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

特許第6474426号
(P6474426)

(45)発行日　平成31年2月27日(2019.2.27)　(24)登録日　平成31年2月8日(2019.2.8)

(51)Int.Cl.			F I		
G06F	9/46	(2006.01)	G06F	9/46	410
G06F	9/455	(2006.01)	G06F	9/455	150
G06F	9/30	(2018.01)	G06F	9/30	350A

請求項の数 14　(全 32 頁)

(21)出願番号　特願2016-558621 (P2016-558621)
(86)(22)出願日　平成27年3月17日(2015.3.17)
(65)公表番号　特表2017-513121 (P2017-513121A)
(43)公表日　平成29年5月25日(2017.5.25)
(86)国際出願番号　PCT/EP2015/055516
(87)国際公開番号　WO2015/144489
(87)国際公開日　平成27年10月1日(2015.10.1)
審査請求日　平成29年11月16日(2017.11.16)
(31)優先権主張番号　14/226,989
(32)優先日　平成26年3月27日(2014.3.27)
(33)優先権主張国　米国(US)

(73)特許権者　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
INTERNATIONAL BUSINESS MACHINES CORPORATION
アメリカ合衆国10504　ニューヨーク州　アーモンク　ニュー　オーチャード　ロード
New Orchard Road, Armonk, New York 10504, United States of America
(74)代理人　100108501
弁理士　上野　剛史

最終頁に続く

(54)【発明の名称】マルチスレッディング能力情報取得

(57)【特許請求の範囲】
【請求項１】
コンピュータシステムであって、
単一スレッド（ＳＴ）モードとマルチスレッディング（ＭＴ）モードとの間で構成可能なコアと、メモリとを含むコンフィギュレーションであって、前記ＳＴモードは、第一スレッドをサポートし、前記ＭＴモードは、前記コアの共有リソース上で前記第一スレッド及び１以上の第二スレッドをサポートする、コンフィギュレーションを備え、
前記コアにより、前記メモリからフェッチしたマルチスレッディング能力情報取得（ＲＭＴＣＩ）命令を実行することを含み、前記実行することは、
前記コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を、前記メモリに格納することと、
前記格納したスレッド識別情報に基づいて、前記コンフィギュレーションは以前ＭＴモードがイネーブルであったか否か判定することと、
を含む、コンピュータシステム。
【請求項２】
前記ＲＭＴＣＩ命令を実行したコアは、ＳＴモードであり、
前記ＳＴモードより以前にＭＴモードがイネーブルであったと判定された場合、当該ＭＴモードをリ・イネーブルにする、請求項１に記載のコンピュータシステム。
【請求項３】
前記ＭＴモードをリ・イネーブルにすると、前記以前のＭＴモードにおける第二スレッ

ドの数が前記コア上にリストアされる、請求項２に記載のコンピュータシステム。
【請求項４】
前記ＲＭＴＣＩ命令を実行したコアは、ＳＴモードであり、
前記ＳＴモードより以前にＭＴモードがイネーブルであったと判定されなかった場合、当該ＳＴモードを保つ、請求項１に記載のコンピュータシステム。
【請求項５】
前記ＲＭＴＣＩ命令が、サービス・コール（ＳＥＲＶＣ）命令又はシステム情報格納（ＳＴＳＩ）命令のいずれか１つであり、前記ＳＥＲＶＣ命令は、前記スレッド識別情報を前記メモリ内の応答ブロックに格納するように構成され、前記ＳＴＳＩ命令は、前記スレッド識別情報を前記メモリ内のシステム情報ブロック（ＳＹＳＩＢ）内に格納するように構成される、請求項１に記載のコンピュータシステム。
【請求項６】
前記応答ブロックが、サポートされるプログラム指定最大スレッド識別値を示すビットのマスクをさらに含む、請求項５に記載のコンピュータシステム。
【請求項７】
前記格納したスレッド識別情報が、前記コアがマルチスレッディングをサポートするか否かを示すＭＴインストール識別子を含む、請求項１に記載のコンピュータシステム。
【請求項８】
前記格納したスレッド識別情報が、前記コアの最大サポートスレッド数を示す最大スレッド識別子を含む、請求項１に記載のコンピュータシステム。
【請求項９】
前記格納したスレッド識別情報が、前記コンフィギュレーションの前記コアに直近に設定されたプログラム指定最大スレッド識別子を含む、請求項１に記載のコンピュータシステム。
【請求項１０】
前記直近に設定されたプログラム指定最大スレッド識別子が示す非ゼロ値に基づいて、前記コンフィギュレーションは以前ＭＴモードがイネーブルであったかを判定すること、をさらに含む、請求項９に記載のコンピュータシステム。
【請求項１１】
前記コンフィギュレーションが複数のコアタイプをサポートし、コアの最大サポートスレッド数を示す最大スレッド識別子が、前記コンフィギュレーション内の前記コアタイプの各々に対して維持される、請求項１に記載のコンピュータシステム。
【請求項１２】
単一スレッド（ＳＴ）モードとマルチスレッディング（ＭＴ）モードとの間で構成可能なコアと、メモリとを含むコンフィギュレーションにおけるマルチスレッディング能力情報取得のためのコンピュータ実装の方法であって、前記ＳＴモードは、第一スレッドをサポートし、前記ＭＴモードは、前記コアの共有リソース上で前記第一スレッド及び１以上の第二スレッドをサポートし、前記方法は、
前記コアにより、前記メモリからフェッチしたマルチスレッディング能力情報取得（ＲＭＴＣＩ）命令を実行すること、
を含み、前記実行は、
前記コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を、前記メモリに格納することと、
前記格納したスレッド識別情報に基づいて、前記コンフィギュレーションは以前ＭＴモードがイネーブルであったか否か判定することと、
を含む、方法。
【請求項１３】
請求項１乃至１１の何れか一項に記載のコンピュータシステムのコアによりマルチスレッディング能力情報取得（ＲＭＴＣＩ）命令の実行をすることを含む方法。
【請求項１４】

単一スレッド（ＳＴ）モードとマルチスレッディング（ＭＴ）モードとの間で構成可能なコアと、メモリとを含むコンフィギュレーションにおけるマルチスレッディング能力情報取得のためのコンピュータ・プログラムであって、前記ＳＴモードは、第一スレッドをサポートし、前記ＭＴモードは、前記コアの共有リソース上に前記第一スレッド及び１以上の第二スレッドをサポートし、前記コンピュータ・プログラムは、

前記コアに、前記メモリからフェッチしたマルチスレッディング能力情報取得（ＲＭＴＣＩ）命令を実行させるためのものであり、前記実行は、

前記コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を、前記メモリに格納することと、

前記格納したスレッド識別情報に基づいて、前記コンフィギュレーションは以前ＭＴモードがイネーブルであったか否か判定することと、

を含む、コンピュータ・プログラム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、一般に多重スレッドをサポートするコンピュータシステムに関し、より詳細には、コンピュータシステムにおけるマルチスレッディング能力情報の取得に関する。

【背景技術】

【０００２】

コンピュータシステムのプロセッサ速度は、過去１０年間にわたって速くなってきているが、このようなコンピュータシステムのメモリにアクセスすることができる速度はそれに比例して速くなっていない。それゆえ、プロセッサのサイクル時間が速くなるにつれ、メモリからデータをフェッチするための待機の遅延がますます顕著になってきている。このような遅延の影響は、種々のレベルのキャッシングにより軽減されており、最近のプロセッサではマルチスレッディング（ＭＴ）により軽減されている。

【０００３】

ＭＴは、プロセッサの種々のコア・リソースを、スレッドとして知られる複数の命令ストリームで共有することを可能にする。コア・リソースは、実行ユニット、キャッシュ、変換ルックアサイド・バッファ（ＴＬＢ）などを含むことができ、これらを集合的に一般にコアと呼ぶことができる。１つのスレッドにおけるキャッシュ・ミス又はその他の遅延により生じる待ち時間の間、１つ又は複数の他のスレッドがコア・リソースを利用することができ、これによりコア・リソースの使用率が高まる。スーパー・スカラー・プロセッサの同時マルチスレッディング（ＳＭＴ）実装では、多重スレッドが、１つ又は複数のコアのコア・リソースによるサービスを同時に受けることができる。

【０００４】

現代のハードウェア・プラットフォームにおいて、ＭＴは、典型的には、ＭＴハードウェア上で実行されるオペレーティングシステム（ＯＳ）に対して透過的な方式で実装される。この特徴の１つの態様は、ＯＳがＭＴハードウェアを利用するのに修正を必要としないことである。しかしながら、ＯＳに関して透過的ＭＴ動作の結果として、応答時間、容量（capacity）の準備、容量の計画、及びビリングの高い変動性が生じることがある。この変動性は、ＯＳが、そのタスクがコアの排他的制御を有しているのか又はそのタスクがコアを共有するスレッドとして実行されているのか気づかない（unaware）ことにより生じる場合がある。設計により、ＭＴ可能ハードウェア上のメモリ集約的作業負荷のための最高容量は、コア使用時に高い平均スレッド密度がある場合に達成可能である。付加的な容量は、ＭＴにより提供されるキャッシュ活用の増大によるものであり得る。ＯＳが、利用されるコアに対して高い平均スレッド密度を一貫して維持しない場合、ＭＴにより提供される付加的な総スループット容量は利用可能にならない。例えば、ハードウェアが、コンピュータ利用率が低いときには１コア当たり単一のＭＴスレッドを実行し、コンピュータ利用率が高いときには高いスレッド密度で実行する場合、作業負荷に対してどれだけの

総ＭＴコンピュート容量が利用可能であるのかを判定することは非常に困難であり得る。このＭＴスレッド活用におけるハードウェア変動性は、トランザクション応答時間及びビリングの両方において、容量に関して先に説明したのと同様の様式で変動性をもたらす場合がある。

【先行技術文献】

【非特許文献】

【０００５】

【非特許文献１】「ｚ／Ａｒｃｈｉｔｅｃｔｕｒｅ Ｐｒｉｎｃｉｐｌｅｓ ｏｆ Ｏｐｅｒａｔｉｏｎ」ＩＢＭ刊行番号ＳＡ２２－７８３２－０９、２０１２年８月

【発明の概要】

【発明が解決しようとする課題】

【０００６】

本発明の目的は、マルチスレッディング能力情報取得のためのシステム、方法、及びコンピュータ・プログラム製品を提供することである。

【課題を解決するための手段】

【０００７】

実施形態は、マルチスレッディング能力情報取得のためのシステム、方法、及びコンピュータ・プログラム製品を含む。１つの態様によれば、コンピュータシステムは、単一スレッド（ＳＴ）モードとマルチスレッディング（ＭＴ）モードとの間で構成可能な１つ又は複数のコアを有するコンフィギュレーションを含む。ＳＴモードは、第一スレッドをサポートし、ＭＴモードは、各コアの共有リソース上で第一スレッド及び１以上の第二スレッドをサポートする。コンピュータシステムはまた、コンフィギュレーションの利用を制御して、コアにより、マルチスレッディング能力情報取得（ｒｅｔｒｉｅｖｅ ｍｕｌｔｉｔｈｒｅａｄｉｎｇ ｃａｐａｂｉｌｉｔｙ ｉｎｆｏｒｍａｔｉｏｎ、ＲＭＴＣＩ）命令を実行することを含む方法を行うように構成された、マルチスレッディング・ファシリティを含む。実行は、コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を取得することと、取得したスレッド識別情報を格納することと、を含む。

【０００８】

別の態様によれば、コンフィギュレーションにおけるマルチスレッディング能力情報取得のためのコンピュータ実装の方法が提供される。コンフィギュレーションは、ＳＴモードとＭＴモードとの間で構成可能な１つ又は複数のコアを含み、ここでＳＴモードは、第一スレッドをサポートし、ＭＴモードは、各コアの共有リソース上で第一スレッド及び１以上の第二スレッドをサポートする。方法は、コアにより、マルチスレッディング能力情報取得命令を実行することを含む。実行は、コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を取得することと、取得したスレッド識別情報を格納することと、を含む。

【０００９】

さらなる態様は、コンフィギュレーションにおけるマルチスレッディング能力情報取得のためのコンピュータ・プログラム製品を含む。コンフィギュレーションは、ＳＴモードとＭＴモードとの間で構成可能な１つ又は複数のコアを含み、ここでＳＴモードは、第一スレッドをサポートし、ＭＴモードは、各コアの共有リソース上で第一スレッド及び１以上の第二スレッドをサポートする。コンピュータ・プログラム製品は、具体化されたプログラム命令を有するコンピュータ可読ストレージ媒体を含み、ここでコンピュータ可読媒体は、信号ではない。プログラム命令は、処理回路により可読であり、処理回路に方法を実行させる。方法は、コアにより、マルチスレッディング能力情報取得命令を実行することを含む。実行は、コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を取得することと、取得したスレッド識別情報を格納することと、を含む。

【００１０】

実施形態であるとみなされる主題は、本明細書の結論部にある特許請求の範囲において

特に指摘され、明確に特許請求される。実施形態の上記及び他の特徴、並びに利点は、添付図面と共に解釈される以下の詳細な説明から明らかである。

【図面の簡単な説明】

【００１１】

【図１】実施形態により実装することができるコンピューティング環境を示す。

【図２】実施形態により実装することができるコンピューティング環境を示す。

【図３】実施形態により実装することができるコアの処理回路を示す。

【図４】実施形態により実装することができるコンピューティング環境を示す。

【図５】実施形態により実装することができるコンピューティング環境におけるハイパーバイザ・コンテキスト保持の例を示す。

【図６】実施形態によるマルチスレッディングの動的イネーブルメントのプロセスフローを示す。

【図７】実施形態によるＣＰＵアドレス拡張プロセスの例を示す。

【図８】実施形態によるＣＰＵアドレス短縮プロセスの例を示す。

【図９】実施形態によるマルチスレッディング設定（ｓｅｔ－ｍｕｌｔｉｔｈｒｅａｄｉｎｇ）指令のプロセスフローを示す。

【図１０】実施形態によるマルチスレッディング能力情報の格納の例を示す。

【図１１】実施形態によるマルチスレッディング能力を判定するためのプロセスフローを示す。

【図１２】実施形態による種々のスレッド・コンテキスト位置の例を示す。

【図１３】実施形態によるマルチスレッディング・レジスタ保存の例を示す。

【図１４】実施形態によるマルチスレッディング・レジスタ保存のためのプロセスフローを示す。

【図１５】実施形態によるマルチスレッディング・レジスタ復元の例を示す。

【図１６】実施形態によるマルチスレッディング・レジスタ復元のためのプロセスフローを示す。

【図１７】実施形態によるコンピュータ可読媒体を示す。

【発明を実施するための形態】

【００１２】

例示的な実施形態は、単一スレッド及びマルチスレッディング動作モードをサポートするコンピュータシステムにおける、マルチスレッディング動作を提供する。本明細書において用いる場合、論理スレッドは、単一命令ストリーム及びそれに関連付けられた状態を指す。すなわち、アーキテクチャ・レベルにおいて、各論理スレッドは、独立した中央処理ユニット（ＣＰＵ）又はプロセッサを表す。ハードウェア・レベルにおいて、スレッドは、論理スレッドに関連付けられた命令ストリームの実行であり、スレッドがディスパッチされたとき、そのゲスト状態の維持と組み合わされる。したがって、「スレッド」及び「ＣＰＵ」という用語は、本明細書において互換的に用いることができる。

【００１３】

例示的な実施形態において、ＣＰＵは、命令実行、割込み動作、タイミング機能、初期プログラム・ロード、及び他のマシン関連機能のための、順番付け及び処理ファシリティを収容する。ＣＰＵは、種々の下層の物理的実装にマップすることができる論理関数を定義する。ＣＰＵは、命令を実行する際に、固定長の２進整数及び浮動小数点数（例えば、２進、１０進、１６進）、可変長の１０進整数、並びに固定長又は可変長いずれかの論理情報を処理することができる。処理は、並列又は直列とすることができる。処理要素の幅、シフティング経路の多重度、及び異なる型式の算術を実行する同時性度は、論理的結果に影響を与えることなく、ＣＰＵの型式ごとに異なったものであり得る。

【００１４】

ＣＰＵが実行する命令は、汎用命令、１０進命令、浮動小数点サポート（ＦＰＳ）命令、２進浮動小数点（ＢＦＰ）命令、１０進浮動小数点（ＤＦＰ）命令、１６進浮動小数点（ＨＦＰ）命令、制御命令、及びＩ／Ｏ命令などの、多数の命令クラスを含むことができ

る。汎用命令は、２進整数算術演算、並びに論理演算、分岐演算、及びその他の非算術演算を行う際に用いることができる。１０進命令は、１０進形式のデータ上で動作する。ＢＦＰ、ＤＦＰ、及びＨＦＰ命令は、それぞれＢＦＰ、ＤＦＰ、及びＨＦＰ形式のデータ上で動作し、一方、ＦＰＳ命令は、形式に無関係に浮動小数点データ上で動作し、又は１つの形式を別の形式に変換する。特権制御命令及びＩ／Ｏ命令は、ＣＰＵがスーパバイザ状態にあるときに実行されることができ、準特権制御命令は、適切な認証機構に従属する、問題状態（problem state）において実行されることができる。

【００１５】

ＣＰＵは、レジスタを提供し、これはプログラムに対して利用可能であるが、主ストレージ内でアドレス指定可能な表現を有さない。レジスタは、例えば、現行プログラム・ステータス・ワード（ＰＳＷ）、汎用レジスタ、浮動小数点レジスタ及び浮動小数点制御レジスタ、ベクトル・レジスタ、制御レジスタ、アクセス・レジスタ、プリフィックス・レジスタ、時刻（ＴＯＤ）プログラム可能レジスタ、並びに時刻コンパレータ及びＣＵＰタイマ用レジスタを含むことができる。このレジスタの組をＣＰＵのアーキテクチャ化レジスタ・コンテキスト（architected register context）と呼ぶことができる。コンフィギュレーション内の各ＣＰＵは、ＴＯＤクロックへのアクセスを提供することができ、これはコンフィギュレーション内の全てのＣＰＵで共有することができる。命令動作コードは、ある動作にどの型式のレジスタを使用すべきかを決定することができる。

【００１６】

各ＣＰＵは、それが機能及びファシリティの完全な一揃い（fullcomplement）を提供するものであるか（例えば汎用ＣＰＵ）又は特定の型式の作業負荷を処理することを意図したものであるか（例えば、専用ＣＰＵ）を示す、型式属性を有することができる。一次ＣＰＵは、汎用ＣＰＵ又は最後の初期プログラム・ロード（ＩＰＬ）動作に続いて始動したＣＰＵ（ＩＰＬ　ＣＰＵ）と同じ型式を有するＣＰＵのいずれかである。二次ＣＰＵは、汎用ＣＰＵ以外の、ＩＰＬ　ＣＰＵとは異なるＣＰＵ型式を有する任意のＣＰＵである。

【００１７】

マルチスレッディング・ファシリティは、サポーティング・アーキテクチャを実装するコンピュータシステム上で利用可能であり得る。マルチスレッディング・ファシリティは、マルチスレッディングのためのサポートを提供して、コアを共有する、ＣＰＵと呼ばれることもある一群のスレッドをイネーブルにする。マルチスレッディング・ファシリティがイネーブルにされている場合、コア内のＣＰＵは、実行ユニット又はキャッシュなどの特定のハードウェアリソースを共有することができる。コア内の１つのＣＰＵがハードウェアリソースを（典型的にはメモリ・アクセスを待機しながら）待機しているとき、コア内の他のＣＰＵは、コア内の共有リソースをアイドルのままにしておくのではなく、むしろ利用することができる。マルチスレッディング・ファシリティがインストールされており、かつイネーブルにされている場合、スレッドは、コアのメンバであるＣＰＵと同義である。マルチスレッディング・ファシリティがインストールされていない場合、又はファシリティはインストールされているがイネーブルにされていない場合、コアは、単一のＣＰＵ又はスレッドを含む。

【００１８】

マルチスレッディング・ファシリティがインストールされている場合、これをマルチスレッディング設定（ｓｅｔ－ｍｕｌｔｉｔｈｒｅａｄｉｎｇ）信号プロセッサ（ｓｉｇｎａｌ　ｐｒｏｃｅｓｓｏｒ、ＳＩＧＰ）指令の実行によりイネーブルにすることができる。例示的な実施形態において、マルチスレッディング・ファシリティがイネーブルにされている場合、コンフィギュレーション内のＣＰＵの数は、倍数で増加し、その値は、プログラム指定最大スレッド識別（ｐｒｏｇｒａｍ－ｓｐｅｃｉｆｉｅｄ　ｍａｘｉｍｕｍ　ｔｈｒｅａｄ　ｉｄｅｎｔｉｆｉｃａｔｉｏｎ、ＰＳＭＴＩＤ）により決定される。コア内のＣＰＵの数は、ＰＳＭＴＩＤよりも１つだけ多くすることができる。この倍数に対応する数のＣＰＵがグループ化されて１つのコアになる。コンフィギュレーション内の同じＣＰＵ型式の各コアは、同数のＣＰＵを有することができる。コア内の各ＣＰＵは、同じ

ＣＰＵ型式のものであるが、モデル及びＣＰＵ型式に基づいて、コア内の幾つかのＣＰＵは、オペレーショナルでなくてもよい。

【００１９】

例示的な実施形態において、制御プログラム、例えばオペレーティングシステム（ＯＳ）は、そのＯＳが管理するコンフィギュレーションがマルチスレッディングを使用できるようにするために、マルチスレッディングを明示的にイネーブルにする。代替的に、ハイパーバイザがマルチスレッディングをイネーブルにすることができ、ハイパーバイザのゲスト及びそれらのアプリケーションは、透過的に利益を得ることできる。アプリケーションプログラムは一般に、マルチスレッディングがイネーブルになっているかどうか気づかない。マルチスレッディングがイネーブルにされるとき、コンフィギュレーション内の全てのＣＰＵのＣＰＵアドレスは、アドレスの左端のビット内にコア識別（又はコアＩＤ）を含み、かつアドレスの右端ビット内にスレッド識別（スレッドＩＤ、又はＴＩＤ）を含むように調整される。コアＩＤは、コア・アドレス値と呼ばれることもあり、ＴＩＤは、スレッド・アドレス値と呼ばれることもある。コア内のＣＰＵは、実行ユニット又は下位レベル・キャッシュなどの特定のハードウェア・ファシリティを共有することができ、それゆえコアの１つのＣＰＵ内の実行がコア内の他のＣＰＵの性能に影響を与えることがある。

【００２０】

コンフィギュレーションの１つ又は複数のコアを、単一スレッド・モードとマルチスレッディング・モードとの間で動的に切り換えることに関連した変化を管理するために、多数のサポート機能が含められる。マルチスレッディングをサポートしないプログラムとの互換性を維持するために、リセット又は非アクティブ化したときに単一スレッド・モードをデフォルト・モードとすることができる。例示的な実施形態は、マルチスレッディング・モードから単一スレッド・モードへの移行後にスレッド・コンテキストの解析及び／又は復元をサポートするために、マルチスレッディング・モードからのコンテキストを保存し、通信し、及び復元する機能を含む。

【００２１】

例示的な実施形態により実装することができるコンピューティング環境は、例えばニューヨーク州Ａｒｍｏｎｋのインターナショナル・ビジネス・マシーンズ・コーポレーションにより提供されるｚ／Ａｒｃｈｉｔｅｃｔｕｒｅに基づくものとすることができる。ｚ／Ａｒｃｈｉｔｅｃｔｕｒｅは、その全体が引用により本明細書に組み入れられる非特許文献１のＩＢＭ（登録商標）刊行物に説明されている。一例において、ｚ／Ａｒｃｈｉｔｅｃｔｕｒｅに基づくコンピューティング環境は、ニューヨーク州Ａｒｍｏｎｋのインターナショナル・ビジネス・マシーンズ・コーポレーションにより提供される、ｅＳｅｒｖｅｒ、ｚＳｅｒｉｅｓを含む。コンピューティング環境は、例えば、１つ又は複数のコア（例えば、プロセッサ・コア）を有する１つ又は複数のパーティション（例えば、論理パーティション）を有するプロセッサ複合体、及び本明細書でさらに説明するような１つ又は複数のレベルのハイパーバイザを含むことができる。

【００２２】

図１は、マルチスレッディング（ＭＴ）をサポートするコンピューティング環境の例としてコンピュータシステム１００を示す。図１の例において、コンピュータシステム１００は、複数のプロセッサ・コア１０２、入力／出力（Ｉ／Ｏ）サブシステム１０４、及びシステムメモリ１６０を含む。Ｉ／Ｏサブシステム１０４は、当該分野で知られたＩ／Ｏデバイスへのアクセスを提供することができる。プロセッサ・コア１０２は、本明細書においては単に「コア」とも呼ばれ、処理回路をサポート要素と共に含むことができる。図１の例において、５つのコア１０２は、コア１　１１０、コア２　１２０、コア３　１３０、コア４　１４０、及びコア５　１５０として示されているが、より多数の又は少数のコア１０２もまた企図される。ＭＴファシリティ１０３は、コア１０２の各々のハードウェア・コンポーネントとすることができる。この例において、コア１０２の各々は、上限４つまでのスレッドをサポートすることが可能である。例えば、コア１　１１０は、スレ

ッド１１１、１１２、１１３及び１１４をサポートすることができる。コア２　１２０は、スレッド１２１、１２２、１２３及び１２４をサポートすることができる。コア３　１３０は、スレッド１３１、１３２、１３３及び１３４をサポートすることができる。コア４　１４０は、スレッド１４１、１４２、１４３及び１４４をサポートすることができる。コア５　１５０は、スレッド１５１、１５２、１５３及び１５４をサポートすることができる。いずれの時点でも、各コア１０２の４つ全てのスレッドがオペレーショナルである必要はないことに留意されたい。例えば、コア３　１３０において、スレッド１３１及び１３２はオペレーショナルであり得るが、一方でスレッド１３３及び１３４はオペレーショナルとなることが許容されない（網掛けで示される）。

【００２３】

図１はまた、コンピュータシステム１００のシステムメモリ１６０示し、ここでシステムメモリ１６０の部分は、論理パーティション１（ＬＰＡＲ１）１７０、ＬＰＡＲ２　１８０、及びＬＰＡＲ３　１９０に分割されている。ＬＰＡＲ１７０、１８０、１９０は、仮想コンピューティングシステム（コンフィギュレーションとしても知られる）を表し、この中でＬｉｎｕｘ（商標）又はＩＢＭ（登録商標）ｚ／ＯＳ（商標）、ｚ／ＶＭ、又はｚＴＰＦオペレーティングシステムなどのオペレーティングシステムを実行することができる。図１はまた、ＬＰＡＲ１７０、１８０、１９０に対するコア１０２の割当ても示す。この図において、コア１　１１０及びコア２　１２０は、ＬＰＡＲ１　１７０による使用専用である。コア３　１３０は、ＬＰＡＲ２　１８０による使用専用であり、コア５　１５０は、ＬＰＡＲ３　１９０による使用専用である。コア４　１４０は、ＬＰＡＲ２　１８０とＬＰＡＲ３　１９０との間で共有することができるが、図１ではＬＰＡＲ２　１８０に割り当てられているように示されている。ＬＰＡＲ３　１９０は、コア１０２の２つの異なる型式がパーティションにより使用されている例を示し、ここでコア４　１４０は、多重スレッドがオペレーショナルであることを許容するが、コア５　１５０は、この例では多重スレッドがオペレーショナルであることを許容しない。図１の例において、ＬＰＡＲ１　１７０は、ＯＳ１７１並びにプログラム１７２、１７３、１７４及び１７５のための処理リソースを提供する。ＬＰＡＲ２　１８０は、ＯＳ１８１並びにプログラム１８２、１８３及び１８４のための処理リソースを提供する。ＬＰＡＲ３　１９０は、ＯＳ１９１並びにプログラム１９２及び１９３のための処理リソースを提供する。

【００２４】

ＬＰＡＲ内で実行するオペレーティングシステムの制御下で、プログラムは、コアのスレッド上で実行される。例示的な実施形態において、個々のスレッドは、一度に１つのプログラムのみを実行するが、しかし、再入可能になるように設計されたプログラムは、多重スレッド又はコア上で同時に実行されることができる。例えば、ＬＰＡＲ１　１７０のＯＳ１７１のプログラム１７２を、コア１　１１０内のスレッド１１１及び１１３上並びにコア２　１２０のスレッド１２１及び１２４内で実行することができる。ＯＳの制御に従って、異なるプログラムを、ディスパッチ規則及びサービス品質協定に従って、同じスレッド又は異なるスレッド上にディスパッチすることができる。

【００２５】

また、システムメモリ１６０内には、例えばミリコード（Ｍｉｌｌｉｃｏｄｅ）１６２及びＬＰＡＲハイパーバイザ１６３を含む、種々のレベルのファームウェアが存在する。ミリコード１６２は、下位レベルのシステム機能をサポートするためのファームウェアとして具体化することができる。ＬＰＡＲハイパーバイザ１６３は、例えば、ＩＢＭ　Ｐｒｏｃｅｓｓｏｒ－Ｒｅｓｏｕｒｃｅ／Ｓｙｓｔｅｍ　Ｍａｎａｇｅｒ（商標）（ＰＲ／ＳＭ（商標））などの、ライセンスされた内部コードとすることができる。ＬＰＡＲハイパーバイザ１６３は、ＬＰＡＲ１７０、１８０、１９０を確立することができ、コア１０２上のディスパッチを管理することができる。ＭＴファシリティ１０３がコンピュータシステム１００にインストールされている場合、ミリコード１６２及びＬＰＡＲハイパーバイザ１６３はまた、ＭＴファシリティ・サポートコード１６４及び１６５をそれぞれ含む。ＭＴをサポートするための論理は、ミリコード１６２と、ＬＰＡＲハイパーバイザ１６３

と、コア１０２との間に分散させることができるので、ＭＴファシリティ・サポートコード１６４及び１６５は、ＭＴファシリティ１０３の一部と考えることができる。図示していないが、ＯＳ１７１、１８１、１９１の各々もまた、それぞれのＬＰＡＲ１７０、１８０、１９０内のＭＴをイネーブルにして活用するためにＭＴファシリティ・サポートコードを含むことができる。

【００２６】

図２は、図１と同じコンピュータシステム１００を示すが、但し、図２のコンピューティング環境において、コア４　１４０は、ここではＬＰＡＲ２　１８０の代わりにＬＰＡＲ３　１９０に割り当てられている。スレッド１４３及び１４４がオペレーショナルではなかった図１とは異なり、図２では、ＬＰＡＲ３　１９０がコア４　１４０上にディスパッチされたとき、４つ全てのスレッド１４１－１４４が現在オペレーショナルであることにもまた留意されたい。コア１０２上にＬＰＡＲをディスパッチすること及びアンディスパッチすること（undispatching）は動的であり、他の時点では他のＬＰＡＲ（図示せず）が同じコア１０２上で動作することができる。

【００２７】

ここで図３を参照すると、実施形態により、処理コア、例えば図１及び図２のコア１０２のうちの１つを実装するための処理回路２００のブロック図が一般的に示される。処理回路２００は、ＭＴ環境内で１つ又は複数のスレッドを同時にサポートすることができる処理回路の一例である。図３に示す処理回路２００は、処理回路２００を他のプロセッサ及び周辺デバイスに結合することができるシステムコントローラ・インタフェース・ユニット２０２を含む。システムコントローラ・インタフェース・ユニット２０２はまた、データ値を読み出して格納するＤキャッシュ２０４、プログラム命令を読み出すＩキャッシュ２０８、及びキャッシュ・インタフェース・ユニット２０６を、外部メモリ、プロセッサ、及び他の周辺デバイスに接続することができる。

【００２８】

Ｉキャッシュ２０８は、命令フェッチ・ユニット（ＩＦＵ）２１０と共同して命令ストリームのローディングを提供することができ、命令フェッチ・ユニットは、命令を予めフェッチし、推論的ローディング及び分岐予測能力を含むことができる。フェッチされた命令は、デコードして命令処理データにするために、命令デコード・ユニット（ＩＤＵ）２１２に提供されることができる。

【００２９】

ＩＤＵ２１２は、命令を発行ユニット２１４に提供することができ、これは、一般演算を実行するための１つ又は複数の固定小数点ユニット（ＦＸＵ）２１６及び浮動小数点演算を実行するための１つ又は複数の浮動小数点ユニット（ＦＰＵ）２１８などの種々の実行ユニットに対する命令の発行を制御することができる。ＦＰＵ２１８は、２進浮動小数点ユニット（ＢＦＵ）２２０、１０進浮動小数点ユニット（ＤＦＵ）２２２、又はその他のいずれかの浮動小数点ユニットを含むことができる。発行ユニット２１４はまた、１つ又は複数のロード／格納ユニット（ＬＳＵ）２１８に１つ又は複数のＬＳＵパイプラインを介して結合することができる。多重ＬＳＵパイプラインは、ロード及び格納並びに分岐のためのアドレス生成を行うための実行ユニットとして扱われる。ＬＳＵ２２８及びＩＦＵ２１０の両方が、変換ルックアサイド・バッファ（ＴＬＢ）２３０を利用して、オペランド及び命令アドレスに対するバッファされた変換を提供することができる。

【００３０】

ＦＸＵ２１６及びＦＰＵ２１８は、汎用レジスタ（ＧＰＲ）２２４及び浮動小数点レジスタ（ＦＰＲ）２２６などの種々のリソースに結合される。ＧＰＲ２２４及びＦＰＲ２２６は、Ｄキャッシュ２０４からＬＳＵ２２８によりロードされて格納されるデータ値のためのデータ値ストレージを提供する。

【００３１】

処理回路２００はまた、システム時間ベースの生成及び診断アクションをサポートするためのカウンタ及び／又はタイマ２５０を含むことができる。例えば、カウンタ及び／又

はタイマ２５０を用いて、時刻、並びに種々の診断及び測定ファシリティをサポートすることができる。

【００３２】

ここで図４を参照すると、図１と類似のコンピューティング環境が示されているが、但し、図４では、第２レベル・ハイパーバイザ３００がコンピュータシステム１００のＬＰＡＲ２　１８０内で実行されている。第２レベル・ハイパーバイザ３００は、例えば、ＩＢＭのｚ／ＶＭオペレーティングシステムであり、ＬＰＡＲ（第１レベル）ハイパーバイザ１６３により提供されるＭＴサポートコード１６５と同様のＭＴサポートコード３０１を含む。第２レベル・ハイパーバイザ３００は、複数の仮想マシン３１０、３２０、及び３３０（コンフィギュレーションとも呼ばれる）のためのサポートを提供し、その中でゲスト・オペレーティングシステム３１１、３２１、及び３３１がそれぞれ動作する。ゲスト・オペレーティングシステム３１１、３２１、及び３３１は、例えば、Ｌｉｎｕｘ（商標）又はＩＢＭ（登録商標）ｚ／ＯＳ（商標）、ｚ／ＶＭ、若しくはｚ／ＴＰＦ　ＯＳを含むことができ、又はＩＢＭ会話型モニタシステム（ＣＭＳ）などのゲスト開発環境を含むことができる。各ゲストＯＳ３１１、３２１、及び３３１は、マルチスレッディングをイネーブルにすることができる場合もできない場合もあり、その場合、第２レベル・ハイパーバイザ３００は、第２レベル・ハイパーバイザ３００が動作しているＬＰＡＲ２　１８０にとって利用可能な物理的処理リソース（コア１３０、１４０及びスレッド１３１－１３４、１４１－１４４）を用いて、ゲストＯＳ３１１、３２１、３３１及び関連付けられたプログラム３１２、３１３、３２２、３２３、３３２、及び３３３のディスパッチを担当することができる。種々の仮想マシン３１０、３２０、３３０のプログラム３１２、３１３、３２２、３２３、３３２、３３３は、それぞれのゲストＯＳ３１１、３２１、及び３３１が利用可能なスレッド１３１－１３４、１４１－１４４上で実行することができる。ゲストＯＳ３１１、３２１、及び３３１は、第２レベル・ハイパーバイザ３００がマルチスレッディングを活用している場合、ＭＴから透過的に利益を得ることができるので、ＭＴサポートコードを含む必要はない。

【００３３】

ここで図５を参照すると、実施形態により実装することができるコンピューティング環境内のハイパーバイザ・コンテキスト保持の例が示される。図５の例において、多数のサポート構造体が、図１及び図２のＬＰＡＲハイパーバイザ１６３内に示されている。例えば、構造体４１０は、図１のＬＰＡＲ１　１７０をサポートすることができ、図１に示すような物理スレッド１１１、１１２、１１３、１１４、１２１、１２２、１２３、１２４上で現在実行されている論理スレッド４１１、４１２、４１３、４１４、４２１、４２２、４２３、４２４のアーキテクチャ化レジスタ・コンテキスト（すなわち、スレッド・コンテキスト）を格納する、状態記述（state description）及びサテライト・ブロックを含む。これらの論理スレッドがディスパッチされている間、物理スレッドは、スレッドの現行アーキテクチャ化レジスタ・コンテキストを保持する。アーキテクチャ化レジスタ・コンテキストは、これらがもはやディスパッチされないときには、状態記述及びサテライト・ブロック内で維持される。構造体４３０は、図１のＬＰＡＲ２　１８０をサポートすることができ、図１に示すように物理スレッド１３１、１３２、１４１、１４２上で現在実行されている論理スレッド４３１、４３２、４４１、４４２のアーキテクチャ化レジスタ・コンテキストを格納する状態記述及びサテライト・ブロックを含む。構造体４５０は、図１のＬＰＡＲ３　１９０をサポートすることができ、図１に示すように物理スレッド１５１上で現在実行されている論理スレッド４５１のアーキテクチャ化レジスタ・コンテキストを格納する状態記述及びサテライト・ブロックを含む。構造体４５０はまた、物理プロセッサ上に現在ディスパッチされていない（網掛けで示される）論理スレッド４６１、４６２、４６３及び４６４のアーキテクチャ化レジスタ・コンテキストを格納する状態記述及びサテライト・ブロックも含む。物理コア上にディスパッチされていないＬＰＡＲをサポートする他の構造体、例えば、論理スレッド４７１、４７２、４７３及び４７４の状態記述及びサテライト構造体を含むＬＰＡＲ　Ａ（図１には示されていない）のための

構造体４７０もまた、ＬＰＡＲハイパーバイザ１６３によって保持することができる。さらなる構造体の例は、論理スレッド４８１及び４８２の状態記述及びサテライト構造体を含む非ディスパッチＬＰＡＲ　Ｂ（図１には示されていない）をサポートする構造体４８０、並びに論理スレッド４８５のための非ディスパッチＬＰＡＲ　Ｃ（図１には示されていない）構造体４８４を含む。

【００３４】

図５には多数の構造体が示されているが、マルチスレッディングを管理するために、付加的な構造体を、ＬＰＡＲハイパーバイザ１６３及びコンピュータシステム１００内のどこか他の場所により、サポートすることができることが理解される。

【００３５】

ここで図６を参照すると、実施形態により、マルチスレッディングの動的イネーブルメントのためのプロセスフロー５００が示される。ブロック５０２において、第一スレッドが単一スレッド（ＳＴ）モードで実行される。ブロック５０４において、マルチスレッディング（ＭＴ）モード設定命令がＳＴモードでフェッチされる。まとめて５０５で示されるこの命令の実行の際に、ブロック５０６において、ＭＴモード設定命令により指定された場所から要求されるスレッドの数が得られる。場所は、ｓｅｔ－ＭＴモード命令を発行するときにパラメータ・レジスタにより指定することができる。ＭＴモード設定命令は、ｓｅｔ－ＭＴ指令と、要求されたスレッドの数に関連付けられるプログラム指定最大スレッドｉｄ（ＰＳＭＴＩＤ）とを含む、信号プロセッサ（ＳＩＧＰ）命令とすることができる。ＳＩＧＰ命令のｓｅｔ－ＭＴ指令に関連付けられたプロセスの例は、図９を参照して本明細書でさらに説明される。

【００３６】

プロセス５００の続きで、ブロック５０８において、要求されたスレッドの数が多重スレッドを示しているか否かの判定が行われる。例えば、多重スレッドは、１より大きい値により示されるものとすることができる。値ゼロが単一スレッドを示す実施形態において、１又は１より大きい値は、多重スレッドを示すことができる。要求されたスレッドの数が多重スレッドを示していないとの判定に基づいて、ブロック５１０において、コアはＳＴモードのままであり、ｓｅｔ－ＭＴモード命令の実行は完了し、制御はブロック５０２に戻る。要求されたスレッドの数が多重スレッドを示しているとの判定に基づいて、ブロック５１２において、ＭＴモードがイネーブルにされ、ｓｅｔ－ＭＴモード命令の実行が完了する。ブロック５１４において、第一スレッド及び１つ又は複数の第二スレッドを含む多重スレッドが実行される。ブロック５１６において、リセット又は非アクティブ化がない場合には、プロセス５００は、ブロック５１４にループバックし、そうでない場合には、ブロック５１８において、ＳＴモードに復帰するコンフィギュレーションのリセット又は非アクティブ化に基づいて、ＭＴモードがディスエーブルにされる。ＭＴモードをディスエーブルにすることの一部として、スレッドの数（ＰＳＭＴＩＤ）は、非クリアリング・リセット（non-clearing reset）の場合には保持され、クリアリング・リセット（clearingreset)の場合にはゼロにされる。プロセス５００は、ブロック５０２に戻る

【００３７】

ＣＰＵは、ｌｏａｄ－ｎｏｒｍａｌ、ｌｏａｄ－ｗｉｔｈ－ｄｕｍｐ、ｌｏａｄ－ｃｌｅａｒ、又はｌｏａｄ－ｃｌｅａｒ－ｌｉｓｔ－ｄｉｒｅｃｔｅｄキーがアクティブ化されたとき、ロード状態に入ることができる。チャネル・コマンド・ワード（ＣＣＷ）型初期プログラム・ローディング動作が成功裡に完了した場合、ＣＰＵは、ロード状態から動作状態に変化する。

【００３８】

ＣＰＵリセットは、破壊される情報量を最小限にして、装置チェック表示及び結果として生じるＣＰＵ状態におけるあらゆる不確定要素をクリアするために用いることができる。特に、これは解析又は動作再開のためにＣＰＵ状態を保存すべき場合に、チェック条件をクリアするために用いることができる。ＣＰＵリセットがｌｏａｄ－ｎｏｒｍａｌ又はｌｏａｄ－ｗｉｔｈ－ｄｕｍｐキーのアクティブ化により引き起こされた場合、（ａ）こ

れはアーキテクチャ・モード（architectural mode）をデフォルト・モードに設定することができ、（ｂ）マルチスレッディング・ファシリティがインストールされており、かつイネーブルにされている場合には、マルチスレッディングがディスエーブルにされる。ＣＰＵリセットがデフォルト・モードを設定する場合、ＰＳＷを復元することができるように現行ＰＳＷを保存することができる。

【００３９】

初期ＣＰＵリセットは、ＣＰＵリセットの機能を、現行ＰＳＷ、ＣＰＵタイマ、クロック・コンパレータ、及び他のレジスタ、例えば中断イベント・アドレス・レジスタ、キャプチャＰＳＷレジスタ、制御レジスタ、浮動小数点制御レジスタ、プリフィックス・レジスタ、及びＴＯＤプログラム可能レジスタなどの初期化と共に、提供する。初期ＣＰＵリセットは、ｌｏａｄ－ｎｏｒｍａｌ又はｌｏａｄ－ｗｉｔｈ－ｄｕｍｐキーのアクティブ化により引き起こされた場合、アーキテクチャ・モードをデフォルト・モードに設定することができる。初期ＣＰＵリセットがｌｏａｄ－ｎｏｒｍａｌ又はｌｏａｄ－ｗｉｔｈ－ｄｕｍｐキーのアクティブ化により引き起こされた場合にマルチスレッディングがイネーブルにされた場合、初期ＣＰＵリセット機能は、コアの最小番号ＣＰＵに対して行われることができ、コア内の他の全てのＣＰＵに対して、ＣＰＵリセットが行われる。クリアリング・リセットは、初期ＣＰＵリセット及びサブシステム・リセットの実行を引き起こし、加えて、ＴＯＤクロック以外の、コンフィギュレーション内の全てのＣＰＵ内の全てのストレージ位置及びレジスタがクリアされ又は初期化される。クリアリングは、外部ストレージ、例えばアドレス指定できないページの内容を保持するために制御プログラムにより用いられる直接アクセス・ストレージ・デバイスなどには影響を及ぼさない。

【００４０】

ＣＰＵ電源投入リセットは、初期ＣＰＵリセットの実行を引き起こし、有効チェック・ブロックコードにより、汎用レジスタ、アクセス・レジスタ、制御レジスタ、及び浮動小数点レジスタの内容をクリアしてゼロ／デフォルト値にする。状態のクリア又は初期化は必ずしもゼロ値になる必要はなく、クリアされた状態においてデフォルトで非ゼロ値にすることができることが理解される。ＣＰＵ電源投入リセットがコンフィギュレーションを確立する場合、アーキテクチャ・モードをデフォルト・モードに設定することができる。そうでない場合には、アーキテクチャ・モードを、すでにコンフィギュレーション内にあるＣＰＵのものに設定することができる。ＣＰＵリセット、初期ＣＰＵリセット、サブシステム・リセット、及びクリア・リセットは、手動で始動することができる。

【００４１】

例示的な実施形態において、各ＣＰＵは、そのＣＰＵアドレスと呼ばれる割り当てられた番号を有する。ＣＰＵアドレスは、コンフィギュレーション内の１つのＣＰＵを一意に識別する。ＣＰＵは、ＳＩＧＰ命令のＣＰＵアドレス・フィールド内でこのアドレスを指定することにより示される。誤動作警報、緊急信号、又は外部コールの信号を送信しているＣＰＵは、このアドレスを、中断を伴うＣＰＵアドレス・フィールド内に格納することにより識別することができる。ＣＰＵアドレスは、コンフィギュレーション定義プロセスにより割り当てられ、典型的には再構成による変化の結果として変更されない。プログラムは、ＣＰＵアドレス格納（ｓｔｏｒｅ　ＣＰＵ　ａｄｄｒｅｓｓ）命令を用いて、ＣＰＵのアドレスを決定することができる。ＣＰＵアドレス格納命令は、また、多重処理コンフィギュレーションにおいてＣＰＵを識別するＣＰＵアドレスを識別するために用いることもできる。

【００４２】

マルチスレッディングがイネーブルにされている場合、ＣＰＵアドレスは、コア内のＣＰＵの識別と連結されるコア識別（コアＩＤ）含むことができる。コア内のＣＰＵ識別は、スレッド識別（スレッドＩＤ、又はＴＩＤ）である。コンフィギュレーション内で、全てのコアは、同数のＣＰＵを提供するが、モデル及びＣＰＵ型式に応じて、コア内の幾つかのＣＰＵがオペレーショナルではない場合がある。

【００４３】

信号プロセッサのマルチスレッディング設定指令により用いられるパラメータ・レジスタのＰＳＭＴＩＤに基づいて、固定数のビットがスレッド識別を表す。このビット数は、ＴＩＤ幅と呼ばれる。

【００４４】

コアＩＤは、マルチスレッディングがイネーブルにされる前のＣＰＵアドレスの右側ビットから形成することができる。コアＩＤは、ＴＩＤ幅ビットだけ左にシフトされ、マルチスレッディングが利用可能になった後でＣＰＵアドレスの左側ビットをもたらす。スレッドＩＤは、同じＴＩＤ幅ビット数を有し、マルチスレッディングがイネーブルにされた後、ＣＰＵアドレスの右側ビットを占める。スレッドＩＤは、数の連続範囲内に割り当てることができる。表１は、ＰＳＭＴＩＤ、ＴＩＤ幅、並びにコア識別及びスレッド識別を含むＣＰＵアドレス・ビットの例示的な関係を示す。

【表１】

ＰＳＭＴＩＤ	ＴＩＤ幅	ＣＰＵアドレス・ビット	
		コアＩＤ	スレッドＩＤ
０	０	０－１５	－
１	１	０－１４	１５
２－３	２	０－１３	１４－１５
４－７	３	０－１２	１３－１５
８－１５	４	０－１１	１２－１５
１６－３１	５	０－１０	１１－１５

表１　例示的なアドレス・ビット・マッピング

【００４５】

アドレス拡張は、実施形態によるＣＰＵアドレス拡張プロセス６００Ａの例として図７に示される。ブロック６０２において、第一スレッドは、ＣＰＵアドレス・ビット数としてコア・アドレス値６０４を用いて、ＳＴモードでアクセスすることができる。矢印６０６は、ＳＴモードからＭＴモードへの切り換えを示す。ブロック６０８において、第一スレッド又は１つ若しくは複数の第二スレッドは、拡張アドレス値６１０を用いて、ＭＴモードでアクセスすることができる。拡張アドレス値６１０は、シフト・コアアドレス値６１２としてシフトされてスレッド・アドレス値６１４と連結された、コア・アドレス値６０４を含む。シフト・コアアドレス値６１２は、コア識別子（コアＩＤ）であり、スレッド・アドレス値６１４は、スレッド識別子（ＴＩＤ）である。シフト・コアアドレス値６１２は、要求された最大スレッド識別子、すなわちＰＳＭＴＩＤに基づく量だけシフトされることができる。スレッド・アドレス値６１４内のＴＩＤビット数は、上記表１に示すように、ＰＳＭＴＩＤに基づいて決定することができる。スレッド・アドレス値６１４をシフト・コアアドレス値６１２の下位桁ビットに連結して、拡張アドレス値６１０を形成することができる。全てがゼロのスレッド・アドレス値６１４は、第一スレッドを示し、ゼロより大きい値は、第二スレッドを識別して、アドレス指定する。

【００４６】

ＭＴモードとＳＴモードとの間で切り換えるとき、コア・アドレス値６０４（ＳＴモード）又は拡張アドレス値６１０（ＭＴモード）のいずれかが選択され、それぞれＳＴモード又はＭＴモードでＣＰＵアドレスとして用いられる。コア・アドレス値６０４は、ＳＴモードで用いられる標準フォーマットアドレスの例であり、コアは、ＭＴモードをディスエーブルにしたことに基づいて、ＭＴモードからＳＴモードへ復帰する。例示的な実施形

態において、第一スレッド（すなわち、第二スレッドではない）のみが、ＭＴモードをディスエーブルにしたことに基づいて、アクセス可能である。図８は、実施形態よるＣＰＵアドレス短縮プロセス６００Ｂの例を示す。図８の矢印６１６は、ブロック６０８のＭＴモードからブロック６０２のＳＴモードへの切換えを示す。ＭＴモードからＳＴモードへの復帰は、拡張アドレス値６１０を右へシフトし、スレッド・アドレス値６１４を排除して、シフト・コアアドレス値６１２からコア・アドレス値６０４（コアＩＤ）をＣＰＵアドレスとして含む標準フォーマットアドレスを形成することを含むことができる。

【００４７】

リセット機能がマルチスレッディングをディスエーブルにするとき、（ａ）スレッドＩＤゼロを有するＣＰＵのＣＰＵアドレスは、イネーブルメントに用いたのと同じＴＩＤ幅ビット数だけ右にシフトされ、（ｂ）アドレスの左側のＴＩＤ幅ビット数にゼロが挿入され、（ｃ）ＣＰＵアドレスは、元の非マルチスレッディング形式（すなわち、標準フォーマットアドレス）に復帰する。マルチスレッディングがイネーブルにされたときに非ゼロスレッドＩＤを有するコア内の全てのＣＰＵは、マルチスレッディングがディスエーブルにされたときにはもはやオペレーショナルではなくなる。

【００４８】

マルチスレッディングがイネーブルにされていないとき、ＣＰＵアドレスは、コンフィギュレーション定義プロセスにより割り当てられた値から変化せずにそのままである。この場合、スレッド識別は存在しない。

【００４９】

多数の信号プロセッサ指令は、ＣＰＵに対して、例えば、始動、停止、再始動、「停止及びステータス格納」（ｓｔｏｐ ａｎｄ ｓｔｏｒｅ ｓｔａｔｕｓ）、初期ＣＰＵリセット（ｉｎｉｔｉａｌ ＣＰＵ ｒｅｓｅｔ）、ＣＰＵリセット（ＣＰＵ ｒｅｓｅｔ）、「アドレスにステータス格納」（ｓｔｏｒｅ ｓｔａｔｕｓ ａｔ ａｄｄｒｅｓｓ）、アーキテクチャ設定（ｓｅｔ ａｒｃｈｉｔｅｃｔｕｒｅ）、実行ステータス検知（ｓｅｎｓｅ ｒｕｎｎｉｎｇ ｓｔａｔｕｓ）、マルチスレッディング設定（ｓｅｔ ｍｕｌｔｉｔｈｒｅａｄｉｎｇ）、「アドレスに追加ステータス格納」（ｓｏｒｅ ａｄｄｉｔｉｏｎａｌ ｓｔａｔｕｓ ａｔ ａｄｄｒｅｓｓ）などを含む指令を与えることができる。初期ＣＰＵリセット又はＣＰＵリセットは、信号プロセッサ命令により起動することができ、アーキテクチャ・モード又は他のＣＰＵに影響を及ぼさず、マルチスレッディングをディスエーブルにせず、Ｉ／Ｏのリセットを引き起こさない。

【００５０】

アーキテクチャ設定指令は、コンフィギュレーション内の全てのＣＰＵが設定されることになるアーキテクチャ・モードを指定する。アーキテクチャの差異は、異なるアドレス指定モード、レジスタ定義、及びＣＰＵによりサポートされる命令を含むことができる。アーキテクチャ・モードが変更されると、レジスタの選択ビットフィールドをデフォルト状態に設定する（例えばゼロにする）ことができ、コンフィギュレーション内の全てのＣＰＵのアクセス・レジスタ変換ルックアサイド・バッファ（ＡＬＢ）及び変換ルックアサイド・バッファ（ＴＬＢ）がクリアされ、コンフィギュレーション内の全てのＣＰＵに対してシリアル化及びチェックポイント同期機能を行うことができる。

【００５１】

実行ステータス検知指令は、アドレス指定されたＣＰＵが実行されているか否かを示すことができる。ＳＴモードでは、実行／非実行ステータスとしてインジケータを返すことができる。ＭＴモードでは、アドレス指定されたＣＰＵがそのメンバであるコアのいずれかのＣＰＵが実行しているか否か、又はアドレス指定されたＣＰＵがそのメンバであるコアの全てのＣＰＵが実行していないか否かを、インジケータを用いて識別することができる。

【００５２】

ｓｅｔ－ＭＴ指令は、マルチスレッディング・ファシリティをイネーブルにする。パラメータ・レジスタのビット位置は、コンフィギュレーション内で提供されるＰＳＭＴＩＤ

を収容することができる。ＰＳＭＴＩＤは、各コア内でアクセス可能にされるＣＰＵの数よりも１だけ少なくなるように定めることができる。例えば、指定されたビット位置における値３は、最大で４スレッドが提供されることを示す。コンフィギュレーション内の全てのＣＰＵはアドレス指定されていると考えることができるので、ＳＩＧＰ命令のＣＰＵアドレス・レジスタの内容は、無視することができる。ｓｅｔ－ＭＴ指令を受け取った場合、これはＳＩＧＰ命令の実行中に全てのＣＰＵにより完了される。図９を参照すると、ＳＩＧＰ　ｓｅｔ－ＭＴ指令７０２のためのプロセス７００が示される。図９のプロセス７００を参照して本明細書でさらに説明されるように、エラー表示を与えることができ、ＳＩＧＰ　ｓｅｔ－ＭＴ指令７０２が、無効指令、不正状態、及び無効パラメータのうちの１つ又は複数を伴って発行されたことの判定に基づいて、ＭＴモードのイネーブルメントが防止される。

【００５３】

ブロック７０４において、マルチスレッディング・ファシリティがインストールされていない場合、又は７０８でＣＰＵが有効アーキテクチャ・モードでイネーブルにされていない場合、ｓｅｔ－ＭＴ指令は受け入れられず、それぞれブロック７０６又は７１０において無効指令表示を返すことができる。ブロック７１２において、コンフィギュレーション内の他のＣＰＵが停止されていない又はチェック・ストップ状態にない場合、又はブロック７１６において、コンフィギュレーションがマルチスレッディングに関して既にイネーブルにされている場合、ｓｅｔ－ＭＴ指令は受け入れられず、それぞれブロック７１４又は７１８において不正状態表示を返すことができる。

【００５４】

ブロック７２０において、ＰＳＭＴＩＤが無効である場合、ｓｅｔ－ＭＴ指令は受け入れられず、ブロック７２２において無効パラメータ表示を返すことができる。ブロック７２４において、ＰＳＭＴＩＤがゼロのとき、コンフィギュレーションはマルチスレッディングに関してイネーブルにされず、ＳＴモードのままであり、ブロック７２８において条件コードとしていずれかのステータスを提供する。例示的な実施形態において、ＰＳＭＴＩＤが有効かつ非ゼロであるとき、ブロック７２６において、コンフィギュレーションはマルチスレッディングに関してイネーブルにされ、その結果、ＣＰＵアドレス拡張が生じ、コンフィギュレーション内の全てのＣＰＵのＡＬＢ及びＴＬＢは、その内容がクリアされ、コンフィギュレーション内の全てのＣＰＵ上でシリアル化及びチェックポイント同期機能が実行される。ブロック７２８において、ステータスが条件コードとして提供される。成功裡に完了すると、ｓｅｔ－ＭＴ指令を実行しているＣＰＵ以外の全てのＣＰＵは、停止された状態又はチェック・ストップ状態のままである。しかしながら、マルチスレッディングがイネーブルにされる前にＣＰＵがチェック・ストップ状態にあった場合、同じコア内で非ゼロスレッドＩＤを有するＣＰＵが停止状態又はチェック・ストップ状態に置かれることが予測できない場合がある。

【００５５】

スレッド・コンテキストは、アーキテクチャ化レジスタ・コンテキストとも呼ばれる。マルチスレッディングがイネーブルにされる前の各ＣＰＵのアーキテクチャ化レジスタ・コンテキスト（すなわち、ＰＳＷ、ＣＰＵタイマ、クロック・コンパレータ、汎用レジスタ、浮動小数点レジスタ及び浮動小数点制御レジスタ、ベクトル・レジスタ、制御レジスタ、アクセス・レジスタ、プリフィックス・レジスタ、並びにＴＯＤプログラム可能レジスタ、等）は、マルチスレッディングがイネーブルにされた後、各コアそれぞれのＴＩＤゼロを有するＣＰＵのアーキテクチャ化レジスタ・コンテキストになる。同様に、ＭＴイネーブルにされたコンフィギュレーションの各コアのＴＩＤゼロを有するＣＰＵのアーキテクチャ化レジスタ・コンテキストは、ｌｏａｄ－ｎｏｒｍａｌ又はｌｏａｄ－ｗｉｔｈ－ｄｕｍｐキーのアクティブ化の結果としてマルチスレッディングがディスエーブルにされたとき、各ＣＰＵそれぞれのアーキテクチャ化レジスタ・コンテキストになる。

【００５６】

非ゼロスレッド表示を有する全てのＣＰＵのアーキテクチャ化レジスタ・コンテキスト

を、ｌｏａｄ－ｎｏｒｍａｌ又はｌｏａｄ－ｗｉｔｈ－ｄｕｍｐキー動作のアクティブ化の結果としてマルチスレッディング・ファシリティがディスエーブルにされるときに保持することができる。マルチスレッディング・ファシリティがその後、介在するクリア・リセットを伴わずに再イネーブルにされた場合、非ゼロ・スレッドを有する全てのＣＰＵのアーキテクチャ化レジスタ・コンテキストは、復元される。

【００５７】

ｌｏａｄ－ｎｏｒｍａｌ又はｌｏａｄ－ｗｉｔｈ－ｄｕｍｐキーのアクティブ化によりディスエーブルにされた後でマルチスレッディングが再イネーブルされたとき、パラメータ・レジスタ内のビットでのＰＳＭＴＩＤの値が以前のイネーブルメントにおいて用いられた値と違う場合、非ゼロ・スレッドＩＤを有する全てのＣＰＵのアーキテクチャ化レジスタ・コンテキストは、予測できない場合がある。

【００５８】

システム情報格納（ｓｔｏｒｅ　ｓｙｓｔｅｍ　ｉｎｆｏｒｍａｔｉｏｎ）命令を用いて、コンフィギュレーションのコンポーネントに関する情報をシステム情報ブロック（ＳＹＳＩＢ）に格納することができる。ＳＹＳＩＢは、ＭＴインストール・フィールド、ＭＴ汎用フィールド、総ＣＰＵ／コアカウント、構成ＣＰＵ／コアカウント、スタンバイＣＰＵ／コアカウント、リザーブＣＵＰ／コアカウント、及びその他のフィールドを含むことができる。ＭＴインストール・フィールドは、マルチスレッディング・ファシリティがインストールされているか否かを示すことができ、また第１のコア型式、例えば専門コア型式に対する最大サポートＴＩＤを示すことができる。ＭＴ汎用フィールドは、第２のコア型式、例えば汎用コア型式に対する最大サポートＴＩＤを示すことができる。ＭＴ汎用フィールド内の最大サポートＴＩＤは、ＭＴインストール・フィールド内の最大サポートＴＩＤを下回るか又は等しいものに限定することができる。総ＣＰＵ／コアカウントは、構成状態、スタンバイ状態、又はリザーブ状態のいずれかにある、コンフィギュレーション内の汎用ＣＰＵ又は汎用ＣＰＵを含むコアの総数を示すことができる。構成ＣＰＵ／コアカウントは、構成状態にある、すなわちコンフィギュレーション内にあり、かつプログラムを実行する準備ができている汎用ＣＰＵ又は汎用ＣＰＵを含むコアの数を示すことができる。スタンバイＣＰＵ／コアカウントは、スタンバイ状態にある、すなわち構成状態に置かれるまでプログラムの実行に用いることができない汎用ＣＰＵ又は汎用ＣＰＵを含むコアの数を示す。リザーブＣＰＵ／コアカウントは、リザーブ状態にある、すなわちプログラムを実行するために使用することはできず、かつ構成状態におくことができない汎用ＣＰＵ又は汎用ＣＰＵを含むコアの数を示す。

【００５９】

図１０は、実施形態によるマルチスレッディング能力情報の格納の例を示す。スレッド、例えばコア８００Ａのスレッド１内で実行されるプログラムは、ＬＰＡＲなどのコンフィギュレーション８５０のメモリ８０１から、ＳＴＯＲＥ　ＳＹＳＴＥＭ　ＩＮＦＯＲＭＡＴＩＯＮ（ＳＴＳＩ）命令８３０をフェッチすることができる。ＳＴＳＩ命令の実行は、システム情報ブロック（ＳＹＳＩＢ）８０２の格納をもたらすことができる。図１０の例において、ＳＹＳＩＢ８０２は、コンフィギュレーション８０５がマルチスレッディングをサポートするか否かを示すＭＴインストール識別子８５０を含む。ＳＹＳＩＢ８０２はまた、コア８００Ａ／８００Ｂの最大サポートスレッドの最大スレッド識別子を含み、これは、専門コアの場合にはコア当たりの最大ＴＩＤ８０６として、及び汎用コアの場合には最大ＴＩＤ８０８として提供されることができる。ＳＹＳＩＢ８０２はまた、現行プログラム指定最大スレッド識別子（ＰＳＭＴＩＤ）８０９を含むことができる。現行ＰＳＭＴＩＤ８０９は、プログラムによりコンフィギュレーション８５０内でイネーブルにされたときのマルチスレッディング・モードを反映する。現行ＰＳＭＴＩＤ８０９は、ＳＴＳＩ命令８３０がベーシック・マシン・レベルで実行される場合は定義されなくてもよい。

【００６０】

スレッド内、例えばコア８００Ｂのスレッド２内で実行されるプログラムもまた、コン

フィギュレーション８５０のメモリ８０１から、ＳＥＲＶＩＣＥ　ＣＡＬＬ（ＳＥＲＶＣ）命令８３４をフェッチし、ここで命令はシステム制御プログラム情報読出し（ｒｅａｄ－ｓｙｓｔｅｍ－ｃｏｎｔｒｏｌ－ｐｒｏｇｒａｍ－ｉｎｆｏｒｍａｔｉｏｎ、ｒｅａｄ－ＳＣＰ－ｉｎｆｏ又はＲＳＣＰＩ）コマンドを指定する。ＲＳＣＰＩコマンドの実行は、サービス・コール制御ブロック（ＳＣＣＢ）８１０をメモリ８０１内に格納８３６させることができる。例示的な実施形態において、ＲＳＣＰＩコマンドの実行により格納されるＳＣＣＢ８１０は、同様の情報及びＳＹＳＩＢ８０２内では利用可能ではない場合もある付加的な情報を提供する。図１０の例において、ＳＣＣＢ８１０は、コア８００Ｂがマルチスレッディングをサポートするか否かを識別するＭＴインストール識別子８１２を含む。ＳＣＣＢ８１０はまた、専門コアの場合にはコア当たりの最大ＴＩＤ８１４として提供され、汎用コア８１６の場合には最大ＴＩＤとして提供されることができる、コア８００Ｂの最大サポートスレッドの最大スレッド識別子も含む。ＳＣＣＢ８１０の値８１２－８１６は、ＳＹＳＩＢ８０２においてアクセス可能であり得る値８０４－８０８と等しい。さらに、ＳＣＣＢ８１０は、コア８００Ｂの最大サポートスレッドの最終設定（last-set）プログラム指定最大スレッド識別子を含むことができ、これは最終設定プログラム指定最大スレッド識別子（ＰＳＭＴＩＤ）８１８とも呼ばれる。ＳＣＣＢ８１０は、ｓｅｔ－ＭＴ指令上で受け入れ可能なＰＳＭＴＩＤ値のマスクをＰＳＭＴＩＤサポート・マスク８２０として含むことができる。ＰＳＭＴＩＤサポート・マスク８２０は、コア当たりの最大ＴＩＤ８１４により定義される数よりも少ない数が所望されるときに、サポートＣＰＵ／スレッドを識別するために用いることができる。

【００６１】

コア８００Ａ及び８００Ｂは、この例で示されていない他の態様を含むことが理解される。さらに、ＳＹＳＩＢ８０２及びＳＣＣＢ８１０は、図１０の例に示された値を超える付加的な値を含むことができる。

【００６２】

図１１は、実施形態によるマルチスレッディング能力を判定するためのプロセスフロー９００を示す。ブロック９０２において、コアは、マルチスレッディング能力情報取得（ｒｅｔｒｉｅｖｅ　ｍｕｌｔｉｔｈｒｅａｄｉｎｇ　ｃａｐａｂｉｌｉｔｙ　ｉｎｆｏｒｍａｔｉｏｎ、ＲＭＴＣＩ）命令を実行し、これはＳＥＲＶＣ命令又はＳＴＳＩ命令のいずれか１つとすることができる。ブロック９０４において、コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報が得られる。ブロック９０６において、得られたスレッド識別情報が格納される。ブロック９０８において、得られたスレッド識別情報に基づいて、コンフィギュレーションが、以前にマルチスレッディングがイネーブルにされていたか否かを判定することができる。

【００６３】

先に説明したように、ＳＥＲＶＣ命令は、スレッド識別情報をメモリ内の応答ブロック（例えば、図１０のＳＣＣＢ８１０）に格納するように構成され、ＳＴＳＩ命令は、スレッド識別情報をメモリ内のＳＹＳＩＢ内（例えば、図１０のＳＹＳＩＢ８０２）に格納するように構成される。得られたスレッド情報は、コアがマルチスレッディングをサポートするか否かを識別するＭＴインストール識別子（例えば、図１０のＭＴインストール識別子８０４又は８１２）を含むことができる。得られたスレッド情報は、コアの最大サポートスレッドの最大スレッド識別子（例えば、図１０の最大ＴＩＤ値８０６、８０８、８１４、又は８１６）を含むことができる。得られたスレッド情報は、現行プログラム指定最大スレッド識別子（例えば、図１０の現行ＰＳＭＴＩＤ８０９）及び最終設定プログラム指定最大スレッド識別子（例えば、図１０のＰＳＭＴＩＤ８１８）を含むことができる。応答ブロックは、個々にサポートされる特定のスレッド識別子を示すビットのマスク（例えば、図１０のＰＳＭＴＩＤサポート・マスク８２０）を含むことができる。コンフィギュレーションが以前にＭＴがイネーブルにされていたことの判定は、最終設定プログラム指定最大スレッド識別子内の非ゼロ値に基づくことができる（例えば最終設定ＰＳＭＩＴＤ＞０）。例示的な実施形態において、コンフィギュレーションは、複数のコア型式をサ

ポートする。

【００６４】

例示的な実施形態において、レジスタ、及びレジスタ内に含めるか又は別個に管理することができるプログラム・カウンタ値などの値は、スレッド・コンテキストとしてキャプチャされる。アドレス拡張がＭＴモードで生じると、付加的なスレッド・コンテキストがアクセス可能になる。図７及び図８を参照して先に説明したように、ＣＰＵアドレスが、コンフィギュレーション内の各コアに対して形成される。ＣＰＵアドレスを、ＣＰＵアドレス格納命令により検査することができ、これは他の構造体の中に現れ、種々のＳＩＧＰ指令において用いられる。ＭＴがイネーブルにされていない場合、このアドレス指定スキームは変更されないままである。ＭＴがイネーブルにされた場合、ＣＰＵアドレスは、拡張プロセスを受ける。先に説明したように、ＣＰＵアドレスの非ＭＴイネーブル部分を、ＴＩＤを収容するのに十分なビットだけ左側にシフトさせることができる。例えば、オペレーティングシステムがＰＳＭＴＩＤ値１でＳＩＧＰ　ｓｅｔ－ＭＴ指令を発行した場合、ＣＰＵアドレスは左に１ビットシフトされ、ＰＳＭＴＩＤが２又は３であった場合、ＣＰＵアドレスは、左に２ビットシフトされ、ＰＳＭＴＩＤが４－７である場合、ＣＰＵアドレスは左に３ビットシフトされ、以下同様である。

【００６５】

マルチスレッディングが、その後、（ｌｏａｄ－ｎｏｒｍａｌ動作に引き起こされるクリア・リセット又はＣＰＵリセットの結果として）ディスエーブルにされると、ＣＰＵアドレス短縮が生じる。ＭＴイネーブルＣＰＵアドレスを、ＭＴをイネーブルにしたＳＩＧＰ　ｓｅｔ－ＭＴ指令で用いられたＰＳＭＴＩＤビットと同じ数だけ右にシフトさせることができ、アドレスのスレッドＩＤ部分が消える。ＭＴモード中にアクセス可能なスレッド・コンテキストは、図１２に示す例のように１つ又は複数の場所に存在することができる。図１２の例において、コンフィギュレーション１０００は、コア１００２を含み、他のコア（図示せず）を含むことができる。メモリ１００６は、コンフィギュレーション１０００の一部としてのコンフィギュレーション・メモリ１００５を含むことができ、コンフィギュレーション１０００から分離したホスト／ファームウェア・メモリ１００７を含むことができる。ホスト／ファームウェア・メモリ１００７は、ホストにより維持される状態識別ブロック１００８を含むことができ、これはスレッドに関する（例えば図１２ではスレッドｎ）スレッド・コンテキスト１０１０を格納することができる。サテライト・ブロック１０１２は、メモリ１００６内でホスト／ファームウェア・メモリ１００７の一部として状態記述ブロック１００８に固定されることができ、ここでサテライト・ブロック１０１２は、スレッド・コンテキスト１０１０の代替として又はスレッド・コンテキスト１０１０との組合せで、スレッド・コンテキスト１０１４を含むことができる。各スレッドは、対応する状態記述ブロック１００８と、随意にサテライト・ブロック１０１２とを有することができ、ここにスレッド・コンテキスト１０１０又はスレッド・コンテキスト１０１４を格納することができる。さらなる代替として、ハードウェア・コンテキスト・レジスタ１０１６を用いて、スレッド・コンテキスト１０１８を、例えばコア１００２内に格納することができる。スレッド・コンテキスト１０１０、１０１４及び１０１８の例は、ストレージ・オプションとして組合せで又は別個に用いることができる。代替的なストレージ・オプションを実施形態において使用することができる。スレッド・コンテキストがどこに維持されているかにかかわらず、アドレス短縮されると、スレッド・コンテキストはもはや直接アクセス可能ではなくなる場合があるが、ダンププログラムにより、アクセスのために保存することができる。

【００６６】

ＭＴがディスエーブルにされているとき、ＣＰＵアドレス短縮プロセスは、コアのスレッド１－ｎをもはやアクセス可能ではなくなるようにする。同様にアーキテクチャ化レジスタを含むスレッド・コンテキストは、もはやプログラムから見えなくなる。非クリアリング・ロード動作の結果生じるＣＰＵリセットの結果としてＭＴがディスエーブルにされた場合、スレッド１－ｎのレジスタ・コンテキストは保持される。これらのデータは、そ

の後、コンフィギュレーションがＭＴモードに戻された場合に検査される。各ゲストスレッドに対するレジスタ・コンテキストは、図１２に示すように、スレッドの状態記述ブロック１００８内（又はベクトル・レジスタの場合、状態記述内に固定されたサテライト・ブロック１０１２内）のホストにより維持することができる。

【００６７】

ＭＴのディスエーブルメント中のスレッド１－ｎのコンテキストの保持は、ＯＳ故障後にスレッドの状態がダンプされるための診断的機能である。ＯＳ故障の後、オペレータは、スタンドアロン・ダンプ（ＳＡＤＭＰ）プログラムを実行することを選択して、故障時点のシステムのメモリ及びスレッド・コンテキストをキャプチャすることができる。しかしながら、ＳＡＤＭＰプログラムをロードすることで、コンフィギュレーションが、ＳＴモードがイネーブルにされるデフォルト・アーキテクチャモードに戻ってしまう場合があり、したがってＭＴがディスエーブルにされる。しかし、非クリアリング・ロード動作によりＳＡＤＭＰがロードされるので、各コアのスレッド１－ｎのレジスタ・コンテキストは保持される。ＳＡＤＭＰは、ＳＥＲＶＣリードＳＣＰ情報コマンドの応答ブロックの結果を検査することにより、ダンプされているコンフィギュレーション内でＭＴがイネーブルにされていたか否かを判定することができる。この数は、その後、以前と同じレベルでＭＴを再イネーブルにするためのＳＩＧＰ　ｓｅｔ－ＭＴ指令への入力として用いることができる。

【００６８】

図１３は、実施形態によるマルチスレッディング・レジスタ保存の例を示す。システム、例えば図１３のコンピュータシステム１１００は、多重コンフィギュレーション１１０２及び１１０４を含むことができる。図１３の例において、コンフィギュレーション１１０２は、コア１１０６及びコア１１０８を含み、コンフィギュレーション１１０４は、コア１１１０及びコア１１１２を含む。コンフィギュレーション１１０２及び１１０４の各々は、異なる時点で、ＳＴモードとＭＴモードとの間で独立に切り換えることができる。コンピュータシステム１１００のコンフィギュレーション１１０２及び１１０４の各々は、コンフィギュレーション１１０２及び１１０４の各々において、イネーブルにされた異なる数のスレッドを同時にサポートするために、異なる数の最大スレッドＩＤで構成することができる。図１３の例において、コンフィギュレーション１１０２がＭＴモード１１１４にあるとき、コア１１０６及び１１０８は各々、最大で２つのスレッドをサポートし、他方、コンフィギュレーション１１０４がＭＴモード１１１６にあるとき、コア１１１０及び１１１２は各々、最大で４つのスレッドをサポートする。

【００６９】

コンフィギュレーション１１０２においてＭＴモード１１１４がイネーブルにされているとき、ＴＩＤ０及びＴＩＤ１の両方が、スレッド・コンテキスト１１１５の別個のインスタンスのように、別個のスレッド・コンテキストとしてアクセス可能である。時刻１１１８において、ＭＴモード１１１４は、コンフィギュレーション１１０２に対するｌｏａｄ－ｎｏｒｍａｌ動作又は非クリアリング・リセットによりディスエーブルにされることができ、これがコア１１０６及び１１０８の両方をＳＴモード１１２０に切り換える。以前に説明したアドレス短縮により、ＴＩＤ０レジスタはＳＴモード１１２０でアクセス可能であるが、ＭＴモード１１１４でアクセス可能であったＴＩＤ１レジスタは、保持されるが、もはやアクセス可能ではない。例えば、ＴＩＤ１レジスタは、図１２のスレッド・コンテキスト１０１０、１０１４又は１０１８として具体化することができ、ここでアドレス拡張により使用可能であったアドレスは、ＳＴモード１１２０へ切り換わったときのアドレス短縮後は、もはやアクセス可能ではなくなる。

【００７０】

コンフィギュレーション１１０４がＭＴモード１１１６をイネーブルにしているとき、ＴＩＤ０、ＴＩＤ１、ＴＩＤ２及びＴＩＤ３レジスタは、図１２のスレッド・コンテキスト１０１０、１０１４、又は１０１８の別個のインスタンスのように、別個のスレッド・コンテキストとしてアクセス可能である。この例において、ＴＩＤ０は、第一スレッドを

表し、ＴＩＤ１－ＴＩＤ３は、コア１１１０及び１１１２の各々に対して別個に維持される第二スレッドを表す。時刻１１２２において、ＭＴモード１１１６は、コンフィギュレーション１１０４に対するクリアリング・リセットによりディスエーブルにされることができ、これがコア１１１０及び１１１２の両方をＳＴモード１１２４に切り換える。時刻１１２２におけるクリアリング・リセットは、ＴＩＤ０、ＴＩＤ１、ＴＩＤ２及びＴＩＤ３のレジスタの全てをクリアすることができる。以前に説明したアドレス短縮により、ＴＩＤ０レジスタはＳＴモード１１２４でアクセス可能であるが、ＭＴモード１１１６でアクセス可能であったＴＩＤ１、ＴＩＤ２及びＴＩＤ３レジスタは、クリアされた状態で保持されるがもはやアクセス可能ではない。図１３に示すように、動作は、各コンフィギュレーション１１０２及び１１０４上で異なる時刻１１１８及び１１２２に独立に行うことができ、効果は各コンフィギュレーション１１０２及び１１０４にローカライズされる。したがって、コンフィギュレーション１１０４がＭＴモード１１１６である一方で、コンフィギュレーション１１０２はＳＴモード１１２０であることが可能であり、ＳＴ／ＭＴモードは、コンピュータシステム１１００の全てのコンフィギュレーションに対して揃っている必要はない。

【００７１】

図１４は、実施形態によるマルチスレッディング・レジスタ保存のためのプロセスフロー１２００を示す。ブロック１２０２において、ＭＴモードのコアによる、そのコア内でＭＴがディスエーブルにされるべきであるとの判定に基づいて、ＭＴモードからＳＴモードへの切換えが行われる。ＭＴモードの第一スレッドは、ＳＴモードの唯一のスレッドとして維持されることができる。第二スレッドのプログラム・アクセス可能なレジスタ値及びプログラム・カウンタ値を含む１つ又は複数のスレッド・コンテキストが、アプリケーションプログラムに対してアクセス不能にされる。ブロック１２０４において、切り換えに基づいて、動作型式（例えば、クリアリング　対　非クリアリング）が決定され、プログラム・アクセス可能レジスタ値をクリアするか又はプログラム・アクセス可能レジスタ値を保持するか、どちらかが行われる。ブロック１２０６において、非クリアリング動作に基づいて、プログラム・アクセス可能レジスタ値が保持されるべきであると判定される。ブロック１２０８において、クリアリング動作に基づいて、プログラム・アクセス可能レジスタがクリアされるべきであると判定される。

【００７２】

先に説明したように、スレッド・コンテキストのプログラム・アクセス可能レジスタ値及びプログラム・カウンタ値は、プログラム汎用レジスタ、浮動小数点レジスタ、制御レジスタ、アクセス・レジスタ、プリフィックス・レジスタ、及びＴＯＤプログラム可能レジスタを含むことができる。制御レジスタは、浮動小数点制御レジスタ、ランタイム計装制御、ＣＰＵ測定制御などを含むことができる。スレッド・コンテキスト内に含めることができるレジスタの他の例は、プログラム・ステータス・ワード（例えば、プログラム・カウンタ／命令アドレス、条件コード、並びに命令順番付けを制御するため及びＣＰＵ状態を決定するためのその他の情報）、ベクトル・レジスタ、ＣＰＵタイマ、クロック・コンパレータ、中断事象アドレス・レジスタ、及び当該分野で知られた他のレジスタを含む。以前に説明したように、ＰＳＭＴＩＤは、最後に成功裡に実行された、ＭＴをイネーブルにさせる信号プロセッサ命令に基づいて設定される。ＭＴモードへの切換えに基づいて、プログラム・アクセス可能レジスタ値は、再イネーブルにされた対応する第二スレッドに基づいて、アプリケーションプログラムに対してアクセス可能になる。例えば、図１３におけるＳＴモード１１２０からＭＴモード１１１４へ戻る切り換えは、ＴＩＤ１レジスタへのアクセスを可能にし、ＴＩＤ１は、再イネーブルにされる。スレッド・コンテキストは、図１２のスレッド・コンテキスト１０１０、１０１４又は１０１８のように、状態記述ブロック、メモリ内で状態技術ブロックに固定されたサテライト・ブロック、又はコンテキスト・レジスタのいずれかにおいて維持されることができる。

【００７３】

第一スレッド・コンテキストは、第一スレッドのプログラム・アクセス可能レジスタ値

及びプログラム・カウンタ値、例えば、図１３のコンフィギュレーション１１０４に対するＴＩＤ０及びＴＩＤレジスタを含むことができ、ここで第一スレッド・コンテキストは、ＳＴモード１１２４及びＭＴモード１１１６の両方においてアプリケーションプログラムに対してアクセス可能である。第二スレッド・コンテキストは、第二スレッドのプログラム・アクセス可能レジスタ値及びプログラム・カウンタ値、例えば、図１３のコンフィギュレーション１１０４に対するＴＩＤ１－ＴＩＤ３及びＴＩＤ１－ＴＩＤ３レジスタを含むことができる。

【００７４】

図１５は、実施形態によるマルチスレッディング・レジスタ復元の例を示す。図１５の例は、単一のコンフィギュレーション１３０２を有するコンピュータシステム１３００を含む。コンフィギュレーション１３０２は、コア１３０４、コア１３０６、及びコア１３０８を含む。コア１３０４－１３０８のコアの各々は、この例おいて最大の４つのスレッド（ＴＩＤ０、ＴＩＤ１、ＴＩＤ２、及びＴＩＤ３）を含む。ＭＴモード１３１０では、コア１３０４－１３０８においてＴＩＤ０－ＴＩＤ３のスレッド・コンテキストの全てが利用可能である。時刻１３１２において、ＭＴモード１３１０は、コンフィギュレーション１３０２のｌｏａｄ－ｎｏｒｍａｌ動作又は非クリアリング・リセットによりディスエーブルにされることができ、これがコア１３０４－１３０８をＳＴモード１３１４に切り換える。ＳＴモード１３１４において、ＴＩＤ０レジスタは、アクセス可能のままであり、ＴＩＤ１－ＴＩＤ３レジスタは、アクセス不能であるが、コア１３０４－１３０８の各々に対して保持されている。時刻１３１６において、ＭＴは、ＳＩＧＰｓｅｔ－ＭＴ指令の実行により再イネーブルにされ、再開されたＭＴモード１３１８に入ることができる。再開されたＭＴモード１３１８において、コア１３０４－１３０８の各々に対して、ＴＩＤ１－ＴＩＤ３レジスタのスレッド・コンテキストへのアクセスが復元される。これは、ダンププログラム、例えばスタンドアロン型ダンププログラム１３２０によるＴＩＤ１－ＴＩＤ３レジスタを含む全てのスレッド・レジスタの検査を可能にして、スレッド・コンテキスト情報を解析のために保存する。

【００７５】

図１６は、図１５のスタンドアロン型ダンププログラム１３２０などのスタンドアロン型ダンピング（ＳＡＤＭＰ）プログラムを使用して、オペレーショナルシステムの故障後にスレッドのアーキテクチャ化レジスタ・コンテキストをキャプチャすることができるような、実施形態によるマルチスレッディング・レジスタ復元のためのプロセスフロー１４００を示す。ブロック１４０５において、ＳＡＤＭＰプログラムが非クリアリング・ロード動作（例えば、ｌｏａｄ－ｎｏｒｍａｌ又はｌｏａｄ－ｗｉｔｈ－ｄｕｍｐ）を介してロードされる。非クリアリング・ロード動作は、図１５のコンフィギュレーション１３０２の場合のＳＴモード１３１４のように、暗黙的にコンフィギュレーションをＳＴモードに戻させる。ＳＡＤＭＰプログラムは、次にブロック１４１０において、ＭＴファシリティがコンフィギュレーション内で利用可能であるかどうかを、ＳＴＳＩ又はＳＥＲＶＣ命令を用いることによりクエリする。ＭＴがインストールされている場合、ＳＡＤＭＰプログラムは、ブロック１４１５において、コンフィギュレーションに対して設定された最終設定プログラム指定最大スレッド識別（ＰＳＭＴＩＤ）をクエリする。ＭＴがそのコンフィギュレーションに対して今まで一度も設定されていなかった場合、最終設定ＰＳＭＴＩＤ値はゼロになる。ＳＡＤＭＰプログラムは、最終設定ＰＳＭＴＩＤが何であっても（ゼロであっても）、次にブロック１４２０においてマルチスレッディングを再イネーブルにする命令を実行することができる。ブロック１４１０でのクエリが、ＭＴがインストールされていないことを明らかにした場合、ブロック１４１５での最終設定ＰＳＭＴＩＤ値のクエリ又はブロック１４２０でのＭＴの再イネーブルの試行はいずれも行われない。

【００７６】

ＳＡＤＭＰプログラムは、コンフィギュレーション内の他のＣＰＵ（スレッド）に信号を送って、そのアーキテクチャ化レジスタ・コンテキストをメモリ内の所定の場所に保存するよう試行する。ＭＴがＳＡＤＭＰをロードする以前にイネーブルにされていなかった

場合、ＣＰＵアドレスは、ノーマルな非拡張形式である。ＭＴが以前にイネーブルにされていた場合、ＣＰＵアドレスは、コアＩＤ及びスレッドＩＤを含む拡張形式である。ＳＡＤＭＰは、ブロック１４２５において、ＣＰＵアドレス（Ｎ）ゼロから開始して、ブロック１４３０においてそのＣＰＵアドレスがＳＡＤＭＰを実行しているＣＰＵを表しているか否かを判定する。表している場合、ＣＰＵ／スレッドはスキップされ、ブロック１４５０においてＮは次の値に増分される。Ｎが現行ＣＰＵアドレスとは異なる場合、ブロック１４３５において、そのアーキテクチャ化レジスタ・コンテキストを、例えばＳＩＧＰ「アドレスにステータス格納」（ｓｔｏｒｅ－ｓｔａｔｕｓ－ａｔ－ａｄｄｒｅｓｓ）指令又はＳＩＧＰ「停止及びステータス格納」（ｓｔｏｐ－ａｎｄ－ｓｔｏｒｅ－ｓｔａｔｕｓ）指令のいずれかを実行することによりメモリに保存するためにＣＰＵ／スレッドに信号が送られる。コンフィギュレーションがベクトル・ファシリティを含む場合、ＳＩＧＰ「アドレスにステータス格納」指令はまた、ＣＰＵ／スレッドのベクトル・レジスタの内容を格納するために実行することもできる。ブロック１４３５の信号が成功したか否かの判定がブロック１４４０で行われる。成功した場合、ブロック１４４５において、ＳＡＤＭＰプログラムは、ＣＰＵ／スレッドのレジスタ・コンテキストをテープ又はディスク上のダンプファイルに保存することができ、ブロック１４５０においてＮを増分することにより処理が続く。ブロック１４３５の信号が成功しなかった（例えばスレッドがオペレーショナルでない場合）と、ブロック１４４０で判定された場合、これはスキップされ、ブロック１４５０でＮを増分することにより処理が続く。信号送信で用いられるＣＰＵアドレスの値（Ｎ）は、ブロック１４５０で増分され、ブロック１４５５において、Ｎがこのときにコンフィギュレーションに対する最大可能ＣＰＵアドレスより大きいか否かの判定が行われる。Ｎがコンフィギュレーションに対する最大可能ＣＰＵアドレスよりも大きくない場合、ブロック１４３０において、Ｎが、ＳＡＤＭＰプログラムを実行している現行ＣＰＵ／スレッドを表しているか否かを判定することにより処理は続く。Ｎがコンフィギュレーションに対する最大可能ＣＰＵアドレスよりも大きい場合、ブロック１４６０において、アーキテクチャ化レジスタ・コンテキスト復元及びダンピングが完了する。

【００７７】

図１６はコンフィギュレーションの１つのコアに関して説明しているが、図１６のプロセスフローは、多重コアを含むコンフィギュレーションの全てのコアにわたって最大ＣＰＵアドレスを通じて実行するように拡張することができる。ＭＴをサポートしないＯＳ又はＭＴアウェア（MT-aware）であるがＭＴを活用しないプログラムに対するダンピングをサポートするために、コンフィギュレーション内で付加的な適応を行うことができる。例えば、コンフィギュレーション内でＭＴをサポートしないＯＳをロードする前にクリアリング・リセットを行って、ＭＴアウェアのスタンドアロン型ダンププログラムが何らかの第二スレッドをコンフィギュレーションからダンプすることを試行することを防止することができる。別の例として、ＭＴアウェアであるがＭＴを活用しないプログラムは、コンフィギュレーションに対するスタンドアロン型ダンププログラムの実行の前に、対応する最大スレッドｉｄがゼロのｓｅｔ－ＭＴ指令を発行することができる。

【００７８】

技術的効果及び利益は、マルチスレッディング能力情報の取得及びマルチスレッディング能力情報を利用可能にすることを含む。マルチスレッディング能力情報は、制御プログラムがマルチスレッディング制約条件を発見することを可能にして、所望の数のスレッドがサポートされることができることを保証する。マルチスレッディング能力情報はまた、以前にマルチスレッディングが最後にイネーブルにされたときに用いられていたスレッドと同数のスレッドで単一スレッド・モードからマルチスレッディング・モードを再イネーブルにすることを補助するために用いることもできる。

【００７９】

本明細書で説明するシステムは、ＯＳがＭＴハードウェアを活用すること明示的に「オプトイン」するよう要求することにより、ハードウェアの変動性を軽減するためのソフトウェアをイネーブルにする。ＯＳが実行環境のＭＴの性質を理解している場合、ＯＳは、

コア当たりのスレッド密度を明示的に管理する能力を有する（所与の作業負荷ディスパッチパターンで、その能力の及ぶ限りで）。ＯＳは、コンピュート・リソースの利用が少ないときでも高スレッド密度を維持するオプションを有し、これにより他のＭＴ実装で見られる総コンピュート容量におけるかなりの変動性を軽減する。高スレッド密度を維持することの直接的結果として、トランザクション応答時間及びビリングの態様の両方をより一貫したものにすることができる。

【００８０】

実施形態は、マルチスレッディング能力情報取得のためのシステム、方法、及びコンピュータ・プログラム製品を含む。１つの態様によれば、コンピュータシステムは、単一スレッド（ＳＴ）モードとマルチスレッディング（ＭＴ）モードとの間で構成可能な１つ又は複数のコアを有するコンフィギュレーションを含む。ＳＴモードは、第一スレッドをサポートし、ＭＴモードは、各コアの共有リソース上で第一スレッド及び１以上の第二スレッドをサポートする。コンピュータシステムはまた、コンフィギュレーションの利用を制御して、コアにより、マルチスレッディング能力情報取得（ｒｅｔｒｉｅｖｅ　ｍｕｌｔｉｔｈｒｅａｄｉｎｇ　ｃａｐａｂｉｌｉｔｙ　ｉｎｆｏｒｍａｔｉｏｎ、ＲＭＴＣＩ）命令を実行することを含む方法を行うように構成された、マルチスレッディング・ファシリティを含む。実行は、コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を取得することと、取得されたスレッド識別情報を格納することと、を含む。

【００８１】

別の態様によれば、コンフィギュレーションにおけるマルチスレッディング能力情報取得のためのコンピュータ実装の方法が提供される。コンフィギュレーションは、ＳＴモードとＭＴモードとの間で構成可能な１つ又は複数のコアを含み、ここでＳＴモードは、第一スレッドをサポートし、ＭＴモードは、各コアの共有リソース上で第一スレッド及び１以上の第二スレッドをサポートする。方法は、コアにより、マルチスレッディング能力情報取得命令を実行することを含む。実行は、コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を取得することと、取得されたスレッド識別情報を格納することと、を含む。

【００８２】

さらなる態様は、コンフィギュレーションにおけるマルチスレッディング能力情報取得のためのコンピュータ・プログラム製品を含む。コンフィギュレーションは、ＳＴモードとＭＴモードとの間で構成可能な１つ又は複数のコアを含み、ここでＳＴモードは、第一スレッドをサポートし、ＭＴモードは、各コアの共有リソース上で第一スレッド及び１以上の第二スレッドをサポートする。コンピュータ・プログラム製品は、具体化されたプログラム命令を有するコンピュータ可読ストレージ媒体を含み、ここでコンピュータ可読媒体は、信号ではない。プログラム命令は、処理回路により可読であり、処理回路に方法を実行させる。方法は、コアにより、マルチスレッディング能力情報取得命令を実行することを含む。実行は、コンフィギュレーションのマルチスレッディング能力を識別するスレッド識別情報を取得することと、取得されたスレッド識別情報を格納することと、を含む。

【００８３】

上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、取得したスレッド識別情報に基づいて、コンフィギュレーションは以前ＭＴモードがイネーブルであったか否か判定する場合を含むことができる。

上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、ＲＭＴＣＩ命令を実行したコアが、ＳＴモードであり、ＳＴモードより以前にＭＴモードがイネーブルであったと判定された場合、当該ＭＴモードをリ・イネーブルにする場合を含むことができる。

上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、ＭＴモードをリ・イネーブルにすると、以前のＭＴモードにおける第二スレッドの数がコア上にリス

トアされる場合を含むことができる。
上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、ＲＭＴＣＩ命令を実行したコアが、ＳＴモードであり、ＳＴモードより以前にＭＴモードがイネーブルであったと判定されなかった場合、当該ＳＴモードを保つ場合を含むことができる。
上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、ＲＭＴＣＩ命令が、サービス・コール（ｓｅｒｖｉｃｅ　ｃａｌｌ、ＳＥＲＶＣ）命令又はシステム情報格納（ｓｔｏｒｅ　ｓｙｓｔｅｍ　ｉｎｆｏｒｍａｔｉｏｎ、ＳＴＳＩ）命令のいずれか１つであり、ＳＥＲＶＣ命令は、スレッド識別情報をメモリ内の応答ブロックに格納するように構成され、ＳＴＳＩ命令は、スレッド識別情報をメモリ内のシステム情報ブロック（ＳＹＳＩＢ）内に格納するように構成される場合を含むことができる。

【００８４】
上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、応答ブロックが、サポートされるプログラム指定最大スレッド識別子値を示すビットのマスクをさらに含む場合を含むことができる。

【００８５】
上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、取得したスレッド識別情報が、コアがマルチスレッディングをサポートするか否かを示すＭＴインストール識別子を含む場合を含むことができる。

【００８６】
上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、取得したスレッド識別情報が、コアの最大サポートスレッド数を示す最大スレッド識別子を含む場合を含むことができる。

【００８７】
上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、取得したスレッド識別情報が、コンフィギュレーションのコアに直近に設定されたプログラム指定最大スレッド識別子を含む場合を含むことができる。

【００８８】
上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、直近に設定されたプログラム指定最大スレッド識別子が示す非ゼロ値に基づいて、コンフィギュレーションは以前にＭＴモードがイネーブルであったかを判定することをさらに含むことができる。

【００８９】
上記の特徴の１つ又は複数に加えて、又は代替的に、さらなる実施形態は、コンフィギュレーションが複数のコアタイプをサポートし、コアの最大サポートスレッド数を示す最大スレッド識別子が、コンフィギュレーション内のコアタイプの各々に対して維持される場合を含むことができる。

【００９０】
本明細書において用いられる用語は、特定の例を説明する目的のためのものにすぎず、本発明を限定することを意図したものではない。本明細書で用いられる場合、単数形「１つの（a）」、「１つの（an）」及び「その（the）」は、文脈が明らかにそうでないことを示していない限り、複数形もまた同様に含むことを意図したものである。「備える、含む」及び／又は「備えている、含んでいる」という用語は、本明細書で用いられる場合、記述された特徴、整数、ステップ、動作、要素、及び／又はコンポーネントの存在を指定するが、１つ又は複数のその他の特徴、整数、ステップ、動作、要素、コンポーネント、及び／又はそれらの群の存在又は付加を排除するものではないことが更に理解されるであろう。

【００９１】
以下の特許請求の範囲における全ての「手段又はステップと機能との組み合わせ（ミーンズ又はステップ・プラス・ファンクション）」要素の対応する構造、材料、動作、及び均等物は、その機能を、明確に特許請求された他の請求要素との組み合わせで実行するた

めのあらゆる構造、材料、又は動作を含むことが意図されている。本発明の説明は、例証及び説明を目的として提示されたものであるが、網羅的であること又は本発明を開示された形態に限定することを意図したものではない。本発明の範囲及び思想から逸脱しない多くの修正及び変形が当業者には明らかとなるであろう。実施形態は、本発明の原理及び実際の用途を最も良く説明するために、そして、当業者が、企図した特定の用途に適した種々の修正を伴う種々の実施形態に関して本発明を理解できるように、選択及び説明したものである。

【００９２】

本発明の種々の実施形態の説明を例証のために提示したが、これらは網羅的であること又は開示された実施形態に限定することを意図したものではない。当業者には、説明した実施形態の範囲及び思想から逸脱しない多くの修正及び変形が明白となるであろう。本明細書で用いた用語は、実施形態の原理、実際的用途、又は市場で見いだされる技術に対する技術的改良点を最も良く説明するように、又は、当業者が本明細書で開示した実施形態を理解することができるように、選択されたものである。

【００９３】

ここで図１７を参照すると、コンピュータ可読媒体１５０２及びプログラム命令１５０４を含む、実施形態によるコンピュータ・プログラム製品１５００が一般的に示されている。

【００９４】

本発明は、システム、方法、及び／又はコンピュータ・プログラム製品とすることができる。コンピュータ・プログラム製品は、本発明の態様をプロセッサに実行させるためのコンピュータ可読プログラム命令を有する１つ又は複数のコンピュータ可読ストレージ媒体を含むことができる。

【００９５】

コンピュータ可読ストレージ媒体は、命令実行デバイスが使用するための命令を保持し、格納することができる有形デバイスとすることができる。コンピュータ可読ストレージ媒体は、例えば、電子ストレージ・デバイス、磁気ストレージ・デバイス、光ストレージ・デバイス、半導体ストレージ・デバイス、又は上記のものの任意の適切な組合せとすることが出来るがこれらに限定されない。コンピュータ可読ストレージ媒体のより具体的な例の非網羅的なリストは、ポータブル・コンピュータ・ディスケット、ハードディスク、ランダム・アクセス・メモリ（ＲＡＭ）、読み出し専用メモリ（ＲＯＭ）、消去可能プログラム可能読み出し専用メモリ（ＥＰＲＯＭ又はフラッシュメモリ）、静的ランダム・アクセス・メモリ（ＳＲＡＭ）、ポータブル・コンパクトディスク読み出し専用メモリ（ＣＤ－ＲＯＭ）、デジタル多目的ディスク（ＤＶＤ）、メモリスティック、フロッピーディスク、パンチカード若しくは記録された命令を有する溝内に隆起した構造のような機械式コード化デバイス、及び上記のものの任意の適切な組合せを含む。コンピュータ可読ストレージ媒体は、本明細書で用いられる場合、無線波若しくは他の自由に伝搬する電磁波、導波路若しくは他の伝送媒体を通って伝搬する電磁波（例えば光ファイバケーブルを通る光パルス）、又は電線を通って伝送される電気信号のような、それ自体が一時的な信号と解釈されるべきではない。

【００９６】

本明細書で説明されるコンピュータ可読プログラム命令は、コンピュータ可読ストレージ媒体からそれぞれのコンピューティング／処理デバイスにダウンロードすることも、又は、例えばインターネット、ローカルエリアネットワーク、広域ネットワーク及び／又は無線ネットワークを経由して、外部コンピュータ若しくは外部ストレージ・デバイスにダウンロードすることもできる。ネットワークは、銅伝送ケーブル、光伝送ファイバ、無線伝送、ルータ、ファイアウォール、スイッチ、ゲートウェイコンピュータ及び／又はエッジサーバを含むことができる。各コンピューティング／処理デバイス内のネットワーク・アダプタ・カード又はネットワークインタフェースは、ネットワークからコンピュータ可読プログラム命令を受け取り、コンピュータ可読プログラム命令をそれぞれのコンピュー

ティング／処理装置内のコンピュータ可読ストレージ媒体にストレージのために転送する。

【００９７】

本発明の動作を実行するためのコンピュータ可読プログラム命令は、アセンブラ命令、命令セット・アーキテクチャ（ＩＳＡ）命令、機械語命令、機械依存命令、マイクロコード、ファームウェア命令、状態設定データ（state-setting data）、又は、Ｓｍａｌｌｔａｌｋ、Ｃ＋＋などのオブジェクト指向プログラミング言語及び「Ｃ」プログラミング言語若しくは類似のプログラミング言語のような従来の手続き型プログラミング言語を含む１つ若しくは複数のプログラミング言語の任意の組合せで記述されたソースコード若しくはオブジェクトコードのいずれか、とすることができる。コンピュータ可読プログラム命令は、完全にユーザのコンピュータ上で実行される場合もあり、一部がユーザのコンピュータ上で、独立型ソフトウェア・パッケージとして実行される場合もあり、一部がユーザのコンピュータ上で実行され、一部が遠隔コンピュータ上で実行される場合もあり、又は完全に遠隔コンピュータ若しくはサーバ上で実行される場合もある。後者のシナリオにおいては、遠隔コンピュータは、ローカルエリアネットワーク（ＬＡＮ）若しくは広域ネットワーク（ＷＡＮ）を含むいずれかのタイプのネットワークを通じてユーザのコンピュータに接続される場合もあり、又は外部コンピュータへの接続が為される場合もある（例えば、インターネット・サービス・プロバイダを用いたインターネットを通じて）。幾つかの実施形態において、例えばプログラム可能論理回路、フィールドプログラム可能ゲートアレイ（ＦＰＧＡ）、又はプログラム可能論理アレイ（ＰＬＡ）を含む電子回路は、本発明の態様を実施するために、コンピュータ可読プログラム命令の状態情報を利用して電子回路を個別化することにより、コンピュータ可読プログラム命令を実行することができる。

【００９８】

本発明の態様は、本明細書において、本発明の実施形態による方法、装置（システム）、及びコンピュータ・プログラム製品のフローチャート図及び／又はブロック図を参照して説明される。フローチャート図及び／又はブロック図の各ブロック、並びにフローチャート図及び／又はブロック図のブロックの組合せは、コンピュータ可読プログラム命令によって実装することができることが理解されよう。

【００９９】

これらのコンピュータ可読プログラム命令を、汎用コンピュータ、専用コンピュータ、又は他のプログラム可能データ処理装置のプロセッサに与えてマシンを製造し、それにより、コンピュータ又は他のプログラム可能データ処理装置のプロセッサによって実行される命令が、フローチャート及び／又はブロック図の１つ又は複数のブロック内で指定された機能／動作を実装するための手段を作り出すようにすることができる。これらのコンピュータ・プログラム命令を、コンピュータ、プログラム可能データ処理装置、及び／又は他のデバイスを特定の方式で機能させるように指示することができるコンピュータ可読媒体内に格納し、それにより、その中に格納された命令を有するコンピュータ可読媒体が、フローチャート及び／又はブロック図の１つ又は複数のブロックにおいて指定された機能／動作の態様を実装する命令を含む製品を含むようにすることもできる。

【０１００】

コンピュータ可読プログラム命令を、コンピュータ、他のプログラム可能データ処理装置又は他のデバイス上にロードして、一連の動作ステップをコンピュータ、他のプログラム可能データ処理装置又は他のデバイス上で行わせてコンピュータ実装のプロセスを生成し、それにより、コンピュータ、他のプログラム可能装置又は他のデバイス上で実行される命令が、フローチャート及び／又はブロック図の１つ又は複数のブロックにおいて指定された機能／動作を実装するようにすることもできる。

【０１０１】

図面内のフローチャート及びブロック図は、本開示の種々の実施形態による、システム、方法、及びコンピュータ・プログラム製品の可能な実装の、アーキテクチャ、機能及び

動作を示す。この点に関して、フローチャート又はブロック図内の各ブロックは、指定された論理機能を実装するための１つ又は複数の実行可能命令を含む、モジュール、セグメント、又は命令の一部を表すことができる。幾つかの代替的な実装において、ブロック内に記された機能は、図中に記された順序とは異なる順序で行われることがある。例えば、連続して示された２つのブロックは、関与する機能に応じて、実際には実質的に同時に実行されることもあり、又はこれらのブロックはときとして逆順で実行されることもある。ブロック図及び／又はフローチャート図の各ブロック、及びブロック図及び／又はフローチャート図中のブロックの組合せは、指定された機能又は動作を実行する専用ハードウェア・ベースのシステムによって実装することもでき、又は専用ハードウェアとコンピュータ命令との組合せを実行することもできることにも留意されたい。

【符号の説明】

【０１０２】

１００、１１００、１３００：コンピュータシステム

２００：処理回路

１１０２、１１０４、１３０２：コンフィギュレーション

１１１４、１１１６、１３１０：ＭＴモード

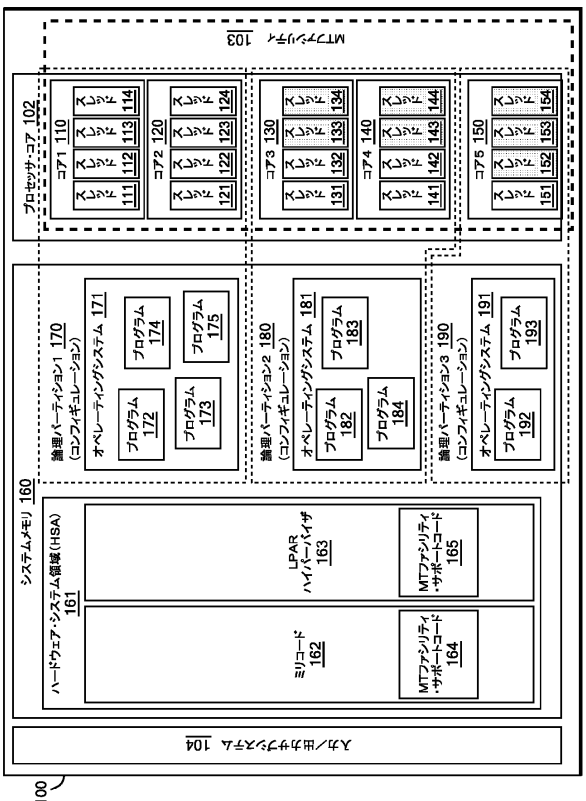
１１１５：スレッド・コンテキスト

１１１８、１１２２、１３１２、１３１６：時刻

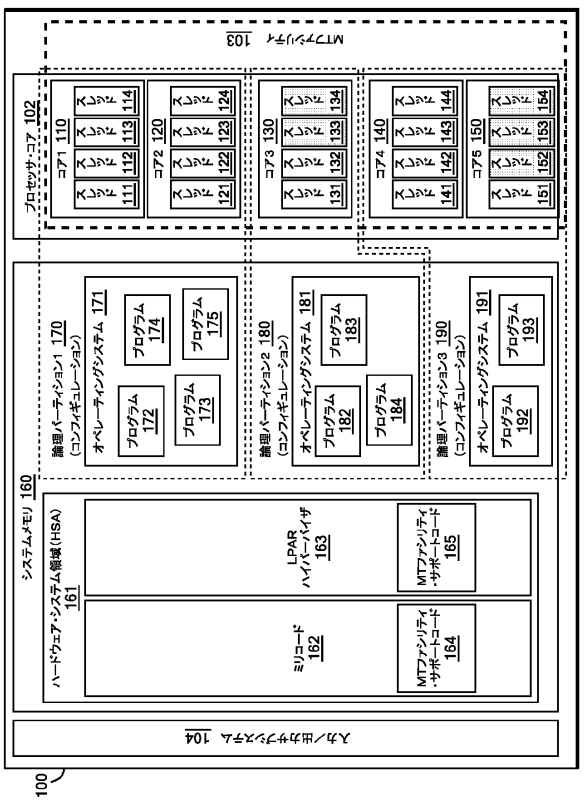
１１２０、１１２４、１３１４：ＳＴモード

１３１８：再開されたＭＴモード

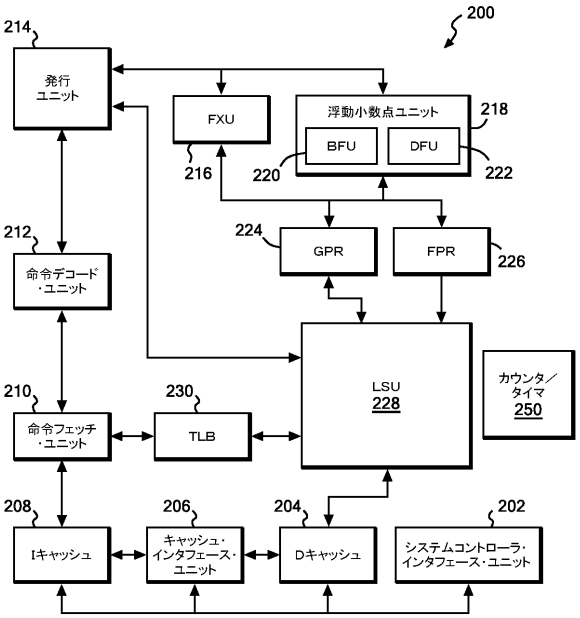
【図１】



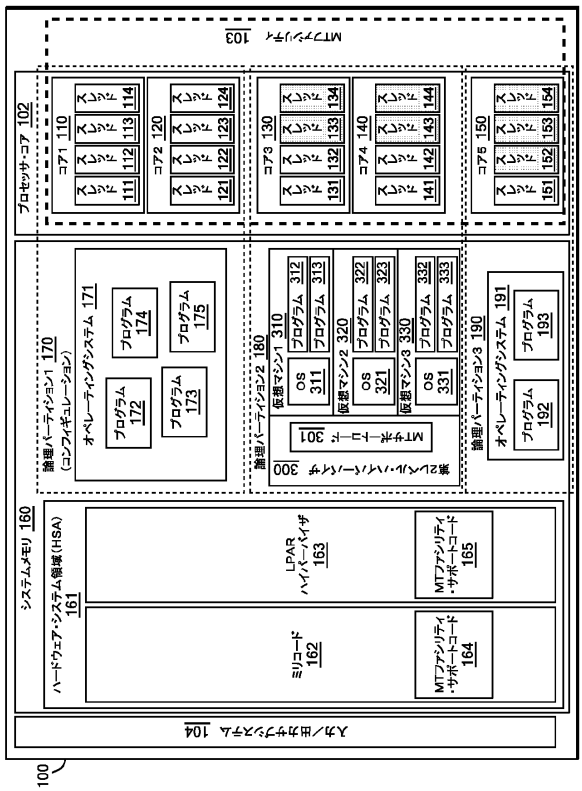
【図２】



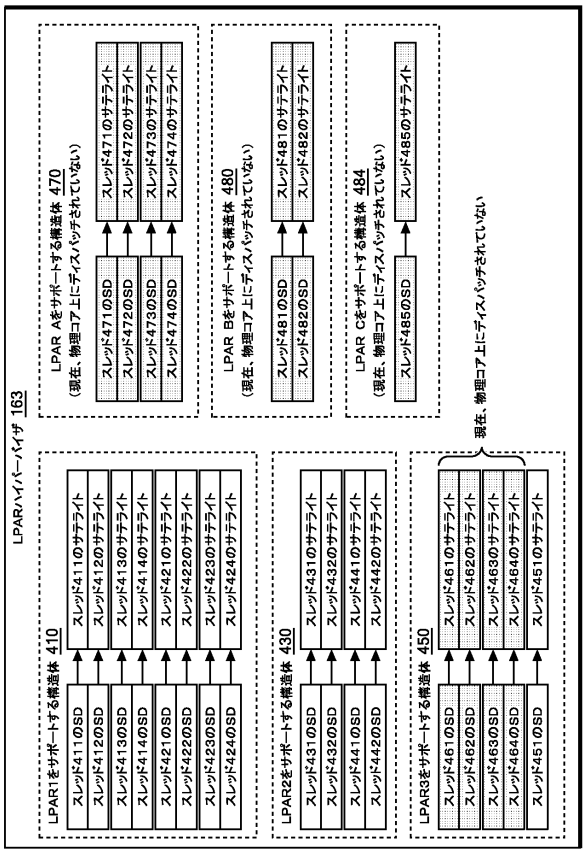
【図３】


200
214
発行ユニット
FXU
216
浮動小数点ユニット
218
BFU
DFU
220
222
212
命令デコード・ユニット
224
GPR
FPR
226
210
命令フェッチ・ユニット
230
TLB
LSU
228
カウンタ／タイマ
250
208
Iキャッシュ
206
キャッシュ・インタフェース・ユニット
204
Dキャッシュ
202
システムコントローラ・インタフェース・ユニット

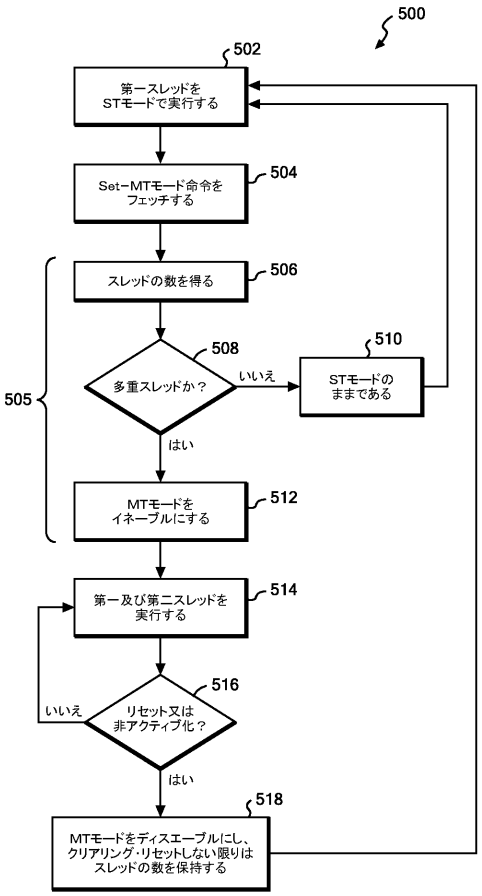
【図４】


100
I/O／出力プロセス 104
システムメモリ 160
ハードウェア・システム領域(HSA) 161
LPAR ハイパーバイザ 163
MTファシリティ・サポートコード 165
ミリコード 162
MTファシリティ・サポートコード 164
論理パーティション1 170
(コンフィギュレーション)
オペレーティングシステム 171
プログラム 174
プログラム 175
プログラム 172
プログラム 173
論理パーティション2 180
仮想マシン1 310
OS 311
プログラム 312
プログラム 313
仮想マシン2 320
OS 321
プログラム 322
プログラム 323
仮想マシン3 330
OS 331
プログラム 332
プログラム 333
第2レベル・ハイパーバイザ 300
MTサポート 301
論理パーティション3 190
オペレーティングシステム 191
プログラム 193
プログラム 192
プロセッサ・コア 102
コア1 110
コア2 120
コア3 130
コア4 140
コア5 150
スレッド
MTプロセッサ 103

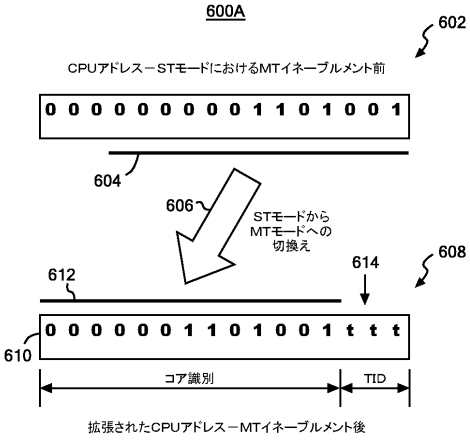
【図５】


LPARハイパーバイザ 163
LPAR Aをサポートする構造体 470
(現在、物理コア上にディスパッチされていない)
スレッド4710のサテライト
スレッド4720のサテライト
スレッド4730のサテライト
スレッド4740のサテライト
スレッド4710のSD
スレッド4720のSD
スレッド4730のSD
スレッド4740のSD
LPAR Bをサポートする構造体 480
(現在、物理コア上にディスパッチされていない)
スレッド4810のサテライト
スレッド4820のサテライト
スレッド4810のSD
スレッド4820のSD
LPAR Cをサポートする構造体 484
(現在、物理コア上にディスパッチされていない)
スレッド4850のサテライト
スレッド4850のSD
現在、物理コア上にディスパッチされていない
LPAR1をサポートする構造体 410
スレッド4110のサテライト
スレッド4120のサテライト
スレッド4130のサテライト
スレッド4140のサテライト
スレッド4210のサテライト
スレッド4220のサテライト
スレッド4230のサテライト
スレッド4240のサテライト
スレッド4110のSD
スレッド4120のSD
スレッド4130のSD
スレッド4140のSD
スレッド4210のSD
スレッド4220のSD
スレッド4230のSD
スレッド4240のSD
LPAR2をサポートする構造体 430
スレッド4310のサテライト
スレッド4320のサテライト
スレッド4410のサテライト
スレッド4420のサテライト
スレッド4310のSD
スレッド4320のSD
スレッド4410のSD
スレッド4420のSD
LPAR3をサポートする構造体 450
スレッド4610のサテライト
スレッド4620のサテライト
スレッド4630のサテライト
スレッド4640のサテライト
スレッド4510のサテライト
スレッド4610のSD
スレッド4620のSD
スレッド4630のSD
スレッド4640のSD
スレッド4510のSD

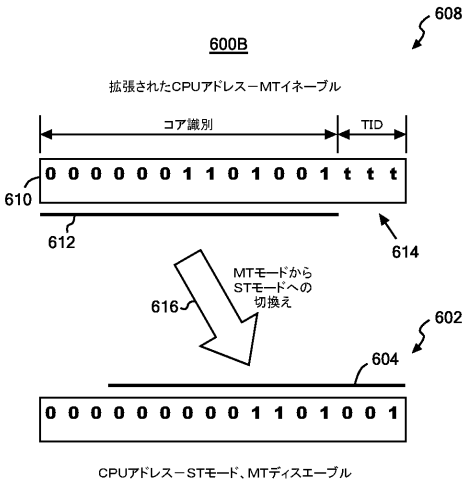
【図６】


500
502
第一スレッドをSTモードで実行する
504
Set−MTモード命令をフェッチする
506
スレッドの数を得る
505
508
多重スレッドか？
いいえ
510
STモードのままである
はい
512
MTモードをイネーブルにする
514
第一及び第二スレッドを実行する
516
リセット又は非アクティブ化？
いいえ
はい
518
MTモードをディスエーブルにし、クリアリング・リセットしない限りはスレッドの数を保持する

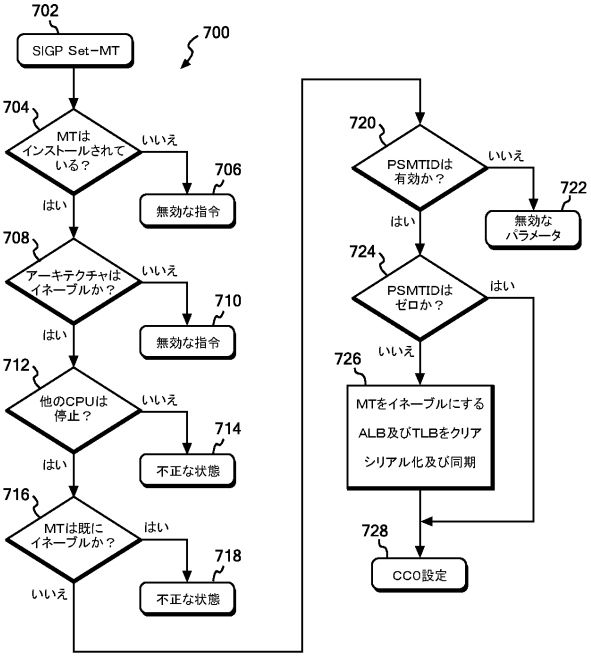
【図７】


600A
602
CPUアドレス－STモードにおけるMTイネーブルメント前
0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1
604
606
STモードから
MTモードへの
切換え
608
612
614
0 0 0 0 0 0 1 1 0 1 0 0 1 t t t
610
コア識別
TID
拡張されたCPUアドレス－MTイネーブルメント後

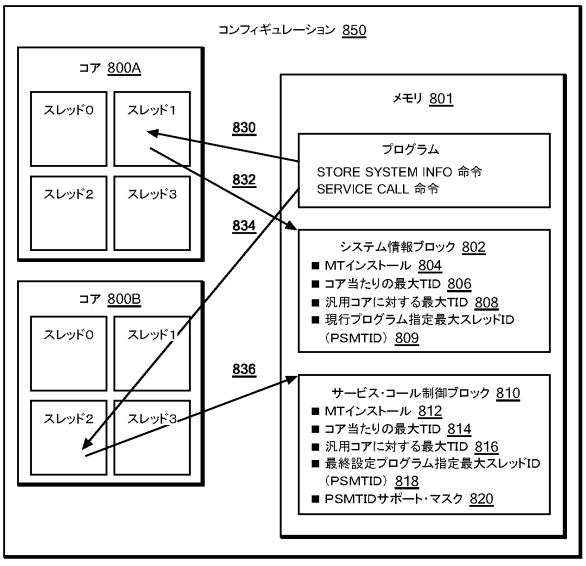
【図８】


608
600B
拡張されたCPUアドレス－MTイネーブル
コア識別
TID
0 0 0 0 0 0 1 1 0 1 0 0 1 t t t
610
612
614
MTモードから
STモードへの
切換え
616
602
604
0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1
CPUアドレス－STモード、MTディスエーブル

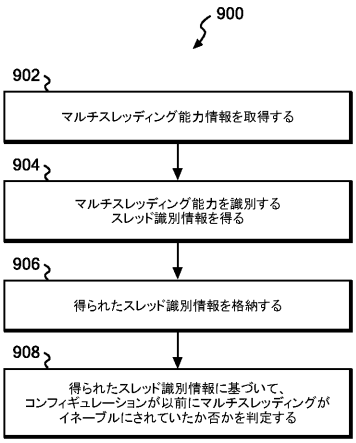
【図９】


702
SIGP Set−MT
700
704
MTは
インストールされて
いる？
いいえ
706
無効な指令
はい
708
アーキテクチャは
イネーブルか？
いいえ
710
無効な指令
はい
712
他のCPUは
停止？
いいえ
714
不正な状態
はい
716
MTは既に
イネーブルか？
はい
718
不正な状態
いいえ
720
PSMTIDは
有効か？
いいえ
722
無効な
パラメータ
はい
724
PSMTIDは
ゼロか？
はい
いいえ
726
MTをイネーブルにする
ALB及びTLBをクリア
シリアル化及び同期
728
CC0設定

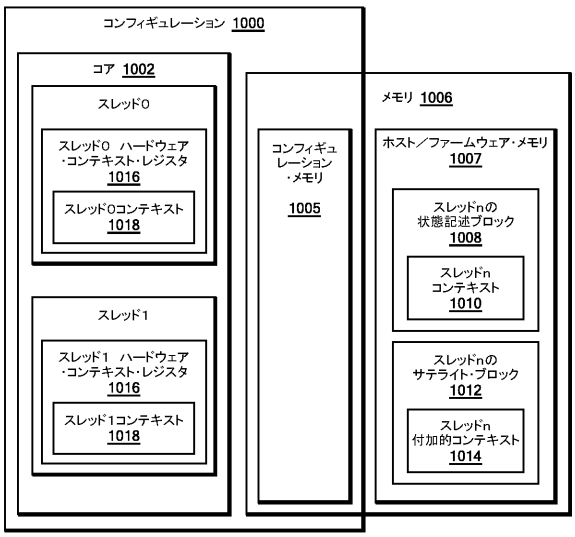
【図１０】


コンフィギュレーション 850
コア 800A
スレッド0
スレッド1
スレッド2
スレッド3
コア 800B
スレッド0
スレッド1
スレッド2
スレッド3
830
832
834
836
メモリ 801
プログラム
STORE SYSTEM INFO 命令
SERVICE CALL 命令
システム情報ブロック 802
■ MTインストール 804
■ コア当たりの最大TID 806
■ 汎用コアに対する最大TID 808
■ 現行プログラム指定最大スレッドID
（PSMTID） 809
サービス・コール制御ブロック 810
■ MTインストール 812
■ コア当たりの最大TID 814
■ 汎用コアに対する最大TID 816
■ 最終設定プログラム指定最大スレッドID
（PSMTID） 818
■ PSMTIDサポート・マスク 820

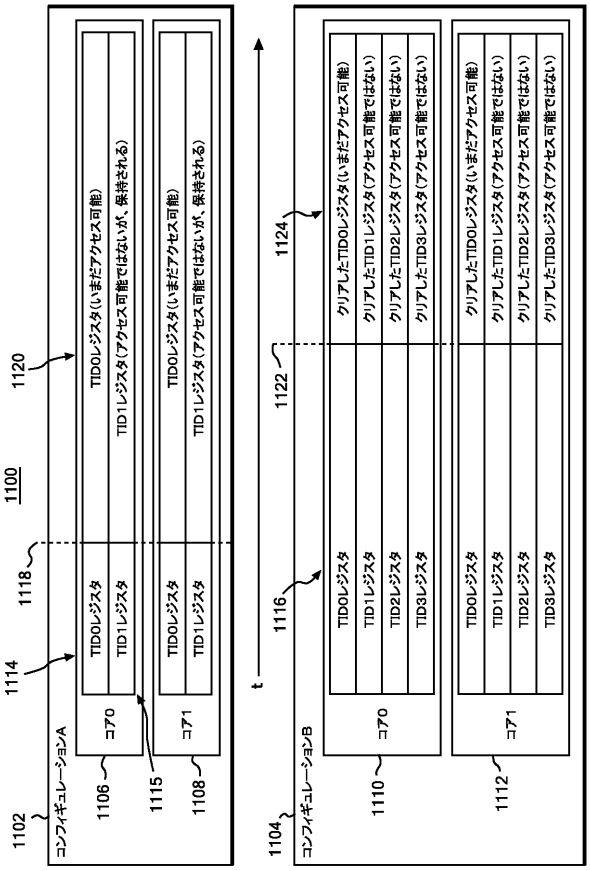
【図１１】



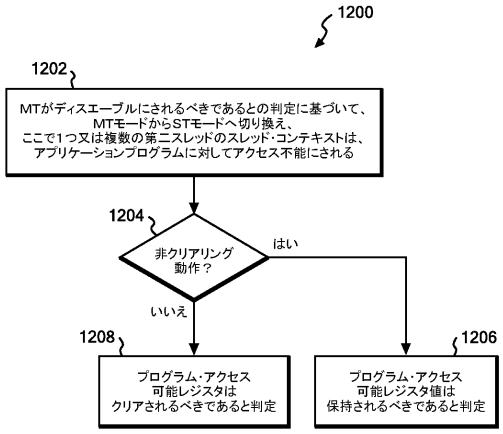
【図１２】



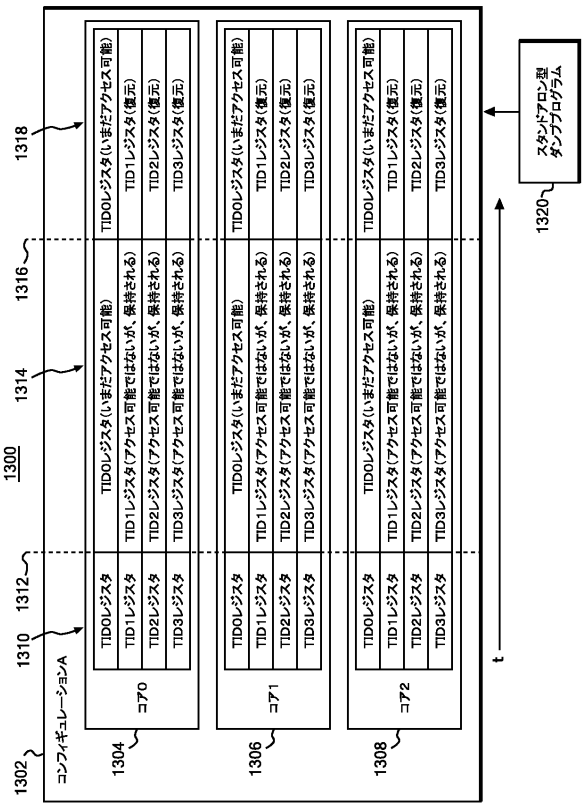
【図１３】



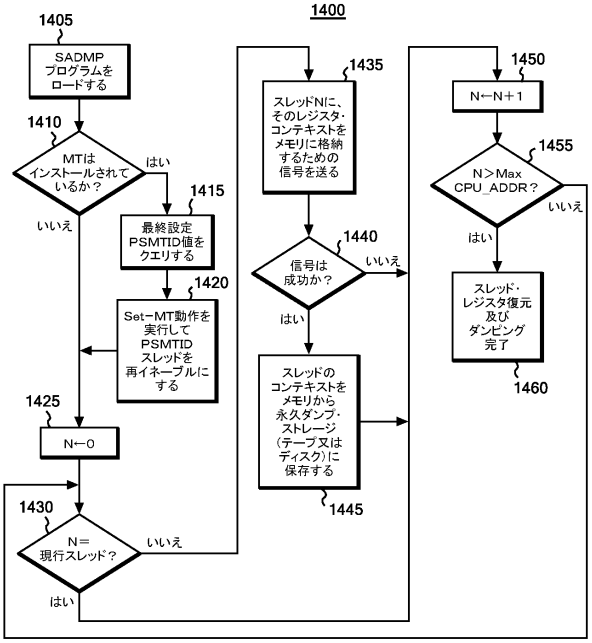
【図１４】



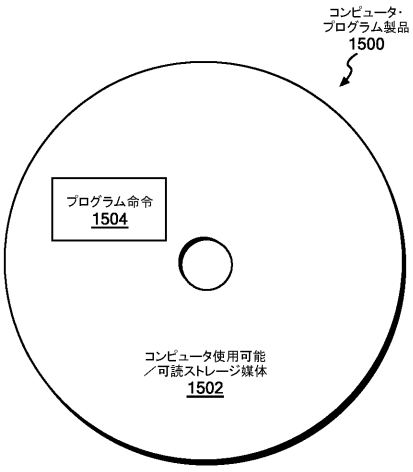
【図１５】


1300
1302
コンフィギュレーションA
1304
コア0
1306
コア1
1308
コア2
1310
1312
1314
1316
1318
TID0レジスタ(いまだアクセス可能)
TID1レジスタ(アクセス可能ではないが、保持される)
TID2レジスタ(アクセス可能ではないが、保持される)
TID3レジスタ(アクセス可能ではないが、保持される)
TID0レジスタ
TID1レジスタ
TID2レジスタ
TID3レジスタ
TID0レジスタ(いまだアクセス可能)
TID1レジスタ(復元)
TID2レジスタ(復元)
TID3レジスタ(復元)
スタンドアロン型ダンププログラム
1320
t

【図１６】


1400
1405
SADMPプログラムをロードする
1410
MTはインストールされているか？
はい
いいえ
1415
最終設定PSMTID値をクエリする
1420
Set−MT動作を実行してPSMTIDスレッドを再イネーブルにする
1425
N←0
1430
N＝現行スレッド？
いいえ
はい
1435
スレッドNに、そのレジスタ・コンテキストをメモリに格納するための信号を送る
1440
信号は成功か？
いいえ
はい
1445
スレッドのコンテキストをメモリから永久ダンプ・ストレージ（テープ又はディスク）に保存する
1450
N←N＋1
1455
N＞Max CPU_ADDR？
いいえ
はい
スレッド・レジスタ復元及びダンピング完了
1460

【図１７】


コンピュータ・プログラム製品
1500
プログラム命令
1504
コンピュータ使用可能／可読ストレージ媒体
1502

フロントページの続き

(74)代理人 100112690
弁理士 太佐 種一
(72)発明者 グレイナー、ダン
アメリカ合衆国 ９５１４１－１００３ カリフォルニア州 サンノゼ サンタ・テレサ・ラボラトリー ベイリー・アベニュー ５５５
(72)発明者 ファレル、マーク
アメリカ合衆国 １２６０１－５４００ ニューヨーク州 ポキプシー サウス・ロード ２４５５
(72)発明者 オシセック、ダミアン、レオ
アメリカ合衆国 １３７６０／２５０－２ ニューヨーク州 エンディコット ノース・ストリート １７０１
(72)発明者 シュミット、ドナルド、ウィリアム
アメリカ合衆国 １２６０１－５４００ ニューヨーク州 ポキプシー サウス・ロード ２４５５
(72)発明者 ブサバ、ファディ、ユスフ
アメリカ合衆国 １２６０１－５４００ ニューヨーク州 ポキプシー サウス・ロード ２４５５
(72)発明者 クバラ、ジェフリー、ポール
アメリカ合衆国 １２６０１－５４００ ニューヨーク州 ポキプシー サウス・ロード ２４５５
(72)発明者 ブラッドベリ、ジョナサン、デイビット
アメリカ合衆国 １２６０１－５４００ ニューヨーク州 ポキプシー サウス・ロード ２４５５
(72)発明者 ヘラー、リサ、クラントン
アメリカ合衆国 １２６０１－５４００ ニューヨーク州 ポキプシー サウス・ロード ２４５５
(72)発明者 スレゲル、ティモシー
アメリカ合衆国 １２６０１－５４００ ニューヨーク州 ポキプシー サウス・ロード ２４５５
(72)発明者 ゲイニージュニア、チャールズ
アメリカ合衆国 １２６０１－５４００ ニューヨーク州 ポキプシー サウス・ロード ２４５５

審査官 井上 宏一

(56)参考文献 特表２００７－５３１１６７（ＪＰ，Ａ）
特表２０１１－５０９４７８（ＪＰ，Ａ）
特開平０６－３４８５８４（ＪＰ，Ａ）
特開２００４－３２６７４９（ＪＰ，Ａ）
米国特許出願公開第２００４／０２１５９３９（ＵＳ，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／４５５－９／５４
Ｇ０６Ｆ ９／３０