【公報種別】特許法第１７条の２の規定による補正の掲載
【部門区分】第６部門第３区分
【発行日】平成30年7月12日(2018.7.12)

【公表番号】特表2017-520857(P2017-520857A)
【公表日】平成29年7月27日(2017.7.27)
【年通号数】公開・登録公報2017-028
【出願番号】特願2017-500877(P2017-500877)
【国際特許分類】
Ｇ０６Ｆ　9/30　(2018.01)
Ｇ０６Ｆ　9/46　(2006.01)
Ｇ０６Ｆ　9/32　(2006.01)
Ｇ０６Ｆ　9/34　(2006.01)
Ｇ０６Ｆ　9/38　(2006.01)
【ＦＩ】
Ｇ０６Ｆ　9/40　３１０Ａ
Ｇ０６Ｆ　9/46　４３０
Ｇ０６Ｆ　9/42　３３０Ａ
Ｇ０６Ｆ　9/34　３５０Ａ
Ｇ０６Ｆ　9/38　３８０Ａ

【手続補正書】
【提出日】平成30年6月1日(2018.6.1)
【手続補正１】
【補正対象書類名】特許請求の範囲
【補正対象項目名】全文
【補正方法】変更
【補正の内容】
【特許請求の範囲】
【請求項１】
データ処理命令を実行するように構成されたプロセッサ回路であり、前記プロセッサ回路が、前記プロセッサ回路が実行する前記データ処理命令に依存する呼出しスタック・データ構造体を維持するように構成され、前記プロセッサ回路が、実行される前記データ処理命令がさらなるプロセッサ回路と共用される記憶されたデータ項目にアクセスするときにトランザクション実行モードで動作するように構成される、プロセッサ回路と、
前記呼出しスタック・データ構造体のスタック深さ指示を記憶するように構成されたトランザクション前スタック・ポインタ記憶回路であり、前記プロセッサ回路が、前記トランザクション実行モードに入る前の前記スタック深さ指示を記憶するように構成される、トランザクション前スタック・ポインタ記憶回路と
を備え、
前記プロセッサ回路が、前記トランザクション実行モードで動作するときに、前記スタック深さ指示に関する前記呼出しスタック・データ構造体への修正の相対的スタッキング位置を判定するように、そして、
前記相対的スタッキング位置が、前記スタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合には、前記修正が非投機的であるという指示を前記修正に関連して記憶するように、そして、
前記相対的スタッキング位置が、前記スタック深さ指示によって指示される前記位置に関して正のスタック成長方向にない場合には、前記修正が投機的であるという指示を前記修正に関連して記憶するように構成される、装置。
【請求項２】

前記トランザクション前スタック・ポインタ記憶回路が、前記呼出しスタック・データ構造体の前記プロセッサ回路によって使用されるスタック・ポインタのコピーを記憶するように構成された、請求項１に記載の装置。
【請求項３】
前記トランザクション前スタック・ポインタ記憶回路が、前記プロセッサ回路にアクセス可能なレジスタを備える、請求項１又は２に記載の装置。
【請求項４】
前記レジスタが、前記プロセッサ回路によって実行されるデータ処理命令にアクセス不能に構成される、請求項３に記載の装置。
【請求項５】
前記相対的スタッキング位置が、前記スタック深さ指示によって指示される前記位置に対して前記正のスタック成長方向になく、前記修正が、前記呼出しスタック・データ構造体へのプッシュである場合に、前記プッシュによって前記呼出しスタック・データ構造体にプッシュされるデータ項目が、投機的に書かれたものとして印を付けられる、請求項１から４までのいずれか一項に記載の装置。
【請求項６】
前記相対的スタッキング位置が、前記スタック深さ指示によって指示される前記位置に関して正のスタック成長方向になく、前記修正が、前記呼出しスタック・データ構造体からのポップである場合に、前記ポップによって前記呼出しスタック・データ構造体からポップされるデータ項目が、投機的に読み取られたものとして印を付けられる、請求項１から４までのいずれか一項に記載の装置。
【請求項７】
前記データ処理命令を実行するときにメモリ内の前記プロセッサ回路によってアクセスされるデータ項目のローカル・コピーを記憶するように構成された記憶回路をさらに備え、前記プロセッサ回路が、前記記憶回路に前記呼出しスタック・データ構造体を維持するように構成された、請求項１から６までのいずれか一項に記載の装置。
【請求項８】
前記記憶回路が、整合性制御ユニットを備え、前記整合性制御ユニットが、非投機的として指示された前記記憶回路の内容が前記メモリに取り出されることを可能にするように、及び、前記記憶回路の内容が取り出すために選択され、投機的として指示される場合に、エラー状態を生成するように構成された、請求項７に記載の装置。
【請求項９】
前記整合性制御ユニットが、前記プロセッサ回路が、前記トランザクション実行モードを終了し、前記トランザクション実行モードで実行される前記データ処理命令がそれらのデータ処理動作を無事に完了したときに、投機的として指示された前記記憶回路の内容を非投機的として指示されるように変更するように構成された、請求項８に記載の装置。
【請求項１０】
前記整合性制御ユニットが、投機的として指示された前記記憶回路の内容を前記さらなるプロセッサ回路から隠すように構成された、請求項７から９までのいずれか一項に記載の装置。
【請求項１１】
中止処理回路をさらに備え、前記中止処理回路が、前記プロセッサ回路が前記トランザクション実行モードで動作しているときに、保留中のトランザクションが失敗したという指示に応答してロール・バック手続きを実行するように構成され、
前記ロール・バック手続きが、投機的として指示された任意の修正を破棄するステップ、及び、前記記憶されたスタック深さ指示を使用して現在のスタック・ポインタをリセットするステップを含む、請求項１から１０までのいずれか一項に記載の装置。
【請求項１２】
データ処理命令を実行するための手段と、
実行される前記データ処理命令に依存する呼出しスタック・データ構造体を維持するた

めの手段であり、前記データ処理命令を実行するための手段が、実行される前記データ処理命令がデータ処理命令を実行するためのさらなる手段と共用される記憶されたデータ項目にアクセスするときにトランザクション実行モードで動作するように構成された、手段と、
前記呼出しスタック・データ構造体のスタック深さ指示を記憶するための手段であり、前記スタック深さ指示が、前記データ処理命令を実行するための手段が前記トランザクション実行モードに入る前に記憶される、手段と、
前記トランザクション実行モードで動作するときに前記スタック深さ指示に関する前記呼出しスタック・データ構造体への修正の相対的スタッキング位置を判定するための手段と、
前記修正に関連して指示を記憶するための手段であり、前記相対的スタッキング位置が前記スタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合には、前記指示は、前記修正が非投機的であると示し、
前記相対的スタッキング位置が前記スタック深さ指示によって指示される前記位置に関して正のスタック成長方向にない場合には、前記指示は、前記修正が投機的であると示す、手段と
を備える、装置。
【請求項１３】
プロセッサ・デバイスにおけるデータ処理の方法であって、
データ処理命令を実行するステップと、
実行される前記データ処理命令に依存する呼出しスタック・データ構造体を維持するステップと、
実行される前記データ処理命令がさらなるプロセッサ・デバイスと共用される記憶されたデータ項目へのアクセスを求めるときにトランザクション実行モードに入るステップと、
スタック深さ指示を記憶するステップであり、前記スタック深さ指示が、前記トランザクション実行モードに入る前に記憶される、ステップと、
前記トランザクション実行モードで動作するときに、前記スタック深さインジケータに関する前記呼出しスタック・データ構造体への修正の相対的スタッキング位置を判定するステップと、
前記相対的スタッキング位置が前記スタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合には、前記修正が非投機的であるという指示を前記修正に関連して記憶するステップと、
前記相対的スタッキング位置が前記スタック深さ指示によって指示される前記位置に関して正のスタック成長方向にない場合には、前記修正が投機的であるという指示を前記修正に関連して記憶するステップと
を含む、方法。
【手続補正２】
【補正対象書類名】明細書
【補正対象項目名】全文
【補正方法】変更
【補正の内容】
【発明の詳細な説明】
【発明の名称】トランザクション・データ処理実行モードのための呼出しスタック維持
【技術分野】
【０００１】
本開示は、データ処理に関する。より詳細には、本開示は、トランザクション実行モードでのデータ処理時に使用される呼出しスタック・データ構造体の維持に関する。
【背景技術】
【０００２】

データ・プロセッサは、特に、データ・プロセッサが、関数呼出しに遭遇したときに呼出しスタックに復帰アドレスをプッシュする場合、それが実行するデータ処理命令に依存する呼出しスタックを維持するように、及び、その関数が終了するときに呼出しスタックからその復帰アドレスをポップするように構成され得る。現代のデータ処理装置では、これがもたらすパフォーマンス強化により２つ以上のデータ・プロセッサ（たとえば、プロセッサ・コア）が提供されることが一般的である。しかし、これらの複数のプロセッサによって実行されるデータ処理の結果が予測可能で決定的であることを確保するために、これらの複数のプロセッサによって実行されるデータ処理に関するある一定の制約が、その場合、順守される必要がある。これのそのような１つの実例は、データ構造体がシステム内の少なくとも１つの他のデータ・プロセッサと共用される、そのデータ構造体（たとえば、システム・メモリに記憶された）にアクセスするときの、データ・プロセッサによるトランザクション実行に関する。このデータ構造体の共用は、共用されるデータ構造体に同時修正を行う異なるデータ・プロセッサによるデータ・ハザードを回避するために、ある種のプロトコルが準拠される必要があることを意味する。共用データ構造体に関するトランザクション実行が、無事に完了しない場合、データ・プロセッサが、その投機的修正を破棄し、データ処理シーケンス内の最後の知られている非投機的ポイントまで「ロール・バック」することができるように、データ処理装置内の各データ・プロセッサは、その場合、トランザクション実行モードで動作するときに（すなわち、共用データ構造体へのアクセスが試みられるが、まだ無事に完了していない間）、投機的形で呼出しスタックを更新するように構成され得る。

【発明の概要】

【課題を解決するための手段】

【０００３】

第１の例示的構成によれば、データ処理命令を実行するように構成されたプロセッサ回路であり、それが実行するデータ処理命令に依存する呼出しスタック・データ構造体を維持するように構成されるプロセッサ回路であり、実行されるデータ処理命令がさらなるプロセッサ回路と共用される記憶されたデータ項目にアクセスするときにトランザクション実行モードで動作するように構成されたプロセッサ回路と、呼出しスタック・データ構造体のスタック深さ指示を記憶するように構成されたトランザクション前スタック・ポインタ記憶回路とを備える装置が提供され、プロセッサ回路は、トランザクション実行モードに入る前のスタック深さ指示を記憶するように構成され、プロセッサ回路は、トランザクション実行モードで動作するときに、そのスタック深さ指示に関する呼出しスタック・データ構造体への修正のための相対的スタッキング位置を判定するように構成され、相対的スタッキング位置がスタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合にはその修正が非投機的であるという指示をその修正に関連して記憶するように、及び、相対的スタッキング位置がスタック深さ指示によって指示される位置に関して正のスタック成長方向にない場合にはその修正が投機的であるという指示をその修正に関連して記憶するように構成される。

【０００４】

もう１つの例示的構成によれば、データ処理命令を実行するための手段と、データ処理命令を実行するための手段が、実行されるデータ処理命令がデータ処理命令を実行するためのさらなる手段と共用される記憶されたデータ項目にアクセスするときにトランザクション実行モードで動作するように構成される、実行されるデータ処理命令に依存する呼出しスタック・データ構造体を維持するための手段と、スタック深さ指示が、データ処理命令を実行するための手段がトランザクション実行モードに入る前に記憶される、呼出しスタック・データ構造体のスタック深さ指示を記憶するための手段と、トランザクション実行モードで動作するときにそのスタック深さ指示に関する呼出しスタック・データ構造体への修正の相対的スタッキング位置を判定するための手段と、相対的スタッキング位置がスタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合には、指示はその修正が非投機的であると示し、相対的スタッキング位置がスタック深さ指示

によって指示される位置に関して正のスタック成長方向にない場合には、指示はその修正が投機的であると示す、修正に関連して指示を記憶するための手段を備える装置が提供される。

【０００５】

もう１つの例示的構成によれば、データ処理命令を実行するステップと、実行されるデータ処理命令に依存する呼出しスタック・データ構造体を維持するステップと、実行されるデータ処理命令がさらなるプロセッサ・デバイスと共用される記憶されたデータ項目へのアクセスを求めるときにトランザクション実行モードに入るステップと、スタック深さ指示を記憶するステップであり、前記スタック深さ指示が、前記トランザクション実行モードに入る前に記憶される、ステップと、トランザクション実行モードで動作するときに、スタック深さインジケータに関する呼出しスタック・データ構造体への修正の相対的スタッキング位置を判定するステップと、相対的スタッキング位置が、スタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合には、その修正が非投機的であるという指示をその修正に関連して記憶し、相対的スタッキング位置が、スタック深さ指示によって指示される位置に関して正のスタック成長方向にない場合には、その修正が投機的であるという指示をその修正に関連して記憶するステップとを含む、プロセッサ・デバイスにおけるデータ処理の方法が提供される。

【０００６】

本発明は、単に実例として、以下のような添付の図面に示されるようなその実施例を参照して、さらに説明される。

【図面の簡単な説明】

【０００７】

【図１】メモリ内の共用データ構造体に各々がアクセスすることができる２つのプロセッサ・コアを示す、１つの実施例におけるデータ処理装置を概略的に示す図である。

【図２Ａ】Ｃライクなコード（C-like code）の例示的シーケンスを示す図である。

【図２Ｂ】１つの実施例において呼出しスタックへのプッシュ及び呼出しスタックからのポップに図２Ａのコードにおける関数呼出し及び復帰がどのように対応するかを概略的に示す図である。

【図２Ｃ】１つの実施例における図２Ａのコードの呼出しスタックの内容及び成長を概略的に示す図である。

【図３】図２Ａに示されるコードの実行に対応する１つの実施例における呼出しスタックの内容の進化をより詳細に概略的に示す図である。

【図４Ａ】１つの実施例における記憶ユニットを概略的に示す図である。

【図４Ｂ】図４Ａの記憶ユニットの例示的使用を概略的に示す図である。

【図４Ｃ】図４Ａの記憶ユニットの例示的使用を概略的に示す図である。

【図５】１つの実施例におけるプロセッサ・コア及び関連Ｌ１キャッシュを概略的に示す図である。

【図６】１つの実施例の方法に従って実行されるステップのシーケンスを概略的に示す図である。

【図７】トランザクション中止を監視し、それに反応するために、１つの実施例において実行されるステップのシーケンスを概略的に示す図である。

【発明を実施するための形態】

【０００８】

第１の例示的構成によれば、データ処理命令を実行するように構成されたプロセッサ回路であり、それが実行するデータ処理命令に依存する呼出しスタック・データ構造体を維持するように構成されたプロセッサ回路であり、実行されるデータ処理命令がさらなるプロセッサ回路と共用される記憶されたデータ項目にアクセスするときにトランザクション実行モードで動作するように構成されたプロセッサ回路と、呼出しスタック・データ構造体のスタック深さ指示を記憶するように構成されたトランザクション前スタック・ポインタ記憶回路とを備える装置が提供され、プロセッサ回路は、トランザクション実行モード

に入る前にスタック深さ指示を記憶するように構成され、プロセッサ回路は、トランザクション実行モードで動作するときに、そのスタック深さ指示に関する呼出しスタック・データ構造体への修正の相対的スタッキング位置を判定するように、及び相対的スタッキング位置がスタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合にはその修正が非投機的であるという指示をその修正に関連して記憶するように、及び、相対的スタッキング位置がスタック深さ指示によって指示される位置に関して正のスタック成長方向にない場合にはその修正が投機的であるという指示をその修正に関連して記憶するように構成される。

【０００９】

本技法の発明者は、プロセッサがトランザクション実行モードで動作している間の呼出しスタックへの修正が投機的として印を付けられる構成は、これが、試行されたトランザクションが無事に完了しない場合にそのプロセッサがトランザクション実行モードに入ったときに有した構成に呼出しスタックの状態をプロセッサが「ロール・バック」することを確実に可能にすることになるという意味で、安全であるが、データ・プロセッサが呼出しスタックへの投機的修正に関して維持しなければならないデータ・セットの結果的なサイズが、不必要に大きくなり得ることに気が付いた。本技法は、この問題に対処し、データ・セットのサイズが低減され得ることを実現する。

【００１０】

それに応じて、呼出しスタックのその管理及び維持においてプロセッサによって使用されるスタック・ポインタによってポイントされるスタック深さの指示を記憶することができる、トランザクション前スタック・ポインタ記憶回路が提供され、このスタック深さ指示は、プロセッサがそのトランザクション実行モードに入る前にトランザクション前スタック・ポインタ記憶回路に記憶される。トランザクション実行モードに入ることは、たとえば、アクセスされようとする記憶されたデータ項目の識別がさらなるプロセッサ回路と共用されるものであることに基づいてそれを行う必要をプロセッサが認識することによって生じることが可能であり、又は、もう１つの実例では、プロセッサをトランザクション実行モードに切り替える明示的な「トランザクション実行モードの開始」命令の実行によって生じ得ることに留意されたい。後者の実例に関連して、「トランザクション実行モードの終了」命令もまた、トランザクション実行モードからプロセッサを切り替えるために後から使用することができる。

【００１１】

さらに、プロセッサは、次いで、それが呼出しスタックにその後に行う（トランザクション実行モードの間に）各修正について、トランザクション前スタック・ポイント・ストレージに記憶された指示に関してその修正がどこで生じることになるかを判定する。修正のこの相対的スタッキング位置は、具体的には、呼出しスタックのスタック成長方向に関して判定され、このスタック成長は、呼出しスタックの個々の構成、呼出しスタックがメモリ・アドレスを増やす方向に成長するか又は減らす方向に成長するかなど、に応じて「上方向」又は「下方向」と記述され得ることが、理解されよう。次いで、呼出しスタックへの修正を行うとき、プロセッサは、修正の相対的スタッキング位置が、トランザクション前スタック・ポインタ・ストレージの内容によって指示される位置に関して正のスタック成長方向にあるかを判定する。修正が、正のスタック成長方向にある相対的スタッキング位置に必要とされることが分かった場合、指示は、この修正が非投機的であることを指示し、呼出しスタックへの修正に関連して記憶される。逆に、修正の相対的スタッキング位置が、トランザクション前スタック・ポインタ・ストレージの内容によって指示される位置に関して正の成長方向にないと判定された場合、呼出しスタックへの修正に関連して記憶される指示は、その修正が投機的であることを指示するようにセットされる。修正に関連して記憶されるこれらの投機的／非投機的指示は、たとえば、ビットがセットされるかセットされないかを単に含み得ることが、理解されよう。

【００１２】

したがって、トランザクション実行モードにあるときに呼出しスタックへのいくつかの

修正の投機的特質を正確に管理するために、プロセッサが呼出しスタックに関連してその場合に維持しなければならないデータ・セットのサイズは、低減され得る。その理由は、それらの修正のサブセットは、投機的として実際にはラベル付けされないが、そのような修正が、プロセッサがトランザクション実行モードにない（及び、ある種の投機的実行を他の方法で実行しない）ときに行われることになる形で、非投機的としてラベル付けされるためである。これは、トランザクション実行モードで試みられるトランザクションが無事に完了しない場合、プロセッサが、呼出しスタックの状態をプロセッサがそのトランザクション実行モードに入ったときにそれが有した状態にロール・バックすることによってこれに対処する、具体的にはスタック・ポインタをデータ処理のそのステージでそれが有した値にリセットする場合には、トランザクション実行モード・エントリでのそのスタック深さに関して正のスタック成長方向にある位置でトランザクション実行モードにおいて行われた呼出しスタックへの修正は、それらが、トランザクション実行モードへのエントリのそのポイント以前に書かれた呼出しスタック内容を上書きする可能性を有さなかったので、単純に破棄することができる、という認識に基づいて可能である。

【００１３】

トランザクション前スタック・ポイント記憶回路は、様々な方法で構成することができるが、１つの実施例では、トランザクション前スタック・ポインタ記憶回路は、呼出しスタック・データ構造体のプロセッサ回路によって使用されるスタック・ポインタのコピーを記憶するように構成される。それにより、たとえば、現在のスタック・ポインタが、プロセッサのレジスタ内の装置によって記憶される場合、そのレジスタの内容は、トランザクション前スタック・ポインタ記憶回路にコピーされ得る。実際には、いくつかの実施例では、トランザクション前スタック・ポインタ記憶回路は、それ自体、プロセッサ回路にアクセス可能なレジスタを備える。そのようなレジスタは、さらに、プロセッサ回路によって実行されるデータ処理命令にアクセスできないように構成することができる。言い換えれば、このレジスタは、プログラマに可視であることができ、その内容は外部のエージェントによって修正され得ない。

【００１４】

呼出しスタックへの修正は、プッシュ及びポップを含み得る。いくつかの実施例では、相対的スタッキング位置が、スタック深さ指示によって指示される位置に関して正のスタック成長方向になく、修正が、呼出しスタック・データ構造体へのプッシュである場合、プッシュによって呼出しスタック・データ構造体にプッシュされるデータ項目は、投機的に書かれたものとして印を付けられる。したがって、プッシュ動作の結果を反映するために装置によって記憶される１つ又は複数のデータ項目は、投機的に書かれたものとしてそのように印を付けられる。逆に、相対的スタッキング位置が、スタック深さ指示によって指示される位置に関して正のスタック成長方向になく、修正が、呼出しスタック・データ構造体からのポップである場合、ポップによって呼出しスタック・データ構造体からポップされるデータ項目は、投機的に読み取られたものとして印を付けられる。したがって、ポップが呼出しスタックから現在取り除こうとしている呼出しスタック内容の追加を反映するために装置によって前に記憶された１つ又は複数のデータ項目は、それ／それらが投機的に読み取られたことを指示するために更新される。呼出しスタックが、それが属するプロセッサ回路にのみアクセス可能である（及び、さらなるプロセッサ回路にはアクセス可能でない）場合、そのとき、このデータ項目又はこれらのデータ項目の投機的読取りは、提供される任意の整合性機構に関してさらなるプロセッサ回路に影響を及ぼし得ないので、投機的に読み取られたというそのような印は、通常は、冗長であるということにさらに留意されたい。

【００１５】

プロセッサ回路は、様々な記憶場所に呼出しスタック・データ構造体を維持するように構成することができる。たとえば、プロセッサ回路は、さらなるプロセッサ回路と共用されるデータ項目が記憶されるメモリに呼出しスタック・データ構造体を維持することができる。しかし、いくつかの実施例では、装置はさらに、データ処理命令を実行するときに

メモリ内のプロセッサ回路によってアクセスされるデータ項目のローカル・コピーを記憶するように構成された記憶回路を備え、プロセッサ回路は、記憶回路に呼出しスタック・データ構造体を維持するように構成される。この記憶回路は、メモリ内のプロセッサ回路によってアクセスされるデータ項目のローカル・コピーを記憶するように構成される、すなわち、言い換えれば、通常は、キャッシュであり、大幅に小さい記憶容量を有するように通常は構成されることになる。本技法は、プロセッサ回路及び関連ローカル・キャッシュを備える装置に関連して、そのようなキャッシュが有し得る比較的限定された記憶容量により、特定の適用性を見出し得る。

【００１６】

本装置が、そのような記憶回路を備えるとき、記憶回路は、整合性制御ユニットを備えることができ、整合性制御ユニットは、非投機的として指示された記憶回路の内容が、メモリに取り出されることと、記憶回路の内容が取り出すために選択され、投機的として指示される場合にはエラー状態を生成することとを可能にするように構成される。整合性制御ユニットは、それにより、特に、記憶回路（キャッシュなど）は比較的限定された記憶容量を有し得るという事実を考慮して、どの内容が任意の所与のステージのデータ処理で記憶回路において保持されるべきかの判定を管理する。したがって、整合性制御ユニットは、投機的でないとして指示された記憶回路の内容が、有効な内容が既に記憶された所定の場所に記憶されることを必要とする（記憶回路構成により）メモリに取り出される（通常は、新しい内容が、記憶回路に記憶されることを必要とするとき）ことを可能にし得る。この既存の有効な内容が、非投機的として印を付けられた場合、次いで、整合性制御ユニットは、そのような取り出しを進められるようにし、一方で、それが投機的として印を付けられた場合、整合性制御ユニットは、エラー状態を生成する。トランザクション実行モードで動作するプロセッサに関連して、呼出しスタックへのさらなる投機的修正が記憶回路に記憶され得ることを保証することはできず、生成されたエラー状態は、次いで、トランザクションの試行の中止をもたらし得るので、これは、通常は、試行されたトランザクションをプロセッサ回路が継続するのを防ぐ開発を表すことになる。

【００１７】

しかし、トランザクション実行モードで試行されたトランザクションが、無事に完了した場合、対応する変更は、無事に行われた決定的変更を表すので、記憶回路の投機的内容は、非投機的に更新され得る。したがって、１つの実施例では、整合性制御ユニットは、プロセッサ回路がトランザクション実行モードを終了し、トランザクション実行モードで実行されるデータ処理命令がそれらのデータ処理動作を無事に完了したときに、投機的として指示された記憶回路の内容を非投機的として指示されるように変更するように構成される。

【００１８】

整合性制御ユニットは、投機的として指示された記憶回路の内容をさらなるプロセッサ回路から隠すように構成され得る。これは、装置において提供される任意の整合性機構が、記憶回路の任意の投機的内容が、そのさらなるプロセッサ回路に関して安全に処理されることをさらに確保する必要がないことを実現する。たとえば、整合性制御ユニットは、さらなるプロセッサ回路が、それが対応する内容にアクセスしていると指示する場合、そのような内容が記憶回路に現在記憶されていないという通知で応答するように構成することができる。

【００１９】

いくつかの実施例では、その装置はさらに、中止処理回路を備え、中止処理回路は、プロセッサ回路がトランザクション実行モードで動作しているときに、保留中のトランザクションが失敗したという指示に応答してロール・バック手続きを実行するように構成され、ロール・バック手続きは、投機的として指示された任意の修正を破棄するステップ、及び、記憶されたスタック深さ指示を使用して現在のスタック・ポインタをリセットするステップを含む。したがって、プロセッサ回路が、トランザクション実行モードを無事に終了せず、試行されたトランザクションが、再び試行されなければならない場合、プロセッ

サ回路によって実行されるデータ処理は、トランザクション実行モードが開始されたポイントまでロール・バック（リセット）することができ、したがって、呼出しスタックを参照するためにプロセッサ回路によって使用される現在のスタック・ポインタは、記憶されたスタック深さ指示の値、すなわち、トランザクション実行モードに入ったときにスタック・ポインタが有した値、にリセットされる。

【００２０】

もう１つの例示的構成によれば、データ処理命令を実行するための手段と、データ処理命令を実行するための手段が、実行されるデータ処理命令がデータ処理命令を実行するためのさらなる手段と共用される記憶されたデータ項目にアクセスするときにトランザクション実行モードで動作するように構成される、実行されるデータ処理命令に依存する呼出しスタック・データ構造体を維持するための手段と、スタック深さ指示が、データ処理命令を実行するための手段がトランザクション実行モードに入る前に記憶される、呼出しスタック・データ構造体のスタック深さ指示を記憶するための手段と、トランザクション実行モードで動作するときにそのスタック深さ指示に関する呼出しスタック・データ構造体への修正の相対的スタッキング位置を判定するための手段と、相対的スタッキング位置が、スタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合には、指示は、その修正が非投機的であると示し、相対的スタッキング位置が、スタック深さ指示によって指示される位置に関して正のスタック成長方向にない場合には、指示は、その修正が投機的であると示す、修正に関連して指示を記憶するための手段を備える装置が提供される。

【００２１】

もう１つの例示的構成によれば、データ処理命令を実行するステップと、実行されるデータ処理命令に依存する呼出しスタック・データ構造体を維持するステップと、実行されるデータ処理命令がさらなるプロセッサ・デバイスと共用される記憶されたデータ項目へのアクセスを求めるときにトランザクション実行モードに入るステップと、スタック深さ指示を記憶するステップであり、前記スタック深さ指示が、前記トランザクション実行モードに入る前に記憶される、ステップと、トランザクション実行モードで動作するときに、スタック深さインジケータに関する呼出しスタック・データ構造体への修正の相対的スタッキング位置を判定するステップと、相対的スタッキング位置が、スタック深さ指示によって指示される位置に関して正のスタック成長方向にある場合には、その修正が非投機的であるという指示をその修正に関連して記憶し、相対的スタッキング位置が、スタック深さ指示によって指示される位置に関して正のスタック成長方向にない場合には、その修正が投機的であるという指示をその修正に関連して記憶するステップとを含む、プロセッサ・デバイスにおけるデータ処理の方法が、提供される。

【００２２】

図１は、１つの実施例による２つのデータ処理装置１２及び１４を備えるデータ処理システム１０を概略的に示す。各データ処理装置１２、１４は、プロセッサ・コア１６、１８をそれぞれ備え、したがって、データ処理システム１０はマルチコア・システムであることが認識されよう。各データ処理装置１２、１４は、それを介して各データ処理装置１２、１４がメモリ２２（数ある目標の中で）にアクセスすることができる同じシステム・バス２０にアクセスする。処理装置１２及び１４の両方は、同じメモリ２２にアクセスすることができるが、知られているメモリ割当て及び制御技法に従ってそれらのデータ処理装置のうちの一方のみがアクセスすることができ、他方のデータ処理装置はアクセスを阻止される（又は、少なくとも恐らくは修正を阻止される）メモリ２２の領域が存在し得る。しかし、図示するように、メモリ２２は、データ処理装置１２及び１４の両方がアクセスすることができる共用データ構造体２４が記憶された少なくとも１つの領域を含む。この共用データ構造体２４は、たとえば、データベースでもよいが、本技法はそのような実例に如何なる点でも限定されない。共用データ構造体２４は、遥かに単純なデータ構造体、実際には原則として両方のデータ処理装置にアクセス可能な単一のデータ項目のみさえ、も表し得る。

【００２３】
データ処理装置１２及び１４の両方がメモリ２２内の同じ共用データ構造体２４にアクセスすることができるという事実は、両方のデータ処理装置、特にそれぞれのプロセッサ・コア、が同じ共用データ構造体２４（及び、前述の共用データベース、たとえば、その共用データベース内の同じデータ項目、との関連で）にアクセスしようとするときに競合が生じないことを確保するために、データ処理装置が動作する方式にある種の制約を課す。そのような競合を避けるために、各プロセッサ・コア１６、１８は、共用データ構造体２４へのアクセスが求められるときにトランザクション実行モードに入るように構成される。たとえば各プロセッサ・コアは、アクセスされるデータ項目が、そこへのアクセスがページ・テーブル内の情報を参照することによって、キャッシュ・コヒーレンス・プロトコル及びキャッシュに記憶された追跡構造体を参照することによって、又は他の知られている追跡及び信号伝達構造体を参照することによってもう１つのプロセッサ・コアと共用されるものであることを認識し得る。別法として又は加えて、明示的「トランザクション実行モードの開始」及び「トランザクション実行モードの終了」命令が、トランザクション実行モード及びモード外にプロセッサを切り替えるために使用され得る。そのようなトランザクション実行モードは、当業者には知られているが、本質的に、トランザクション実行モードで動作するときに、プロセッサ・コアは、メモリ２２により保守的な方式でアクセスすることを余儀なくされ、その方式によれば、共用データ構造体２４（又はその部分）へのアクセスが行われるとき、プロセッサ・コアは、データ構造体又はデータ項目のロックを先ず取得しようとし、次いで無事に取得しなければならず、それが取得された後は、それが一時的に属するプロセッサが、ロックが放棄されるまで、そのデータ構造体又はデータ項目への排他的アクセスを有するように、このロックは設定される。これは、様々なタイプの知られているデータ・ハザードが、次いで、２つ以上のプロセッサが同じデータ構造体又はデータ項目に一斉にアクセスする結果として生じ得ないことを確保し、全体としてのデータ処理システム１０の異なる処理結果が、プロセッサ１６及び１８がその共用データ構造体にアクセスする（及び、特に、修正する）特定の順番の結果として生じ得る。

【００２４】
本明細書に記載の技法は、特に、そのトランザクション実行モードで動作するプロセッサに関連して識別される問題、及び、そのプロセッサが次いで呼出しスタックを維持する方式に関し、これは、関数呼出し及び関数復帰が、プロセッサが実行するデータ・プログラム命令のシーケンスにおいて遭遇されるときに、プロセッサが、正しいメモリ・アドレスへと正しくナビゲートする（及び、対応するプロセッサ状態情報を更新する）ことができるように、復帰アドレス（及び、恐らくは、さらなる関連プロセッサ状態情報）を記憶するためにデータ処理デバイスによって使用される、知られているデータ構造体である。プロセッサ・コア１６及び１８が本技法を実装するために構築される方式の１つの特徴は、各々が、それが維持している呼出しスタックを参照するためにそれが使用するスタック・ポインタ（ＳＰ：stack pointer）の、コピーがその中に記憶され得る追加のレジスタを、それぞれにその方式が行われる内部レジスタ２６及び２８のセット内に、備えるということである。このスタック・ポインタ・コピーが、これを目的として提供される専用レジスタ３０、３２に記憶されるときについてさらに詳しくは、以下の図を参照して、より詳細に以下に説明される。各データ処理装置１２、１４はさらに、それぞれ、レベル１（Ｌ１）キャッシュ３４及び３６を備える。プロセッサ・コアがメモリ２２の内容にアクセスしようとするときに、システム・バス２０を介する及びメモリ階層（レベル２（Ｌ２）キャッシュなど　－　図示せず）のさらなるメンバを恐らくは介するメモリ２２に記憶されたデータ項目へのアクセスに関連する待機時間が大幅に回避され得るように、これらのＬ１キャッシュの各々は、その関連プロセッサ・コアの使用のためにメモリ２２の内容の小さいサブセットをキャッシュに格納するように構成される。各Ｌ１キャッシュ３４、３６は、キャッシュの全体的制御を維持するそれぞれの制御ユニット３８、４０を備え、特に、図示された実施例では、本技法をサポートするために以下の図を参照してより詳細に

説明される方式で構成される。
【００２５】
図２Ａは、図１に示されるデータ処理システム１０内のプロセッサ・コア１６、１８のうちの１つによって実行することができるデータ処理命令のシーケンスの構造を示すいくつかのＣライクなコードを示す。図２Ａに示されるコードは、関数呼出しの比較的単純なセットであり、その中で関数呼出しのうちのいくつかは、互いの中に組み込まれ、たとえば、関数ｆ２（）は関数ｆ１（）の中に組み込まれ、一方、関数ｆ４（）は関数ｆ３（）の中に組み込まれ、関数ｆ３（）自体は関数ｆ１（）の中に組み込まれる。本技法の特別な意義は、そのポイントでこれらの命令を実行するプロセッサが、メモリ内の記憶されたデータ構造体又は項目へのアクセスを求めるために、そのトランザクション実行モードに入ることになる、動作ｔｂｅｇｉｎ（）を含む関数ｆ２（）の定義である。したがって、トランザクション実行モードに入ること及びそれに続くトランザクション動作は、ライブラリ呼出し内から抽出されることに、特に留意されたい。
【００２６】
図２Ａに示される定義されたコードを介して正しく進むために、プロセッサが、プログラマによって定義されたようなプログラムされたシーケンスの命令に正しく従うことができるように、これらの命令を実行するプロセッサは、それがプログラム命令のシーケンスにおいて遭遇する関数呼出しの復帰アドレスがそこにプッシュされ、その後に関数が完了したときにそこからポップされる、呼出しスタックを維持する。
【００２７】
図２Ｂは、プロセッサが図２Ａに示されるデータ処理命令を実行するときの呼出しスタックへのプッシュ及び呼出しスタックからのポップを説明する。したがって、プロセッサが、関数ｆ１（）内の関数ｆ２（）に遭遇するとき、復帰アドレスｒｅｔ＿ｆ２が、呼出しスタックにプッシュされ、関数ｆ２（）の最後に、このアドレス（及び、恐らくは、いくつかの追加の構成データ）が、スタックからポップされる。関数ｆ２（）の最後のこのポップは、以下でさらに詳しく説明されるように、プロセッサがトランザクション実行モードにある間にこのポップが生じるという事実を示すｔｘ＿ｐｏｐとして図２Ｂにおいて示されることに留意されたい。同様に、プロセッサが、関数ｆ１（）内のさらに後に関数ｆ３（）に遭遇するとき、復帰アドレスｒｅｔ＿ｆ３が、呼出しスタックにプッシュされる（ｔｘ＿ｐｕｓｈ）。しかし、関数ｆ３（）内のこれに続いて、関数ｆ４（）に遭遇するとき、復帰アドレスｒｅｔ＿ｆ４はまた呼出しスタックにプッシュされるが、以下でさらに詳しく説明されるように、これは、図２Ｂにおいてｎｏｎ＿ｔｘ＿ｐｕｓｈとしてラベル付され、この復帰アドレスｒｅｔ＿ｆ４が処理される特定の方式は、復帰アドレスｒｅｔ＿ｆ３が処理される方式とは異なることに留意されたい。
【００２８】
図２Ｃは、図２Ａに示される命令を実行するプロセッサが、それがそれらの命令を介して進むときに維持する、呼出しスタックの内容及び発展を概略的に示す。図２Ｃに示すように、呼出しスタックは、ページの下方への正のスタック成長方向を有するが、これは、プロセッサが呼出しスタック・データ構造体を維持するために使用するメモリ・アドレスの特定の順番付けとして解釈されるべきではない　－　これらは、個々の実装形態に応じて、正のスタック成長とともに増加するメモリ・アドレスに又は減少するメモリ・アドレスに同様に十分に進み得る、ことに留意されたい。本論考の特別な意義は、図２Ｃの右に示され、図２Ａに示されるようなコード内の特定のポイントに対応する、ラベリング（１）（２）、（３）及び（４）である。これらのポイントは、コードのその特定のステージが到達される時点で呼出しスタックが有するスタック深さを示す。
【００２９】
プロセッサは、専用レジスタがそのために提供される、スタック・ポインタ（ＳＰ）を用いて現在のスタック深さの知識を維持する。加えて、プロセッサは、それがトランザクション実行モードに入るポイントで、すなわち、図２Ａ及び２Ｂに示される図解される実例では、それがｔｂｅｇｉｎ（）関数に遭遇するときに、ＳＰのコピーを記憶するように

構成される。図１に示される実施例では、スタック・ポインタのこのコピーは、図２Ａに示される命令を実行しているそれぞれのプロセッサ・コア１６、１８によって専用レジスタ３０、３２に記憶される。その後、プロセッサが、まだそのトランザクション実行モードにある間に、プロセッサは、それが呼出しスタックを修正するときに、スタック・ポインタのこの記憶されたコピーを参照し、呼出しスタックへの修正が、記憶されたコピーによって指示されるスタック深さ位置に関して正のスタック成長方向にあるかどうかを判定する。それに応じて、図２Ｃによって示される図解を参照すると、ＳＰ＠ｔｂｅｇｉｎとして指示される深さによって指示される深さを下回るスタック深さでの呼出しスタックへの修正は、記憶されたＳＰ値に対して正のスタック成長方向にあると判定されることになり、一方、このポイントを上回るスタック深さ位置での呼出しスタックへの修正は、正のスタック成長方向にない。図２Ａに示される命令を実行するプロセッサは、関連Ｌ１キャッシュを提供され、その制御ユニットは、呼出しスタック・データ構造体の部分を形成する、及び、その呼出しスタックへの修正（プロセッサがそのトランザクション実行モードにある間）がＳＰコピー・レジスタに記憶されたスタック・ポインタ値ＳＰ＠ｔｂｅｇｉｎに関して正のスタック成長方向にあるスタック深さで生じたかどうかに応じて非投機的又は投機的のいずれかとしてＬ１キャッシュに記憶される、データ項目にラベルを付けるように構成される。修正が、そのＳＰコピー値に対して正のスタック成長方向で行われた場合、キャッシュ制御ユニットは、Ｌ１キャッシュ内の対応するエントリに非投機的としてラベルを付け、一方、修正が、そのＳＰコピー値に対して正のスタック成長方向にない場合、対応するデータ項目は、投機的としてラベル付けされる。

【００３０】

これは、図２Ｂにおいて、「ｔｘ＿ｚｏｎｅ」及び「ｎｏｎ＿ｔｘ＿ｚｏｎｅ」にプロセッサのトランザクション実行モードに対応する図の部分を分ける破線による図の垂直の分割によって示される。そのように、「ｔｘ＿ｚｏｎｅ」で行われる呼出しスタックへの修正は、投機的としてラベル付けされ、一方、「ｎｏｎ＿ｔｘ＿ｚｏｎｅ」における呼出しスタックへの修正は、非投機的としてラベル付けされる。これは、図２Ｃを参照してさらに理解することができ、そこで、関数ｆ２（）の復帰アドレスｒｅｔ＿ｆ２は、関数ｆ３（）の復帰アドレスｒｅｔ＿ｆ３によって原則として上書きされることになることが理解されよう。しかし、プロセッサがそのトランザクション実行モードで動作するときに実行しようとするトランザクション実行は、無事に完了しないことがあるので、もう１度そのトランザクション実行を試行するために、プロセッサは、それがｔｂｅｇｉｎ（）命令に遭遇したときにそれが有したその実行状態にロール・バックする必要があり得ることを、本技法は認識する。プロセッサがそのトランザクション実行モードにある間に試行するトランザクションは、いくつかの理由で、たとえば、動的にトリガされる、たとえば別のプロセッサが所望のデータ項目に現在アクセスしているため、又は手動で、たとえば、図２Ａにおいて関数ｆ１（）内に示されるｔａｂｏｒｔ（）呼出しにより、無事に完了しないことがあることに留意されたい。ｆ３（）復帰アドレスの追加を含んだ呼出しスタックの修正が、Ｌ１キャッシュ内の対応するエントリを決定的に修正することを可能にされた場合、そのとき、この修正は、永続的であったことになり、対応するデータ項目はまた、その間にメモリに取り出された可能性があり、プロセッサは、新しい試行（トランザクション実行モードで）をトリガして必要とされるトランザクションを実行するためのｔｂｅｇｉｎ（）関数を再実行するために正しいステータスに呼出しスタックの状態をロール・バックする方法を有さないことになる。これは、関数ｆ２（）の復帰アドレスは関数ｆ３（）の復帰アドレスによって書き換えられるので、図２Ｃの図解におけるポイントＳＰ＠ｔｂｅｇｉｎ（１）より上の呼出しスタックの内容は、その場合、関数ｆ３（）の復帰アドレスを含むが、関数ｆ２（）の復帰アドレスを含まないことになるためである。したがって、プロセッサは関数ｆ２（）から後で正しく復帰することが不可能となり、これは必要な復帰アドレスがもはや使用可能でなくなっているからである。

【００３１】

本技法は、プロセッサがトランザクション実行モードにあるときに行われる呼出しスタ

ック・データ構造体への修正が、それらが投機的としてラベル付けされるべきか又は非投機的としてラベル付けされるべきかを判定するために、記憶された値ＳＰ@ｔｂｅｇｉｎと比較されることを実現することによって、この問題に対処する。しかし、これは、「ｔｘ＿ｚｏｎｅ」における呼出しスタックへの修正の投機的特質、すなわち図２Ｃに示されるハッチングを掛けた上部の領域、が正しく処理されるが、非ｔｘゾーン内の呼出しスタック修正の処理、すなわち図２Ｃの下部に示される明るい網掛けの領域、にも関連することを確保するための唯一の技法ではないことを理解されたい。この後者の態様は、これはまた、トランザクション実行モードの完了の不成功の場合に、すなわち試行されたトランザクションが無事に完了しないときに、プロセッサがｔｂｅｇｉｎ（）呼出しのポイントで必要とされる状態に正しくロール・バックすることも可能であることを確保することになるので、プロセッサがそのトランザクション実行モードにある間に呼出しスタックへのすべての修正を投機的として単にラベル付けさせることが可能であることになる（そして、実際にこれは先行技術で取られる手法である）ため、特別な意義をここで有する。しかし、データ処理命令の比較的単純なセットのみが図２Ａの実例では示され、図２Ｃに示される対応する呼出しスタックの比較的単純な及び限定された進化を結果としてもたらすが、プロセッサがそのトランザクション実行モードにある間に実行することができるデータ処理命令及び行われる実際のさらなる関数呼出しの数の制限は原則として存在しない。呼出しスタックへのこれらの対応する修正のすべてが、投機的として単純にラベル付けされる場合、そのとき、呼出しスタックへの投機的（トランザクション実行モードにある）修正及び非投機的（トランザクション実行モードにはない）修正の両方のコピーは、維持される必要があることになり、トランザクション実行モードに関連するかなり大きいライトセットサイズを潜在的にもたらすので、呼出しスタックの管理は、かなり手間のかかるものになり得る。これは、データ記憶容量が限定される場合、任意のデータ処理システムにおいて重大であり得るが、記憶容量が通常は、もちろん、メイン・メモリにおいて使用可能なものよりも遥かに限定される、その関連Ｌ１キャッシュ内で第１のインスタンスにおいて呼出しスタック・データ構造体がプロセッサによって維持される図１に示されるものなどの実施例において特別な意義を有し得る。したがって、本技法によれば、プロセッサがそのトランザクション実行モードにあるにもかかわらず、プロセッサ回路が維持する呼出しスタック・データ構造体への修正のいくつかは、非投機的としてラベル付けされる。これは、キャッシュにおけるこれらのエントリが、試行されたトランザクションが無事に完了しないときにその実行状態を正しくロール・バックするためのプロセッサの能力に影響を及ぼすことなくメイン・メモリに無事に取り戻され得るという結果を有する。さらに、図１に示される実施例における一方のプロセッサ・コアは、メモリ２２に記憶されたコミットされたデータ項目によって表されるように他方のプロセッサ・コアの呼出しスタックを見ることを許され得るが、そのプロセッサ・コアに関連するＬ１キャッシュの投機的ラベル付けされた内容を見ることが許されない場合、そのとき、データ処理システム１０内で実行されるトランザクションの正しい原子性（atomicity）が維持される。トランザクション実行が無事完了したとき、呼出しスタック全体は、非投機的（非トランザクション）としてラベル付けされるように構成され、これらのキャッシュされたデータ項目は、次いで、メモリ２２に自由に取り出され得ることに留意されたい。しかし、前述のように、動的に又は手動でトリガされる、たとえば、ｆ４（）内のｔａｂｏｒｔ（）呼出しによる、中止の場合、トランザクションにおいて投機的に修正されたすべてのアーキテクチャの状態及びメモリ（それが進行した範囲で）は、ｔｂｅｇｉｎ（）への呼出しで存在した状態にロール・バックされ、そのようなシナリオでは、スタックを指し示すレジスタ２６、２８のセット内の任意のレジスタは、それらの以前の値にロール・バックすることになり、スタック・ポインタは、図２Ａ及び２Ｃに示されるポイント（１）で取られるスナップショットの値にリセットされることになる。

【００３２】

図３は、図２Ａに示されるデータ処理命令のシーケンスを実行するプロセッサ回路によって維持される呼出しスタックの進化のより明示的な表現を示す。ステージＡは、関数ｆ

１（）に遭遇するポイントでの呼出しスタックを表し、スタック・ポインタＳＰは、そのとき、復帰アドレスｒｅｔ＿ｆ１が現在のスタック深さを表すことを指示する。その後、ステージＢで、ｆ２（）関数に遭遇するとき、復帰アドレスｒｅｔ＿ｆ２は、スタックにプッシュされ、そして、スタック・ポインタは、現在のスタック深さがこの復帰アドレスの位置に対応することを指示する。ステージＣでｔｂｅｇｉｎ（）呼出しに遭遇するとき、スタック・ポインタは、いくらかの追加の構成情報が、ｔｂｅｇｉｎ（）命令自体に遭遇する前に、呼出しスタックに追加されたことにより、正のスタック成長方向にわずかに移動していることに留意されたい。これは、その場合、トランザクション実行モードに入ることに対応するそのＳＰコピー・レジスタ（３０又は３２）に装置が記憶するスタック深さ（ＳＰ値）である。その後、プロセッサは（主に、そのＬ１キャッシュの制御ユニットへのこのタスクの委任によって）、これらが投機的としてラベル付けされるべきか非投機的としてラベル付けされるべきかを判定するために、それが呼出しスタックに行うさらなる修正をそのＳＰコピー・レジスタに記憶されたスタック・ポインタの値と比較する。したがって、図３のステージＤで、ｒｅｔ＿ｆ２命令に遭遇するとき、関数ｆ１（）の復帰アドレスｒｅｔ＿ｆ１を指示するために現在のスタック・ポインタを返すための呼出しスタック内容のポップは、投機的としてラベル付けされる。しかし、図４Ａ～４Ｃを参照して以下でさらに詳しく論じるように、これは、プロセッサについての投機的読取りを表すのみなので、呼出しスタックへのポップ修正を投機的としてラベル付けする必要はないことがあり、そして、別のプロセッサが、図示された命令を実行するプロセッサ・コアに関連するキャッシュの内容にアクセスすることができないこともまた確保される限り、キャッシュ内のこのエントリが投機的に読み取られたもの（及びまた投機的に書かれたものではない）として単にラベル付けされる限り、それは、必要とされるトランザクションの原子性を脅かすことなくメイン・メモリに実際に安全に取り出され得る。投機的書込みは、ｔｂｅｇｉｎ（）で取られる記憶されたＳＰスナップショットに関して正のスタック成長方向にない呼出しスタック内の相対的スタッキング位置において行われるので、投機的書込みは、実際には、ｆ３（）関数に遭遇する図３のステージＥで行われる。したがって、復帰アドレスｒｅｔ＿ｆ３を保持するキャッシュにおいて行われるエントリは、投機的としてラベル付けされる。これに反して、ステージＦでは、ｆ４（）への呼出しに遭遇するとき、ｆ４（）復帰アドレスｒｅｔ＿ｆ４を呼出しスタックにプッシュするために必要とされる修正は、ＳＰスナップショットに関して正のスタック成長方向にあるものとして識別され、それに応じて、この復帰アドレス値を保持するキャッシュにおいて行われるエントリは、非投機的としてラベル付けされる。その後、ステージＧで、ｔａｂｏｒｔ（）呼出しに遭遇するとき、キャッシュの任意の投機的ラベル付けされた内容が、破棄され、非投機的にラベル付けされた内容のみが、保持され（すなわち、実際には、キャッシュ内の任意の投機的にラベル付けされたキャッシュ・ラインは、無効なものとして印を付けられる）、それにより、呼出しスタック・データ構造体は、潜在的にはＬ１キャッシュに記憶された内容の組合せから、そしてさらに、何らかのキャッシュ内容がその間に取り出され得る先の、メイン・メモリ２２などからの、メモリ階層に、ｔｂｅｇｉｎ（）呼出しに遭遇したときのステージＣでのそれの状態に再構築される。

【００３３】

図４は、１つの実施例におけるキャッシュ５０を概略的に示し、図１に示されるＬ１キャッシュ３４又は３６はそれに従って構成することができる。Ｌ１キャッシュ５０は、制御ユニット５２と、そこにデータが記憶され得るいくつかのキャッシュ・ライン５４とを備える。制御ユニット５２は、キャッシュ５０の全体的制御を維持し、キャッシュ・ライン５４への更新を管理する。それはさらに、キャッシュ５０が属するデータ処理システム内にある残りのメモリ・システムに関して正しい整合性制御をさらに維持する。したがって、キャッシュ５０は、プロセッサと残りのメモリ階層との間に位置付けられ、このメモリ階層は、潜在的に、図１に示されるものなどのメイン・メモリ２２のみを備えるが、さらなるキャッシュ・レベルもまた備え得る。キャッシュ・ライン５４は、このデータ・ラインの内容において「修正済み」（ＭＯ：modified）としてラベル付けされた復帰アドレ

ｓｒｅｔ＿ｆ２の記憶に対応する例示的エントリとともに示され、このデータがプロセッサによって修正済みであり、残りのメモリ階層において、特にこのデータ・ラインが取り出される場合にはメイン・メモリ２２において、最終的に更新されるべきであることを指示する。ｒｅｔ＿ｆ２のこのデータ・ラインは、これはプロセッサがそのトランザクション実行モードに入る前に起こった呼出しスタックへの修正を表すため、投機的に読み取られた（ＳＲ：speculatively read）又は投機的に書かれた（ＳＷ：speculatively written）としてラベル付けされないことに留意されたい。

【００３４】

図４Ｂは、復帰アドレスｒｅｔ＿ｆ２によって投入された後に、プロセッサが関数呼出しｆ３（）に遭遇し、それにより、復帰アドレスｒｅｔ＿ｆ３を呼出しスタックにプッシュする、図４Ａに示される例示的キャッシュ・ライン・エントリのその後の進化を示す。偶然に、この図示された実例では、復帰アドレスｒｅｔ＿ｆ２及びｒｅｔ＿ｆ３のメモリ・アドレスは、それらが両方とも同じキャッシュ・ライン内のキャッシュ５０によって記憶されるのに十分な程類似している。したがって、ｒｅｔ＿ｆ３が、プロセッサによって呼出しスタックにプッシュされるとき、キャッシュ５０のキャッシュ制御ユニット５２は、この復帰アドレスｒｅｔ＿ｆ３にそのキャッシュ・ラインの現在の内容（すなわち、復帰アドレスｒｅｔ＿ｆ２）を取り出させる。内容ｒｅｔ＿ｆ２を有する、このキャッシュ・ラインは、内容ｒｅｔ＿ｆ２とともに、投機的に書かれた（ＳＷ）ものとしてラベル付けされないので、それは、メモリ階層に、たとえば、先ずＬ２キャッシュに、そして潜在的には次いでさらにメイン・メモリ２２に、取り出され得る。プロセッサは現在トランザクション実行モードにあり、この修正は、プロセッサがｔｂｅｇｉｎ（）呼出しに遭遇した時点で取られたスタック・ポインタＳＰのスナップショットに関して正のスタック成長方向で起こらなかったので、復帰アドレスｒｅｔ＿ｆ３は、次いで、このキャッシュ・ラインを投入し、「ＭＯ」ステータス「修正済み」で同様に印を付けられるが、さらに投機的に書かれた（ＳＷ）としてラベル付けされる。その後、トランザクションが中止する（命令ｔａｂｏｒｔ（）に遭遇するプロセッサによって）とき、このキャッシュ・ラインは、投機的に書かれたとしてラベル付けされるので、その内容は無効にされる（ステータス「Ｉ」）。

【００３５】

図４Ｂに示されるキャッシュ・ライン内容の進化を図４Ｃに示されるそれと比較すると、図４Ｃは、ステージＤが図３において到達されるキャッシュ動作を示し、そこでは、関数ｆ２（）の復帰アドレスｒｅｔ＿ｆ２が呼出しスタックからポップされ、さらに、呼出しスタックへのこの修正は、ｔｂｅｇｉｎ（）でのＳＰのスナップショットに関して正のスタック成長方向において起こらず、したがって（プロセッサはトランザクション実行モードにあるので）、これは、投機的修正として処理される。したがって、このキャッシュ・ラインの内容がプロセッサに返されるとき、キャッシュ・ラインは、次いで、投機的に読み取られた（ＳＲ）としてラベル付けされる。しかし、前述のように、投機的に読み取られたとしてのキャッシュ内のキャッシュ・ラインのラベル付けは、キャッシュ５０の制御ユニット５２が、キャッシュの内容（又は少なくとも修正済み内容）がデータ処理システム内の別のプロセッサによって読み取ることができないことを確保する限り、原則として必要とされない。これは、たとえば、制御ユニット５２が、必要とされるシステム全体のメモリ整合性を強化する際に、そのような内容がキャッシュ５０内の任意の有効な状態において現在存在することを否定することによって、このメモリ・アドレスに関連する整合性プロトコルの部分として受信される任意の問合せに単純に応答する、構成によって強化され得る。

【００３６】

図５は、プロセッサ・コア６０と、図１において示される対応する構成要素がそれに従って構成され得る構成を示すＬ１キャッシュ６２とを概略的に示す。Ｌ１キャッシュ６２のキャッシュ制御６４は、プロセッサ・コア６０から様々な情報を受信するように及びプロセッサ・コアに何らかの情報を返すように構成される。具体的には、プロセッサ・コア

が現在そのトランザクション実行モード（ＴＥＭ：transactional execution mode）にあるかどうかに応じて、キャッシュ制御６４がＬ１キャッシュ６２の内容の更新を適切に管理することができるように、キャッシュ制御６４は、それが現在動作している処理モードのプロセッサ・コアから指示を受信する。これは、連続的信号である必要はないが、プロセッサ・コア６０は、それが実行モードを変更するときにキャッシュ制御ユニット６４に単純に指示することができる。プロセッサ・コア６０はまた、キャッシュ・コントローラ６０にトランザクション中止を指示すること（たとえば、ｔａｂｏｒｔ（）呼出しがプロセッサ・コアによって遭遇されるときに）、及び、たとえば、投機的に書かれた（ＳＷ）として印を付けられたＬ１キャッシュ６２の内容が、取り出すために選択されるときに（それの対応するメモリ・アドレスに基づいて）、キャッシュ制御ユニット６４から中止指示を受信することの両方を行うように構成された、中止処理ユニット６６を備える。残りのメモリ階層へのこの内容の取り出しは、データ処理システムの別の部分によって読み取られるようにそれを公開する可能性があり、これは、試行されたトランザクションの原子性を潜在的に脅かし得るので、これは、試行されたトランザクションが中止されるべき状況である。別法として、この問題はまた、投機的に書かれたものとしてのこのキャッシュ・エントリのラベル付けが、それがメモリ階層の別の部分に取り出されるときに維持されることを確保することによって、対処され得るが、これは、投機的に書かれたものとして印を付けられたキャッシュ・エントリの取り出しが試行されるときにトランザクションを中止する比較的単純な機構によって回避することができる構成のさらなる複雑さを招き得る。トランザクション中止の原因が何であれ、中止処理ユニット（ここでは専用回路として具体化されるが、プロセッサ・コア６０が実行するデータ処理命令内にプログラムされた専用中止処理ルーチンでもよい）は、次いで、図３のステージＣでの、すなわちｔｂｅｇｉｎ（）呼出しに遭遇したときの、それらの状態へのレジスタ６８の内容のロール・バックをもたらす。具体的には、本技法に関連して、これは、ｔｂｅｇｉｎ（）が最初に遭遇され、ＳＰコピー・レジスタ７２に記憶されたときに取られたスタック・ポインタのスナップショットで現在のスタック・ポインタ・レジスタ７０の内容を置き換える中止処理ユニット６６を含む。

【００３７】

図６は、１つの実施例の方法において取られるステップのシーケンスを概略的に示す。その流れは、ステップ１００で開始すると考えることができ、そこで、プロセッサは、それの命令シーケンスを実行する。ステップ１０２で、トランザクション開始命令（たとえば、ｔｂｅｇｉｎ（））に遭遇するかが、判定される。遭遇しない場合、流れは、単純に、ステップ１００に折り返して戻る。そのようなトランザクション開始命令に遭遇した後は、次いで、流れは、ステップ１０４に進み、そこで、スタック・ポインタの現在の値が、専用スタック・ポインタ・コピー・レジスタに記憶され、その後、ステップ１０６で、プロセッサは、そのトランザクション実行モードに入る。ステップ１０８に続いて、プロセッサは、メモリ内の共用データ項目又は構造体へのそれが求めるトランザクション・アクセスの開始を含めて、その命令シーケンス実行を継続する。ステップ１１０で、データ処理命令のシーケンスの継続の部分として呼出しスタックへの修正が必要とされるかが判定される。必要とされない場合、次いで、流れは、単純に、ステップ１０８に折り返して戻る。しかし、そのような修正が必要とされるとき、次いで、ステップ１１２で、呼出しスタックへのこの修正が、専用ＳＰコピー・レジスタに記憶されたスタック・ポインタ・コピーによって指示されるスタック深さに対して正の成長方向で行われるべきかが判定される。この修正が、この正の成長方向にある場合、流れは、ステップ１１４に進み、そこで、呼出しスタック・データ構造体への修正が行われ、キャッシュ内の１つ又は複数の任意の対応するエントリを非投機的とする。別法として、呼出しスタック・データ構造体に必要とされる修正が、ステップ１１２で、スタック・ポインタ・コピーによって指示されるスタック深さに対して正の成長方向にないと判定された場合、流れはステップ１１６に進み、そこで、呼出しスタック・データ構造体への修正が行われ、そして、キャッシュ内の１つ又は複数の任意の対応するエントリが、投機的として印を付けられる。ステップ１

１４又はステップ１１６のいずれかの後、流れはステップ１１８に進み、そこで、試行されたトランザクションが無事に完了したかが判定される。無事完了しなかった限り、次いで、流れは、トランザクション実行モードでの継続する実行を進めるためにステップ１０８に戻る。しかし、試行されたトランザクションが、無事に完了した後は、次いで、流れはステップ１２０に進み、そこで、プロセッサは、そのトランザクション実行モードを終了し、そして、ステップ１１２で、任意の投機的にラベル付けされたキャッシュ内容が、更新されて、非投機的として印を付けられる。次いで、流れは、ステップ１００に戻る。

【００３８】

図６に示されるステップのシーケンスは、試行されたトランザクションが中止されるときに取られるステップを含まず、そのようなステップが、図７に示され、そこでは、プロセッサがそのトランザクション実行モードにある間に、これらのステップは、図６に示されるそれらに並行して実行すると考えられ得ることに留意されたい。そのトランザクション実行モードにあるプロセッサによる命令の実行は、ステップ１３０によって表される。ステップ１３２で、トランザクション中止が生じたかが継続的にチェックされる。それが生じない限り、流れは、単純に、ステップ１３０に折り返して戻る。しかし、トランザクションが中止したとき、次いで、流れはステップ１３４に進み、そこで、投機的として印を付けられた任意のキャッシュ・コンタクトは、無効にされる。さらに、ステップ１３６で、現在のスタック・ポインタが、プロセッサがそのトランザクション実行モードに入った時点で取られたスタックされたポインタの記憶されたコピーから更新され（すなたち、それにロール・バックされ）、そして、ステップ１３８で、プロセッサの状態をそれがそのトランザクション実行モードに入ったときにそれが有したものに戻すために必要とされるレジスタの任意の他の内容（現在のスタック・ポインタ・レジスタ以外）もまた、復元される。流れは、次いで、プロセッサがトランザクション実行モードに留まり、トランザクションをもう１度試行するためのステップ１３０に戻る。

【００３９】

特定の実施例が、本明細書において説明されたが、本発明は、それに限定されず、それへの多数の修正及び追加が、本発明の範囲内で行われ得ることが、理解されよう。たとえば、以下の従属請求項の特徴の様々な組合せが、本発明の範囲を逸脱することなしに独立請求項の特徴とともに行われ得る。