な慣習である。これら及び類似の用語は、適切な物理量に関連していてよく、またこれらの量に適用される便利な指標にすぎない。特に断りのない限り、以下の議論から明らかであるように、「処理すること」又は「コンピューティング」又は「計算すること」又は「決定すること」又は「表示すること」等の用語を用いる議論は、コンピュータシステムのレジスタ及びメモリ内で物理的な（電子的な）量として表されるデータを、コンピュータシステムのメモリ若しくはレジスタ又は他のそのような情報記憶デバイス、伝送デバイス若しくは表示デバイス内で物理量として同様に表される他のデータへと操作し及び変換するコンピュータシステム又は同様の電子的コンピューティングデバイスの動作及び処理を参照することが理解される。

【００１８】

図２を参照すると、本発明の選択された実施形態に従う分散型多重コアメモリ初期化を伴うコンピューティングシステムアーキテクチャ２００が示されている。図示されるコンピューティングシステム２００においては、リンク２０３を介して互いに通信する複数の処理ノード２０２［０：３］がある。各処理ノード２０２は、例えば、プロセッサコア、メモリ制御器、及びリンクインタフェース回路を含む。リンク２０３は、例えばハイパートランスポート(HyperTransport)（ＨＴ）プロトコル等の分割トランザクション(split-transaction)バスプロトコルに従うデュアル点対点(point to point)リンクであってよい。リンク２０３は、上流へのデータフロー及び下流へのデータフローを含んでいてよい。リンク信号は、典型的には、クロック、制御、コマンド、アドレス、及びデータの情報等のトラフィック、並びにデバイス間で流れるトラフィックを制限し(qualify)また同期させるリンクサイドバンド信号を含む。処理ノード２０２の各メモリ制御器は、対応するメモリアレイ２０６［０：３］と通信する。処理ノード２０２及びメモリアレイ２０６はシステムの「コヒーレントな(coherent)」部分内にあり、ここでは全てのメモリトランザクションがコヒーレントである。処理ノード２０２の１つ以上の内部にはノースブリッジが含まれていてよく、あるいはノースブリッジは、別のＨＴリンクを介して処理ノード２０２の１つに結合される別個のノースブリッジデバイス２０８として設けられていてよい。いずれの場合にも、ノースブリッジデバイスは別のＨＴリンクを介してサウスブリッジ２１０に結合されていてよい。加えて、１つ以上のＩ／Ｏデバイス２１２がサウスブリッジ２１０に結合されていてよい。ＢＩＯＳ＿ＲＯＭ２１４はサウスブリッジ２１０に結合されていてよい。ノースブリッジ２０８、サウスブリッジ２１０、及びＩ／Ｏデバイス２１２は、システムの「非コヒーレントな」部分内にある。

【００１９】

各メモリアレイ２０６は、メモリモジュールの追加又は交換のための、データ投入された又は未投入の(populated or unpopulated)幾つかのメモリスロットから構成され得る。例えばサーバシステム２００においては、各メモリスロットは、全体的なサーバシステム２００が大きなメモリ、例えば３２ギガバイト（ＧＢｙｔｅｓ）の記憶容量を有するように、５１２メガバイト（Ｍｂｙｔｅｓ）の記憶容量を提供することができる。各処理ノード２０２のメモリ制御器は、異なってプログラムされ得るが、関連する処理ノード２０２に結合されるメモリモジュール２０６のローカル種とインタフェースするようにプログラムされる必要がある。システム２００は図示されるよりも複雑であり得ることが理解されるはずである。例えば、システムのコヒーレントな部分には追加的な処理ノード２０２があってよい。また、処理ノード２０２は梯子型アーキテクチャで図示されているが、処理ノード２０２は種々の方法で相互接続されてよく、また追加的な単一又は複数のＨＴリンクを介するより複雑な相互間の結合を有していてよい。

【００２０】

図３は本発明の選択された実施形態に従う例示的な処理ノード２０２を示している。図示されるように、処理ノード２０２は、プロセッサコア３０２、多重ＨＴリンクインタフェース３０４、及びメモリ制御器３０６を含む。プロセッサコア３０２は、プロセッサノードによって行われるメモリ初期化及び試験タスクのためのコード命令を実行する。例えばプロセッサノードが制御ノード又はマスターノードとして機能する場合には、プロセッサコア３０２は、メモリ初期化及び試験タスクを、他のプロセッサノードに割り当てられまた分散させられる複数のサブタスクへと分割するためのコード命令を実行する。一方、プロセッサノードが実行ノード又はスレーブノードの１つとして機能する場合には、プロセッサコア３０２は、割り当てられたメモリ初期化及び試験サブタスクを獲得し、スケジューリングし、また実行し、そして結果を制御／マスターノードへ戻すためのコード命令を実行する。クロスバー３０８は、要求、応答、及びブロードキャストメッセージをプロセッサ３０２及び／又は適切な単一若しくは複数のＨＴリンクインタフェース３０４へ転送する。要求、応答、及びブロードキャストメッセージの転送は、ＢＩＯＳにより構成される必要のある各処理ノード２０２内の多重構成ルーティングテーブル(multiple configuration routing tables)によって直接的に指令される(directed)。また、メモリ制御器３０６は、ＢＩＯＳによってプログラムされる必要のある動作パラメータのための多重構成レジスタを含む。

【００２１】

図４は例示的なＤＩＭＭ（デュアルインラインメモリモジュール）４００を示している。理解されるであろうように、メモリアレイ２０６は、幾つかの、典型的には８つのＤＩＭＭから構成されてよく、ここで各ＤＩＭＭ４００は、ＤＤＲ（ダブルデータレート）メモリチップ等の多重ランダムアクセスメモリ（ＲＡＭ）集積回路又はチップ４０２［１：Ｎ］を含む。また、ＤＩＭＭ４００はＥＣＣ（誤り訂正符号）回路４０４を有していてよい。ＥＣＣ回路４０４は、メモリ誤りを発見して且つ訂正することを可能にする誤り訂正符号を記憶する。更に、ＤＩＭＭ４００はＳＰＤ（シリアルプレゼンスディテクト）回路４０６を有していてよい。ＳＰＤ回路４０６は、ＤＩＭＭ４００の動作範囲を指定するリードオンリ情報及びデータシートにおいて見出されるであろう他の同様の情報を含む。例えばＳＰＤ回路４０６は、ＤＩＭＭ４００のメモリ記憶容量、最小サイクル時間等の動作パラメータ、ＣＡＳ待ち時間、等を識別する。

【００２２】

システム初期化に際しては、各メモリモジュール２０６は初期化され且つ試験される必要がある。このことは、メモリモジュールのデータ投入状況(population)を検証することと、メモリの適切な動作（スタックビット無し）を検証することと、メモリを既知の値に初期化し又はクリーニングすることと、を含んでいてよい。各メモリモジュールは、任意のメモリ誤りを訂正するためにＥＣＣを周期的に利用することに基づいて洗われ(scrubbed)得る。

【００２３】

図５は本発明の選択された実施形態に従う最適化されたメモリ初期化及び試験シーケンス５００のフロー図を示している。シーケンスは、ステップ５０２で、各処理ノードが他の処理ノード及びメモリと通信することができるようにシステムリンクを初期化することによって開始する。このことは、典型的には、ブートストラッププロセッサ上で動作しているＢＩＯＳによって行われる。代替的には、ハードワイヤードのシステム(hard wired system)においては、処理ノードは、システム電源投入に際して互いに及びメモリと通信するように自動的に構成されてよい。

【００２４】

ステップ５０４では、ブートストラッププロセッサ（ＢＳＰ）として指定された処理ノードは、メモリ初期化及び／又は試験プロセスが、メモリ初期化及び／又は試験タスクを複数のサブタスクに分割し、次いで各タスクを実行処理ノードに割り当てると共に分散させることよって開始するようにさせる。例えばＢＳＰは、割り当てられたサブタスクと共に開始メッセージを各処理ノードへ送ることができる。追加的に又は代替的に、割り当てられたサブタスクがＢＳＰによって送り届けられ得るように又は割り当てられたサブタスクが単一若しくは複数の実行処理ノードによってレトリーブされ(retrieved)得るように、メモリ初期化及び試験プロセスが開始すべきであることを示すビットが各処理ノード内に書き込まれてよい。代替的には、任意のプロセッサ上でのコード実行が１つ以上のレジスタに書き込む場合等に、指令された割り込みがプロセスを開始させることができる。書き込まれた値は、割り当てられたサブタスクに対して１つ以上のプロセッサが特定のアクションをとるようにそれらに指令することができる。例えば書き込まれた値は、指令された割り込みの目標を識別するノード識別子と、どこでコード実行を開始するのかを直接的に又は間接的に指定することを介して実行するための一連の命令及びサブタスクを示すベクターと、を含むことができる。必要であれば、目標とされているプロセッサがベクターにより指定される割り当てられたサブタスクの実行を開始することを促す開始メッセージを、目標とされているプロセッサにハードウエアメカニズムが送り届けることができる。

【００２５】

ステップ５０６では、各処理ノード２０２は、その割り当てられたメモリ初期化及び試験サブタスクを行い、それらは関連するメモリ２０６に対応していてもしていなくてもよい。図２に示される例示的な実装においては、コンピューティングシステム２００は、４つのコア、１つの制御コア２０２［０］、及び３つのスレーブコア２０２［１：３］を含む。メモリ初期化／試験タスクは、制御コア２０２［０］によって複数のサブタスク（例えばサブタスク１、サブタスク２、サブタスク３等）に分割される。制御コア２０２［０］は、サブタスク１を第１のスレーブコア２０２［１］へ送り、サブタスク２を第２のスレーブコア２０２［２］へ送り、そしてサブタスク３を第３のスレーブコア２０２［３］へ送る。ステップ５０６では、各スレーブコア２０２［１：３］はそのサブタスクを受け取り、そして実行のためにそれをスケジューリングする。また、制御コア２０２［０］は、異なるサブタスクの実行を継続してよく、あるいはスレーブコア上での実行が完了するのを待ってよく又はそれ自身の別個のサブタスクを実行してよい。各スレーブコア２０２［１：３］は、それがタスクを処理する資源を有しているタイミングでサブタスクを実行し、その結果、実行の時間は他のスレーブコアと重なるかもしれないし重ならないかもしれない。結果として、割り当てられたメモリ初期化／試験サブタスクは、スレーブコアの資源の能力に応じて、コア２０２によってシリアルに又は並列に実行されることが可能である。サブタスク実行は、処理ノードによって又は各処理ノードのノースブリッジ内の回路によって行われ得る。

【００２６】

理解されるであろうように、制御コア及びスレーブコアは、独立した複数のプラットフォームシステム上にあってよい。そのような実装においては、制御コアは、各参加しているプラットフォームシステム上の１つ以上のコアが、その割り当てられている単一又は複数のサブタスクを各々処理し得るように、サブタスクを当該システムへ送る。このシナリオにおいては、実行はシリアルに又は並列になされてよい。メモリ初期化タスク実行の分

散させられた性質の結果として、実行が並列に又はシリアルになされるという事実は、タスクの完了とは関係がない。タスクが並列に実行される場合、タスクはより速く完了するであろうが、データを転送する待ち時間は、制御コアでのシリアル化を結果としてもたらす可能性がある。

【００２７】

ステップ５０８では、各処理ノードは状況をＢＰＳに報告する。状況報告することは、例えば、メモリ初期化及び試験の間に連続的であってよく、あるいは周期的であってよく、あるいはメモリ初期化及び試験が完了したときであってよい。また、状況報告は種々の方法で報告され得る。例えば、ブートストラッププロセッサは、各処理ノードへ周期的にクエリーを送り、割り当てられたサブタスクが実行され終わったかどうかを問い合わせることができる。代替的には、各処理ノードは、その状況をブートストラッププロセッサに示すメッセージを送ることができる。別の代替案としては、各処理ノードは、ローカルレジスタに又はブートストラッププロセッサのローカルメモリに書き込んで状況を報告することもできる。

【００２８】

全ての処理ノードがそれらの割り当てられたメモリ初期化／試験を完了した後に、ステップ５１０では、ブートストラッププロセッサがシステムブートを継続し、そして他の処理ノードは停止する。メモリ初期化／試験タスクが実行されるべきである場合（決定５１２が肯定）には、処理はステップ５０４で再び始まる。そうでない場合には、処理はステップ５１４で終了する。

【００２９】

メモリ初期化／試験サブタスクを異なる処理ノードに割り当てることによって、システムブートは極めて速く完了することができる。幾つかのサブタスクは先行するサブタスクの完了に依存し、また他のタスクは先行するサブタスクが完了するのを待たずにシーケンスからは自由に完了され得るのではあるけれど、タスクの依存性の有無を問わずにサブタスクを分割し、割り当て、そして分散させることによって、全体的な性能及び起動時間を改善することができる。このことは、サブタスクの幾つかは、タスクが実行順序になる前にそれらが現れ且つそれらの前に依存性のある実行を有していないことを待つ必要性なしに第１の処理ノードによって実行されるという理由で「システムブート(system boot)」又は「システムをブートすること(booting the system)」の時間が低減されるという事実に由来している。言うまでもなく、各サブタスクに対する個々の依存性を考慮に入れることによってブートストラッププロセッサがメモリ初期化／試験タスクをサブタスクに分割すれば、メモリ初期化は更に改善され得る。例えば、１つ以上の指定された処理ノードが他のサブタスクの結果に依存するサブタスクを割り当てられてよい一方で、１つ以上の他の指定された処理ノードは、先行する単一又は複数のサブタスクが完了するのを待たずにシーケンスからは自由に完了され得るサブタスクを割り当てられてよい。

【００３０】

ここに説明されるように、分散させられたメモリ初期化ルーチンは、分散及び別個のノードによる実行のためにメモリタスクを分割することによって多重プロセッサシステム内の多重ノード間で分散させられるメモリを構成し、初期化し、そして試験することが可能なＢＩＯＳコードと共に実装されてよく、結果として改良された実行時間をもたらすことができる。例えば、１つのコアはＳＭバス(SMBus)（低速バス）からＳＰＤを読み出すタスクを課せられ得る一方で、別のコアはＳＰＤ依存でない複雑な初期化タスクを実行することができる。開示される分散型処理手法はまた、特定のＢＩＯＳアルゴリズムの精度を改善することができる。例えば、より速い実行のためにＢＩＯＳが最適化されるようにアルゴリズムをトレーニングすることが開発されてきており、それにより遅延設定の最適配置を決定するために用いられ得るデータアイが生成される。しかし、生成されるデータアイは、トレーニングアルゴリズムを実行するコアにより処理され得るデータの量によって制限される。ここに開示される分散型処理アプローチは、主コアを用いてデータを集めそして情報を処理しまた結果を主コアに戻すシステム内の他のコアへの分散のために当該デ

ータを分割することによって精度を改善するために用いられ得る。開示される分割及び分散型スキームはまた結果として、割り当てられたサブタスクが順序を問わず実行される場合に、より速いブート時間をもたらす。更に別の可能な利益は、ブートストラッププロセッサにＢＩＯＳ初期化タスクの全てを行わせることに代えて、サブタスクが各コアに分散させられるようにすることによって、処理コアがＢＩＯＳ初期化の間に効率的に使用され得ることである。

【００３１】

更なる例、及びメモリ初期化タスクがどのようにして分割されそして複数の処理ノードにわたって分散させられ得るのかの説明を提供するために、以下のメモリ初期化ルーチン又はアルゴリズムが次に説明され、ここでサブタスクｓｔの総数はシステム内のコアの総数以下である。

【００３２】

Ｔ＝　｛ｓｔ［１］＋ｓｔ［２］＋ｓｔ［３］＋…ｓｔ［Ｘ］｝、ここでｓｔはメモリ初期化サブタスクである；

【００３３】

Ｃ０＝　「ｓｔ」を結合して「Ｔ」を生成することに関与する主初期化コア；

【００３４】

ｃ［ｎ］＝　サブタスクコア、但しｎ＝｛１，…，Ｎ｝；

【００３５】

Ｎ＝　Ｎ≧Ｘとなるコアの最大数；

【００３６】

Ｘ＝　任意のＴに対するサブタスクの最大数；

【００３７】

ｎ－　現在のコア数；

【００３８】

ｘ－　現在のサブタスク数；

【００３９】

ステップ１－　サブタスクをコアに分散させる；

【００４０】

ステップ２－　ｃ［ｎ］に対して；

【００４１】

ステップ３－　ｓｔ［ｘ］に対して；

【００４２】

ステップ４－　ｃ０からｃ［ｎ］までｓｔ［ｘ］初期化情報及びコードを送る；

【００４３】

ステップ５－　ｃ［ｎ］はｓ［ｘ］の実行を開始；

【００４４】

ステップ６－　ｃ０はｃ［ｎ］からｓｔ［ｘ］結果を受け取り実行を開始；

【００４５】

ステップ７－　ｃ０はｎ及びｘをインクリメント；

【００４６】

ステップ８－　ｃ０は全てのＸに対してステップ２～７を繰り返す；

【００４７】

ステップ９－　ｎ＝Ｎの場合にステップ１０へ移動；

【００４８】

ステップ１０－　ｃ０は分散させられたサブタスクデータを収集；

【００４９】

ステップ１１－　ｃ［ｎ］に対して；

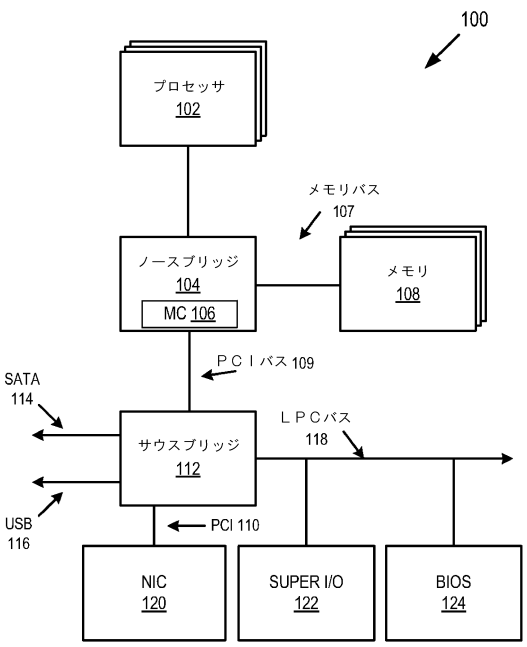
【００５０】

ステップ１２－　ｃ０はｃ［ｎ］からｓｔ［ｘ］データを要求；

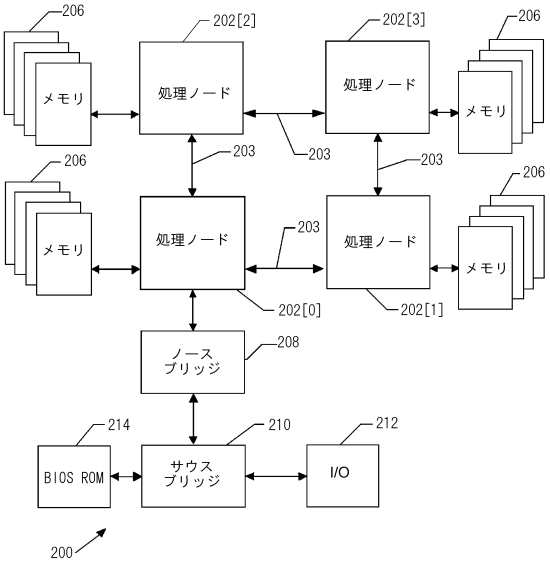
【００５１】
ステップ１３－　ｃ［ｎ］は実行を完了して結果をｃ０へ送る；
【００５２】
ステップ１４－　ｃ０はｎ及びｘをインクリメント；
【００５３】
ステップ１５－　ｃ０は全てのｎ＝Ｎになるまで全てのＸに対してステップ１１～１４を繰り返す；
【００５４】
ステップ１６－　ｃ０は全てのｓｔ［１－Ｘ］からのデータを再結合することによってＴを生成；
【００５５】
ステップ１７－　全てのステップが次のタスクのために繰り返される。
【００５６】
理解されるであろうように、前述のルーチンは、疑似コード形態で表現され得るし且つ／又は特定の命令セットアーキテクチャのためのそれらが使用している対応するアセンブリ言語若しくは高レベル言語コードへとトランスレートされ得る。また、ここに説明される動作は、コンピュータシステムユーザによって直接エントリされたコマンド及び／又はソフトウエアモジュールによって実行されるステップを含み得る。ここに参照される任意のステップの機能性は、モジュール又はモジュールの一部の機能性に対応していてよい。ソフトウエアモジュールに加えて、上述のフロー又はフローの一部は、アプリケーション命令として実装され得る。ここに参照される動作は、モジュール又はモジュールの一部（例えばソフトウエア、ファームウエア、又はハードウエアのモジュール）であってよい。例えば、ここで論じられるソフトウエアモジュールは、スクリプト、バッチ若しくは他の実行可能なファイル又はそのようなファイルの組み合わせ及び／若しくは一部を含んでいてよい。ソフトウエアモジュールは、コンピュータ可読媒体上でエンコードされるコンピュータプログラム又はそのサブルーチンを含んでいてよい。
【００５７】
加えて、当業者であれば、モジュール間の境界は単に例示的なものであり、また代替的な実施形態はモジュールを合併し又はその代わりにモジュールの機能性の分解を課する場合があることを理解するはずである。例えば、ここで論じられるモジュールは、多重コンピュータ処理として実行されるべきサブモジュールへと分解されてよい。また、代替的な実施形態は特定のモジュール又はサブモジュールの多重インスタンスを結合してよい。更に、当業者であれば、例示的な実施形態において説明される動作は例示のみを目的としていることを認識するはずである。本発明に従う追加的な動作においては、動作は結合されてよく、あるいは動作の機能性は分散させられてよい。従って、ここに説明されるフロー、その動作、及びそのためのモジュールは、フローの動作を実行するように構成されるコンピュータシステム上で実行されてよく且つ／又はコンピュータ可読媒体から実行されてよい。フローは、フローを実行するコンピュータシステムを構成するための機械可読媒体及び／又はコンピュータ可読媒体において具現化されてよい。従って、ソフトウエアモジュールは、コンピュータシステムメモリ内に記憶され且つ／又はコンピュータシステムメモリへ伝送されて、モジュールの機能を行うためのコンピュータシステムを構成してよい。
【００５８】
上に開示される特定の実施形態は例示のみを目的としており、また本発明は異なる方法であるがここでの教示の利益を有する当業者に明らかな均等な方法において修正されそして実施され得るので、上に開示される特定の実施形態は本発明を限定するものとして解釈されるべきではない。従って、前述の説明は、記載されている特定の形態に本発明を限定することを意図するものではないが、一方で、本発明の精神及び範囲の最も広い形態から逸脱することなしに当業者が種々の変更、置換及び代替をなし得ると当業者が理解するはずであるように添付の特許請求の範囲によって定義される本発明の精神及び範囲内に含ま

れるであろう代替、修正、及び均等なものに及ぶことが意図されている。

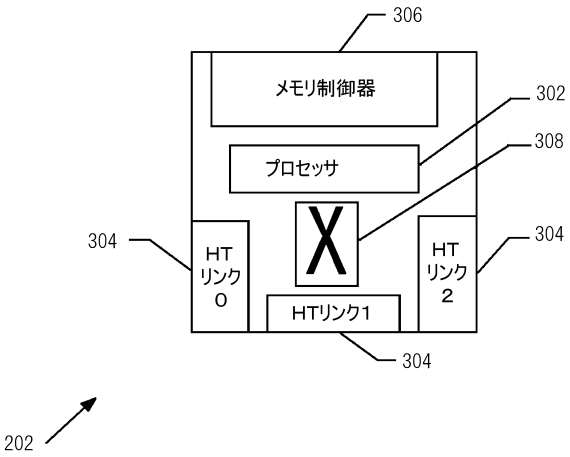
【図１】



【図２】



【図３】



【図４】

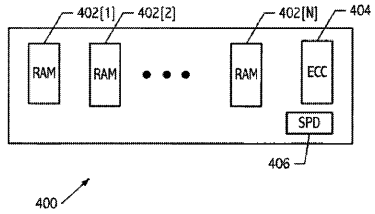
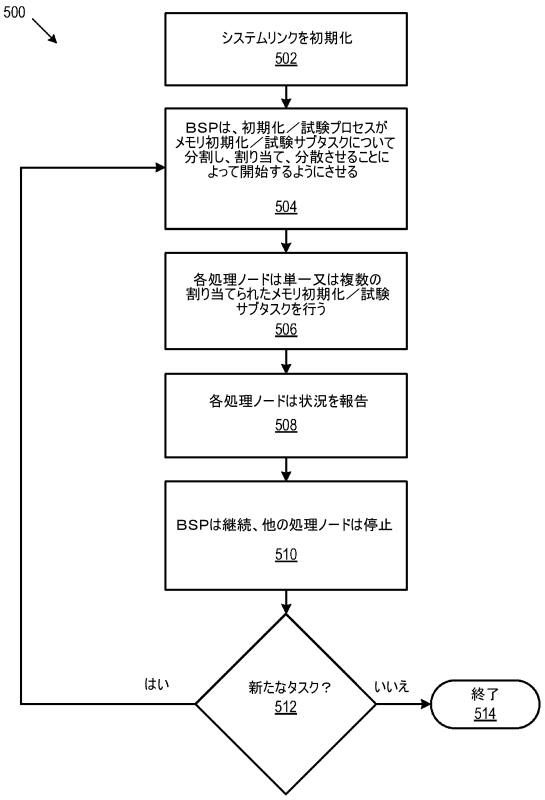


Figure 4

【図５】



フロントページの続き

(74)代理人 100162156
弁理士 村雨 圭介
(72)発明者 オズウィン ハウスティ
アメリカ合衆国 ７８７４９ テキサス州、オースティン、センデラ メサ ドライブ ８８１１

審査官 三坂 敏夫

(56)参考文献 国際公開第００／０１７７５０（WO，Ａ１）
米国特許出願公開第２００６／０１４９９５９（ＵＳ，Ａ１）
米国特許出願公開第２００８／０１６２８７８（ＵＳ，Ａ１）
米国特許第０５６７３３８８（ＵＳ，Ａ）
米国特許第０６１５８０００（ＵＳ，Ａ）
特開２００９－２１７２９７（ＪＰ，Ａ）
特開２００６－３０９３３２（ＪＰ，Ａ）
特開２００４－１４５７３３（ＪＰ，Ａ）
特開２００９－１１０４２９（ＪＰ，Ａ）
特表２００２－５２５７４５（ＪＰ，Ａ）
特表２００９－５０６４３６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／５０
Ｇ０６Ｆ １２／００
Ｇ０６Ｆ １５／１７７