

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7645262号
(P7645262)

(45)発行日 令和7年3月13日(2025.3.13)

(24)登録日 令和7年3月5日(2025.3.5)

(51)国際特許分類		F I	
G 0 6 F	21/74 (2013.01)	G 0 6 F	21/74
G 0 6 F	9/30 (2018.01)	G 0 6 F	9/30 3 8 0 Z
G 0 6 F	9/48 (2006.01)	G 0 6 F	9/48 1 0 0 Z
		G 0 6 F	9/48 3 5 0 B
請求項の数 24 (全70頁)			
(21)出願番号	特願2022-532799(P2022-532799)	(73)特許権者	500395107
(86)(22)出願日	令和2年11月11日(2020.11.11)		アーム・リミテッド
(65)公表番号	特表2023-505149(P2023-505149 A)		グレート・ブリテン及び北部アイルランド連合王国、イングランド、シー・ビー1、9エヌ・ジェイ、ケンブリッジ、フルボーン・ロード 110
(43)公表日	令和5年2月8日(2023.2.8)	(74)代理人	110000855
(86)国際出願番号	PCT/GB2020/052855		弁理士法人浅村特許事務所
(87)国際公開番号	WO2021/116654	(72)発明者	グロカット、トーマス クリストファー
(87)国際公開日	令和3年6月17日(2021.6.17)		イギリス国 ケンブリッジシャー CB1
審査請求日	令和5年11月2日(2023.11.2)		9NJ ケンブリッジ、フルボーン ロード 110, アーム リミテッド宛
(31)優先権主張番号	1918216.1	審査官	上島 拓也
(32)優先日	令和1年12月11日(2019.12.11)		
(33)優先権主張国・地域又は機関	英国(GB)		
		最終頁に続く	

(54)【発明の名称】 ドメイン遷移無効化構成パラメータ

(57)【特許請求の範囲】

【請求項1】

装置であって、

少なくともセキュアドメイン及び低セキュアドメインを含む複数のセキュリティドメインのうちの1つでデータ処理を実行するための処理回路と、

前記処理回路の現在のセキュリティドメインに応じて、メモリアクセスが許可されているかどうかをチェックするためのメモリアクセスチェック回路であって、前記メモリアクセスチェック回路は、前記処理回路が前記セキュアドメインでデータ処理を実行している場合、前記処理回路が前記メモリのセキュア領域および前記メモリの低セキュア領域にアクセスすることを許可し、前記メモリアクセスチェック回路は、前記処理回路が前記低セキュアドメインでデータ処理を実行している場合、前記処理回路が前記メモリの前記セキュア領域にアクセスすることを許可しない、前記メモリアクセスチェック回路と、

前記セキュアドメインと前記低セキュアドメインとの間のドメイン遷移が前記処理回路の少なくとも1つのモードにおいて有効にされるか無効にされるかを指定するドメイン遷移無効化構成パラメータを記憶するための制御記憶位置と、を備え、

前記処理回路の前記少なくとも1つのモードにおいて、前記ドメイン遷移無効化構成パラメータが、前記少なくとも1つのモードにおいて前記ドメイン遷移を無効にすると指定する場合、前記処理回路は、

前記セキュアドメインから前記低セキュアドメインへの遷移の試行に応答して、無効化ドメイン遷移障害のシグナリングをトリガし、

前記低セキュアドメインから前記セキュアドメインへの遷移の試行に应答して、前記無効化ドメイン遷移障害のシグナリングをトリガする、ように構成され、
前記処理回路が、特権状態および低特権状態のうちの1つでデータ処理を実行するように構成され、
前記処理回路が前記特権状態にあるとき、前記処理回路の少なくとも1つのモードにおいて、前記セキュアドメインと前記低セキュアドメインとの間のドメイン遷移が有効であるか無効であるかを指定するドメイン遷移無効化構成パラメータが、前記メモリの前記セキュア領域へのアクセスが許可されていない前記低セキュアドメインにおいて変更可能である、装置。

【請求項2】

前記セキュアドメインにおいて、前記処理回路は、少なくとも1つのセキュア構成レジスタに記憶された情報を使用してメモリへのアクセスを制御するように構成されており、
前記低セキュアドメインにおいて、前記処理回路は、少なくとも1つの低セキュア構成レジスタに記憶された情報を使用してメモリへのアクセスを制御するように構成されている、請求項1に記載の装置。

【請求項3】

前記セキュアドメインに関連付けられた少なくとも1つのスタックポインタレジスタと、前記低セキュアドメインに関連付けられた少なくとも1つのスタックポインタレジスタとを含む、スタックデータ構造を指すためのスタックポインタを記憶するための複数のスタックポインタレジスタを備える、請求項1又は2に記載の装置。

【請求項4】

前記少なくとも1つのモードで前記低セキュアドメインから前記セキュアドメインへの少なくとも1つのタイプの遷移を実行しようとする試行に应答して、前記処理回路は、前記低セキュアドメインから前記セキュアドメインへの遷移の前記試行が許可されているかどうかを判定するために少なくとも1つの他のセキュリティチェックを実行する前に、前記無効化ドメイン遷移障害のシグナリングをトリガするかどうかを判定するために、前記ドメイン遷移無効化構成パラメータを使用して無効化ドメイン遷移チェックを実行するように構成されている、請求項1から3のいずれか一項に記載の装置。

【請求項5】

前記処理回路は、スレッドモード及びハンドラモードを有し、
例外条件に应答して、例外制御回路は、前記処理回路が、前記ハンドラモードにおける例外ハンドラの処理に切り替わるように制御するように構成されており、
前記処理回路の前記少なくとも1つのモードは、前記スレッドモードを含む、請求項1から4のいずれか一項に記載の装置。

【請求項6】

前記ハンドラモードにおいて、前記処理回路は、前記ドメイン遷移は前記ドメイン遷移無効化構成パラメータとは無関係に許可されているかどうかを判定するように構成されている、請求項5に記載の装置。

【請求項7】

分岐ターゲットアドレスを指定するインターモダル呼び出し分岐命令に应答して、少なくとも前記処理回路が前記ハンドラモードにあるとき、命令デコーダは、前記処理回路を、

関数戻りアドレスを関数戻りアドレス記憶位置に保存し、
前記処理回路の現在のモードを前記スレッドモードに切り替え、
前記ドメイン遷移無効化構成パラメータを設定して、前記ドメイン遷移が前記少なくとも1つのモードにおいて有効化又は無効にされるかどうかを指定し、
前記分岐ターゲットアドレスによって識別された命令に分岐する、ように制御するように構成されている、請求項5又は6に記載の装置。

【請求項8】

前記インターモダル呼び出し分岐命令の第1の変形に应答して、前記命令デコーダは

10

20

30

40

50

、前記処理回路を、前記ドメイン遷移が前記少なくとも1つのモードにおいて無効にされることを指定するように、前記ドメイン遷移無効化構成パラメータを設定し、

前記インターモーダル呼び出し分岐命令の第2の変形に応答して、前記命令デコーダは、前記処理回路を制御して、前記ドメイン遷移が前記少なくとも1つのモードにおいて有効にされることを指定するように、前記ドメイン遷移無効化構成パラメータを設定する、ように制御するように構成されている、請求項7に記載の装置。

【請求項9】

インターモーダル関数戻り命令に応答して、前記処理回路は、

前記処理回路の前記現在のモードを前記ハンドラモードに切り替えさせ、

前記ドメイン遷移無効化構成パラメータを更新させ、

関数戻りアドレスによって識別された命令に分岐させる、ように構成されている、請求項5から8のいずれか一項に記載の装置。

【請求項10】

前記インターモーダル関数戻り命令に応答して、前記処理回路は、前記ドメイン遷移が前記少なくとも1つのモードにおいて有効にされることを指定するように、前記ドメイン遷移無効化構成パラメータを設定するように構成されている、請求項9に記載の装置。

【請求項11】

前記インターモーダル関数戻り命令に応答して、前記処理回路は、前記インターモーダル関数戻り命令の前に、前記ドメイン遷移が前記少なくとも1つのモードにおいて有効化又は無効にされたことを、前記ドメイン遷移無効化構成パラメータが指定したかどうかのインジケーションを記録するように構成されている、請求項9又は10に記載の装置。

【請求項12】

前記無効化ドメイン遷移障害が、第1のセキュリティドメインから第2のセキュリティドメインへの遷移の試行に応答してシグナリングされるとき、前記処理回路が前記少なくとも1つのモードにあるとき、例外制御回路は、前記処理回路を、前記第1のセキュリティドメイン内の例外ハンドラを処理するように制御する、ように構成される、請求項1から11のいずれか一項に記載の装置。

【請求項13】

例外制御回路は、例外条件に応答して、前記ドメイン遷移が前記少なくとも1つのモードにおいて有効にされることを指定するように前記ドメイン遷移無効化構成パラメータを更新する、請求項1から12のいずれか一項に記載の装置。

【請求項14】

例外制御回路は、例外条件に応答して状態保存を実行して、前記例外条件が発生する前に実行された処理に関連するアーキテクチャ状態を保存し、前記状態保存に保存された前記アーキテクチャ状態は、前記例外条件の前の前記ドメイン遷移無効化構成パラメータを示す値を含む、請求項1から13のいずれか一項に記載の装置。

【請求項15】

前記状態保存において、前記例外制御回路は、例外戻りアドレス及び第1のサブセットのアーキテクチャ状態を含む第1のスタックフレームをスタックデータ構造に保存し、前記ドメイン遷移無効化構成パラメータを示す前記値を、前記例外条件に応答して実行される例外ハンドラにアクセス可能な所定のレジスタに保存する、ように構成されている、請求項14に記載の装置。

【請求項16】

少なくとも前記セキュアドメインから前記低セキュアドメインへの遷移をトリガする例外条件に応答して、前記例外制御回路は、前記第1のスタックフレームと、第2のサブセットのアーキテクチャ状態と、前記ドメイン遷移無効化構成パラメータをチェックするためのクロスチェック情報とを含む第2のスタックフレームと、を前記スタックデータ構造に保存するように構成されており、

少なくとも前記低セキュアドメインから前記セキュアドメインへの例外戻り遷移において、前記例外制御回路は、前記ドメイン遷移無効化構成パラメータを示す前記値と、前記

10

20

30

40

50

第2のスタックフレーム内の前記クロスチェック情報との間に不一致が検出された場合に、障害のシグナリングをトリガするように構成されている、請求項15に記載の装置。

【請求項17】

前記セキユアドメインから前記低セキユアドメインへの遷移を要求するドメイン間呼び出し分岐命令に応答して、命令デコーダは、前記処理回路を、

関数戻りアドレスを関数戻りアドレス記憶位置に保存することと、

前記処理回路を前記低セキユアドメインに切り替えることと、

前記ドメイン間呼び出し分岐命令によって指定された分岐ターゲットアドレスによって識別された命令に分岐することと、を含むドメイン間呼び出し分岐動作を実行するように制御する、ように構成されている、請求項1から16のいずれか一項に記載の装置。

10

【請求項18】

前記ドメイン間呼び出し分岐命令に応答して、前記命令デコーダは、前記処理回路を、

前記ドメイン遷移無効化構成パラメータが、前記ドメイン遷移が前記少なくとも1つのモードにおいて有効にされることを指定するか、又は前記処理回路が前記少なくとも1つのモード以外のモードにある場合、前記ドメイン間呼び出し分岐動作を実行し、

前記処理回路が前記少なくとも1つのモードにあり、前記ドメイン遷移無効化構成パラメータが、前記ドメイン遷移が前記少なくとも1つのモードにおいて無効にされることを指定する場合、前記無効化ドメイン遷移障害のシグナリングをトリガする、ように制御するように構成されている、請求項17に記載の装置。

【請求項19】

20

前記セキユアドメインから前記低セキユアドメインへの遷移を要求するドメイン間呼び出し分岐命令に応答して、命令デコーダは、前記処理回路を、関数戻りアドレスをスタックデータ構造に保存させ、ダミー関数戻りアドレスをリンクレジスタに保存するように制御する、ように構成されており、

前記ダミー関数戻りアドレスがドメイン間関数戻りを示すとき、前記ダミー関数戻りアドレスに分岐しようとする試行に応答して、前記処理回路は、前記処理回路を、前記処理回路が前記少なくとも1つのモードにあり、前記ドメイン遷移無効化構成パラメータが、前記ドメイン遷移が前記少なくとも1つのモードにおいて無効にされることを指定するとき、前記無効化ドメイン遷移障害のシグナリングをトリガするように制御する、ように構成されている、請求項1から18のいずれか一項に記載の装置。

30

【請求項20】

前記メモリアクセスチェック回路は、それぞれのアドレス領域がセキユアアドレス領域又は低セキユアアドレス領域であるかどうかを示すセキュリティ属性データを維持するように構成されており、

前記装置は、命令デコーダによる復号のためにキャッシュ又はメモリから命令をフェッチするための命令フェッチ回路を備えており、

セキユアアドレス領域内のアドレスを有する命令をフェッチする試行に応答して、前記処理回路が前記低セキユアドメイン内にあり、前記処理回路が前記少なくとも1つのモードにあり、前記ドメイン遷移無効化構成パラメータが、前記少なくとも1つのモードにおいて前記ドメイン遷移が無効にされることを指定する場合、前記命令フェッチ回路は、前記無効化ドメイン遷移障害のシグナリングをトリガするように構成されている、請求項1から19のいずれか一項に記載の装置。

40

【請求項21】

前記無効化ドメイン遷移障害のシグナリングをトリガすることは、無効化ドメイン遷移障害を示すシンδροームフラグを設定することを含む、請求項1から20のいずれか一項に記載の装置。

【請求項22】

データ処理方法であって、

少なくともセキユアドメイン及び低セキユアドメインを含む複数のセキュリティドメインのうちの1つにおいてデータ処理を実行するための処理回路と、前記処理回路の現在の

50

セキュリティドメインに応じてメモリアクセスが許可されたかどうかをチェックするメモリアクセスチェック回路と、を含む装置を使用してデータ処理を実行することであって、前記メモリアクセスチェック回路は、前記処理回路が前記セキュアドメインでデータ処理を実行している場合、前記処理回路が前記メモリのセキュア領域および前記メモリの低セキュア領域にアクセスすることを許可し、前記メモリアクセスチェック回路は、前記処理回路が前記低セキュアドメインでデータ処理を実行している場合、前記処理回路が前記メモリの前記セキュア領域にアクセスすることを許可しない、前記実行することと、

制御記憶位置において、前記セキュアドメインと前記低セキュアドメインとの間のドメイン遷移が前記処理回路の少なくとも1つのモードにおいて有効にされるか無効にされるかを指定するドメイン遷移無効化構成パラメータを記憶することと、

前記処理回路の前記少なくとも1つのモードにおいて、前記セキュアドメインと前記低セキュアドメインとの間のドメイン遷移の試行に応答して、前記ドメイン遷移無効化構成パラメータが、前記ドメイン遷移が前記少なくとも1つのモードにおいて無効にされることを指定した場合に、試行された前記ドメイン遷移が前記セキュアドメインから前記低セキュアドメインへのものであるか、前記低セキュアドメインから前記セキュアドメインへのものであるかにかかわらず、無効化ドメイン遷移障害のシグナリングをトリガすることと、

を含み、

前記処理回路が、特権状態および低特権状態のうちの1つでデータ処理を実行するように構成され、

前記処理回路が前記特権状態にあるとき、前記処理回路の少なくとも1つのモードにおいて、前記セキュアドメインと前記低セキュアドメインとの間のドメイン遷移が有効であるか無効であるかを指定するドメイン遷移無効化構成パラメータが、前記メモリの前記セキュア領域へのアクセスが許可されていない前記低セキュアドメインにおいて変更可能である、データ処理方法。

【請求項23】

命令を実行するための命令実行環境を提供するために、ホストデータ処理装置を制御するコンピュータプログラムであって、

前記コンピュータプログラムが、

少なくともセキュアドメイン及び低セキュアドメインを含む複数のセキュリティドメインのうちの1つでデータ処理を実行するための処理プログラムロジックと、

前記処理プログラムロジックの現在のセキュリティドメインに応じて、メモリアクセスが許可されているかどうかをチェックするためのメモリアクセスチェックプログラムロジックであって、前記メモリアクセスチェックプログラムロジックは、前記処理プログラムロジックが前記セキュアドメインでデータ処理を実行している場合、前記処理プログラムロジックが前記メモリのセキュア領域および前記メモリの低セキュア領域にアクセスすることを許可し、前記メモリアクセスチェックプログラムロジックは、前記処理プログラムロジックが前記低セキュアドメインでデータ処理を実行している場合、前記処理プログラムロジックが前記メモリの前記セキュア領域にアクセスすることを許可しない、前記メモリアクセスチェックプログラムロジックと、

制御記憶位置において、前記セキュアドメインと前記低セキュアドメインとの間のドメイン遷移が前記処理プログラムロジックの少なくとも1つのモードにおいて有効にされるか無効にされるかを指定するドメイン遷移無効化構成パラメータを設定するための構成パラメータ設定プログラムロジックと、を備え、

前記処理プログラムロジックの前記少なくとも1つのモードにおいて、前記ドメイン遷移無効化構成パラメータが、前記少なくとも1つのモードにおいて前記ドメイン遷移を無効にすると指定する場合、前記処理プログラムロジックは、

前記セキュアドメインから前記低セキュアドメインへの遷移の試行に応答して、無効化ドメイン遷移障害のシグナリングをトリガし、

前記低セキュアドメインから前記セキュアドメインへの遷移の試行に応答して、前記

10

20

30

40

50

無効化ドメイン遷移障害のシグナリングをトリガする、ように構成され、
前記処理プログラムロジックが、特権状態および低特権状態のうちの1つでデータ処理を実行するように構成され、
前記処理プログラムロジックが前記特権状態にあるとき、前記処理プログラムロジックの少なくとも1つのモードにおいて、前記セキュアドメインと前記低セキュアドメインとの間のドメイン遷移が有効であるか無効であるかを指定するドメイン遷移無効化構成パラメータが、前記メモリの前記セキュア領域へのアクセスが許可されていない前記低セキュアドメインにおいて変更可能である、コンピュータプログラム。

【請求項 2 4】

請求項 2 3 に記載の前記コンピュータプログラムを記憶する、コンピュータ可読記憶媒体。

【発明の詳細な説明】

【技術分野】

【0 0 0 1】

本技術は、データ処理分野に関する。

【0 0 0 2】

処理回路は、セキュアドメイン及び低セキュアドメインを含む複数のセキュリティドメインを有することがある。メモリアクセス回路は、現在の動作ドメインに基づいてメモリアクセスが許可されるかどうかを制御することができる。メモリのセキュア領域に記憶されたデータ又はプログラムコードは、低セキュアドメインで動作するコードによるアクセスが許可されない場合がある。これは、低セキュアアプリケーションによる不適切なアクセスからセキュアアプリケーションを隔離するのに役立ち得る。

【0 0 0 3】

少なくともいくつかの例は、少なくともセキュアドメイン及び低セキュアドメインを含む複数のセキュリティドメインのうちの1つでデータ処理を実行するための処理回路と、処理回路の現在のセキュリティドメインに応じて、メモリアクセスが許可されているかどうかをチェックするためのメモリアクセスチェック回路と、セキュアドメインと低セキュアドメインとの間のドメイン遷移が処理回路の少なくとも1つのモードにおいて有効にされるか無効にされるかを指定するドメイン遷移無効化構成パラメータを記憶するための制御記憶位置と、を含む装置を提供し、処理回路の当該少なくとも1つのモードにおいて、ドメイン遷移無効化構成パラメータが、当該少なくとも1つのモードにおいて当該ドメイン遷移を無効にすると指定する場合、処理回路は、セキュアドメインから低セキュアドメインへの遷移の試行に回答して、無効化ドメイン遷移障害のシグナリングをトリガし、低セキュアドメインからセキュアドメインへの遷移の試行に回答して、無効化ドメイン遷移障害のシグナリングをトリガする、ように構成されている。

【0 0 0 4】

少なくともいくつかの例は、データ処理方法を提供し、少なくともセキュアドメイン及び低セキュアドメインを含む複数のセキュリティドメインのうちの1つにおいてデータ処理を実行するための処理回路と、処理回路の現在のセキュリティドメインに応じてメモリアクセスが許可されたかどうかをチェックするメモリアクセスチェック回路と、を含む装置を使用してデータ処理を実行することと、制御記憶位置において、セキュアドメインと低セキュアドメインとの間のドメイン遷移が処理回路の少なくとも1つのモードにおいて有効にされるか無効にされるかを指定するドメイン遷移無効化構成パラメータを記憶することと、処理回路の当該少なくとも1つのモードにおいて、セキュアドメインと低セキュアドメインとの間のドメイン遷移の試行に回答して、ドメイン遷移無効化構成パラメータが、当該ドメイン遷移が当該少なくとも1つのモードにおいて無効にされると指定した場合に、試行されたドメイン遷移がセキュアドメインから低セキュアドメインへのものであるか、低セキュアドメインからセキュアドメインへのものであるかにかかわらず、無効化ドメイン遷移障害のシグナリングをトリガすることと、

を含む。

【0005】

少なくともいくつかの実施例は、命令を実行するための命令実行環境を提供するように、ホストデータ処理装置を制御するためのコンピュータプログラムを提供し、コンピュータプログラムは、少なくともセキュアドメイン及び低セキュアドメインを含む複数のセキュリティドメインのうちの1つでデータ処理を実行するための処理プログラムロジックと、処理プログラムロジックの現在のセキュリティドメインに応じて、メモリアクセスが許可されているかどうかをチェックするためのメモリアクセスチェックプログラムロジックと、制御記憶位置において、セキュアドメインと低セキュアドメインとの間のドメイン遷移が処理プログラムロジックの少なくとも1つのモードにおいて有効にされるか無効にされるかを指定するドメイン遷移無効化構成パラメータを設定するための構成パラメータ設定プログラムロジックと、を含み、処理プログラムロジックの当該少なくとも1つのモードにおいて、ドメイン遷移無効化構成パラメータが、少なくとも1つのモードにおいて当該ドメイン遷移を無効にすると指定する場合、処理プログラムロジックは、セキュアドメインから低セキュアドメインへの遷移の試行に応答して、無効化ドメイン遷移障害のシグナリングをトリガし、低セキュアドメインからセキュアドメインへの遷移の試行に応答して、無効化ドメイン遷移障害のシグナリングをトリガする、ように構成されている。

10

【0006】

コンピュータ可読記憶媒体は、上述のコンピュータプログラムを記憶することができる。記憶媒体は、非一時的記憶媒体であってもよい。

20

【図面の簡単な説明】

【0007】

本技術の更なる態様、特徴、及び利点は、添付の図面と併せて読まれる以下の例の説明から明らかとなる。

【0008】

【図1】異なる動作モードをサポートする処理回路を有するデータ処理システムの例を概略的に示す。

【0009】

【図2】処理回路の異なるドメイン及び動作モードの例を示す。

【0010】

【図3】処理回路のレジスタの一例を示す。

30

【0011】

【図4】実行可能命令に使用することができない予約済みアドレス範囲の提供を示す。

【0012】

【図5】ダミー関数戻りアドレス及びダミー例外戻りアドレスを示す。

【0013】

【図6】関数呼び出し又は例外に응答してスタックに保存され得る異なるスタックフレームを示す。

【0014】

【図7】比較のために、インターモーダル呼び出し分岐命令がサポートされなかった場合の例外特権解除の例を示す。

40

【0015】

【図8】インターモーダル呼び出し分岐命令を使用した例外特権解除の例を示す。

【0016】

【図9】禁止された組み合わせドメイン／モード切り替えの例を示す。

【0017】

【図10】許可された組み合わせドメイン／モードの切り替えの例を示す。

【0018】

【図11A】図10の組み合わせドメイン／モード切り替えが有用であり得る末尾呼び出しの例を示す。

【図11B】図10の組み合わせドメイン／モード切り替えが有用であり得る末尾呼び出

50

しの例を示す。

【図 1 1 C】図 1 0 の組み合わせドメイン／モード切り替えが有用であり得る末尾呼び出しの例を示す。

【0 0 1 9】

【図 1 2】関数が呼び出されたときに実行されるステップを示すフロー図である。

【0 0 2 0】

【図 1 3】例外エントリで実行されるステップを示すフロー図である。

【0 0 2 1】

【図 1 4】関数呼び出し以外の分岐命令に応答して実行されるステップを示すフロー図である。

【0 0 2 2】

【図 1 5】ダミー関数戻りアドレスへの分岐に応答して実行されるステップを示すフロー図である。

【0 0 2 3】

【図 1 6 A】図 1 5 に示す完全性クロスチェックを使用して検出できる潜在的な攻撃を示す図である。

【図 1 6 B】図 1 5 に示す完全性クロスチェックを使用して検出できる潜在的な攻撃を示す図である。

【図 1 6 C】図 1 5 に示す完全性クロスチェックを使用して検出できる潜在的な攻撃を示す図である。

【図 1 6 D】図 1 5 に示す完全性クロスチェックを使用して検出できる潜在的な攻撃を示す図である。

【0 0 2 4】

【図 1 7】スレッドモードにおけるドメイン遷移の無効化を示すフロー図である。

【0 0 2 5】

【図 1 8】ドメイン遷移無効化構成パラメータを使用する例を示す。

【図 1 9】ドメイン遷移無効化構成パラメータを使用する例を示す。

【図 2 0】ドメイン遷移無効化構成パラメータを使用する例を示す。

【0 0 2 6】

【図 2 1】セキュアアドレス領域内のアドレスを有する命令のフェッチ時のドメイン遷移無効化構成パラメータのチェックを示すフロー図である。

【0 0 2 7】

【図 2 2】使用され得るシミュレータの例を示す。

【0 0 2 8】

インターモダル呼び出し分岐命令

処理回路がハンドラモード及びスレッドモードをサポートするシステムでは、一般に、例外ハンドラ（例えば、オペレーティングシステム又は他の監視ソフトウェアに関連付けられる）の処理にハンドラモードが使用され、オペレーティングシステム又は監視ソフトウェアによって管理されるスレッドのバックグラウンドコードを処理するためにスレッドモードが使用されることが期待される。しかしながら、USBコントローラ又はワイヤレスネットワークインタフェースなど、信頼できない可能性があるデバイスに向けられる例外のタイプが存在する場合がある。そのようなデバイスに向けられた割り込みは、処理システムの一般的な動作を管理するオペレーティングシステムを提供する当事者とは異なる、サードパーティによって提供される例外ハンドラコードによって処理され得る。オペレーティングシステムプロバイダは、他のパーティによって提供されたコードを信頼しない可能性があり、そのため、そのような信頼できないデバイスを制御するための信頼できない例外ハンドラコードがハンドラモードに関連する完全な特権を持たないことを望む可能性がある。従って、例外ハンドラが、ハンドラモードのコードが通常アクセスすることができる全てのレジスタ又はメモリにアクセスする能力を有さないように、例外を「非特権化」することができることが望まれる場合がある。また、デバイス上で実行されている

10

20

30

40

50

全てのソフトウェアが同じベンダからのものである場合、特権のあるソフトウェアの複雑さ、従って、特権のあるソフトウェアにセキュリティの脆弱性が存在する可能性を低減することが望まれる場合がある。そのため、綿密な戦術における防御の一部として、全てのソフトウェアが単一のベンダからのものである場合でも、例外を「非特権化」したいという要望が依然としてあり得る。

【0029】

しかしながら、ハンドラモード及びスレッドモードを有する典型的なシステムでは、モードを切り替えるための機構は、スレッドモードからハンドラモードに切り替えるために例外を使用し、例外が戻ってスレッドモードに切り替えることであり得る。これは通常、元の例外に関連する優先度レベルを低下させ、異なる割り込み又は例外を適切に優先順位付けすることが困難になるため、スレッドモードに戻る例外を単にトリガすることは適切ではない場合がある。従って、多くの場合、第2の例外からの戻り時に元の例外がハンドラモードではなくスレッドモードで処理されることを引き起こす例外戻り制御情報をセットアップするためだけに、「偽の」第2の例外を生成する必要がある。第2の例外の呼び出しは、偽の例外戻り動作をセットアップするための動作の性能コストだけでなく、追加の状態保存及び復元動作のコストを招く可能性があるため、この機構は遅くなり得る。追加の状態保存及び復元動作は、通常、例外の前に実行されるプロセスに関連するアーキテクチャ状態を保存するために、例外エントリ及び戻りに対して実行される。従って、例外ハンドラを非特権化するために例外エントリ及び戻り機構が使用される場合、これは全体的な割り込み／例外処理待ち時間を増加させる可能性があり、これは一部のシステム、特にリアルタイムアプリケーション向けのシステムでは許容できない場合がある。また、この手法は、コンパイラをコード化することが困難であり、エラーが発生する可能性がある。

【0030】

以下に説明する例では、分岐ターゲットアドレスを指定するインターモダル呼び出し分岐命令が提供される。インターモダル呼び出し分岐命令に応答して、少なくとも処理回路がハンドラモードにあるとき、命令デコーダは、処理回路を、関数戻りアドレスを関数戻りアドレス記憶位置に保存し、処理回路の現在のモードをスレッドモードに切り替え、分岐命令によって指定された分岐ターゲットアドレスによって識別される命令に分岐する、ように制御する。従って、命令を、ハンドラモードではなくスレッドモードで処理される命令の分岐ターゲットアドレスに直接分岐させることができる、関数呼び出し機構が提供される。これは、例外を非特権化するためのはるかに高速で単純な機構を提供し、上述のような偽の第2の例外の必要性を排除する。

【0031】

ハンドラモード及びスレッドモードに加えて、処理回路はまた、少なくともセキュアドメイン及び低セキュアドメインを含む多数のセキュリティドメインのうちの1つにおけるデータ処理をサポートすることができる。処理回路の現在のセキュリティドメインに応じて、メモリアクセスが許可されているかどうかをチェックするためのメモリアクセスチェック回路が提供され得る。例えば、メモリアクセスチェック回路は、それぞれのアドレス領域をセキュアドメインに関連付けられたセキュア領域又は低セキュアドメインに関連付けられた低セキュア領域のいずれかとして定義するセキュリティ属性を提供するテーブルを維持することができる。セキュアドメインで動作するとき、処理回路は、メモリのセキュア領域と低セキュア領域の両方にアクセスすることができ、低セキュアドメインでは、処理回路は、低セキュア領域にアクセスすることができるが、セキュア領域にアクセスすることを許可されない場合がある。

【0032】

命令デコーダ及び処理回路は、分岐ターゲットアドレスを指定するドメイン間呼び出し分岐命令をサポートすることができる。ドメイン間呼び出し分岐命令に応答して、命令デコーダは、処理回路を、関数戻りアドレスを関数戻りアドレス記憶位置に保存し、分岐ターゲットアドレスによって識別される命令に分岐し、処理回路の現在のセキュリティド

メインを切り替える、ように制御し得る。従って、セキュリティドメインの変更はまた、1つのドメインから別のドメインへの直接的な分岐を許可することによって、より高速に行うことができる。いくつかのシステムでは、ドメイン間呼び出し分岐命令は、セキュアドメインから低セキュアドメインへ、及び低セキュアドメインからセキュアドメインへの両方の方向でサポートされ得る。しかしながら、他の実施態様では、ドメインを切り替える両方向に対してドメイン間呼び出し分岐命令を提供する必要はない場合がある。例えば、いくつかの実装形態では、ドメイン間呼び出し分岐命令は、セキュアドメインから低セキュアドメインへの切り替えをトリガするセキュアから低セキュア呼び出し分岐命令であってもよく、低セキュアドメインからセキュアドメインに遷移するために他の機構が提供されてもよい（例えば、代わりに、セキュアゲートウェイ命令は、ドメインを低セキュアドメインからセキュアドメインに切り替えることができ、その結果、セキュアドメインを入力することができる有効な位置は、セキュアゲートウェイ命令が、メモリの低セキュア領域からの命令を実行した後に、メモリのセキュア領域からフェッチされた第1のアドレスに存在することを要求することによって制限することができる）。いくつかの実装形態は、特権のみとしてマークされたメモリの領域にアクセスする低特権状態で実行されるコードを防止するために同様の方式を使用することができる。

【0033】

従って、処理回路に、関数戻りアドレスを保存し、分岐ターゲットアドレスで命令に分岐する関数呼び出し動作を実行させる、いくつかの呼び出し分岐命令をサポートすることができる。これらは、同じモード及びドメイン内に留まる分岐命令、並びにスレッドモードへの切り替えをトリガするインターモーダル呼び出し分岐命令及びセキュリティドメインの変更をトリガするドメイン間呼び出し分岐命令を呼び出すことを含むことができる。しかしながら、同じ呼び出し分岐命令に応答したモードとドメインの両方の組み合わせた切り替えは禁止され得る。従って、インターモーダル呼び出し分岐命令に응答して、命令デコーダは、処理回路の現在のセキュリティドメインを変更することなく、現在のモードをスレッドモードに切り替えるように処理回路を制御することができる。これは、許可されるセキュリティドメインの有効な遷移の数を減らすことによってセキュリティを向上させることができ、セキュリティドメインの変更が安全であることを検証することをより簡単にする。

【0034】

インターモーダル関数戻り命令に응答して、処理回路は、処理回路の現在のモードをハンドラモードに戻し、インターモーダル呼び出し分岐命令によって以前に保存された関数戻りアドレスによって識別される命令に分岐することができる。従って、例外ハンドラの非特権部分が完了すると、インターモーダル関数戻り命令を使用して、対応する例外が取られたときに処理回路が最初に実行していたハンドラモードへの切り替えをトリガすることができる。

【0035】

いくつかの例では、インターモーダル関数戻り命令は、関数戻り動作を必要とするものとして命令デコーダによって識別される符号化を有する専用の関数戻り命令であり得る。

【0036】

しかしながら、これは必須ではなく、場合によっては、命令がインターモーダル関数戻りを引き起こすかどうかは、実行時にレジスタ又はメモリに記憶された情報に依存し得る。例えば、インターモーダル関数戻りは、予約済み実行不可能アドレス（以下ではダミー関数戻りアドレス、例えば図5に示すFNC__RETURNアドレス140として参照されるが、これについては後でより詳細に説明する）に分岐しようとする試行によってトリガされ得る。インターモーダル関数戻り命令は、例外ハンドラの非特権セクション内の信頼できないコードの一部であり得るため、インターモーダル関数戻り命令が、スレッドモードからハンドラモードへのスイッチバックをトリガすることを意図しない非インターモーダル関数戻り命令と同じ命令符号化を有する場合に有用であり得る。例外ハンドラの非特権セクションは、その例外ハンドラが非特権化されていることを知らないサードパー

10

20

30

40

50

ティによって提供され得るからである。

【0037】

いくつかの例では、インターモーダル呼び出し分岐命令に応答して、命令デコーダは、ダミー関数戻りアドレスをリンクレジスタに保存するように処理回路を制御することができる。ダミー関数戻りアドレスは、有効な実行可能アドレスではない予約済みアドレス範囲内のアドレスとすることができる。ダミー関数戻りアドレスは、ダミー関数戻りアドレスがインターモーダル呼び出し分岐命令に응答して保存されたことを示す値を示すインターモーダル呼び出しインジケーション値を含むことができる。命令がダミー関数戻りアドレスに分岐しようとcする場合、ダミー関数戻りアドレスがインターモーダル呼び出しインジケーション値を含むとき、処理回路はその命令をインターモーダル関数戻り命令として扱い、ハンドラモードに切り替えると共に、スタックから取得された実関数戻りアドレスに分岐することができる。従って、ダミー関数戻りアドレス内インターモーダル呼び出しインジケーション値を記憶することによって、これは、インターモーダル関数戻りを非インターモーダル戻りと区別することができ、その結果、同じ命令符号化を両方のタイプの関数戻りに使用することができる。これは、上述のインターモーダル分岐技術をサポートするために、サードパーティ非特権例外ハンドラコードの修正が必要ないことを意味する。また、リンクレジスタ内の実行不可能なダミー関数戻りアドレスを使用して、関数戻り時に実関数戻りアドレスがスタックからアクセスされるべきであることをシグナリングすることによって、これは、インターモーダル呼び出し分岐命令に続いてスレッドモードで実行される非特権コードから実関数戻りアドレスを隠すことができる。

【0038】

インターモーダル関数戻り命令は、ダミー関数戻りアドレスへの分岐を引き起こし得るいくつかのタイプの命令、例えば分岐命令、又はプログラムカウンタをダミー関数戻りアドレスに設定させる非分岐命令（レジスタ移動命令、ロード命令、又はスタックポップ命令など）、のうちの1つであり得る。従って、いくつかの実装形態では、インターモーダル関数戻りが実行されるべきであり、従ってこれもハンドラモードへの切り替えをトリガすべきであることを識別するのは（命令デコーダではなく）処理回路であってもよい。いくつかの例では、全てのタイプの分岐がダミー関数戻りアドレスを分岐ターゲットアドレスとして指定する場合、それらは関数戻り処理を引き起こす可能性がある。他の例では、ダミー関数戻りアドレスを分岐対象アドレスとして指定しても、関数戻り処理が実行されない分岐の種類が存在してもよい（例えば、即値を使用してターゲットアドレスを指定する分岐は、関数呼び出し命令に응答して即時指定分岐ターゲットアドレスを保存することができなかったため、関数戻りを実行する必要がないため、関数戻り処理を実行させない場合がある）。

【0039】

インターモーダル呼び出し分岐命令とは異なり、インターモーダル関数戻り命令の場合、ドメインとモードの両方の組み合わせた切り替えを許可することが望ましい場合がある。従って、インターモーダル関数戻り命令に응答して、処理回路は、第1のセキュリティドメイン及びスレッドモードでの処理から第2のセキュリティドメイン及びハンドラモードでの処理への組み合わせたドメイン／モード切り替えを実行することができてよい。これは、第1の関数に対して実行されるコード内の最後のアクションとして第2の関数が呼び出されるシナリオにおいて、一部のコンパイラによって使用される性能最適化である末尾呼び出しをサポートするのに有用であり得る。第2の関数は第1の関数とは異なるセキュリティドメインで処理することができるので、インターモーダル関数戻り命令が組み合わせドメイン／モード切り替えをサポートするのに有用であり得、その結果、末尾呼び出しで以前にコンパイルされたレガシーコードが依然として正しく機能する。

【0040】

しかしながら、場合によっては、第1のセキュリティドメインがセキュアドメインであり、第2のセキュリティドメインが低セキュアドメインである場合、インターモーダル関数戻りでのドメイン／モードの組み合わせた切り替えは許可され得るが、第1のセキュ

リティドメインが低セキュアドメインであり、第2のセキュリティドメインがセキュアドメインである場合には禁止され得る。これがセキュアドメインへの遷移を引き起こす場合、関数戻りにおける組み合わせたドメイン／モード切り替えを禁止することによって、セキュアドメインへの有効なエントリに利用可能な手段の数を減らすことによってセキュリティを向上させることができる。

【0041】

以下で更に説明するように、スレッドモードにある間のセキュアドメインと低セキュアドメインとの間の遷移が無効にされるか有効にされるかを示すために、ドメイン遷移無効化構成値が提供され得る。これは、セキュアドメイン又は低セキュアドメインのうちの1つに関連するリソースの遅延割り当てをサポートするのに有用であり得る。インターモーダル呼び出し分岐命令を実行している例外ハンドラ又は他のプロセスは、両方のドメインで動作するために必要なリソースを既に構成しているかどうかを認識できることが多いため、インターモーダル呼び出し分岐命令に応答して、ドメイン遷移無効化構成値を設定することが有用であり得る。これは、ドメイン遷移無効化構成値を設定するために別個の命令を実行する必要性を回避することによって性能を改善する。インターモーダル呼び出し分岐命令の1つ以上の変形。インターモーダル呼び出し分岐命令の第1の変形例に
10 応答して、命令デコーダは、スレッドモードにある間のセキュアドメインと低セキュアドメインとの間の遷移が無効にされることを示すドメイン遷移無効化構成値を設定するように、
20 処理回路を制御することができる。インターモーダル呼び出し分岐命令の第2の変形例に
20 応答して、命令デコーダは、スレッドモードにある間のセキュアドメインと低セキュアドメインとの間の遷移が有効にされることを示すドメイン遷移無効化構成値を設定する
ように、処理回路を制御することができる。いくつかのシステムは、インターモーダル呼び出し分岐命令の両方の変形をサポートすることができる。他のシステムは、インターモー
ダル呼び出し分岐命令の第1の変形／第2の変形のうちの1つのみをサポートすることが
30 できる。

【0042】

インターモーダル呼び出し分岐命令に
30 応答して、命令デコーダは、処理回路の現在の
モードがスレッドモードである場合に、障害のシグナリングをトリガすることができる。
インターモーダル呼び出し分岐命令は、ハンドラモードのスレッドモードへの遷移を引き
起こすことを意図しているため、スレッドモードからそれを実行することは、エラーの
インジケーションであり得る。この場合、障害が発生すると誤用が防止される。例えば、
インターモーダル呼び出し分岐命令の実行は、対応する関数戻り上でスレッドモードから
ハンドラモードへの遷移が行われるべきであることを示す状態の保存をトリガする可能性
があるため、スレッドモードでプロセスを実行している攻撃者が、対応する関数戻りに対
する特権を得ようと試行してインターモーダル呼び出し分岐命令を使用しようとするこ
40 となりかねない。スレッドモードでのインターモーダル呼び出し分岐命令の実行の成功を
防止し、これが試行される場合に障害をシグナリングすることによって、この種の攻撃が
成功するのを防止することによってセキュリティを向上させる。

【0043】

一般に、関数戻り命令は、呼び出し分岐命令に
40 応答して以前に設定された関数戻りア
ドレスによって識別された命令に分岐を引き起こす任意の命令であり得る。インター
モーダル関数戻り命令は、現在のモードでスレッドモードからハンドラモードへの切り
替えもトリガする特定のタイプの関数戻り命令である（例えば、インターモーダル関
数戻りは、レジスタ若しくはメモリに記憶された情報によって、又は分岐ターゲットが
予約済みダミー関数戻りアドレスであることによって、非インターモーダル関数戻
りと区別することができる）。現在のモードがハンドラモードであるときにインター
モーダル関数戻り命令に遭遇した場合、障害がシグナリングされ得る。これにより、
セキュリティを向上させることができる。インターモーダル関数呼び出しでは、いく
50 つかの情報がメモリ内のスタック
データ構造上のスタックフレームに保存され得る。現在のモードがハンドラモード
であるときにインターモーダル関数戻りが試行される場合、これは、関数戻り遷移が
そのスタッ

クフレームを使用することが期待される意図された遷移ではないという徴候である可能性があり、スタックフレーム上の情報が誤用される場合にセキュリティを損なう危険性がある。ハンドラモードにあるときにのみインターモーダル戻り分岐命令が正常に実行され得ることをエンフォースすることによって、これはそのような攻撃に対する脆弱性を低減する。

【0044】

プロセッサアーキテクチャは、呼び出し分岐命令に応答して関数戻りアドレスを記憶するために指定されるリンクレジスタを提供することができる。リンクレジスタはまた、汎用レジスタとして使用するために利用可能であり得るが、関数呼び出しでは、関数の完了時に処理が分岐する関数戻りアドレスを記憶するために使用される。インターモーダル呼び出し分岐命令以外の少なくとも1つの呼び出し分岐命令の場合、リンクレジスタは、関数戻りアドレス記憶位置として使用され得る。しかしながら、インターモーダル呼び出し分岐命令の場合、関数戻りアドレス記憶位置は、メモリに記憶されたスタックデータ構造上の位置であり得る。スタックデータ構造を含むメモリの領域は、スレッドモードにアクセスできない領域とすることができる（例えば、対応するアドレス領域に対するアクセス許可を定義するメモリ保護データが、スレッドモードに関連付けられた特権レベルを有するコードにアクセスできないことを示すため、及び／又はスタックデータ構造を指すためのスタックポインタがスレッドモードでアクセスできないレジスタ内にあるため）。関数戻りアドレスをリンクレジスタではなくスタックに保存することによって、インターモーダル関数呼び出し後にスレッドモードで実行する非特権コードに関数戻りアドレスが見えることを防ぎ、攻撃者がハンドラモードで実行するより特権的なコードの動作に関する情報を見たり変更したりする機会を減らすことによってセキュリティを向上させる。

【0045】

上述したように、インターモーダル呼び出し分岐命令に応答して、ダミー関数戻りアドレスがリンクレジスタに保存され得る。しかしながら、対応する関数戻り命令がダミー関数戻りアドレスに分岐しようと試行する時点で、このダミー関数戻りアドレスはもはやリンクレジスタに記憶されない可能性があることに留意されたい。いくつかのコンパイラは、ダミー関数戻りアドレスがリンクレジスタ内に保持されるようにコードを生成することができ、別の関数が以前に呼び出された関数内にネストされている場合、コンパイルされたコードは、ダミー関数戻りアドレスをメモリ内のスタックに保存し、ネストされた関数が完了すると、ダミー関数戻りアドレスをリンクレジスタに復元することができ、それにより、第1の関数からの戻り時点で、ダミー関数戻りアドレスがリンクレジスタ内に再び存在する。しかしながら、これは必須ではない。他のコンパイラは、ネストされた関数呼び出しからの戻り時にダミー関数戻りアドレスをリンクレジスタに復元しないコードをコンパイルすることができるが、代わりに、対応する関数戻りが必要なときに、ダミー関数戻りアドレスをスタックからプログラムカウンタレジスタに直接ロードし、ダミー関数戻りアドレスへの分岐を効果的に引き起こすことによって性能を向上させ、これは、実関数戻りアドレスをスタックから取得する必要があるとき検出され得る。従って、関数戻り時に、ダミー関数戻りアドレスがリンクレジスタにまだ記憶されていることは必須ではない。なお、関数呼び出しのネストを可能にするためにダミー関数戻りアドレスがスタックに保存される場合、ダミー関数戻りアドレスを記憶するために使用されるスタックは、インターモーダル呼び出し分岐命令が実行されるときに実関数戻りアドレスを記憶するために使用されるスタックデータ構造とは異なるスタックデータ構造であってもよい。例えば、インターモーダル呼び出し分岐命令に응答して実関数戻りアドレスを記憶するために使用されるスタックは、ハンドラモードに関連付けられた主スタックであり得るが、スレッドモードで実行する非特権コード内に関数呼び出しのネストがある場合、以前にリンクレジスタ内あるダミー関数戻りアドレスは、スレッドモードに関連付けられたプロセススタックデータ構造に保存され得る。

【0046】

ダミー関数戻りアドレスは、対応する関数戻りが試行されたときに動作を制御するた

めの様々な情報を含み得る。例えば、ダミー関数戻りアドレスは、ダミー関数戻りアドレスへの分岐を試行する命令に回答してスレッドモードからハンドラモードへの切り替えが要求されるかどうか、ダミー関数戻りアドレスへの分岐を試行する命令に回答して、処理回路の現在のセキュリティドメインにおける切り替えが要求されるかどうか、ダミー関数戻りアドレスへの分岐を試行する当該命令に回答して、関数戻りアドレスが取得されるスタックデータ構造として、複数のスタックデータ構造のうちのどれが使用されるべきか、のうちの少なくとも1つを示すことができる。従って、関数戻り時に、処理回路は、ダミー関数戻りアドレス内の情報を使用して、モードを切り替えるかどうか、ドメインを切り替えるかどうかを判定する、及び／又は関数戻りアドレスを取得するためにどのスタックデータ構造を使用するかを判定することができる。いくつかの例では、ダミー関数戻りアドレスは、2ビットを使用してこれらの3つの情報を示すことができ、第1のビットは、ダミー関数戻りアドレスがインターモーダル呼び出し分岐命令に回答して設定されたかどうかを示し、第2のビットは、インターモーダル関数呼び出しが行われたセキュリティドメインを示す。どのスタックデータ構造を使用すべきかは、これらの2つのビットから推測することができるため、明示的なインジケーションを必要としない場合がある。しかしながら、これはダミー関数戻りアドレス内の情報を符号化する単なる1つの方法であり、他の手法は異なる選択肢を取るることができることが理解されよう。また、いくつかの実装形態は、上述の3つのタイプの情報の全てを示すとは限らない場合がある。

【0047】

ダミー関数戻りアドレスの所定の部分は、ダミー関数戻りアドレスへの分岐を試行する命令に回答して、処理回路の現在のセキュリティドメイン内の切り替えが要求されるかどうかを示すことができる。例えば、この所定の部分は、ダミー関数戻りアドレスの最下位ビットとすることができる。現在のセキュリティドメインが低セキュアドメインであるときにセキュアゲートウェイ命令に回答して、処理回路は、戻りアドレスの一部分（例えば、リンクレジスタ内のアドレス）を低セキュア状態を示す値に設定してもよく、その部分は、ダミー関数戻りアドレス内のドメインの切り替えを示す部分と、戻りアドレス値内の同じ相対位置にある。この手法は、低セキュアドメイン内のアドレスに戻るのではなく、セキュアコード内の対応する戻り分岐をセキュアドメイン内の任意の位置にジャンプさせる試行において、低セキュアドメインからセキュア関数を呼び出す際に、セキュアコードに偽の戻りアドレスを渡す低セキュアコードに基づく攻撃を減らすことによって、セキュリティを向上させるのに役立つ。セキュアゲートウェイ命令は、低セキュアドメインから実行されると、低セキュアドメインからセキュアドメインへの切り替えをトリガする命令であってもよく、処理回路は、低セキュアドメインで命令を実行した後に、セキュアドメインからフェッチされた第1の命令がセキュアゲートウェイ命令ではない場合、障害のシグナリングをトリガしてもよい。従って、セキュアゲートウェイ命令は、セキュアドメインへの有効なエントリポイントに存在することが期待され得るため、戻りアドレスがセキュアドメイン内の任意のアドレスへの分岐を引き起こすように信頼され得ないことを示すために、セキュアドメインに入る時点で低セキュアコードによって設定された戻りアドレスをサニタイズすることができる。低セキュア状態を示す値に設定された部分を有するアドレスを、分岐ターゲットアドレスとして指定する所定のタイプの分岐命令が、セキュアドメイン内で実行されると、これにより、ターゲットアドレスがメモリ内のセキュア領域に対応するかそれとも低セキュア領域に対応するかにかかわらず、処理回路は低セキュアドメインに戻る。従って、低セキュアドメインによって制御される分岐ターゲットアドレスを使用していると予想される分岐に、所定のタイプの分岐命令を含むようにセキュアコードを書き込むことができるため、これは上述したタイプの攻撃に対する保護を提供する。インターモーダル関数戻り分岐は、同様の攻撃のリスクがある可能性があるため、ダミー関数戻りアドレスを符号化することによって、セキュリティドメインの切り替えが、セキュアゲートウェイ命令に回答して低セキュア状態を示す値に設定された戻りアドレスの部分と同じ相対位置で要求されるかどうかをシグナリングするために使用される部分がまた、インターモーダル関数戻り分岐を攻撃から保護する。

【0048】

リンクレジスタは、スレッドモードで実行するコードにアクセス可能なレジスタであってもよい。従って、対応する関数戻りをインターモーダル呼び出しとして扱うべきかどうかを示す情報を提供するために、リンクレジスタに記憶されたダミー関数戻りアドレスを使用することが便利であるが、ダミー関数戻りアドレスに配置された情報は、スレッドモードを実行している低特権コード実行によって修正されるように脆弱である可能性がある。ダミー関数戻りアドレスの悪意のある変更に基づく攻撃から保護するために、インターモーダル呼び出し分岐命令に 응답して、命令デコーダは、スタックデータ構造に保存されたスタックフレーム内の第1の相対位置の少なくとも一部にクロスチェック値を保存するように、処理回路を制御することができる。クロスチェック値が保存されるスタックデータ構造は、スレッドモードで実行されるコードにアクセスできないスタックであり得る。アドレスのダミー関数に分岐しようと試行する少なくとも1つのタイプの命令に 응답して、処理回路は、インターモーダル呼び出しインジケーション値を符号化するためのダミー関数戻りアドレスの一部と、スタックデータ構造から取得されたスタックフレーム内の第1の相対位置の少なくとも一部の値とを比較し、比較に基づいて障害のシグナリングをトリガするかどうかを判定することができる。従って、ダミー関数戻りアドレスは、スタックデータ構造上のスタックフレームにアクセスする必要があるかどうかをシグナリングすることができるが、スタックフレーム内のクロスチェック値は、セキュリティを高めるために、関数戻りのモードの変更を示す任意の情報を信頼できるという二重チェックを提供することができる。ダミー関数戻りアドレス内のセキュリティドメインインジケーションについても同様のチェックを行うことができる。

10

20

【0049】

インターモーダル呼び出し分岐命令以外の少なくとも1つのタイプの呼び出し分岐命令に 응답して、命令デコーダは、スタックデータ構造に保存されたスタックフレーム内の第1の相対位置の少なくとも一部におけるクロスチェック値以外の値を保存するように、処理回路を制御し得る。従って、クロスチェックはまた、非インターモーダル呼び出し分岐命令を介して入力されたコードから戻るための機構としてインターモーダル関数戻りを使用しようとする攻撃から保護することができる。このタイプの不一致関数呼び出し及び関数戻りは、メモリ内の不一致スタックフレームに基づく脆弱性をもたらす可能性がある。従って、インターモーダル関数戻りを示すためにダミー関数戻りアドレスを修正しようと試行したが、メモリ内の対応するスタックデータ構造が非インターモーダル呼び出し分岐命令によってメモリに実際に保存された場合、クロスチェックは失敗する可能性があり、そのため、障害がトリガされる可能性がある。

30

【0050】

インターモーダル呼び出し分岐命令に 응답して、命令デコーダは、スタックデータ構造に保存されたスタックフレーム内の第2の相対位置の少なくとも一部にハンドラモードを示す値を保存するように、処理回路を制御することができる。第2の相対位置は、第1の相対位置と同じアドレスであってもよく（この場合、値を示すインターモーダル呼び出しインジケーション値とハンドラモードを示す値のクロスチェック情報は、スタック上の同じデータワード内の異なる部分のビットにあってもよい）、又は第1の相対位置に対する異なるアドレスに対応するスタックフレーム内の異なる相対オフセットであってもよい。ダミー関数戻りアドレスがインターモーダル呼び出しインジケーション値を含むときにダミー関数戻りアドレスに分岐しようと試行する少なくとも1つのタイプの命令に 응답して、処理回路は、スタックフレーム内の第2の相対位置の関連部分内の値がハンドラモードを示さない場合に障害のシグナリングをトリガすることができる。従って、ダミー関数戻りアドレスがハンドラモード関数戻りへのスレッドモードを示すが、メモリ内のスタックフレームが、対応する関数呼び出しが元々ハンドラモードからのものではなかったことを示す場合、障害がトリガされ得る。やはり、これはいくつかの形態の攻撃から保護する。

40

【0051】

いくつかの実装形態は、第1の相対位置におけるクロスチェック値と、スタック上の

50

第2の相対位置における値を示すハンドラモードとの両方を提供することができる。他の実装形態は、これらのタイプのクロスチェックのうちの1つを省略することができ、クロスチェック値又はハンドラモードを示す値のいずれかのみを提供することができる。

【0052】

ダミー関数戻りアドレスがインターモーダル呼び出しインジケーション値を含むときに、ダミー関数戻りアドレスに分岐しようと試行する少なくとも1つのタイプの命令に回答して、処理回路は、スタックデータ構造から取得された関数戻りアドレスのチェックを実行することができる。処理回路の現在のモードをハンドラモードに切り替える前に、処理回路は、スタックデータ構造から取得された関数戻りアドレスが、命令実行が禁止される予約済みアドレスの所定の範囲内にあるかどうかをチェックすることができる。関数戻りアドレスが予約済みアドレスの所定の範囲内にある場合、スレッドモードを実行するプログラムコードに起因する障害がトリガされ得る。このタイプのチェックは、インターモーダル呼び出し分岐命令が実行されると、関数戻りアドレスが、スタックデータ構造に保存されたスタックフレーム内の第3の相対位置に保存され得、例外条件によってトリガされる少なくとも1つのタイプの例外遷移について、例外制御回路が、スタックフレーム内の第3の相対位置に保存された予約済みアドレスの所定の範囲のうちの1つを含むスタックフレームを保存し得るため、有用であり得る。従って、インターモーダル関数戻りをチェックすることによって、関数戻りアドレスが予約済みアドレスの所定の範囲のうちの1つであるかどうかをチェックすることによって、これは、そうでなければセキュリティを損なう予測不可能な結果をもたらし得る、関数呼び出し遷移ではなく例外エントリ遷移に回答して、実際に保存されたスタックフレームに基づいて関数戻りを実行しようと試行することに基づく攻撃から保護する。関数戻りアドレスのチェックはまた、他のタイプの攻撃を防止するのに役立ち得る。例えば、スタックの先頭がスタックフレーム内の第3の相対位置において（予約済みアドレスの所定の範囲から）別の予約済み値でシールされている場合、スタックをアンダーフローさせようとする試行（すなわち、スタックが空である場合にスタックフレームを使用する）も検出され、障害をシグナリングする。

【0053】

予約済みアドレスの所定の範囲からの命令を実行しようとする試行によって引き起こされる障害は、予約済みアドレスのうちの1つから命令をフェッチしようとする試行する時点でトリガされるべきであると予想されるため、インターモーダル関数戻り命令のチェックの一部として、関数戻りアドレスが予約済みアドレス範囲内にあるかどうかをチェックすることは冗長であるように思われ得る。しかしながら、本発明者は、インターモーダル関数戻りについて、インターモーダル関数戻りが正常に完了することが許可され、その結果、モードがハンドラモードに切り替えられ、その後、関数戻りアドレスから次の命令をフェッチしようとする試行すると障害がトリガされる場合、この障害は、ハンドラモードに切り替える前にスレッドモードで実行されているプログラムコードではなく、ハンドラモードで実行されているコードに起因する可能性があることを認識した。これは、ハンドラコードがオペレーティングシステムに関連付けられている可能性があるため、問題となる可能性があり、そのため、この障害は、オペレーティングシステム動作が障害していると疑われる可能性があり、フルシステムリセットなどの比較的侵入的なアクションを実行する必要がある可能性がある。従って、この手法では、これは、（例外スタックフレームに基づいて関数戻りを繰り返しトリガすることによって）処理システムの正しい機能を妨害することを目的としたサービス拒否攻撃を攻撃者が実行する方法を提供することができる。

【0054】

対照的に、（スレッドモードからハンドラモードへのモードの任意の切り替えをトリガする前に）インターモーダル関数戻り命令を処理するときに関数戻りアドレスが予約済みアドレス範囲内にあるかどうかの早期チェックが行われる本明細書に記載の手法では、障害はスレッドモードコードに起因する可能性があり、従って障害処理アクションはシステム機能にはるかに低い影響を与える可能性があり（例えば、単一のスレッドをキルし再開するだけである）、サービス拒否攻撃の機会を低減し、従ってセキュリティを向上させ

る。

【0055】

信頼性及び安全性の目的のために、いくつかの実装形態は、ハードウェア障害又は一時的な欠陥によって引き起こされる一時的又は永続的なエラーを検出するためのエラー検出／訂正回路を有することができる。例えば、エラー検出／訂正回路は、エラー検出／訂正コードを使用してレジスタ又はメモリに記憶されたデータのエラーを検出する回路を含むことができ、これは、例えば、粒子衝突又は記憶素子が0又は1でスタックすることによって引き起こされ得る。エラー検出／訂正回路はまた、それらの動作の冗長結果の比較からエラーを検出することができるように、同じ動作を2回以上実行することができる冗長処理ロジックを含むことができる。エラー検出／訂正回路の特定の形態は、そのシステムの必要性に応じてシステムごとに大きく異なり得る。上記のサービス拒否攻撃シナリオと同様の理由で、検出された任意のエラーを特定のデータ処理コンテキストに帰属させることができることが有用であり得、その結果、エラーの解決をより効率的に行うことができる（例えば、エラーが特定のスレッドに関連するデータ処理内で実際に発生した場合、オペレーティングシステムをキルし、侵入的システムリセットを実行する必要性を回避することによって）。

【0056】

エラーの分離を可能にするために、インターモーダル呼び出し分岐命令に応答して、命令デコーダは、エラー同期バリア動作の前に実行されたデータ処理に関連するエラーの検出を、エラー同期バリア動作の後に実行されたデータ処理に関連するエラーの検出から分離するために、エラー同期バリア動作を実行するように、処理回路を制御することができる。同様に、インターモーダル関数戻り命令の場合、これはまた、エラー同期バリア動作をトリガし得る。例えば、エラー同期バリア動作は、未処理の動作が完了し、エラー同期バリア動作が利用可能になる前に実行された任意の動作に対するエラーチェックの結果が得られ、以前の処理の結果が正しいかどうか分かるまで、処理を停止することを含むことができる。従って、これにより、発生するエラーを、エラーが発生した特定のコードに、より良好に特定することができる。いくつかのシステムでは、インターモーダル呼び出し分岐命令及びインターモーダル関数戻り命令は、常にエラー同期バリア動作をトリガすることができる。他のシステム（例えば、信頼性／安全性エラーチェック回路のいずれも有さないシステム）は、エラー同期バリア動作をサポートしない場合がある。他の例では、インターモーダル呼び出し分岐命令及びインターモーダル関数戻り命令が、エラー同期バリア動作をトリガするかどうかを制御するいくつかの構成状態データが提供されてもよい。エラー同期バリア動作の存在をソフトウェアによって構成することを可能にすることにより、単一のチップは、例えば、性能がエラー分離よりも重要であるか、又はその逆である、異なる市場のニーズにより良好に対処することができる。

【0057】

いくつかの実装形態は、セキュリティドメインごとにプロセススタックポインタレジスタと、セキュリティドメインごとにメインポインタレジスタとを含む、2つ以上のスタックポインタレジスタを提供することができる（1つのセキュリティドメインのみを有するシステムでは、単一のプロセススタックポインタレジスタ及び単一の主スタックポインタレジスタのみが存在してもよく、複数のセキュリティドメインをサポートするシステムでは、スタックポインタレジスタはセキュリティドメインごとにバンクされてもよい）。選択回路は、各セキュリティドメインごとに、メモリ内のスタックデータ構造にアクセスするためのスタックポインタを提供するために、どのスタックポインタレジスタを使用すべきかを選択することができる。ハンドラモードでは、選択回路は、現在のセキュリティドメインの主スタックポインタレジスタを選択することができる。スレッドモードでは、選択回路は、現在のドメインのスタックポインタ選択値に基づいて、現在のドメインの主スタックポインタレジスタとプロセススタックポインタレジスタとの間で選択することができる。ハンドラモード及びスレッドモード用の別々の主スタック及びプロセススタックをそれぞれ提供することは、例外ハンドラが主スタックデータ構造上に残し得る任意のデ

ータが、一般に、例外戻り後にスレッドモードで実行しているスレッドにアクセスできない可能性があることを意味するので、例外ハンドラコードの開発をより簡単にすることができる。いくつかの実装形態では、スレッド及びハンドラモードに別々のスタックを使用することはまた、偽の例外戻りスタックフレームの偽装に基づく特定の種類の攻撃から保護するのに役立ち得る。しかしながら、いくつかのシナリオ（例えば、利用可能なメモリが限られている場合）では、スレッドモードコードが、別個のプロセススタックを使用するスレッドモードの代わりに、主スタックポインタによって識別される主スタックをハンドラモードと共有することを可能にすることが望ましい場合がある。従って、どのスタックがスレッドモードで使用されるべきかを定義するために、スタックポインタ選択値が提供され得る（より特権的なコードによって制御される）。

10

【0058】

インターモダル呼び出し分岐命令に応答して、命令デコーダは、スレッドモードにあるときにプロセススタックポインタレジスタが選択されるべきであることを示すために、現在のセキュリティドメインのスタックポインタ選択値を設定するように、処理回路を制御することができる。これにより、より特権的なコードが主スタック構造に残っている可能性がある情報にアクセスするインターモダル関数呼び出しの後に例外ハンドラの非特権セクションが実行されるのを防ぐことができ、セキュリティが向上する。

【0059】

ホストデータ処理装置を制御して、ターゲットプログラムコードからの命令を実行するための命令実行環境を提供するためのシミュレータコンピュータプログラムを提供することができる。コンピュータプログラムは、処理プログラムロジック、例外制御プログラムロジック、並びに上述の処理回路、例外制御回路、及び命令デコーダの機能をエミュレートする命令復号プログラムロジックを有することができる。これは、上述したようなインターモダル呼び出し分岐命令のサポートを含む。従って、そのようなシミュレータコンピュータプログラムは、シミュレータコンピュータプログラムを実行しているホストコンピュータにこれらの機能を提供する実際のハードウェアがなくても、実際のハードウェア装置によって提供されるものと同様の命令環境を、シミュレータコンピュータプログラム上で実行されるターゲットコードに提示することができる。これは、そのアーキテクチャを実際にサポートしていないホストプラットフォーム上で1つの命令セットアーキテクチャに対して書き込まれたコードを実行するのに有用であり得る。また、シミュレータは、命令セットアーキテクチャの新しいバージョンをサポートするハードウェアデバイスの開発と並行してソフトウェア開発が実行されている場合に、命令セットアーキテクチャの新しいバージョン用のソフトウェアの開発中に有用であり得るので、開発中のソフトウェアをシミュレーションでテストして、命令セットアーキテクチャの新しいバージョンをサポートするハードウェアデバイスの準備が整う前に、ソフトウェア開発を開始することができる。

20

30

【0060】

ドメイン遷移無効化構成パラメータ

上述したように、処理回路は、少なくともセキュアドメイン及び低セキュアドメインを含むいくつかのセキュリティドメインを有することができ、処理回路の現在のセキュリティドメインに応じてメモリアクセスが許可されているかどうかをチェックするためにメモリアクセスチェック回路を設けることができる。例えば、このメモリアクセスチェックは、メモリ領域が安全であるか、又は安全でないと見なされるかを定義する属性データに基づいてもよく、低セキュアドメイン内で動作する場合、セキュア領域へのアクセスは禁止されてもよい。

40

【0061】

制御記憶位置は、処理回路における少なくとも1つのモードにおいて、セキュアドメインと低セキュアドメインとの間のドメイン遷移が有効にされるか無効にされるかを指定するドメイン遷移無効化構成パラメータを記憶するために提供され得る。少なくとも1つのモードにおいて、ドメイン遷移無効化構成パラメータが、ドメイン遷移を無効にすると

50

指定する場合、処理回路は、セキュアドメインから低セキュアドメインへの遷移の試行に
応答して、無効化ドメイン遷移障害のシグナリングをトリガし得、低セキュアドメインから
セキュアドメインへの遷移の試行に応答して、無効化ドメイン遷移障害のシグナリング
をトリガし得る。この手法は直感に反する可能性があり、なぜなら、低セキュアドメイン
からセキュアドメインへの遷移を防ぐことができることがセキュリティ上の理由から有用
であると期待するかもしれないが、セキュアドメインから低セキュアドメインへの遷移を
選択的に無効にすることができることがなぜ有用であるのかを疑問に思う場合があるため
である。しかしながら、本発明者は、これがセキュアドメイン及び低セキュアドメインの
うちの単一のドメインに特定の処理スレッドを含める方法を提供し、これは、ドメインを
切り替える試行が無効化ドメイン遷移障害をトリガすることが知られている場合、スレッド
が両方のドメインをまたぐ場合に必要とされ得る性能集約的動作の一部を抑制すること
ができるため、性能に優れ得ることを認識した。これにより、スレッドがドメインをまた
ぐかどうかにかかわらず、全てのスレッドに対して推測でこれらの動作を実行する性能コ
ストを回避するために、ドメインを切り替える試行があるときに必要な場合にのみ実行さ
れる、これらの性能集約的動作を緩慢に実行することが可能になる。従って、処理回路が
少なくとも1つのモードにあるときにセキュアドメインと低セキュアドメインとの間の双
方向の遷移を無効にするドメイン遷移無効化構成パラメータを提供することは、性能にと
って有益であり得る。

【0062】

この技術は、セキュアドメインにおいて、処理回路が、少なくとも1つのセキュア構
成レジスタに記憶された情報を使用してメモリへのアクセスを制御し、低セキュアドメイ
ンにおいて、処理回路が、少なくとも1つの低セキュア構成レジスタに記憶された情報
を使用してメモリへのアクセスを制御する場合に、特に有用であり得る。例えば、セキュ
ア／低セキュア構成レジスタは、メモリ内のそれぞれのセキュア又は低セキュアスタックデ
ータ構造にアクセスするためのスタックポインタを記憶するスタックポインタレジスタ、
メモリの領域へのアクセス許可を定義するためのメモリ保護属性を定義するメモリ保護ユ
ニット（MPU）内のレジスタ、及び／又はメモリ保護領域属性を提供するメモリ内のテ
ーブルへのポインタを提供するMPUポインタレジスタを含むことができる。セキュアド
メインと低セキュアドメインとの間の分離を保証するために、セキュアドメインと低セキュ
アドメインとでそれぞれ使用するためのMPU属性又はスタックデータ構造の別個のセ
ットを定義することが望ましい場合がある。しかしながら、スタックポインタを構成し、
スタックデータ構造のためにメモリアドレス空間内に空間を割り当て、及び／又はセキュ
アドメイン及び低セキュアドメインのためにMPU領域属性を構成するための動作は、比
較的性能集約的であり得、従って、他のドメインで動作する必要があるだけのスレッドの
ために、セキュアドメイン及び低セキュアドメインのうちの1つに対してこれらの動作を
実行することを回避することが望ましい場合がある。これは、割り込み処理のようなタイ
ムクリティカルな動作を実行する必要がある場合に特に重要であり得る。従って、ドメイ
ン遷移無効化構成パラメータを提供することによって、これは、これらの構成動作が実行
されていないスレッドが他のドメインに切り替えを試みる場合を検出するために使用する
ことができる障害を提供し、その結果、これらの構成動作はオンデマンドで緩慢に実行す
ることができる。一例では、構成されたリソースは、セキュアドメインに関連付けられた
少なくとも1つのスタックポインタレジスタ及び低セキュアドメインに関連付けられた少
なくとも1つのスタックポインタレジスタを含む、スタックデータ構造を指すためのスタ
ックポインタを記憶するスタックポインタレジスタを含み得る。

【0063】

少なくとも1つのモードで低セキュアドメインからセキュアドメインへの少なくとも
1つのタイプの遷移を実行しようとする試行に応答して、処理回路は、無効化ドメイン遷
移障害のシグナリングをトリガするかどうかを判定するために、ドメイン遷移無効化構成
パラメータを使用して無効化ドメイン遷移チェックを実行し得る。この無効化ドメイン遷
移チェックは、低セキュアドメインからセキュアドメインへの遷移の試行に対して、低セ

10

20

30

40

50

セキュアドメインからセキュアドメインへの遷移の試行が許可されているかどうかを判定するために少なくとも1つの他のセキュリティチェックを実行する前に、実行することができる。少なくとも1つの他のセキュリティチェックの少なくとも一部は、スタックポイントレジスタによって参照されるMPU構成又はスタックデータ構造など、ドメインを切り替えることが期待されていなかったスレッドに対してまだ構成されていない情報又はリソースに依存する可能性がある。従って、無効化ドメイン遷移チェックの前に他のセキュリティチェックが実行される場合、これは、何らかの他のセキュリティリスクが識別されたためではなく、単にチェックすべき関連データがまだ構成されていなかったために、そのセキュリティチェックが失敗することにつながる可能性がある。セキュリティチェックは、（リソースの遅延構成を単にトリガすることができる）無効化ドメイン遷移障害に関連付けられた障害ハンドラよりも劇的なアクションをトリガする障害ハンドラに関連付けられ得るので、最初に無効化ドメイン遷移チェックを実行することが望ましい場合があり、その結果、リソースが利用可能でない場合には、他のタイプのセキュリティチェックを実行し続ける前にそれらを構成することができる。例えば、いくつかの実装形態は、チェックが失敗した場合にセキュア障害を引き起こすセキュリティチェックを実行することができる。セキュア状態の完全性を確保するために、セキュア障害に関連付けられた例外ハンドラは、低セキュア状態全体の実行を終了し得る。そのような障害は、低セキュア状態で動作しているオペレーティングシステムに対してサービス拒否攻撃を実行する手段として、低セキュア及び低特権状態で動作しているソフトウェアによって意図的にトリガされる可能性がある。

10

20

【0064】

セキュアドメインから低セキュアドメインへの試行された遷移の場合、ドメイン遷移無効化構成パラメータを使用して、無効化ドメイン遷移チェックも実行され得る。しかしながら、この遷移がセキュアドメインから低セキュアドメインへの遷移であるため、他のセキュリティチェックを実行する必要がないことがあるため、無効化ドメイン遷移チェックと他のセキュリティチェックとの間の順序は関連しない場合がある。あるいは、いくつかのシステムは、セキュアドメインから低セキュアドメインに戻る際にセキュリティチェックを依然として実行することができ、その場合、無効化ドメイン遷移チェックと他のセキュリティチェックとの間の順序は、無効化ドメイン遷移障害に応答して構成された任意の情報がその他のセキュリティチェックで使用されると予想されるかどうか依存し得る。

30

【0065】

無効化ドメイン遷移チェックは、低セキュアドメインからセキュアドメインへの、又はセキュアドメインから低セキュアドメインへの、少なくとも1つのタイプの遷移を実行しようとする試行によってトリガされ得る。場合によっては、無効化ドメイン遷移チェックは、必ずしも実際にはセキュアドメインと低セキュアドメインとの間の遷移を引き起こさない可能性があるイベント又は命令によってトリガされ得る。例えば、場合によっては、セキュリティドメイン間の遷移を潜在的に引き起こす可能性があるイベント又は命令に遭遇したときに、そのイベント又は命令が実際にドメインの変更をトリガするかどうか識別される時点の前に、無効化ドメイン遷移チェックが実行され得る。従って、ドメイン遷移無効化構成パラメータは、ドメインを変更する実際の試行に回答して少なくともチェックされるが、これは、無効化ドメイン遷移チェックが合格し、ドメイン遷移が有効にされると判定した場合に、実際にはドメインの変更を引き起こさないイベントに対してチェックされることもあることを除外しない。

40

【0066】

ドメイン遷移無効化構成パラメータは、セキュアドメイン及び低セキュアドメインの両方において修正可能であってもよい。いくつかの実装形態では、特定のプログラムコードがドメイン遷移無効化構成パラメータを変更することを許可するかどうか、他の条件も課され得る。例えば、特権状態及び低特権状態を有するシステムでは、ドメイン遷移無効化構成パラメータは、特権状態でのみ修正可能であり得る。それにもかかわらず、処理回路は、現在のセキュリティドメインが低セキュアドメインであっても、ドメイン遷移無

50

効化構成パラメータを変更することを許可され得る。これは、セキュリティドメインの変更を制限する通常の構成パラメータがセキュアドメインから構成されると予想されるため、直感に反するように思われるかもしれない。しかしながら、無効化ドメイン遷移障害のトリガは、セキュリティではなく性能を改善するための機構として使用され得るので、ドメイン遷移無効化構成パラメータが低セキュアドメインから変更されることは許容可能であり得る。これは、適切なセキュアリソースが確立されていない場合にセキュリティドメインの変更を可能にするためのスレッドの利点がほとんどない可能性があるという事実を利用しており、その場合、セキュアドメインに切り替えることを試行した場合、そのようなリソースは確立されていないため、処理は失敗する。

【0067】

上述したように、処理回路は、スレッドモード及びハンドラモードを有することができ、これらのモードは、セキュアドメインにおいて処理回路がスレッドモード又はハンドラモードのいずれかで動作することができ、同様に低セキュアドメインにおいて処理回路がスレッドモード又はハンドラモードのいずれかで動作することができるという点で、セキュリティドメインと直交していてもよい。上述した処理回路の少なくとも1つのモードは、スレッドモードであってもよい。従って、スレッドモードにあるとき、ドメイン遷移無効化構成パラメータは、セキュリティドメインの遷移を選択的に無効にすることができる。しかしながら、少なくとも1つのモードはハンドラモードを除外することができ、その結果、ハンドラモードにおいて、処理回路は、ドメイン遷移がドメイン遷移無効化構成パラメータとは無関係に許可されるかどうかを判定することができる。従って、ドメイン遷移無効化構成パラメータが、セキュアドメインと低セキュアドメインとの間の遷移がスレッドモードで無効にされることを示す場合でも、そのようなセキュア／低セキュアドメイン遷移は依然としてハンドラモードで許可され得る。セキュアドメイン及び低セキュアドメインを有するシステムでは、ハンドラモードで実行されるハンドラコード（例えば、オペレーティングシステムに関連付けられ得る）は、セキュア処理及び低セキュア処理の両方を含む可能性があり、両方のドメインからのリソースを必要とするため、ハンドラモードでセキュア／低セキュアドメイン遷移を無効にする価値は少ない。また、ハンドラモードで実行されるハンドラコードは、スレッドモードコードが無効化ドメイン遷移障害をトリガした場合に他のドメインのリソースをステップインして緩慢に構成するコードであってもよく、そのため、スレッドモードがドメイン遷移を無効化した場合でも、ハンドラモードが両方のドメインで動作し続けることを可能にすることが有用であり得る。

【0068】

前述したように、命令デコーダは、関数戻りアドレスを保存し、分岐ターゲットアドレスに分岐することに加えて、処理回路の現在のモードをハンドラモードからスレッドモードに切り替えるインターモーダル呼び出し分岐命令をサポートすることができる。そのような命令に応答してドメイン遷移無効化構成パラメータを設定して、ドメイン遷移が有効か無効かを指定することも有用であり得る。インターモーダル呼び出し分岐命令の第1及び第2の変形のうちの少なくとも1つを定義することができ、第1の変形は、ドメイン遷移が少なくとも1つのモードで無効にされることを指定するようにドメイン遷移無効化構成パラメータを設定させ、第2の変形は、ドメイン遷移が少なくとも1つのモードで有効にされることを指定するようにパラメータを設定させる。両方の変形をサポートするシステムは、ハンドラコードが、例外ハンドラコードの非特権セクションを呼び出す同じインターモーダル呼び出し分岐命令において、両方のドメインで動作するように非特権セクション用のリソースを既に構成しているかどうかをシグナリングすることを可能にするので、特に有用であり得る。

【0069】

同様に、前述したようなインターモーダル関数戻り命令の場合、これはまた、設定されるドメイン遷移無効化構成パラメータをトリガすることができる。関数戻りの場合、ドメイン遷移無効化構成パラメータが、ドメイン遷移が少なくとも1つのモードで有効にされることを指定する値に設定されることが特に有用であり得る。これは、上述したように

、インターモーダル関数呼び出し分岐及びインターモーダル関数戻りの1つのユースケースは、例外ハンドラがスレッドモードで非特権化される関数を呼び出すことを可能にすることであり得、このコードはサードパーティプロバイダによって提供され得るため、有用である。ハンドラモードのより特権的なコードは、コードの非特権セクションの前後に実行される「ラップ」コードを実行することができる。インターモーダル関数戻り命令に 응답してドメイン遷移無効化構成パラメータをデフォルト値に設定することにより、これは、非特権セクションが実際にセキュアドメインと低セキュアドメインの両方を使用したかどうかにかかわらず、非特権セクションの最後のラップコードが常に一貫した既知の状態で行われることを意味する。また、「無効化」インジケーションではなく、インターモーダル関数戻りに続くデフォルト値セットとして「有効化」インジケーションを使用することによって、これは、（スレッドモードのドメイン遷移が許可されることを期待する）ドメイン遷移無効化構成パラメータをサポートしないシステムのために書き込まれた後続のレガシーコードが、追加の障害をトリガすることなく依然として機能することができることを意味する。

10

【0070】

また、インターモーダル関数戻り命令に 응답して、並びに少なくとも1つのモードでドメイン遷移が有効にされたことを示すためにドメイン遷移無効化構成パラメータを更新して、処理回路は、インターモーダル関数戻り命令を実行する前に、ドメイン遷移が少なくとも1つのモードで有効にされたか無効にされたかを指定したドメイン遷移無効化構成パラメータのインジケーションを記録することができる。ドメイン遷移無効化構成パラメータの前の状態のこのインジケーションは、後続のハンドラコードが、他のセキュリティ状態（MPU状態又はスタックポインタなど）に属する構成レジスタ内の情報が、インターモーダル呼び出しとインターモーダル関数戻りとの間に実行されたコードによって変更された可能性があるかどうかを判定することを可能にするのに役立ち、その結果、他のセキュリティ状態の構成レジスタが変更されていない場合、性能を改善するために前の構成を復元するための任意の動作を省略することができる。

20

【0071】

ドメイン遷移無効化構成パラメータのチェックに続いて、第1のセキュリティドメインから第2のセキュリティドメインへの遷移を試行するときに無効化ドメイン遷移障害がシグナリングされる場合、例外制御回路は、第1のセキュリティドメイン内の例外ハンドラを処理するように処理回路を制御することができる。従って、障害が低セキュアドメインからセキュアドメインに切り替える試行によってトリガされた場合、障害は低セキュアドメインで処理され、試行がセキュアドメインから低セキュアドメインに遷移しようとした場合、障害はセキュアドメインで処理される。これは、典型的には、対応するスレッドがドメイン切り替え前に動作していたドメインは、スレッドが最初に初期化されたドメインであり、従ってそのスレッドを管理する管理プロセスに関連付けられたドメインであるため、有用である。例えば、セキュアドメインで作成されたセキュアスレッドはセキュアオペレーティングシステムによって管理されてもよく、低セキュアドメインで作成された低セキュアスレッドは低セキュアオペレーティングシステムによって管理されてもよく、そのため、無効化ドメイン遷移障害が、無効化ドメイン遷移を試行したスレッドに関連付けられた管理プロセスに向けられることが望ましい場合がある。障害は第1のセキュリティドメインで処理され得るが、障害に 응답して例外ハンドラによって実行されるリソースの構成は、第2のセキュリティドメインに関連付けられたリソースの構成であり得る。無効化ドメイン遷移障害ターゲットのドメインは、リソースを緩慢な構成にする必要がある状態とは異なるため、低セキュア状態で動作するプログラムコードとセキュア状態との間の密接な調整がハンドラモードで必要とされ得る。従って、無効化ドメイン遷移チェックが実行される少なくとも1つのモードからハンドラモードを除外することが有利であり得る。

30

40

【0072】

例外条件の発生時に、例外制御回路は、ドメイン遷移無効化構成パラメータを更新し

50

て、セキュア／低セキュアドメイン遷移が少なくとも1つのモードで有効にされることを指定することができる。ここでも、これは、例外ハンドラに安全な既知のデフォルト状態を提供し、その結果、例外ハンドラは、例外が取られる時点で処理されているバックグラウンドコードに関連付けられたドメイン遷移無効化構成パラメータの値にかかわらず、一貫した方法で機能することができる。また、例外条件に応答して遷移を有効にすることをデフォルトにすることによって、これは、ドメイン遷移無効化機能をサポートしなかったシステムのために書き込まれた可能性がある例外ハンドラ内のレガシーコードとの互換性を改善する。

【0073】

例外条件が発生すると、例外制御回路は何らかの状態保存をトリガして、例外条件が発生する前に実行された処理に関連するアーキテクチャ状態を保存することができる。例外制御回路のハードウェアが以前の処理に関連する全てのアーキテクチャ状態を保存する必要はなく、場合によっては、ハードウェアはアーキテクチャ状態のサブセットのみを保存することができる。他の状態が存在するレジスタが例外ハンドラによって上書きされる場合、ハードウェアに保存されていない他のアーキテクチャ状態を保存することは、後続の例外ハンドラの責任であり得る。

【0074】

例外に応答してそのような状態保存を実行するとき、状態保存に保存されたアーキテクチャ状態は、ドメイン遷移無効化構成パラメータを含むことができる。従って、ドメイン遷移無効化構成パラメータは、例外が発生したときにハードウェアによって自動的に保存することができ、その結果、例外処理が完了し、処理が例外が発生する前に実行されていた前の処理に戻ると、ドメイン遷移無効化構成パラメータを制御記憶位置に復元でき、その結果、バックグラウンドコードの後続の処理は、例外が発生しなかった場合と同じように機能することができる。

【0075】

いくつかの実装形態では、ドメイン遷移無効化構成パラメータは、例外条件に応答して、スタックデータ構造に保存されるスタックフレームの一部として保存することができ、スタックフレームはまた、例えば汎用レジスタコンテンツ又は例外戻りアドレスなどの他のアーキテクチャ状態も含む。しかしながら、いくつかの実装形態では、ドメイン遷移無効化構成パラメータを符号化するためのスタックフレーム内にスペア空間がほとんどない場合があり、他の目的のために任意の予備の符号化空間を保存することが好ましい場合がある。

【0076】

従って、いくつかの例では、状態保存中に、例外制御回路は、例外戻りアドレスとアーキテクチャ状態の第1のサブセットを含む第1のスタックフレームをスタックデータ構造に保存することができるが、ドメイン遷移無効化構成パラメータを示す値は、第1のスタックフレームに含まれる代わりに、所定のレジスタに保存されてもよい（例えば、所定のレジスタは、リンクレジスタであってもよい）。これにより、スタックフレーム内で余分なビットを消費する必要がなくなる。ドメイン遷移無効化構成パラメータを示す保存値は、ドメイン遷移無効化構成パラメータ自体と同一の符号化を有する必要はなく、例外が発生したときにドメイン遷移無効化構成パラメータがどのような値を有していたかの判定を可能にする任意の値をスタックに保存することができることに留意されたい。

【0077】

しかしながら、所定のレジスタは、例外条件に応答して実行される例外ハンドラにアクセス可能であってもよい。少なくとも、セキュアドメインから低セキュアドメインへの遷移を引き起こす例外については、例外の後に実行する低セキュアコードによる、例外の前に実行されている処理でセキュリティドメイン遷移が無効にされたかどうかを示す値の変更を許可するのは望ましくない場合がある。

【0078】

このシナリオから保護するために、少なくともセキュアドメインから低セキュアドメ

10

20

30

40

50

インへの遷移をトリガする例外条件に応答して、例外制御回路は、上述の第1のスタックフレームだけでなく、アーキテクチャ状態の第2のサブセットと、ドメイン遷移無効化構成パラメータを示す値をチェックするためのクロスチェック情報とを含む第2のスタックフレームもスタックデータ構造に保存してもよい。少なくとも低セキュアドメインからセキュアドメインへの例外戻り遷移において、例外制御回路は、ドメイン遷移無効化構成パラメータを示す値と、第2のスタックフレーム内のクロスチェック情報とを比較してもよく、不一致が検出された場合に、障害のシグナリングをトリガしてもよい。ドメイン遷移無効化構成パラメータを示す値は、初期例外に応答して所定のレジスタに保存されている可能性があるが、例外戻り遷移を実行する時点で、ドメイン遷移無効化構成パラメータは、もはや所定のレジスタ内になくてもよく、メモリ内のスタックデータ構造からロードされた値内、又は例外戻りをトリガするために分岐されたダミー実行不可能アドレス内のフィールド内にあってもよいことに留意されたい。第2のスタックフレームにクロスチェック情報を提供することにより、セキュリティ違反を防止するために、低セキュア例外ハンドラによる、所定のレジスタに初期に配置されたドメイン遷移無効化構成パラメータの保存値の変更の試行が検出されることを可能にすることができる。この手法はより複雑に見えるかもしれないが、第2のスタックフレームは、低セキュア例外ハンドラから隠すためにメモリに保存された追加の状態を記録することができ、全ての例外エントリに保存された状態のサブセットを提供する第1のスタックフレームよりも、追加情報のためのより多くの空き容量を有することができる。従って、第2のスタックフレームにクロスチェック情報を記録することは、第1のスタックフレームのみが必要とされる例外については、（この場合は所定のレジスタを信頼することが安全であるため）ドメイン遷移無効化構成パラメータ内に保持するための追加のスタック状態を保存するメモリコスト及び割り込み待ち時間を被る必要がないが、第2のスタックフレームを保存する必要があるセキュアから低セキュア例外遷移では、セキュリティを高めるためにスタックにクロスチェックを保存することができることを意味する。

【0079】

処理回路がセキュアドメインと低セキュアドメインとの間のドメイン遷移の実行を許可し、ドメイン遷移無効化構成パラメータのチェックが、セキュリティドメイン間の遷移にどのルートが使用されるかに応じてわずかに異なる方法で実行され得る、いくつかの方法があり得る。

【0080】

前述のように、ドメインを切り替える1つの方法は、セキュアドメインから低セキュアドメインへの遷移を要求するドメイン間呼び出し分岐命令を使用することであり、これに応答して、命令デコーダは、ドメイン間呼び出し分岐動作を実行するように処理回路を制御することができ、ドメイン間呼び出し分岐動作は、関数戻りアドレスを関数戻りアドレス記憶位置（例えば、メモリ内のスタックデータ構造）に保存することと、ダミー関数戻りアドレスを所定のレジスタ（例えば、リンクレジスタ）に保存することと、処理回路を低セキュアドメインに切り替えることと、ドメイン間呼び出し分岐命令によって指定された分岐ターゲットアドレスによって識別される命令に分岐することを含む。セキュアドメインから低セキュアドメインへの直接関数呼び出しを許可することにより、ドメイン遷移を管理するために例外ハンドラ又はモニタコードを呼び出す必要がなくなり、性能が向上する。

【0081】

ドメイン間呼び出し分岐命令の場合、ドメイン遷移無効化構成パラメータをチェックすることができ、ドメイン遷移無効化構成パラメータが、ドメイン間遷移が少なくとも1つのモードで有効にされることを指定すると、ドメイン間呼び出し分岐動作が実行され得る。また、ドメイン遷移無効化構成パラメータの値にかかわらず、処理回路が現在少なくとも1つのモード以外のモードにある場合（例えば、現在のモードがハンドラモードである場合）、ドメイン間呼び出し分岐動作を実行することができる。逆に、少なくとも1つのモード（例えば、スレッドモード）において、ドメイン遷移無効化構成パラメータが、

その少なくとも1つのモードにおいてドメイン遷移が無効にされることを指定する場合、無効化ドメイン遷移障害のシグナリングがトリガされ得る。

【0082】

ドメイン間を遷移する別の方法は、セキュアドメインから低セキュアドメインへの遷移を要求し、ドメイン間戻り分岐命令によって指定された分岐ターゲットアドレスによって識別される命令に分岐するドメイン間戻り分岐命令を実行しようと試行することであり得る。命令を実行しようとする試行は、ドメイン遷移無効化構成パラメータがチェックされるのをトリガしてもよく、ドメイン遷移無効化構成パラメータが、ドメイン間遷移が少なくとも1つのモードで有効にされることを指定すると、ドメイン間戻り分岐動作が実行され得る。また、ドメイン遷移無効化構成パラメータの値にかかわらず、処理回路が現在少なくとも1つのモード以外のモードにある場合（例えば、現在のモードがハンドラモードである場合）、ドメイン間戻り分岐動作を実行することができる。逆に、少なくとも1つのモード（例えば、スレッドモード）において、ドメイン遷移無効化構成パラメータが、その少なくとも1つのモードにおいてドメイン遷移が無効にされることを指定する場合、無効化ドメイン遷移障害のシグナリングがトリガされ得る。

10

【0083】

ドメイン間を遷移する別の方法は、前述したように、以前の関数呼び出し分岐命令によってセットアップされたダミー関数戻りアドレスに分岐しようと試行することであり得る。セキュアドメインから低セキュアドメインへの遷移を要求するドメイン間呼び出し分岐命令が実行されると、関数呼び出しのための関数戻りアドレスは、低セキュアコードから見えることから保護するためにスタックデータ構造に保存されてもよく、ダミー関数戻りアドレスは、低セキュアコードから見えるリンクレジスタに保存されてもよい。ダミー関数戻りアドレスは、対応する関数戻りにおいて、セキュアドメインに戻るためにドメイン遷移が実行されるべきであることを示す情報を含むことができる。従って、後でダミー関数戻りアドレスに分岐しようとする試行があり、ダミー関数戻りアドレスがドメイン間関数戻りを示すとき、処理回路が現在少なくとも1つのモードにあり、ドメイン遷移無効化構成パラメータが、その少なくとも1つのモードでドメイン遷移を無効にすべきであると指定する場合、処理回路は、無効化ドメイン遷移障害のシグナリングをトリガすることができる。

20

【0084】

低ドメインからセキュアドメインに遷移する別の方法は、命令が低セキュアドメイン領域からフェッチされる場合であってもよい。メモリアクセスチェック回路によって維持されるセキュリティ属性データは、アドレス領域を低セキュアドメイン領域又はセキュアドメイン領域のいずれかとして定義することができる。メモリのセキュアドメイン領域に記憶された命令は、セキュアドメインからのみ実行され得る。現在低セキュアドメインにあるときに、セキュアドメイン領域内のアドレスを有する命令をフェッチしようと試行すると、セキュアドメインへの遷移が許可され得るかどうかをチェックするためにいくつかのセキュリティチェックを実行する必要がある。例えば、チェックは、セキュアドメイン領域内のアドレスにおける命令が、セキュアドメインへの有効なエントリポイントをマークする特定のタイプのセキュアゲートウェイ命令であるかどうかをチェックすることを含むことができる。場合によっては、セキュアドメイン領域内のアドレスにおける命令がセキュアゲートウェイ命令である場合、セキュアゲートウェイ命令の復号は、低セキュアドメインからセキュアドメインへのセキュリティドメインの変更をトリガすることができる。他の場合には、命令をフェッチしようとする試行が、低セキュアドメイン呼び出し可能セキュアドメインメモリであってもよい特定のタイプのセキュアドメインメモリをターゲットとする場合にのみ、低セキュアドメインからセキュアドメインへの遷移が許可されてもよい。いくつかの実装形態は、セキュアゲートウェイ命令チェックと低セキュアドメイン呼び出し可能セキュアドメインメモリチェックの両方を実行することができる。

30

40

【0085】

しかしながら、前述のように、低セキュアドメインからセキュアドメインへの遷移の

50

ために、セキュリティチェックは、単一のセキュリティドメイン内に留まることが予想されたスレッドに対して構成されていない可能性があるリソースに依存する場合がある（例えば、チェックは、セキュア領域からフェッチされた命令がフェッチ及び復号され得るように設定されているセキュア領域のメモリ保護ユニット属性に依存し得る）。従って、低セキュアドメインにのみ存在すると予想されるスレッドが状態の切り替えを試行する場合、セキュリティチェック自体が失敗することがある。これらのセキュリティチェックが失敗しないようにするために、可能な限り早くドメイン遷移無効化構成パラメータのチェックを実行することが望ましい場合がある。

【0086】

従って、このチェックは、その命令がフェッチされて復号されるまで待つのではなく、セキュアアドレス領域内のアドレスを有する命令をフェッチする試行に応答して実行されてもよい。ドメイン遷移無効化構成パラメータのチェックをパイプラインの初期段階で実行することにより、構成されたリソースの欠如に起因してセキュリティチェックが失敗する可能性を低減する。このチェックが行われた時点では、フェッチされている命令が実際にセキュアドメインへの遷移をトリガしたであろう命令であるかどうかはまだ分からないことがある（例えば、フェッチされた命令がセキュアゲートウェイ命令でなかった場合、それはセキュアドメインへのエントリをトリガしていない可能性がある）。従って、ドメイン遷移無効化構成パラメータのチェックは、命令がセキュアアドレス領域内のアドレスからフェッチされ、現在のドメインが低セキュアドメインである場合に、フェッチされた命令のタイプに関係なくチェックされ得るという点で保守的であり得る。いくつかの実装形態では、パイプライン内の様々なチェックの位置は異なり得るが、最初にドメイン遷移無効化チェックを実行する同じ効果は、パイプライン内で後でこのチェックを実行するが、結果として生じるドメイン遷移無効化障害のシグナリングを他の障害のシグナリングよりも優先することによって達成できることが理解されよう。

【0087】

前述のインターモダル呼び出し分岐命令に関して、ドメイン遷移無効化構成パラメータは、ホストデータ処理装置上で実行されるシミュレーションコンピュータプログラムが、上述のようにドメイン遷移無効化構成パラメータと、パラメータを設定及び使用するための関連ロジックとを記憶する制御記憶位置の特徴を有する実際のハードウェアデバイス上で、ターゲットプログラムコードが実行された場合に期待されるものと同等の命令実行環境をシミュレートするシミュレーション環境で使うことができる。従って、コンピュータプログラムは、処理プログラムロジックと、メモリアクセスチェックプログラムロジックと、処理回路、メモリアクセスチェック回路、及び上述のように制御記憶位置内のドメイン遷移無効化構成パラメータを維持するための回路の機能をエミュレートする構成パラメータ設定プログラムロジックと、を有することができる。

【0088】

具体例の説明

図1は、メモリシステム6からフェッチされた命令に応答してデータ処理を実行するための処理回路4を含むデータ処理システム2の一例を概略的に示す。メモリシステム6は、主メモリに加えてキャッシュ（例えば、1つ以上のレベルのデータキャッシュ及び／又は命令キャッシュ）を含んでもよい。処理回路4はいくつかのパイプライン段を含む処理パイプラインを含み、パイプライン段は、例えば、メモリシステム6から実行される命令をフェッチするためのフェッチ段8と、データ処理を実行するために残りのパイプライン段を制御するための制御信号を生成するためにフェッチされた命令を復号するための復号段10と、データ処理動作を実行するために復号された命令を実行するための実行段12と、を含む。レジスタ14は、パイプライン4によって処理される命令の入力オペランドを記憶するために設けられる。レジスタ14は、整数値を記憶するための汎用（整数）レジスタ16と、処理回路4による命令の実行及び例外処理、セキュリティチェックなどの他の動作を制御するための制御パラメータを記憶するための制御レジスタ20と、を含むいくつかのタイプのレジスタを含む。例えば、浮動小数点値を記憶するための浮動小数

点レジスタ、又は複数の独立したデータ値を含むベクトルオペランドを記憶するためのベクトルレジスタなど、他のタイプのレジスタも提供され得る。

【0089】

図1に示すパイプライン段は簡略化された表現であり、レジスタリネーミングを実行するためのリネーム段、実行を待機する命令を待ち行列に入れ、それらの必要なオペランドが利用可能なときに実行のためにそれらを発行する発行段、及び命令のコミットメントを処理し、結果をレジスタ14に書き戻すためのライトバック段などの、他のタイプのパイプライン段も提供され得ることが理解されよう。パイプラインは、インオーダーパイプライン又はアウトオブオーダーパイプラインであり得る。

【0090】

信頼性、可用性、及び保守性(RAS)エラー検出／訂正回路29は、処理システムの動作においてランダムに生じるハードウェア障害を検出又は訂正するためのエラー検出又は訂正動作を実行するために設けられる。例えば、記憶されたビットの状態又は処理ロジック経路上の信号を0と1との間で反転させる粒子衝突により、ハードウェア障害が発生する可能性がある。また、ハードウェア障害は、経時的な電子回路の物理的劣化に起因して生じる可能性があり、例えば、エレクトロマイグレーションは、最終的に、記憶素子に書き込まれたビット値にかかわらず、記憶素子が0又は1で固定する可能性がある。このような障害を検出するために、RASは、もう1つのエラー検出／訂正機構を含むことができる。例えば、1つの機構は、処理回路4と同じ動作を冗長に実行するための冗長処理ロジック、及び主処理と冗長処理の結果を比較して障害を検出するための比較ロジックを提供することであり得る。別の機構は、メモリに書き込まれたデータに関連して記憶されたエラー検出コード又はエラー訂正コードの設定及びチェックを管理することであり得、メモリからデータを読み出すと、対応するエラー検出／訂正コードを再計算し、記憶されたエラー検出／訂正コードと比較してエラーを検出する(可能であれば、修正する)ことができる。RASエラーが検出された場合、可能であればエラーが訂正され、処理は前進して継続する(例えば、これは、トリプルコア・ロックステップ技術又はエラー訂正コードの使用で可能であり得る)。あるいは、エラーが訂正可能でない場合、処理のスレッドを終了させて再開させることがある障害がシグナリングされ得る。エラー検出／訂正及び回復のための任意の既知の技術を使用することが理解されよう。

【0091】

システム2はまた、メモリアドレス空間の様々な領域に対して指定された属性データに基づいて、メモリシステム6へのアクセスが許可されているかどうかをチェックするためのメモリアクセスチェック回路22を含む。メモリアクセスチェック回路は、メモリアドレス空間の各それぞれの領域に関連付けられたセキュリティドメインを定義するセキュリティドメイン定義データを記憶するためのセキュリティ属性ユニット(SAU)24を含む。セキュリティ属性データに基づいて、セキュリティ属性ユニット24は、処理回路の動作の現在のセキュリティドメインと、メモリアクセスのターゲットアドレスを含む領域に関連付けられたセキュリティドメインとに依存して、メモリアクセスが許可されているかどうかをチェックすることができる。他の実施形態では、SAUは、セキュリティドメイン定義データを直接記憶するのではなく、代わりに、メモリアクセスチェックを実行するために、他の場所に記憶されたセキュリティドメイン定義データにアクセスすることができる。いくつかのシステムでは、セキュリティドメイン定義データは、メモリシステム6に記憶されてもよく、システム2内の他の場所の構成レジスタであってもよい。処理回路4は、現在実行されている命令のアドレスに関連付けられたセキュリティドメインに一般に対応することができる動作の現在のセキュリティドメインで動作することができる(ただし、いくつかの例外があり得、例えばドメイン間の遷移を処理する場合)。命令が、SAU24によってあるドメイン内にあると指定された領域内のアドレスから、異なるドメインに関連付けられた領域内のアドレスに分岐すると、これは、処理回路4が動作している現在のセキュリティドメインの遷移をトリガすることができる。他の実施形態では、そのような分岐は、現在のセキュリティドメインの遷移を直接トリガしなくてもよく、

10

20

30

40

50

代わりに、ゲートウェイ命令が存在するかどうかなどの追加のセキュリティチェックをトリガしてもよく、これらの実施形態では、現在のセキュリティドメインの変更をトリガするのはゲートウェイ命令自体であってもよい。一般に、セキュアドメインで動作している間、処理回路は、セキュアドメインと低セキュアドメインの両方に関連するメモリ領域内のデータにアクセスすることができ、低セキュアドメインで動作しているとき、処理回路は、低セキュアドメインに関連する領域内のデータにアクセスすることができるが、SAU24がセキュアドメインに関連付けられていると指定するアドレス空間の領域内のデータにアクセスすることは許可されない場合がある。これにより、低セキュアドメインで動作するコードからの不正アクセスに対する機密データの保護が可能になる。

【0092】

また、メモリアクセスチェック回路は、メモリシステム6へのメモリアクセスがアクセス許可を満たすかどうかをチェックするメモリ保護ユニット(MPU)26を含むことができ、アクセス許可は、例えば、処理回路4のどの特権レベルがメモリの所与の領域にアクセスすることを許可されるかを指定することができ、又はアドレス空間のメモリ領域が読み出し及び書き込み動作の両方によってアクセス可能であるかどうか、又は書き込みが禁止される読み出し専用領域であるかどうかを指定することができる。MPU26によって使用されるアクセス許可は、例えば、低特権プロセス(アプリケーションなど)がアクセスを許可されるメモリの領域、及び(読み出しのみ又は読み出し/書き込み、及びメモリが実行可能であるかどうか)方法を制御するために、より特権的なプロセス(ハイパーバイザ又はオペレーティングシステムなど)によって指定され得る。いくつかの実装形態では、MPU26はまた、どのメモリ領域に特権プロセスがアクセスすることを許可されているか、及びその方法(読み出しのみ又は読み出し/書き込み、及びメモリが実行可能であるかどうか)のチェックを適用することができる。MPU26は、低特権アクセスと特権アクセスに対して異なる許可を指定することを可能にすることができる。MPU26によって提供される許可は、SAU24によって提供される許可と直交してもよく、その結果、所与のメモリアクセスが許可されるためには、MPU26及びSAU24の両方に対して定義されたアクセス許可に基づいてチェックに合格する必要がある。MPU26は、各々がセキュリティドメインのうちの1つに関連付けられた別個のセキュアMPU26-S及び低セキュアMPU26-NSを含むことができ、その結果、現在のドメインがセキュアドメインであるか低セキュアドメインであるかに応じて、メモリの所与の領域に対して異なるメモリアクセス許可を指定することができる(例えば、領域は、低セキュアドメインでは読み出しのみ可能であるが、セキュアドメインでは読み出し可能及び書き込み可能の両方が可能である)。

【0093】

従って、図2に示すように、処理回路4は、少なくともセキュアドメイン(S)及び低セキュアドメイン(非セキュアドメイン、NSとも呼ばれる)を含む複数のセキュリティドメインのうちの1つでデータ処理を実行することをサポートすることができる。図2は、2つのセキュリティドメインのみを有するシステムを示しているが、異なるレベルのセキュリティに関連付けられた3つ以上のドメインを提供することが可能であり、その場合、示されているSドメイン及びNSドメインは、提供された3つ以上のドメインのうちの任意の2つであってもよい。

【0094】

また、図2に示すように、所与のセキュリティドメイン内で、処理回路は、例外処理のためのハンドラモード(H)及びバックグラウンド処理のためのスレッドモード(T)を含むいくつかのモードのうちの1つでデータ処理を実行することができる。図1に示すように、例外エントリ遷移及び例外戻り遷移、並びにそのような遷移中のアーキテクチャ状態の保存/復元を含む例外処理動作を制御するために、例外制御回路28を設けることができる。例外処理を専用ハンドラモードHに分離することは、どのレジスタが処理回路によってアクセスされることを許可されるかの管理を単純化するのに有用であり得る(例えば、例外処理に使用されるいくつかのレジスタは、ハンドラモードHではアクセス可能

10

20

30

40

50

であってもよいが、スレッドモードTではアクセス不可能であってもよい)。一般に、ハンドラモードHで動作するとき、処理回路4はデフォルトで、メモリ及びレジスタへのアクセスが最小特権レベル以外の特定の特権レベルに従って制御されるように、より特権的な動作モードを有すると仮定することができ、スレッドモードTでは、処理回路は、制御レジスタ20に記憶された他のアーキテクチャ状態に応じて、いくつかの異なる特権レベルのうちの1つを有することができる。

【0095】

従って、図2に示すように、セキュリティドメインと処理回路4が動作しているモードとの組み合わせは、処理回路4によって処理がどのように実行されるかの態様を決定することができる。図2は、以下を含むこれらのモードの4つの異なる組み合わせを示す。

●セキュアスレッドモード（セキュアドメインとスレッドモードとの組み合わせの省略表現）、

●低セキュアスレッドモード（低セキュアドメインとスレッドモードの組み合わせ）、

●セキュアハンドラモード（セキュアドメインとハンドラモードの組み合わせ）、

●低セキュアハンドラモード（低セキュアドメインとハンドラモードとの組み合わせ）。

【0096】

以下で説明するように、例外条件及び例外戻り条件に応答して実行されるアクションは、セキュリティドメインとモードのこれらのそれぞれの組み合わせの間で行われる特定の遷移に依存し得る。セキュア属性ユニット(SAU)24の属性は、セキュア及び低セキュアドメイン間の境界を監視する。セキュアMPU26-Sの属性を使用して、セキュアスレッドモードで動作する異なるスレッド間、及びこれらのスレッドとセキュアオペレーティングシステムとの間の境界を監視することができる。低セキュアMPU26-NSの属性を使用して、低セキュアスレッドモードで動作する異なるスレッド間、及びこれらのスレッドと低セキュアオペレーティングシステムとの間の境界を監視することができる。

【0097】

図3は、処理システムのレジスタ14のいくつかを示す。この表現は、提供され得る可能なレジスタの全てを示すわけではなく、多くの他のレジスタも提供され得ることが理解されよう。図3に示すように、レジスタ14は、汎用レジスタ16と、いくつかの制御レジスタ20を含む（制御レジスタの一部のみが図3に示されており、他のものも提供することができる）。この例では、R0～R15とラベル付けされた16個の汎用レジスタが設けられている。いくつかの実施形態では、汎用レジスタ16はまた、条件フラグ値及び現在のコンテキストに関連する他の情報を保持することができるプログラム状態レジスタ(XPSR)を含むことができる。

【0098】

汎用レジスタR0～R15のうち、レジスタR0～R12は、算術若しくはロジック命令のための入力オペランド、又はメモリシステム6にアクセスするためのロード／記憶命令のためのアドレスを導出するために使用されるオペランドなどの汎用オペランドを記憶するために使用される。汎用レジスタR13～R15をそのような汎用オペランドに使用することが可能であり得るが、それらは他の制御機能も果たす。

【0099】

レジスタR15は、処理回路4によって到達されたプログラムの現在点のインジケーションを提供するプログラムカウンタを記憶するプログラムカウンタ(PC)レジスタとして使用される。プログラムカウンタレジスタは、メモリシステムからどの命令をフェッチするかを決定するためにフェッチ段8によって使用され得る。

【0100】

レジスタR14は、関数に関連する処理が完了したときに、リンクレジスタ内のアドレスを使用して、関数呼び出し命令の後の次の命令に、プログラムフローを転送することができるように、関数が呼び出されたときに関数戻りアドレスを記憶するために使用されるリンクレジスタ(LR)として使用される。また、リンクレジスタは、例外の発生時に

10

20

30

40

50

使用されて、対応する例外戻り条件に遭遇したときに、アーキテクチャ状態のアンスタックを制御するための情報を提供する例外戻り値を記憶することができる。同様に、リンクレジスタはまた、インターモダル又はドメイン間関数呼び出しが実行されるときにダミー関数戻りアドレスを記憶するために使用されてもよい。これらについては、以下でより詳細に説明する。

【0101】

レジスタR13は、メモリ内のスタックデータ構造のアドレスを示すスタックポインタを提供するスタックポインタレジスタとして使用される。スタックデータ構造は、例外条件が発生したときにアーキテクチャ状態を保存し、例外戻り条件が発生したときにアーキテクチャ状態を復元するために使用することができる。図3に示すように、レジスタR13は、各々がレジスタ指定子R13を使用してアクセス可能ないくつかの異なる物理レジスタがハードウェアに提供されるようにバンクされ、R13が指定されたときに特定のバンクされたレジスタがどのように選択されるかは、処理回路の現在のセキュリティドメイン及びモード、並びに構成レジスタの値を含むいくつかの要因に依存し得る。

【0102】

例えば、バンクされたスタックポインタレジスタは、セキュア主スタックポインタレジスタ(MSP__S)、セキュアプロセススタックポインタレジスタ(PSP__S)、低セキュア主スタックポインタレジスタ(MSP__NS)、及び低セキュアプロセススタックポインタレジスタ(PSP__NS)を含むことができる。一般に、セキュアスタックポインタレジスタMSP__S及びPSP__Sは、セキュアドメインS内にあるときは処理回路4にアクセス可能であるが、低セキュアドメインNS内にあるときはアクセス不可能である。低セキュアスタックポインタレジスタMSP__NS及びPSP__NSは、低セキュアドメインでアクセス可能である。いくつかの実装形態は、バンクされたスタックポインタにアクセスする追加の方法を提供することができ、これは、低セキュアドメインに関連するスタックポインタ(MSP__NS及びPSP__NS)へのセキュアドメインアクセスを与えることができる。一般に、プロセススタックポインタPSP__S又はPSP__NSは、スレッドモードT内で使用されると予想され、主スタックポインタMSP__S又はMSP__NSは、ハンドラモードH内で使用されると予想される。ハンドラモード及びスレッドモード用の別々の主スタック及びプロセススタックをそれぞれ提供することは、例外ハンドラが主スタック上に残す任意のデータが、一般に、例外戻り後にスレッドモードで実行しているスレッドにアクセスできない可能性があることを意味するため、例外ハンドラコードの開発をより簡単にすることができる。これにより、例外ハンドラが主スタックデータ構造からそのデータの全てをクリアするための命令を含む必要がないため、例外ハンドラコードの開発を簡素化することができる。しかしながら、スレッドモードTのプロセスは、スレッドモードTのときに主スタックポインタ又はプロセススタックポインタが使用されるべきかどうかを制御する制御レジスタ32に記憶されたスタックポインタ選択値30と共に、主スタックポインタを使用することが可能である。従って、どのスタックポインタが使用されるべきかは、現在のモード(スレッド又はハンドラ)と現在のドメイン(セキュア又は低セキュア)の組み合わせだけでなく、スタックポインタ選択値30にも基づいて決定され得る。図3に示すように、スタックポインタ選択値30は、セキュリティ状態間でバンクされてもよく、その結果、セキュアドメイン及び低セキュアドメインは、スタックポインタ選択値30の別個のセキュアバージョン30-S及び低セキュアバージョン30-NSを使用して、セキュア及び低セキュアスレッドモードが主スタックポインタ又はプロセススタックポインタをそれぞれ使用するかどうかを独立して制御することができる。現在のセキュリティドメインは、どのバンクされたバージョン30-S、30-NSが読み出し／書き込みされるかを制御する。

【0103】

一般に、ハンドラモードHで動作する例外ハンドラ又はスレッドモードTで動作するスレッドの本体内で、セキュア又は非セキュアスタックポインタの選択は、処理回路4がコードを実行している現在のセキュリティドメインに依存し得る。例外エントリ及び戻り

10

20

30

40

50

の場合、状態のスタック及びアンスタックは、例外が発生する前に実行されていたバックグラウンドコードのセキュリティドメインに関連付けられたスタックポインタによって識別されるスタックから実行される。

【0104】

制御レジスタ32はまた、スレッドモードで実行されるコードが特権的であるか非特権的であるかを制御する特権制御値（`nPRIV`）31を含むことができる。特権制御値は、セキュアドメインと低セキュアドメインとで異なるように設定することができる（例えば、セキュアスレッドモードは非特権的であるが、低セキュアスレッドモードは特権的であってもよく、又はその逆も可能である）。あるいは、セキュアスレッドモードと低セキュアスレッドモードの両方が同じレベルの特権で動作することができる。特権制御値31は単なる一例であり、セキュア／低セキュアスレッドモードでコードに割り当てられた特権のレベルを制御するために制御レジスタに記憶されたアーキテクチャ状態を使用する他の方法があり得ることが理解されよう。図3の例では、特権制御値31は、セキュアドメイン及び低セキュアドメインでスレッドモードによって使用される特権レベルをそれぞれ制御するために、別個のセキュア特権制御値31-S及び低セキュア特権制御値31-NSが提供されるようにバンクされる。しかしながら、別の手法は、ドメイン遷移で切り替わる単一の制御ビットを提供することであり得る。

【0105】

図3に示すように、制御レジスタ32はまた、スレッドモードにおけるセキュアドメインと低セキュアドメインとの間の遷移が有効にされるか無効にされるかを制御するための、ドメイン遷移無効化構成パラメータとも呼ばれるスレッドモード遷移無効化（`thread mode interstating disable`、`TMID`）パラメータ34を含む。これについては、以下でより詳細に説明する。スタックポインタ選択値30及び特権値31とは異なり、`TMID`34値はセキュリティ状態間で共有される。`TMID`パラメータ34は、スレッドモードでセキュアドメインと低セキュアドメインとの間の遷移を無効にするために使用することができ、これは、以下で更に説明するように、ドメイン遷移を無効化すること（無効にされたドメイン遷移を実行する試行が行われた場合に障害をトリガする）が、スレッドを単一のドメインに限定することができることを意味するため、セキュア／低セキュアドメインのうちの1つのためのリソースの遅延構成を有効にするのに有用であり得る。

【0106】

図3に示すように、制御レジスタ20はまた、例外番号36を提供するレジスタを含む。例外番号36は、この制御レジスタに記憶される唯一の値ではないことがあり、他の状態は同じレジスタに記憶することができる。例外番号は、システムがハンドラモードにあるときに現在実行中の例外ハンドラに関連付けられた例外を識別する。システムがスレッドモードにあるとき、例外番号36は、現在処理されている例外がないことを示す特定のデフォルト値（例えば0）に設定される。従って、例外番号36は事実上、システムが現在スレッドモードにあるかハンドラモードにあるかのインジケーションでもあり、登録された例外番号レジスタ36が特別な値（例えば0）を指定する場合、システムはスレッドモードにあり、他の値が指定される場合、システムはハンドラモードにあり、特定の非ゼロ値は処理されている現在の例外を識別する。従って、システムが動作している現在のモードを示す別個のレジスタを必要としない場合がある。図3には示されていないが、現在のドメインがセキュアドメインであるか低セキュアドメインであるかを示す制御レジスタが存在してもよい。あるいは、処理されている命令のアドレスが、`SAU24`によって定義されたメモリのセキュア領域又は低セキュア領域にあるかどうかの判定から暗黙的であり得るため、そのようなレジスタは不要であり得る。

【0107】

図4は、メモリシステム6をアドレス指定するために使用されるメモリアドレス空間の一例を示す。メモリアドレス空間100は、セキュアドメインと低セキュアドメインの両方でアクセス可能な1つ以上の低セキュア領域102と、セキュアドメインでアクセス

10

20

30

40

50

可能であり且つ低セキュアドメインではアクセス不可能な1つ以上のセキュア領域104とを含むことができる。説明を容易にするために、図4は、1つの低セキュア領域102及び1つのセキュア領域104のみを示しているが、各タイプの複数の領域が存在し得、これらの領域は互いに散在していてもよく、従って、セキュア領域の全てが連続ブロックにある必要はなく、低セキュア領域の全てが連続ブロックにある必要はない。低セキュア領域は、低セキュアプログラムコード110、スタックデータ構造112（低セキュアスタックポインタMSP__NS及びPSP__NSを使用してアクセスされる）、及び低セキュアデータ114を含み得る。同様に、セキュア領域104は、セキュアプログラムコード122スタックデータ構造124（セキュアスタックポインタMSP__S及びPSP__Sを使用してアクセスされる）及びセキュアデータ126を含み得る。ここでも、図4に示すように、セキュア領域104又は低セキュア領域102内のプログラムコード、スタック及びデータを連続ブロックに編成する必要はなく、実際には、スタック及びプログラムコードは、他のデータが散在するいくつかの不連続ブロックに分散されてもよい。実行可能なアドレス範囲内のメモリの特定の割り当ては、処理システム上で実行されるソフトウェアによって選択されてもよく、ハードウェアによって固定されない。

【0108】

図4に示すように、メモリアドレス空間100は、この例では0xF0000000（16進数で表され、他の例では予約済み領域130として異なるアドレス範囲を選択できることが理解されよう）以上のアドレスを含む予約済み領域130を含む。予約済み領域130は、実行可能命令を提供することを許可されていないアドレス範囲を表す。これらの予約済みアドレスのいくつかは、以下で更に説明するように、関数戻り又は例外戻りを示すなど、特別な目的のために割り当てられてもよい。予約済みアドレス130のうちの1つから命令をフェッチしようと試行すると、命令フェッチ段8は障害をシグナリングすることができる。これらの予約済みアドレスは、これらのアドレスのうちの1つから命令をフェッチしようとする試行が、所与のイベントが処理されることをシグナリングするものとして解釈され得るように、アーキテクチャの将来のバージョンが特別な目的のために追加のアドレスを割り当てる機会を提供し得る。

【0109】

図5に示すように、予約済みアドレス範囲の一部は、特定のドメイン／モード遷移及び例外戻り動作のための関数戻りを示すために使用される、特別なダミー関数戻りアドレス140及びダミー例外戻りアドレス142を表すために割り当てられる。ダミー関数戻りアドレス140は、ビット144の最上位部分が特定の関数戻りプレフィックス値144に設定されている予約済み範囲130内のアドレスのグループのいずれかである。この例では、ダミー関数戻りアドレスが予約済み範囲130内にあるように、関数プレフィックスは0xF0以上の値を有する。ダミー関数戻りアドレスは、そのダミー関数戻りアドレスから命令をフェッチ又は実行しようとする試行する場合、関数戻り動作が実行されるべきであることをシグナリングするために使用される。いくつかの実装形態では、プログラムカウンタ（PC）を（分岐、又はPCをターゲットとするロードのような何らかの他の命令のいずれかによって）ダミー予約済みアドレスに設定しようとする試行すると、命令フェッチの代わりに関数戻り動作がトリガされる場合がある。ハードウェアを単純化するために、PCを設定できる命令のサブセット（関数戻りを実行するために通常使用されるもの）のみが実際に関数戻り動作をトリガすることができてよい。これは、リンクレジスタR14内のインターモダル又はドメイン間関数呼び出しのための実関数戻りアドレスを公開する必要性を回避するための機構を提供し、その結果、それは低セキュア又は低特権コードによってアクセスされ得ない。

【0110】

ダミー関数戻りアドレス140は、関数呼び出しのプロパティ及び／又は対応する関数戻りがどのように処理されるべきかに関する情報を示すために、インターモダル又はドメイン間関数呼び出しを実行する際に設定することができるいくつかの状態情報を含む。この例では、ダミー関数戻りアドレス140は、ダミー関数戻りアドレスが、ハンドラ

10

20

30

40

50

モードからスレッドモードへのインターモーダル関数呼び出し、又は同じモード内に留まった（ハンドラモードのままであるか、又はスレッドモードのままである）非インターモーダル呼び出しに応答して設定されたかどうかを示すインターモーダル呼び出しフラグ（M）146を含む。この例では、インターモーダル呼び出しフラグ146は、0の値を有する場合にインターモーダル呼び出しを示し、フラグ146が1である場合に対応する関数呼び出し中にモードが変更されないままであったことを示すように符号化される。インターモーダル呼び出しフラグ146は、現在のモードがスレッドモードからハンドラモードに戻されるべきか、又は変更されないままであるべきかを判定するために、関数戻り時に使用することができる。

【0111】

また、ダミー関数戻りアドレス140は、対応する関数呼び出しが低セキュアドメインから行われたかセキュアドメインから行われたかを示すセキュリティドメインインジケータ（S）148を含む。この例では、セキュリティドメインインジケータは、呼び出しがセキュアドメインから行われた場合は1に設定され、呼び出しが低セキュアドメインから行われた場合は0に設定されるように符号化される。ダミー関数戻りアドレス140に示されるフラグ146、148の特定の符号化は単なる一例であり、他の手法は異なる符号化を使用することができることが理解されよう。また、ダミー関数戻りアドレスは、図5に示されていない追加のステータスフラグを含むことができる。いくつかの実装形態では、低セキュアドメインから呼び出すことができるセキュア関数の開始時に、セキュアゲートウェイ命令が出される。このセキュアゲートウェイ命令は、処理回路に、低セキュアドメインからセキュアドメインへの遷移を実行させることができる。従って、後続の関数戻りは、その関数がセキュアドメインから呼び出されたかそれとも低セキュアドメインから呼び出されたかを判定することができ、従って、どのドメイン実行に戻るべきかを判定することができ、セキュアゲートウェイ命令が低セキュアドメインからセキュアドメインへの遷移を実行した場合、セキュアゲートウェイ命令は、リンクレジスタ内の戻りアドレスの最下位ビットをクリアすることができる。後述するように、ダミー関数戻りアドレスの最下位ビットにS148を配置し、0が低セキュアドメインを示すようにS148インジケータを符号化することは、上述したセキュアゲートウェイ命令動作と組み合わせて使用される場合、追加のセキュリティ保護を提供することができる。上述した実装形態では、S148の値0は低セキュア状態を示し、セキュアゲートウェイ命令は戻りアドレスの最下位ビットをクリアするが、S148フィールドの他の符号化によって所望の動作も達成できることが理解されよう。例えば、S148についての1の値が、低セキュア状態を示している場合、セキュアゲートウェイ命令は、戻りアドレスの最下位ビットを1に設定する。

【0112】

ダミー例外戻りアドレス142は、特定の例外エントリ遷移に応答してリンクレジスタに保存され、ダミー関数戻りアドレス140と同様に、実際の例外戻りアドレスをスタックに保存（その後にスタックから復元）して、例外後に実行される低セキュアプロセスから隠すことを可能にするための機構を提供する。ダミー例外戻りアドレス142は、ダミー例外戻りアドレス142の最上位ビット部分に例外戻りプレフィックス150を含み、これも0xF0以上の値を有して例外戻りアドレスが予約済み範囲130内にあることを保証する。例外戻りプレフィックス150は、ダミー関数／例外戻りアドレス140、142を互いに区別できるように、関数プレフィックス144とは異なる。

【0113】

ダミー例外戻りアドレス142は、例外が取られたモード（スレッド又はハンドラ）を示すモードフラグ152と、例外が取られたセキュリティドメイン（セキュア又は低セキュア）を示すセキュリティドメインフラグ（S）156とを含む。いくつかの実装形態では、例外が最初に取り込まれたセキュリティドメイン（セキュア又は低セキュア）を示す例外セキュリティドメインフラグ（ES）157もあり得る。また、例外戻り値はスレッドモード遷移有効化（thread mode interstating enabled

10

20

30

40

50

、TMIE) 値158を含み、これは、例外を取る前に、例外が取られたコンテキストにおいて、スレッドモードでドメイン遷移が無効又は有効にされたことをTMIDフラグ34が示したかどうかを示す。過去との互換性の理由から(158に対応するビットを1に設定すると予想されるレガシーコードがスレッドモードでドメイン遷移を有効にするものとしてTMIEフラグ158を表すように)、ダミー例外戻りアドレス内のTMIEフラグ158は、制御レジスタ32内のTMIDフラグ34と反対の符号化を有し、その結果、ダミー例外戻りアドレス142内のTMIEフラグ158を設定するときに現在のセキュリティドメインのTMIDフラグの値が反転する。TMIEフラグ158自体は、例外戻り動作を制御するために必要ではないが、TMIEフラグ158をダミー例外戻りアドレスに保存することは、コンテキスト保存／復元機能を効果的に提供し、その結果、例外から戻ると、戻されるプロセスのための制御レジスタ32内のTMIDフラグ34の値を、例外が取られる前の状態に復元することができる。例外から戻ると、再びTMIEフラグ158の値が反転され、制御レジスタ32内のTMIDフラグ34に書き込まれる。図5は、TMIEフラグ158の特定の符号化を示しているが、この符号化は必須ではなく、TMID34に復元される値を表す他の方法を使用することができる。

【0114】

なお、ダミー関数戻りアドレス140及びダミー例外戻りアドレス142に記録されるステータス情報146、148、152、156、157、158は、異なる値を取ることができ、これは、予約済み範囲130内に、ダミー関数戻りアドレス140として各々動作する複数の異なるアドレスと、ダミー例外戻りアドレス142として各々振る舞う複数の異なるアドレスとが存在することを意味する。従って、関数の呼び出し時又は例外の開始時にリンクレジスタR14に保存される特定のダミーアドレスは、ステータスフラグに符号化される情報に基づいて選択される。

【0115】

図6は、インターモーダル若しくはドメイン間関数呼び出しを実行する際、又は例外に入る際にメモリ内のスタックデータ構造に保存され得る異なるスタックフレームの例を示す。スタックフレームが保存される特定のスタックデータ構造は、関数呼び出し又は例外時の現在のモード及びセキュリティドメインに依存し、現在スレッドモードにある場合、選択されるスタックはまた、前述のように現在のドメインに関連付けられたスタックポインタ選択値30に依存する。

【0116】

図6の部分170に示すように、ドメイン間／インターモーダル関数呼び出し以外の関数呼び出し(モード又はドメインの遷移をトリガしない)の場合、スタックフレームをスタックに記憶する必要はなく、同じモード及びドメイン内の関数呼び出しの場合、実関数戻りアドレスをリンクレジスタR14に単に保存することができ、関数呼び出し後に実行されているコードから関数戻りアドレスを隠す必要はない。

【0117】

セキュアドメインから低セキュアドメインへのドメイン間関数呼び出し又はインターモーダル関数呼び出しの場合、関数戻りスタックフレーム172がスタックに保存される。セキュアドメインから低セキュアドメインへのドメイン間関数呼び出しの場合、スタックフレーム172は、セキュアスタックポインタのうちの1つによって示されるセキュアスタックに保存される。ドメイン間関数呼び出しがハンドラモードから行われる場合、これはセキュア主スタックポインタMSP__Sとなり、ドメイン間呼び出しがスレッドモードから行われる場合、スタックフレームはセキュア主スタック(MSP__S)と、ポインタがセキュアドメインのスタックポインタ選択値30-Sによって示されるセキュアプロセススタック(PSP__S)と、のうちの1つに保存される。インターモーダル関数呼び出しの場合、これらはデフォルトでハンドラモードから行われるため、現在のセキュリティドメインに関連付けられた主スタックポインタが使用される。

【0118】

図6に示すように、関数戻りスタックフレーム172は、呼び出された関数から戻る

ときに処理が戻るべき実関数戻りアドレスを表す戻りアドレス174を含む。実関数戻りアドレスは、関数呼び出し分岐命令のアドレスから順に続く次の命令のアドレスに設定することができる。また、関数戻りスタックフレームは、対応する関数戻りをどのように処理するかを決定するための情報を提供する状態データワード176 (FRETPSR) を含む。この例では、ハンドラモードからスレッドモードへのインターモダル関数呼び出しに回答して関数戻りスタックフレーム172が保存されたかどうかを示す、スレッドモード非特権ハンドラ (HDT) フラグ178を含む。例えば、HDTフラグは、関数呼び出しがインターモダル関数呼び出しではなく、そのためハンドラモードがスレッドモードに対して非特権化されなかった場合には0の値を有することができ、関数呼び出しがインターモダル関数呼び出しであった場合には1の値を有することができる。また、状態データワード176は、関数が呼び出された時点での例外番号レジスタ36の値に対応する例外番号値179を含む。これは、ハンドラモード又はスレッドモードから呼び出された関数に応じて関数戻りスタックフレームが保存されたかどうかを示す「ハンドラモードを示す値」として機能する。他の情報もまた、関数戻り状態ワード176内に、又は関数戻りスタックフレーム172の他の部分内に記憶することができることが理解されよう。

【0119】

例外スタックフレーム180、182の例は、図6の上部に示されている。例外に回答して、第1の（呼び出し側）スタックフレーム180及び第2の（被呼び出し側）スタックフレーム182の一方又は両方がスタックに保存される。性能上の理由から、例外エントリイベントに回答して保存する必要がある可能性があるレジスタを、「呼び出し側」レジスタ及び「被呼び出し側」レジスタと呼ばれる2つのグループに分割することが有用であり得る。呼び出し側レジスタは、ハードウェアに実装された例外制御回路28がスタックへの保存を担当するレジスタである。従って、例外ハンドラのソフトウェアは、呼び出し側レジスタをスタックデータ構造に保存するための命令を含む必要はない。

【0120】

図6に示すように、（呼び出し側スタックフレーム180に含まれる）呼び出し側レジスタは、汎用レジスタのサブセットを含むことができる。この例では、呼び出し側状態は、汎用レジスタR0、R1、R2、R3及びR12と、リンクレジスタLR (R14) と、例外の処理後に処理が戻される戻りアドレス181のインジケーション（例外が取られた時点でプログラムカウンタレジスタR15から導出された値に設定することができる）と、条件付き命令の結果を制御するための条件コード及び現在の処理状態に関する他の状態情報を提供することができるプログラム状態レジスタXPSR内の値に基づいて設定される例外戻りプログラム状態値 (ERETPSR) と、を含む。ERETPSRはまた、例外が取られる前に実行されていた処理の例外番号36を指定することができ、これは例外が取られる前に処理回路がスレッドモードで動作していた場合、これはデフォルト値0を有することができる。これは、どのレジスタ状態が呼び出し側レジスタ状態に含まれるかの単なる特定の一例であることが理解されよう。いくつかの例はまた、呼び出し側スタックフレーム内に浮動小数点レジスタ状態を含むことができ、浮動小数点レジスタ状態が含まれるべきかどうか（例えば、実行されているスレッドが浮動小数点状態を有効にしているかどうかに応じて）を構成できることがある。

【0121】

呼び出し側スタックフレーム180のみがスタックに保存される例外の場合、呼び出し側スタックフレーム180が例外イベントに回答して例外制御回路28によってスタックに保存された後、例外処理コードの実行が開始され、次に例外ハンドラソフトウェアは、実行中にこの状態を上書きすることが分かっている場合、追加の被呼び出し側レジスタから状態をスタックに保存することができる。しかしながら、例外ハンドラによるこの状態保存は、例外ハンドラコードで提供されるロード／記憶命令に回答して関連データをメモリに保存するように処理回路4のロード／保存ユニットを制御することによって実行され、これは、例外制御回路28がハードウェアでスタックを制御した場合よりも遅くなり得る。しかしながら、例外に入った後に実行されるソフトウェアが、どのレジスタを保存

する必要があるかに影響を及ぼすことができるように、呼び出し側レジスタ状態と被呼び出し側レジスタ状態との間の区分を可能にすることは有益であり得る。例外ハンドラが特定の被呼び出し側レジスタ（例えば汎用レジスタ R 7）に決してアクセスしない場合、例外ハンドラは、対応するアーキテクチャ状態を保存する必要がなく、例外制御回路 2 8 及びハードウェアでこの状態保存を行ったことに関連するエネルギー及び時間も回避された。例外ハンドラのためのコードのライター、又はそのようなコードのコンパイラは、ソフトウェアにおける追加の状態保存が必要とされる見込みを低減するために、追加の被呼び出し側レジスタを使用する前に呼び出し側レジスタを最初に使用するように奨励され得る。

【0122】

しかしながら、例外エントリが、セキュアドメインから低セキュアドメインへの遷移を引き起こし、元のバックグラウンド処理が例外前にセキュアドメインで実行される場合、呼び出し側レジスタのみが例外制御回路 2 8 によってメモリに保存されると、これは、被呼び出し側レジスタを、最近の例外の後に実行されるべき低セキュア例外ハンドラにとってレジスタバンク 1 4 内でアクセス可能なままにし得る。これは、被呼び出し側レジスタが、低セキュア例外ハンドラがアクセスすることを SAU 2 4 が通常ならば防ぐ、メモリアドレス空間のセキュア領域から導出された情報を含むことができるので、望ましくないことがある。

【0123】

従って、バックグラウンド処理がセキュアドメインで前に実施された、セキュアドメインから低セキュアドメインへのいくつかの遷移について、例外制御回路 2 8 はまた、呼び出し側レジスタの保存に加えて、関連のあるスタックデータ構造に被呼び出し側レジスタを保存し得、（呼び出し側レジスタと被呼び出し側レジスタとを含む）保存されていたレジスタのコンテンツをクリアし得る。従って、ハードウェアにおいて追加の状態保存を実施することによって、これは、被呼び出し側レジスタを通常保存する低セキュアソフトウェアが、例外が生じる前にこれらのレジスタに記憶された潜在的にセキュア情報へのアクセスを、この場合獲得することができないことを意味する。

【0124】

従って、いくつかの例外エントリ遷移について、呼び出し側スタックフレーム 1 8 0 を記憶することに加えて、第 2 の（被呼び出し側）スタックフレーム 1 8 2 も（例外制御回路 2 8 によるハードウェア制御下で、すなわち、明示的ロード／記憶命令が処理回路 4 によって実行されることを必要とすることなしに）保存され得る。図 6 では、被呼び出し側スタックフレームは呼び出し側スタックフレームとは別のスタックフレームとして参照されているが、他のものは呼び出し側スタックフレームの拡張として被呼び出し側スタックフレームを見なすことができ、1 8 0、1 8 2 の組み合わせは単一のスタックフレームとして見なされることに留意されたい。両方の手法は同等と見なすことができる。

【0125】

被呼び出し側スタックフレーム 1 8 2 は、呼び出し側スタックフレーム 1 8 0 に保存されなかったレジスタ R 4 ~ R 1 1 を含む追加の被呼び出し側状態を含む。任意選択的に、浮遊点コンテキストもまた、被呼び出し側スタックフレーム 1 8 2 に含まれ得る。ハードウェアによって保存された追加の状態はまた、所定の完全性署名 1 9 0 を含み得る。完全性署名 1 9 0 は、呼び出し側状態と被呼び出し側状態の両方が例外制御ハードウェア 2 8 によって保存されるとき、被呼び出し側スタックフレーム 1 8 2 中に含まれるが、呼び出し側例外スタックフレーム 1 8 0 からは消失している。完全性署名 1 9 0 は、命令をそこからフェッチしようと試行すると障害がシグナリングされるように、実行不可能アドレス 1 3 0 の予約済み範囲のうちの 1 つに対応する値を有する。完全性署名 1 9 0 は、有効なダミー関数戻りアドレス 1 4 0 又はダミー例外戻りアドレス 1 4 2 のいずれとも一致し得ない値を有し得る。完全性署名 1 9 0 は、ハードウェアがセキュアドメインと低セキュアドメインとの間の区分をエンフォースすることによって与えられるセキュリティ保護を免れることを試みるために処理回路上でしかけられ得るいくつかの形式の攻撃を検出するために使用され得る。攻撃のための 1 つの可能な手段は、攻撃者が、例外の処理に入るた

10

20

30

40

50

めに例外エントリ遷移を実施することを試行するが、次いで、後で、関数戻り遷移を実施することなど、例外エントリから復帰することが予想されるものへの異なるタイプの復帰遷移を捏造することであり得る。署名190は、関数戻りアドレス174を記憶するために関数戻りスタックフレーム172で使用される相対位置と同じスタックフレーム上の相対位置に配置される。従って、戻りアドレス174が完全性署名190に一致する（又は実行不可能な予約済みアドレス130のいずれかに対応する）ことが関数戻りにおいて検出された場合、それは、関数戻りを実施するために使用されているスタックフレームが、例外エントリイベントに応答してスタックに実際に保存されたことを意味するので、障害がトリガされ得、これは、攻撃が実施されているというインジケータであり得る。また、署名の欠如は、関数呼び出しに응答してメモリに保存されたスタックフレームに基づいて、例外戻りイベントが実行されていることを示す可能性があり、これはやはりセキュリティ違反の危険性があるため、被呼び出し側レジスタに関連付けられた追加のアーキテクチャ状態がスタック上に存在する例外から戻る際に、セキュリティチェックの一部は、完全性署名190に対応するスタック内の関連するオフセットの値が期待値を有するかどうかをチェックすることであり得る。

【0126】

図6に示すように、所定の署名値190の一部として、TMIEクロスチェック値192を所定の署名190のビットのうちの1つ内に記憶することができる（従って、効果的には、各々がアドレスの予約済み範囲130内にあるが、ダミー関数戻り値及びダミー例外戻り値に使用されるアドレスとは別個の所定の署名190の2つ以上の有効値が存在してもよい）。被呼び出し側スタックフレーム182が保存された例外を取る場合、TMIEクロスチェック値192は、リンクレジスタ内のダミー例外戻りアドレス142に書き込まれたTMIEフラグ158の状態を示す値に設定される（例えば、クロスチェック値192は、TMIEフラグ158に等しくすることもでき、TMIEフラグ158に対して反転することもでき、あるいは別の符号化を使用して、TMIEフラグの状態を表すこともできる）。被呼び出し側スタックフレーム182がスタックに保存される場合、これは、例外の前に実行された元のバックグラウンド処理がセキュアドメイン内にあった場合に、セキュアドメインから低セキュアドメインに例外が取られていることを示し、そのため、例外の後に実行された低セキュアドコードが、例外戻りの後に正しく機能するセキュアドコードを防止しようと試行して、ダミー例外戻りアドレス142内のTMIE値を変更する可能性があるリスクがあり得る。TMIEクロスチェック値192は、TMIE値158による改ざんを検出するためのクロスチェックを提供する。

【0127】

いくつかのシステム実装形態では、特定のタイプの例外ハンドラが非特権状態で実行されることが望ましい場合がある。これは、信頼できないサードパーティによって提供されたライブラリコードによって制御されるデバイスに関連付けられた特定のタイプの例外の場合、例えば、信頼できないサードパーティに提供された例外ハンドラが特定の状態情報を変更しないようにすることはできないため、信頼できないサードパーティに提供された例外ハンドラがハンドラモードに利用可能な完全な特権を有することを可能にすることは望ましくない可能性があるため、有用であり得る。しかしながら、図2に示すように、ハンドラモードはデフォルトで特権状態で動作するため、例外ハンドラが非特権状態で動作するためには、スレッドモードに戻る切り替えが必要になる。しかしながら、典型的な実施態様では、ハンドラモードからスレッドモードに切り替えるための機構は、例外戻りを介することになる。元の例外の実行優先度が低下するため、元の例外が例外戻りをトリガすることは一般に望ましくない（戻りアドレスを変更した後、戻りは前のバックグラウンド処理ではなく非特権ライブラリコードに進む）。このように実行優先度を下げることにより、例外処理ルーチンが正しく優先順位付けされるのを防ぐことができる。従って、元の例外を処理しているハンドラコードが例外戻りをトリガすることなくスレッドモードに切り替えることが可能であれば好ましい。

【0128】

典型的なシステムでは、ハンドラからスレッドモードに切り替えるための唯一の機構は例外戻りを介するものであるため、図7に示すように、元の例外を処理するために例外ハンドラ非特権化することができるようにするために、これは更なる例外を生成する必要がある（例えば、スーパーバイザ呼び出し例外、SVC）、その結果、例外戻りスタックフレームはメモリ内で設定され得、この例外戻りスタックフレームは、例外の前に実行されている実際の処理コンテキストを表すものではないが、SVC例外が返されたときに処理がスレッドモードに切り替わることを保証するように偽装され、以前に発生した例外の本体を非特権状態のスレッドモードで処理することができる。

【0129】

例えば、図7に示すように、スレッドモードのスレッドT1内の何らかのバックグラウンドコードを処理するとき、非特権ライブラリに渡されるタイプの割り込みIRQが発生する。図7のステップ1において、割り込みは、例外制御回路28にプロセススタックへのレジスタ状態のスタックをトリガさせて、スレッドT1のレジスタ状態を維持させる。ハンドラモードでは最初にデフォルトで例外が取られるため、割り込みはスレッドモードからハンドラモードへの遷移を引き起こす。IRQ例外を処理するための例外ハンドラの本体200は信頼できないライブラリによって実行されることになるが、IRQ例外を処理するための例外ハンドラには、主例外ハンドラ本体200の前後に実行されるラップコード202が設けられており、ラップコードはハンドラモードで実行され、本体がスレッドモードで処理され得ることを保証するためにSVC例外を生成するための動作を制御する。

【0130】

従って、図7のステップ2において、ハンドラモードのラップコードはSVC例外を生成し、SVC例外は、実際には元のIRQ割り込みを処理しないが、単に、IRQハンドラがスレッドモードに切り替わることを可能にするために生成される。SVC例外は、例外制御回路28に、現在のドメインに関連付けられた主スタックポインタMSPによって指し示される主スタックへのレジスタ状態のスタックを実行させる（現在のモードはハンドラモードであるため、主スタックが使用される）。SVC例外に応答して実行されるSVC例外ハンドラは、プロセススタックポインタを、元のIRQ例外を処理するために主例外ハンドラ本体200を実行することになるスレッドT1のIRQに関連付けられた異なるスタック構造PSP IRQを指すように切り替える。SVC例外ハンドラは、その例外戻り状態情報（ERETPSR）において、対応する例外戻りがスレッドモードに戻るべきであることを示すために例外番号0を示すスタックフレームをプロセススタック上に偽装する。SVC例外の前に実行されていた処理は実際にはハンドラモードであったため、この値は偽であることに留意されたい。

【0131】

SVC例外ハンドラが完了すると、ステップ2で主スタック上に作成された例外スタックフレームではなく、PSP IRQによって指し示されるスタック上に作成された偽の例外戻りスタックフレームを使用して例外戻りをトリガする偽のダミー例外戻りアドレスへの分岐を引き起こす。次に、例外戻りは例外制御回路28によって実行され、図7のステップ3で、PSP IRQスタックからハードウェアをアンスタックさせる。ハードウェアは、偽のダミー例外戻りアドレス142内のMビット152が、戻りがスレッドモードになるべきであることを示すことを検出し、従って、ステップ4において、IRQ例外を処理するための例外ハンドラ200の後続の本体は、非特権ライブラリによってスレッドモードで実行され得る。これは元のIRQ例外から戻ることなく行われており、従って、IRQ例外を処理するための例外ハンドラの優先度は低下していないことに留意されたい。

【0132】

主例外ハンドラ本体200が完了すると、ステップ5で別のSVC例外呼び出しが行われ（SVCは、予期しない又は障害したイベントによってトリガされる例外のタイプではなく、ソフトウェアによって自発的にトリガされ得る例外のタイプである）、これにより再びPSP IRQが指すプロセススタックへの更なるハードウェアスタッキングが行わ

10

20

30

40

50

れ、呼び出されたSVCハンドラは次に、PSPIRQが指すプロセススタック上に作成されたばかりのスタックフレームを破棄し、次いで、現在のセキュリティドメインのプロセススタックポインタを更新して、バックグラウンドスレッドT1に関連付けられた以前のスタックポインタPSP1を復元する。図7のステップ6において、SVC例外は例外戻りを生成し、これは次に、MSPを使用して主スタックからの更なるハードウェアアンスタックをトリガし（すなわち、ステップ2の間に作成されたスタックフレームをアンスタックする）、次いで、ハンドラモードで残りのラップ関数202を実行した後、ステップ7において、更なる例外戻りが生成され、これは、（すなわち、ステップ1の間に作成されたスタックフレームをアンスタックする）PSP1によって指し示されるプロセススタックからスレッドT1に関連するレジスタ状態のハードウェアアンスタックを引き起こす。その後、処理は、スレッドモードで実行しているスレッドT1のバックグラウンドコードにおいて継続する。

【0133】

図7に示すプロセスは、例外ハンドラがスレッドモードで動作するように、特定のタイプの例外を非特権化することを可能にする。しかしながら、それにはいくつかの欠点がある。第1に、必要とされる例外スタックフレームの偽装に起因して、このプロセスを制御するためのソフトウェアコードを生成することは困難であり、そのため、プログラマによるエラーが発生しやすく、攻撃者が悪用する可能性のある、又はシステムが正しく機能するのを停止する可能性のある脆弱性を提供する可能性があり、又は攻撃者が暗号鍵などの秘密値を抽出する方法を提供する可能性がある。

【0134】

別の欠点は、この手法がSVC例外の例外優先度を機能させるためには（主例外ハンドラ本体200の先頭及び末尾で実行されて偽のスタックフレームを作成し、元の例外から戻る前にクリーンアップする）、非特権化される必要がある全ての割り込みIRQの優先度よりも高くなければならないことである。これは、リアルタイムシステムの設計をはるかに複雑にする可能性があり、なぜなら、SVC例外は常に同じ優先度に取りられるため、他のIRQ例外の最悪の場合の割り込み待ち時間は、最も長いSVC呼び出しの持続時間の分だけ長くなるからである。これは望ましくない場合がある。

【0135】

また、図7に示す手法は非常に低速であり、これは重大な割り込み待ち時間を有する非特権ライブラリの使用に対する主要なブロッカーである。この不十分な性能は、図7のステップ2と3との間でSVC例外を処理するときの例外スタックフレームの手動作成、及び図7のステップ2、3、5、及び6で4つの追加のレジスタ状態スタック及びアンスタックイベントをトリガする2つの異なるSVC例外が必要であるというイベントを含む複数の要因に起因し、これらは、IRQを処理するための例外ハンドラが単にハンドラモードで実行された場合には必要とされないメモリアクセス待ち時間を表す。従って、性能コストは、一部のシステム設計者が、信頼できないライブラリコードに関連付けられたワイヤレスネットワークインタフェース又はUSBコントローラなどのデバイスを単に使用することができないと感じる可能性があることを意味する可能性があり、システム設計の柔軟性を制限する可能性がある。図7の下部に示すように、この例では、非特権プロセス全体が低セキュアドメインで実行されるが、プロセスはセキュアドメイン内で実行されてもよいことが理解されよう。

【0136】

図8は、本出願で説明したようなインターモダル呼び出し分岐命令を使用して、例外特権解除の性能を改善する代替手法を示す。命令によって指定されたターゲットアドレスで命令に分岐し、関数が完了した後に処理が戻るべきアドレスを示すように関数戻りアドレスを設定することができるインターモダル呼び出し分岐命令を提供すること、並びにハンドラモードからスレッドモードへの切り替えをトリガすることによって、これは、割り込みをはるかに高速に非特権化ことができ、性能ヒット、複雑さ、及び図7に示す手法に関連する上記の他の問題を低減することを意味する。この手法では、非特権状態で処

10

20

30

40

50

理されるべきである I R Q 例外に対する応答をはるかに簡単にすることができる。

【0137】

図8のステップ1は、図7と同じである。ステップ2において、主割り込みハンドラ本体201の実行を準備するための何らかのラップコード203が実行され、これは、主割り込みハンドラ本体201の前の最後の命令として、主例外ハンドラ本体201内で実行されるべき最初の命令のアドレスを分岐ターゲットアドレスとして指定するインターモダル呼び出し分岐命令を有する。いくつかのオペレーティングシステムでは、ラップ203は、SVC例外をトリガする代わりにインターモダル分岐命令を実行することを除いて、ラップ202と同一であり得る。いくつかの実装形態では、インターモダル呼び出し分岐命令は、ハンドラからスレッドモードに遷移しない非インターモダル呼び出し分岐命令の命令符号化とは異なる専用命令符号化であってもよい。従って、図8のステップ3で、インターモダル呼び出し分岐命令の実行はハンドラモードからスレッドモードへの切り替えをトリガし、次に主例外ハンドラ本体201はスレッドモードで図8のステップ4で実行し続ける。SVC例外が呼び出される必要はない。

【0138】

インターモダル関数呼び出しを行うと、関数呼び出しがインターモダル呼び出しであったことを（インターモダル呼び出しインジケータ146を使用して）示すためにダミー関数戻りアドレス140に情報を記憶することができ、その結果、図8のステップ5で対応する関数戻りがトリガされると、処理回路4はインジケータ146を検出し、（関数戻りスタックフレーム178に記憶された）以前の関数呼び出しに関連する関数戻りアドレス174に分岐することに加えて、現在のモードもスレッドモードからハンドラモードに戻されるべきであることを検出することができる。次に、いくつかの残りのラップコード203は、図8のステップ6においてハンドラモードで実行され、その後ステップ7が続き、例外戻りが実行されて、ハンドラから、I R Q 例外が発生する前に実行されていたバックグラウンドスレッドT1に戻る。いくつかのシステムでは、主割り込みハンドラ本体201（図8）は、非特権主割り込みハンドラ本体から戻るためにSVC例外をトリガする代わりに、図8のステップ3でリンクレジスタ内に配置されたダミー関数戻りアドレス140に単に分岐することによって戻りを要求することができるという点で、主割り込みハンドラ本体200（図7）の例外と同一であってもよい。従って、主割り込みハンドラ本体201（図8）は、標準的なC関数であってもよく、特別な取り扱いを必要としない。

【0139】

従って、この手法では、必要とされるハードウェアのスタック及びアンスタック動作は、バックグラウンドコードT1に関連するレジスタ状態を保存及び復元するためのもののみであり、2つのSVC例外に関連する更なるハードウェアスタッキング及びアンスタック動作は必要ない。また、この手法は、コンパイラ又はプログラマのために符号化するのがはるかに容易であり、エラーの可能性を低減し、SVC例外が非特権化される全ての例外よりも高い優先度を有することを必要としない。

【0140】

セキュリティドメインを切り替える機会の数を制限することによってセキュリティを向上させるために、インターモダル関数呼び出しは、セキュリティドメインを変更することができず、モード間でのみ変更することができる。従って、インターモダル関数呼び出し分岐命令を実行するとき、処理回路4は、現在のセキュリティドメインをインターモダル関数呼び出し分岐命令の実行前と同じに保つことができる。図9に示すように、セキュリティを維持するために、低セキュアスレッドモードからセキュアハンドラモードへの組み合わせたドメイン／モード関数戻り遷移を禁止することができる。これは、以下の図15を参照して説明するように、セキュアドメインへの戻りがインターモダル関数戻りで要求されているかどうかの明示的なチェックによって実施することができ、これが検出された場合に障害をトリガすることができる。

【0141】

しかしながら、インターモーダル関数戻りの場合、図10に示すように、少なくともインターモーダル関数戻りがセキュアスレッドモードから低セキュアハンドラモードにある場合には、組み合わせモード／ドメイン遷移を許可することが望ましい場合がある。これにより、コンパイラは、図11A～図11Cに示すように、末尾呼び出しとして知られる技術を使用することができる。いくつかの実装形態では、セキュリティ理由のために、末尾呼び出しは、セキュアから低セキュア関数呼び出しのために禁止されている可能性がある。このように、図10に示すセキュアスレッドから低セキュアハンドラインターモーダル戻り遷移をサポートする必要があるが、図9に示す低セキュアスレッドからセキュアハンドラインターモーダル戻り遷移をサポートする必要はない場合がある。

【0142】

図11Aに示すように、（処理システムの命令デコーダ10によってサポートされるアセンブリコードにコンパイルされる）いくつかの高レベルコードは、第2の関数 f_{n2} が第1の関数 f_{n1} の内部で呼び出されるネストされた関数呼び出しを含むことができ、関数 f_{n2} への呼び出しの位置は、 f_{n2} の終わりと f_{n1} の終わりととの間に f_{n1} の更なる命令が存在しないような位置である。

【0143】

このシナリオでは、図11Bに示すように、アセンブリコードを生成するコンパイラの1つの選択肢は、 f_{n1} 及び f_{n2} への呼び出しの各々について、リンクレジスタLR（R14）への関数戻りアドレスの保存と、その関数の関数コードの開始を表すターゲットアドレスへの分岐と、をトリガする呼び出し分岐命令（BL、分岐及びリンク命令としても知られている）を生成し、その後、リンクレジスタに以前に保存された戻りアドレスに処理が戻るべきであることを示す関数戻り命令（そのアドレスオペランドとしてリンクレジスタを指定する間接分岐命令BX）が続くことであり得る。 f_{n1} を呼び出すための呼び出し分岐命令は、 f_{n1} を呼び出したコードの一部になるので、図11Bには示されておらず、図11Bは、 f_{n1} 及び f_{n2} の関数コードのみを示している。従って、 f_{n1} に対して、コンパイラは（ f_{n2} を呼び出す）BL命令283及び（ f_{n1} から f_{n1} を呼び出したコードに戻る）BX命令286を生成する。 f_{n2} の場合、コンパイラは、 f_{n2} から f_{n1} に戻るためのBX命令284を生成する。 f_{n2} を呼び出すときに f_{n1} の戻りアドレスが保存されることを保証するために、 f_{n1} コードはまた、 f_{n2} を呼び出す前にリンクレジスタの内容をスタックにプッシュするプッシュ命令282と、 f_{n1} から戻る前にリンクレジスタをスタックから復元するためのポップ命令285を含む。実際には、 f_{n2} は f_{n1} 内で実行される最後のアクティビティであるため、スタックとの間の関数1のリンクレジスタのこのプッシュ及びポップは、一部の不要なメモリ及び性能オーバーヘッドを導入する。

【0144】

従って、図11Cに示すように、関数をコンパイルするための代替手法は、リンクレジスタを更新しない非呼び出し分岐命令287を使用して f_{n2} に関連付けられたコードを呼び出し、 $f_{n1} \sim f_{n2}$ に対してリンクレジスタに保存された戻りアドレスを効果的に渡すことであり得る。これは、最終戻り分岐288が実行されると、処理が f_{n1} の前に処理されていたバックグラウンドコードに戻ることを意味する。この手法は末尾呼び出しとして知られており、プッシュ282、ポップ285、及びBX 286命令の必要性を回避することによって性能を改善することができる。

【0145】

しかしながら、図10の例に示すように、関数1が低セキュアスレッドモードにあり、インターモーダル関数呼び出しによって呼び出され、関数2がセキュアスレッドモードにあり、ドメイン間関数呼び出し（低セキュアドメインNSからセキュアドメインSへの呼び出し）で呼び出されるシナリオでは、これは、末尾呼び出しをサポートするために、後続の戻り分岐命令288が、セキュアスレッドモードから低セキュアハンドラモードへの直接戻りをトリガする必要があることを意味する。組み合わせのドメイン／モード遷移がサポートされていない場合、末尾呼び出しは可能ではなく、これは多くの既存のレガシ

10

20

30

40

50

ーコードが再コンパイルされることを必要とする可能性があり、そのため、セキュアスレッドモードから低セキュアハンドラモードである場合に、組み合わせのドメイン／モード遷移をサポートすることがより効率的であり得る。それにもかかわらず、低セキュアドメインからセキュアドメインへのインターモーダル関数戻りを禁止することによって、これは、セキュアドメインに入ることができる手段を制限するので、セキュリティを向上させる。先に説明したように、セキュアドメインから低セキュアドメインへの末尾呼び出しがレガシーコードで既に禁止されている可能性があるため、この遷移の組み合わせは必要ない可能性がある。

【0146】

図12は、関数呼び出し分岐命令（BLX）に応答して実行されるステップを示すフロー図である。これらのステップは、関数呼び出し分岐命令として識別された命令の復号に
10 応答して、命令デコーダ10の制御下で処理回路4によって実行される。ステップ300において、関数呼び出し分岐命令が、他のタイプの関数呼び出し分岐命令とは異なる命令符号化によって区別される、特定のタイプの関数呼び出し分岐命令であるインターモーダル呼び出し分岐命令であるかどうか
20 が判定される。例えば、インターモーダル呼び出し分岐命令は、非インターモーダル呼び出し分岐命令とは異なるオペコードを有することができ、又は同じオペコードを有するが、それがインターモーダル呼び出し分岐命令であるかどうかを区別する命令符号化で指定された別個のパラメータを有することができる。図12に示す例示的な実施形態では、インターモーダル呼び出し分岐命令の2つの変形、BLXT及びBLXTIがあり、これらは両方ともインターモーダル関数呼び出しをトリガするが、制御レジスタ32にTMIDパラメータ34を設定する方法が異なる。いくつかのシステム実装形態は、インターモーダル呼び出し分岐命令の両方の変形をサポートせず、BLXT及びBLXTIの変形のうちの1つのみを有することができ
30 ことが理解されよう。また、いくつかの実装形態は、インターモーダル呼び出し分岐命令の更なる変形形態を有することができる。

【0147】

関数呼び出し分岐命令がインターモーダル呼び出し分岐命令ではない場合、ステップ302において、処理回路4は、（命令デコーダ10の制御下で）関数呼び出しがセキュアドメインから低セキュアドメインへのドメイン間関数呼び出しであるかどうかを判定する。いくつかの実装形態では、そのようなセキュアから低セキュアドメイン間呼び出しは、関数呼び出し分岐命令の更に別の変形BLXNSによってシグナリングされ得、これは他のタイプの関数呼び出し分岐と区別する専用命令符号化を有する。また、いくつかの実装形態では、BLXNS命令がドメイン間呼び出しをトリガするかどうかは、命令によって指定された分岐ターゲットアドレス、特に分岐ターゲットアドレスの最下位ビットが0であるかどうかにも依存し得る。いくつかの例では、有効な関数呼び出し又は低セキュアコードからのセキュアドメインへの分岐では、関数呼び出し後にセキュアコード内の第1の命令として実行されたセキュアゲートウェイ命令に
40 応答して、関数戻りアドレスの最下位ビットを0にサニタイズして、後続の関数戻りが低セキュアドメインに戻ることを確実にして、その結果、低セキュアコードが、セキュア領域内にある関数戻りアドレスを提供することによって、SAU24によって定義されたセキュアアドレス領域内の任意の位置にセキュアコードを不正に分岐させることができないようにしてもよい。分岐ターゲットアドレスの最下位ビットをサニタイズすることは、（BLXNS命令を使用して）セキュアから低セキュアドメイン間関数戻りのセキュリティを保証するために使用され得るので、（BLXNS命令を使用して）セキュアから低セキュアドメイン間関数呼び出しのセキュリティを保証するために同様のスキームを使用することが便利であり得る。従って、分岐ターゲットアドレスの最下位ビットが0に設定されているドメイン間呼び出し命令BLXNSがあることを検出すると、これは、セキュアから低セキュアドメイン間呼び出しとして扱われ得る。いくつかの実装形態では、セキュアから低セキュアドメイン間呼び出し命令（BLXNS）は、処理回路4がセキュアドメイン内で動作している場合に、命令セット内でのみ利用可能（従って、デコーダ段10によって検出可能）であり得る。そのよう
50

な実施態様では、低セキュア状態からBLXNS命令を実行しようとする試行は、その命令を現在の状態に対する有効な命令として認識しない復号段10の結果に終わることがあり、従って処理はステップ300に到達しない。

【0148】

関数呼び出しがセキュアから低セキュアドメイン間呼び出しでない場合、ステップ304において、関数呼び出しは、いかなる特定のセキュリティ上の影響もなしに処理することができ、ステップ304において、リンクレジスタ(R14)が関数戻りアドレスに設定される。関数戻りアドレスは、アドレス空間100内の関数呼び出し命令後の命令のアドレス(すなわち、関数呼び出し命令のアドレス+関数呼び出し命令オペコードの長さ)の値を有する。ステップ306において、プログラムカウンタレジスタR15は、関数呼び出し分岐命令によって指定された分岐ターゲットアドレスに基づいて更新され、その結果、処理は分岐ターゲットアドレスで命令になるように分岐する。これは、ドメインの変更又はモードの変更をトリガしない関数呼び出しのための分岐処理の通常の場合を表す。

【0149】

ステップ302において、関数呼び出し分岐がセキュアから低セキュアドメイン間呼び出しであると判定された場合、ステップ307において、処理回路4は、(例外番号36が0の特別なスレッドモード値であるかどうかに基づいて)現在のモードがスレッドモードであるかどうかを判定し、また、スレッドモード遷移無効化(TMID)パラメータ34が、ドメイン遷移がスレッドモードにおいて無効にされることを示すかどうかを判定する。現在のモードがスレッドモードであり、スレッドモードにおいてドメイン間遷移が無効にされている場合、ステップ308において、処理回路は障害をシグナリングする。シグナリングされる特定のタイプの障害は、TMIDフラグ34がこれが無効にされていることを示す場合に、スレッドモードでセキュリティドメインを切り替える試行によってトリガされる例外にのみ使用される専用の障害タイプであってもよい。あるいは、障害は、様々な障害状態を処理するために使用される例外ハンドラに向けられてもよく、そのような場合、ドメイン間遷移無効化チェックに関連するシンδροームフラグが設定されてもよく、例外処理ルーチンは、障害の原因、及び実行される必要がある後続のアクションを容易に識別することができる。図12では、これはINV TMI(無効なスレッドモード遷移)Usage Faultとして示されているが、このタイプの障害の正確な命名規則は当然異なり得る。セキュアから低セキュアドメイン間呼び出しの場合、INV TMI障害はセキュアドメイン(呼び出しを試行する前の現在のドメイン)で処理され、これは、セキュアから低セキュア遷移を要求したスレッドを管理するためのプロセスが、セキュアスレッドを管理するセキュアオペレーティングシステムであるため有用であり、これは、そのスレッドが低セキュアドメインで動作するために必要なリソースを構成できるように、INV TMI障害を処理するためにより適切に配置され得る。

【0150】

ステップ307で、現在のモードがハンドラモード(例えば、例外番号36が非ゼロである場合)であるか、又はスレッドモードでドメイン遷移が有効にされていることを示すためにTMIDパラメータ34が0であると判定された場合、ステップ310で、処理回路は、図5に示すように、リンクレジスタR14を有効なダミー関数戻りアドレス値140のうちの1つに設定する。ダミー関数戻りアドレス140において、インターモジュールフラグ(M)146は、これがインターモジュール関数呼び出しではなかったことを示すために1に設定され、また、セキュリティドメインフラグ(S)148は、呼び出しがセキュアドメインから行われたことを示すために1に設定される。ステップ312において、関数戻りスタックフレーム172は、セキュア主スタックポインタMSP__S又はセキュアプロセススタックポインタPSP__Sのいずれかによって指し示されるスタックデータ構造に保存される(現在のモードがハンドラモードであるかスレッドモードであるかに応じて、スレッドモードである場合には、セキュアドメイン13__Sのスタックポインタ選択値30に応じて)。保存された関数戻りスタックフレーム172は、これがインター

10

20

30

40

50

モーダル関数呼び出しではないことを示すために、HDTフラグ178が0に等しいと指定する。また、関数戻りスタックフレーム172内の例外番号フィールド179は、ハンドラモード又はスレッドモードから呼び出しが行われたかどうかを示すために、制御レジスタ20内の例外番号36の現在の値に等しく設定される。この例外番号フィールド179は、現在のモードがハンドラモードであるかどうかを示す値を示すハンドラモードとして機能する。関数戻りスタックフレームはまた、ステップ304と同じ方法で設定された関数戻りアドレス174を含むが、今度はリンクレジスタの代わりにスタックに保存されるため、ドメイン間呼び出し後に実行される低セキュアコードは、関数呼び出し前にセキュアコードでどの命令が実行されていたかを知ることができない。ステップ314において、処理回路は、次に、現在のドメインを低セキュアドメインに切り替え、次いで、方法はステップ306に進み、関数呼び出し分岐命令によって指定された分岐ターゲットアドレスへの分岐が、前述と同じ方法で実行される。

【0151】

再びステップ300を参照すると、関数呼び出し分岐命令がインターモーダル呼び出し分岐命令である場合には、ステップ320において、処理回路は、現在の動作モードがスレッドモードであるかどうかを判定し、現在のモードがスレッドモードである場合には、ステップ322において、使用障害がシグナリングされる。これは、INVTMI Usage Faultとは異なるタイプの使用障害であり得る。インターモーダル呼び出し分岐命令は、ハンドラモードからスレッドモードへの切り替えをトリガして、割り込み非特権化をサポートすることを意図しており、スレッドモードからそれを実行する試行は、エラーである。スレッドモードにあるときにインターモーダル呼び出し分岐の命令実行を抑制すると、セキュリティの脆弱性につながる可能性がある、呼び出し分岐に応答して誤ったスタックフレームを設定することによって引き起こされる潜在的に予測不可能な結果が回避される。従って、インターモーダル呼び出し分岐命令は、ハンドラモードからのみ実行することができる。

【0152】

ステップ320において、現在のモードがハンドラモードであると判定された場合、方法はステップ324に進み、リンクレジスタが特殊ダミー関数戻りアドレス値140のうちの1つに設定されるが、ステップ310とは異なり、対応する関数戻りがインターモーダル関数戻りとして扱われるべきであることを示すために、ステップ324においてインターモーダルフラグ(M)146が0に設定される。また、関数戻り値140内のセキュリティドメインフラグ(S)148は、インターモーダル呼び出し分岐命令が実行された時点の動作の現在のドメインを示す(インターモーダル呼び出しは、セキュアドメイン又は低セキュアドメインのいずれかから行うことができる)。

【0153】

ステップ326において、処理回路は、インターモーダル呼び出し命令及びスタックポインタ選択値30を実行する前に、ドメイン及び動作モードに基づいて選択されたスタックに関数戻りスタックフレーム172を保存する。関数戻りスタック値は再び、(ステップ304又は312と同じ方法で設定される)関数戻りアドレス174、及び例外番号36の現在の値に対応する値を示すハンドラモードを含む。インターモーダル呼び出しがハンドラモードから行われると、例外番号フィールド179は非ゼロになると予想される。インターモーダル呼び出しの場合、関数戻りスタックフレーム内のHDTフラグ178は、この関数戻りスタックフレームがインターモーダル呼び出しに응答して保存されたことを示すために1に設定され、ダミー関数戻りアドレス140を使用して対応する関数戻りで、インターモーダルフラグ146が正しいか、信頼できないコードによって改ざんされているかをチェックするためのクロスチェック情報を提供することができる。ステップ312又は326のいずれかにおける関数戻りスタックフレームの保存に続いて、現在保存されているスタックフレームがアンスタックされる前に更なるスタックが必要とされる場合、対応するスタックポインタも、更なるスタック情報がスタックに保存されるべき位置を指し示すように更新されることに留意されたい。

【0154】

ステップ328において、任意選択的に、処理回路は、インターモーダル呼び出し分岐命令に応答して実行されるエラー同期バリア動作をトリガする。例えば、エラー同期バリア動作は、任意の事前に実行された命令の妥当性をチェックするための少なくともいくつかのタイプのエラー検出又は訂正動作の結果が完了するまで、処理回路4が命令の更なる実行を中断することを含むことができ、それらの動作がエラーフリー、訂正され得るエラーを含む、又はハンドラからスレッドモードへの変更前の命令の実行に正しく起因するエラーを含む、いずれかであったことが確認される。エラー同期バリア動作は、信頼性、可用性、及び保守性(RAS)エラー検出／訂正回路29によって検出された少なくともいくつかのタイプのエラーが、インターモーダル呼び出し分岐命令によって引き起こされるモード変更の前に実行されているコードに起因し、インターモーダル呼び出し分岐命令によって引き起こされるモード変更の後に実行されたコードを処理するときに発生するエラーとは区別され得ることを保証するための動作である。インターモーダル呼び出し分岐の場合、検出されたRASエラーが、インターモーダル関数呼び出しによって引き起こされるモード変更の前に実行されたより特権的なコードに影響を与えたか、又はモード変更の後に実行された低特権コードに影響を与えたかを正確に特定することが有用であり得る。これは、多くの場合、低特権コードに関連するエラーを解決することは、より特権的なコードに関連するエラーを解決することよりもシステムの可用性にはるかに低い影響を及ぼし得るからである。例えば、より特権的なコードはオペレーティングシステムであり得るため、オペレーティングシステムに起因するエラーを処理するには完全なシステムリセットが必要になる場合があるが、個々のスレッドに関連付けられた低特権コードに起因するエラーは、そのスレッドをキルするが、他のスレッド及びオペレーティングシステムに影響を与えないままにすることによって簡単に処理することができる。従って、インターモーダル呼び出しに응答してエラー同期バリア動作を実行することにより、これは、インターモーダル関数呼び出しによって引き起こされたモード変更後に実行される低特権コードに安全に起因する可能性がある場合に、検出されたエラーを処理する影響を制限することによってシステムの可用性を向上させることができる。エラー同期バリアの範囲外であり得るいくつかのタイプのエラーが存在する可能性があり、従って、そのようなバリアの使用によっては含まれないことがあることが理解されよう。これらの封じ込められないエラーは、依然として完全なシステムリセットを必要とする可能性がある。しかしながら、エラー同期バリアの使用は、単一のスレッドに起因するエラーの可能性を高め、従って、完全なシステムリセットが必要とされる可能性を低減することができる。

【0155】

全てのシナリオでエラー同期バリア動作がトリガされることは必須ではない。一部のシステムは、インターモーダル関数呼び出しの後に実行される後続の命令の実行を人為的に抑制しないことによって、定期的なプログラム実行の性能が改善されることを好む場合がある。例えば、特定のリアルタイムクリティカルな割り込みの場合、割り込み処理待ち時間は可能な限り短くなければならない可能性があるため、RASエラーが検出された場合、より大きな性能コストを犠牲にして、エラー同期バリア動作をトリガしないことが好ましい場合があるが、低リアルタイムクリティカルな要件を有する他のタイプの割り込み、又は安全クリティカルな動作に関連する他のタイプの割り込みの場合、エラー同期バリア動作をトリガすることが好ましい場合があり、その結果、通常の割り込み待ち時間は、エラー検出／訂正結果が利用可能になるまでインターモーダル呼び出し後に命令の実行を保留するためにわずかに遅くなり得るが、エラーが検出された場合、エラーの位置を特定することができ、システムリセットの完全な影響を可能な限り回避することができる。従って、いくつかの実装形態では、制御レジスタ20は、インターモーダル呼び出し命令に응答してエラー同期バリア動作をトリガすべきかどうかを指定する構成パラメータを記憶することができる。いくつかのシステムでは、エラー同期バリア動作が実行されるべきかどうかを指定する構成パラメータは、セキュリティドメイン間でバンクされ、低セキュアドメインとセキュアドメインが、バリア動作が実行されるべきかどうかを独立して選択す

ることができる。

【0156】

エラー同期バリア動作が実行されるかどうかにかかわらず、ステップ330において、インターモーダル呼び出し分岐命令に応答して、処理回路はまた、現在のセキュリティドメインのスタックポインタ選択値30-S又は30-NSを設定して、インターモーダル関数呼び出し後にスレッドモードで実行される後続の動作のためにプロセススタックポインタが選択されるべきであることを示す。これにより、たとえ例外ハンドラによって例外が処理される前に実行されているスレッドモードコードが主スタックポインタにアクセスしたとしても、インターモーダル呼び出し後に実行される例外ハンドラの低特権部分が、インターモーダル呼び出し命令の前にハンドラモードで実行されたラップコードのより特権的なセクションによって使用される主スタックポインタにアクセスしたり破損したりすることができないことが保証される。

【0157】

ステップ332において、処理回路4は、実行されているインターモーダル呼び出し分岐命令が命令のBLXTI変形であるかどうかを判定する。そうである場合、ステップ334において、セキュアドメインと低セキュアドメインとの間のドメイン遷移がスレッドモードで有効にされることを示すために、TMIDパラメータ34が0に設定される。これは、前述したように、インターモーダル呼び出し分岐命令の第2の変形例に対応する。インターモーダル呼び出し分岐命令がBLXT変形（第1の変形）である場合には、ステップ336において、スレッドモードでのドメイン遷移が無効にされていることを示すためにTMIDパラメータ34が1に設定される。プログラマは、インターモーダル関数呼び出しに続いて実行されるスレッドに対して、セキュアドメインと低セキュアドメインの両方のリソースが既に確立されているかどうかに応じて、BLXT又はBLXTI変形を使用するかどうかを選択することができる。BLXTI変形は、必要に応じてセキュアドメインと低セキュアドメインとを切り替えることが安全であるように、両方のドメインのリソースが既に確立されている場合に選択することができ、一方、BLXT変形は、S/NSドメインのうちの1つのみのリソースが構成されている場合に選択することができる。これらのリソースは、セキュアMPU26-S及び低セキュアMPU26-NSのうちの関連する1つ内にMPU構成を含むことができる（MPU構成は、それぞれのメモリ領域へのアクセス許可を定義するいくつかのメモリ領域属性及び／又はそれらのメモリ属性を提供したメモリシステム6内のテーブルへのポインタの一方又は両方を含むことができる）。また、リソースは、セキュア又は低セキュアスタックポインタレジスタ（図3に示すR13のバンクバージョン）にスタックポインタを含むことができ、メモリ内の適切なスタック構造がまだ利用可能でない場合、メモリに記憶された対応するスタック構造の割り当ても含むことができる。スレッドモードでドメイン遷移を無効にする機能を提供することにより、これは、スレッドが一方のドメインに対してのみそのリソースを構成することを可能にし、その後、ドメインを切り替える試行がある場合にのみ他方のドメインのリソースを緩慢に割り当てる必要があり、そうでなければ上記のステップ308に示すように、又は他のタイプのドメイン間遷移について前述したように障害がトリガされる。BLXTI又はBLXT命令を使用してスレッドに最遷移無効化パラメータ34も設定することにより、これはフラグを別々に設定するための別個の命令の必要性を回避する。

【0158】

インターモーダル呼び出し分岐命令のどの変形が実行されるかにかかわらず、ステップ338において、処理回路は、非例外処理値（例えば0）を示すために例外番号36を更新し、そのため、プロセッサはスレッドモードに切り替わる。再び、方法は次にステップ306に進み、関数呼び出し分岐命令によって指定された分岐ターゲットアドレスに分岐する。その後のプログラム実行は、分岐ターゲットアドレスに記憶された命令から継続する。分岐ターゲットアドレスは、関数呼び出し分岐命令によっていくつかの方法で指定できることが理解されよう。間接分岐の場合、ターゲットアドレスを含むレジスタを選択するためにオペランドを使用して分岐ターゲットアドレスを定義することができる。直接

10

20

30

40

50

分岐は、命令符号化において直接符号化された即値を使用してターゲットアドレスを指定することができる。プログラムカウンタ相対分岐（間接的又は直接的であり得る）は、分岐ターゲットアドレスを取得するために現在のプログラムカウンタ値に追加されるオフセットを指定することができる。

【0159】

図13は、例外に入る際に実行されるステップを示す。ステップ350において、例外が取られる。例外は、スーパーバイザコール（SVC）などのソフトウェアによってトリガされる例外、又は未定義の命令を実行しようとする試行、予約済みアドレス範囲130から命令をフェッチしようとする試行、又はメモリアクセスチェック回路22でのメモリ許可チェックの失敗に起因してアドレス障害をトリガするメモリへのアクセスなど、ソフトウェアによって直接引き起こされる障害の発生によって引き起こされる例外とすることができる。また、例外は、外部デバイス又は周辺機器からの信号又はメッセージの受信によって引き起こされるハードウェアトリガされた割り込みであり得る。

【0160】

例外の原因にかかわらず、例外が取られると、ステップ352で、処理回路4は例外制御回路28によって制御され、リンクレジスタR14を前述の特殊ダミー例外戻り値142に設定する。より具体的には、いくつかのダミー例外戻り値のうちの1つが選択され、現在のモード及び現在のドメインフラグ152、156の値は、例外が発生した時点の現在のモード及び動作ドメインに基づいて選択される。また、スレッドモード遷移有効化（TMIE）フラグ158は、制御レジスタ32内のTMIDフラグ34の現在値の逆数に設定される。これにより、例外処理中にTMIDフラグ内の情報が保存され、例外戻り時に復元できる。TMIEフラグを設定するためにTMIDフラグを反転させる実装形態は、1つの可能な実装形態にすぎず、他の例は、TMID値と同じ符号化でTMIE値を符号化し、それを直接コピーするだけでよく、又は単一ビットフラグで表される必要はないが、複数ビット値を使用することができる完全に異なる符号化を使用することができることが理解されよう。

【0161】

ステップ354において、例外制御回路28は、レジスタ1から、バンクされたスタックポインタレジスタ内のスタックポインタの1つによって識別されるメモリ内のスタックデータ構造へのレジスタ状態のハードウェアトリガされた状態保存をトリガする。選択する特定のスタックポインタは、例外が発生する前に使用されている現在のドメイン及びモード並びにスタックポインタ選択値30に依存する。状態保存は、図6に示す呼び出し側スタックフレーム180のみを記憶すること、又は呼び出し側スタックフレーム180と被呼び出し側スタックフレーム182の両方を記憶することを含むことができる。呼び出し側スタックフレームのみを保存するか、又は呼び出し側及び被呼び出し側スタックフレームの両方を保存するかの判定は、例外が取られた時点の現在の動作ドメイン及び例外が処理されるドメインに依存し、場合によっては制御レジスタ20に記憶された状態データに基づいてもよい。

【0162】

それ自体が例外を処理していないバックグラウンドコードから例外が取られ、以前に発生した例外が既に処理されており、バックグラウンドコードが低セキュアドメインで処理される場合、呼び出し側スタックフレーム180のみを保存する必要があるが、被呼び出し側スタックフレーム182を保存する必要はない。また、バックグラウンドコードはセキュアドメインで処理されたが、例外はセキュアドメインで取られるべきである場合、被呼び出し側スタックフレーム182を保存する必要はない。

【0163】

例外がセキュアドメインで処理されたバックグラウンドコードから取られるが、例外が低セキュアドメインで処理されることになる場合、低セキュア例外ハンドラが、例外の前に動作するセキュアコードによってレジスタ14に配置されたレジスタ状態にアクセスできるリスクがある。従って、この場合、呼び出し側及び被呼び出し側スタックフレーム

10

20

30

40

50

180、182の両方が、セキュア主スタックポインタ及びセキュアプロセススタックポインタの一方によって識別されるセキュアスタックに保存され、汎用レジスタ16をクリアして、セキュアデータが低セキュアハンドラからアクセスされるのを隠すことができる。
【0164】

以前の段落では、バックグラウンドコードから例外を取るときに追加の被呼び出し側スタックフレームを保存するかどうかを判定する一般的な原理について説明している。しかしながら、更なる例外がその例外ハンドラをまだ完了していない以前の例外を先取りする場合、又は例外の末尾チェーンが実行され、1つの例外を完了すると、バックグラウンドコードに戻る前に別の例外が取られる場合、呼び出し側スタックフレーム及び／又は被呼び出し側スタックフレームを保存するかどうかについてより複雑な判定があり得る。例外に応答して呼び出し側及び被呼び出し側状態のいずれの状態を保存するかを決定するための決定プロセスの例は、公開されたPCT出願国際公開第2013/117899号パンフレット及び国際公開第2014/053804号パンフレットに記載されており、その内容は参照により組み込まれ、本出願の図13のステップ354でどのレジスタをスタックすることができるかを決定するために使用することができる。特に、国際公開第2013/117899号パンフレットの図12及び関連する説明を参照されたい。

【0165】

ステップ354において、被呼び出し側スタックフレーム182をスタックに保存することが決定された場合には、所定の署名190内で、TMIEクロスチェック情報192が指定され、これは、ステップ352で設定されたダミー例外戻りアドレス内のTMIEフラグ158と同じ値に設定される（あるいは代替の実施態様では、TMIEフラグ158が正しく設定されているかどうかをチェックするために使用することができる別の値に）。これは、TMIEフラグ158が使用されてTMIDフラグ34が制御レジスタ32に復元されると、例外から戻った後に不正な値の遷移無効化フラグTMID34を受信した後にセキュアコードが処理されるリスクがあるような、例外の発生後に動作する低セキュアコードがダミー例外戻りアドレス内のTMIEフラグ158を改ざんする可能性があるリスクに対する保護を提供する。これは、TMIEフラグ34で改ざんされたことに基づいて設定されたTMID34に基づいて生成されたスプリアス障害によって引き起こされるサービス拒否攻撃から保護する。

【0166】

バックグラウンドコード用のTMIDフラグ34を保持する値が呼び出し側スタックフレーム180に保存された場合、例外戻り値142内のTMIEフラグ158の場合のように、リンクレジスタ内の例外ハンドラにアクセス可能無しでセキュアに保つことができると考えるかもしれないため、呼び出し側スタックフレーム180ではなく被呼び出し側スタックフレーム182にTMIEクロスチェック値190が指定される理由を疑問に思うかもしれない。しかしながら、実際には、ステータスフィールドERETPSRが符号化されるべき多くの情報を既に有している可能性があり、従ってクロスチェック情報192のための空間がない可能性があるため、呼び出し側スタックフレーム180は、追加の情報のための多くの予備容量を有さない可能性がある。いくつかの実装形態では、TMIEフラグを追加するのに十分な空間がERETPSR値にあるが、ERETPSR値の残りの空間は依然として制限されており、アーキテクチャへの将来の追加のためにその空間を予約することが好ましい場合がある。被呼び出し側スタックフレーム182は、全てのタイプの例外について保存されるわけではないため、例外戻り時にTMIDフラグ34に復元される値を示すTMIE値158の主コピーを記憶することに依拠することはできない。従って、TMIEフラグ158がダミー例外戻りアドレスの一部としてリンクレジスタに配置されるが、クロスチェック情報190が被呼び出し側スタックフレーム182に含まれる図5及び図6に示される手法は、呼び出し側スタックフレーム180内に空間がないにもかかわらず、全てのタイプの例外がTMIDフラグ34の値を保持することを可能にするが、TMIEフラグ158が改ざんされるリスクがあるシナリオでは、これらはまた、追加の被呼び出し側スタックフレーム182を保存することによってレジスタ状

態が保護されている同じシナリオであり、従って、被呼び出し側スタックフレーム 182 に保存されたクロスチェック情報 192 は、T M I E フラグ 158 を改ざんから保護する。

【0167】

ステップ 356 において、例外エントリ遷移の一部として、例外制御回路 28 は、処理回路 4 をトリガして、制御レジスタ 32 内の T M I D フラグ 34 を 0 に設定し、ドメイン遷移がスレッドモードで有効にされていることを示す。これにより、例外の前に実行するコードに対してドメイン遷移が無効にされたかどうかにかかわらず、ドメイン遷移がスレッドモードで有効にされているかどうかの一貫したビューが全ての例外ハンドラに表示されることが保証される。このようにしてスレッドモードでドメイン遷移を有効にすることは、T M I D フラグ 34 が作成される前に書き込まれたソフトウェアとの過去との互換性のために必要とされ得る。

10

【0168】

ステップ 358 において、例外制御回路 28 は、例外処理ルーチンに分岐するように処理回路 4 を制御する。例外処理ルーチンの開始時に実行される命令のアドレスは、異なるタイプの例外をそれぞれの例外処理ルーチンアドレスにマッピングする例外ベクトルテーブルに基づいて決定することができる。例外処理ルーチンはハンドラモードで処理されるので、処理回路 4 が現在スレッドモードにあった場合、例外が取られたことに応答して、現在のモードがハンドラモードに切り替えられる。例外が取られた時点の現在のモードが既にハンドラモードであった場合、現在のモードへの変更は行われない。従って、ハンドラモードでいくつかのコードを処理した後、スレッドモードに切り替えるために前述したようにインターモダル関数呼び出しを実行することは可能であるが、最初にスレッドモードに例外を取ることは不可能である。

20

【0169】

また、例外エントリでは、例外制御回路 28 は、必要に応じて、処理回路 4 を任意選択的にトリガしてセキュリティドメインを切り替えることができる。異なる例外は、異なるセキュリティドメインに関連付けられ得る。いくつかの例外は、セキュアドメイン又は低セキュアドメインなどの特定のドメインで常に取りられる必要がある場合がある。例外が発生した時点で、現在のドメインで他の例外を取ることができる。従って、例外が現在のドメインとは異なるドメインで処理される場合、例外処理ルーチンに分岐するだけでなく、例外制御回路 28 はセキュリティドメインの変更もトリガすることができる。

30

【0170】

図 14 は、図 12 に示すタイプの関数呼び出し分岐命令ではない分岐命令に応答して実行されるステップを示すフロー図である。ステップ 400 において、処理回路は、ターゲットアドレスが予約済みアドレス範囲 130 内にあるかどうかをチェックする。そうではない場合、これは実行可能命令のアドレスへの通常に分岐命令である。ステップ 402 において、分岐がセキュアドメインから低セキュアドメインへの遷移を要求しているかどうかを検出される。これは、処理回路 4 の現在の動作ドメインをチェックし、分岐ターゲットアドレスの最下位ビットをチェックすることによって識別することができる。分岐ターゲットアドレスの最下位ビットが 0 である場合、これは、低セキュアドメインに戻る分岐を示す。あるいは、いくつかの実装形態では、セキュアドメインから低セキュアドメインへの遷移を要求する分岐は、専用ドメイン遷移分岐命令（B X N S と呼ばれ得る）の使用によって示され得る。専用ドメイン遷移分岐命令は、他のタイプの分岐命令とは異なるオペコード値によって検出可能であってもよく、又は他のタイプの分岐命令と同じオペコードであるが、セキュアドメインから低セキュアドメインへの遷移が要求されていることを示すパラメータ値を有してもよい。いくつかの実装形態は、セキュアから低セキュアドメインへの遷移要求が、分岐ターゲットアドレスの最下位ビットが 0 であることと、専用ドメイン遷移命令の使用との両方を必要とするように、2 つの手法を組み合わせることができる。

40

【0171】

分岐がセキュアから低セキュア分岐を要求していない場合、ステップ 404 において

50

、分岐は通常通り処理することができ、プログラムカウンタレジスタ R 1 5 は分岐命令によって指定された分岐ターゲットアドレスに基づいて更新され、プログラムフローは分岐ターゲットアドレスでの命令の実行を継続する（分岐の実行前と同じドメイン内で）。いくつかの実装形態では、全てのタイプの非関数呼び出し分岐が関数戻りに使用されるとは限らないことが理解されよう。例えば、命令オペコード内に符号化された即値に基づいてターゲットアドレスを決定する分岐は、常に同じ位置に分岐することができるため、プログラムフローが、そこから返される関数を呼び出した様々な異なる関数に分岐して戻る必要がある関数戻り分岐には適していない可能性がある。設計を単純化するために、いくつかの実装形態は、関数戻りを実行するために使用することができる非関数呼び出し分岐のサブセットについて図 1 4 に記載されたステップのみを実行することを選択することができる。上述したように、命令が専用の分岐命令でなくても（例えば、プログラムカウンタにロードされた値を書き込むロード命令であっても）、分岐を実行する効果を有することができる多くのタイプの命令が存在し得る。図 1 4 に記載のステップはまた、分岐を実行する効果を有する、そのような非分岐命令の少なくともサブセットに適用され得る。

【0172】

ステップ 4 0 2 において、分岐がセキュアから低セキュア分岐であると判定された場合、ステップ 4 0 6 において、処理回路は、現在のモードがスレッドモードであるかどうか、及びドメイン遷移無効化構成パラメータ（T M I D）が、ドメイン間遷移がスレッドモードで無効にされていることを示す 1 であるかどうかをチェックする。現在のモードがスレッドモードであり、T M I D が、ドメイン遷移がスレッドモードで無効にされていることを示す場合には、ステップ 4 0 8 において、セキュアドメインで処理される I N V T M I U s a g e F a u l t がトリガされる。これにより、セキュアスレッドを管理するセキュアオペレーティングシステムは、必要に応じて任意の M P U 又はスタックリソースを構成して、セキュアスレッドが低セキュアドメインでも動作することを可能にする。

【0173】

あるいは、ステップ 4 0 6 で、現在のモードがハンドラモードであるか、又は制御レジスタ 3 2 内の T M I D 値 3 4 が、ドメイン遷移（i n t e r s t a t i n g）が有効にされていることを示すために 0 であると判定された場合、ステップ 4 1 0 で、セキュリティドメインは低セキュアドメイン（N S）に切り替えられ、次いで、ステップ 4 0 4 で、ターゲットアドレスへの分岐が前述のように実行される。

【0174】

ステップ 4 0 0 において、ターゲットアドレスが予約済み範囲内にあると判定された場合、これは、分岐が関数戻り又は例外戻りのいずれかを表すことを意図され得ることを示す。ステップ 4 1 2 において、処理回路 4 の実行段 1 2 は、分岐のターゲットアドレスが、その最上位ビット部分において関数戻りプレフィックス 1 4 4 によって示される有効なダミー関数戻りアドレス 1 4 0 のうちの 1 つであるかどうかを判定する。ターゲットアドレスが有効なダミー関数戻りアドレス 1 4 0 のうちの 1 つである場合、ステップ 4 1 4 において、関数戻り分岐処理が実行され、これについては以下の図 1 5 を参照してより詳細に説明する。

【0175】

あるいは、分岐のターゲットアドレスが有効なダミー関数戻りアドレスのうちの 1 つではない場合、ステップ 4 1 6 において、図 5 に示すように、ターゲットアドレスが有効なダミー例外戻りアドレス 1 4 2 のうちの 1 つであるかどうか判定される。そうでない場合、アドレスは実行不可能な命令アドレスを表し、例外関数戻りをシグナリングするために使用されるダミーアドレスではないため、ターゲットアドレスに分岐しようとする試行はエラーであり、ステップ 4 1 8 で障害がトリガされ得る。あるいは、分岐が実行不可能アドレスへのものであることを検出したことに応答して障害を直接トリガする代わりに、分岐を通常通り実行して、プログラムカウンタレジスタを分岐ターゲットアドレスに設定することができ（そのため、図 1 4 はステップ 4 1 6 の「N」判定の場合にステップ 4 1 6 からステップ 4 0 4 に遷移する）、その後、命令フェッチ回路 8 が実行不可能なアド

10

20

30

40

50

レスから命令をフェッチしようと試行すると、これにより障害がシグナリングされ得る。従って、障害の生成は、分岐によって間接的に引き起こされ得るが、分岐命令自体に応答して直接シグナリングされない場合がある。いずれの手法も有効であり得る。

【0176】

ステップ416において、ターゲットアドレスが有効なダミー例外戻りアドレス142のうちの1つであると判定された場合には、ステップ420において、少なくとも例外戻りが低セキュアドメインからセキュアドメインへの遷移を含む場合（ダミー例外戻りアドレス内のセキュリティドメインフラグ156から判定することができる）、ダミー例外戻りアドレス142内のTMIEフラグ158と、呼び出しスタックフレーム182の一部として保存されたTMIEクロスチェック情報190との間でクロスチェック比較が行われる。不一致が検出された場合、ステップ422において、これは、リンクレジスタ内のTMIEフラグ158を改ざんしようとする試行がなされたこと、又は代替として、不注意によるエラーが生じたことを示し得るので、障害がトリガされる。TMIEフラグ158がクロスチェック情報192と一致するか、あるいは例外戻り遷移がこのクロスチェックを必要としないものである場合、方法はステップ424に進み、そこで例外戻りが処理される。

【0177】

ステップ420及び424における例外戻り処理はまた、先に参照されたPCT国際公開第2013/117899号パンフレット及び国際公開第2014/053804号パンフレットに記載されているような他の動作も含み得る。ステップ420でTMIE値をチェックする前に、ステップ424での例外戻り処理は、スタックからレジスタ14へのレジスタ状態のアンスタックを含むことができる。アンスタックが呼び出し側スタックフレーム180のみのアンスタックを含むべきか、又は呼び出し側及び被呼び出し側スタックフレーム180、182の両方を含むべきかの判定は、例外を取ったときに例外戻り値142の一部として記憶され得るステータスフラグに基づいて行うことができる。どのレジスタをアンスタックする必要があるかについての決定の詳細については、国際公開第2013/117899号パンフレットの図13又は国際公開第2014/053804号パンフレットの図13を参照されたい。

【0178】

また、例外戻り処理は、所定の署名190が被呼び出し側スタックフレーム182の一部としてスタック上に存在すると予想されるかどうかのチェックを含むことができる。これは、国際公開第2014/053804号パンフレットの図14に従って決定することができる。所定の署名190が存在すると予想されるが、所定の署名と比較したスタック上の位置の内容が一致しない場合、これは、関数呼び出しに回答して保存されたスタックフレームに基づいて例外戻り機構が試行された事例を検出することができるため、障害がトリガされ得る。障害の発生は、国際公開第2014/053804号パンフレットの図7に示されている。

【0179】

例外戻り処理動作はまた、ロードされたスタックフレーム内のERETPSR値に基づいて復元された例外番号値36を更新すること、及び復帰される処理の優先度に基づいて例外優先度レベルを変更することなどの他の動作を含むことができる。

【0180】

更に、ステップ424において、分岐されたダミー例外戻りアドレス内に含まれるTMIEフラグ158に基づいて、制御レジスタ32内のTMID値34を復元することができる。TMIEDフラグ34の復元値は、対応する例外が取られる前にTMID34が有していた値と一致するように、スレッドモードにおけるドメイン遷移が有効にされるべきか無効にされるべきかを示すために、TMIEフラグ158の値の逆数に対応することができる。ここでは説明されていない例外戻りの処理では、他の多くの動作も実行できることが理解されよう。

【0181】

図15は、図14のステップ414において関数戻りアドレスへの分岐に対して実行される動作をより詳細に示す。ステップ440において、処理回路は、分岐の分岐ターゲットアドレスとして検出されたダミー関数戻りアドレス140がインターモーダル関数戻りを示すかどうかを判定する。例えば、これは、対応する関数戻り値がハンドラからスレッドモードへのインターモーダル呼び出しに設定されたことを指定するために、インターモーダルインジケータ146が0であるかどうかのチェックであり得る。試行された関数戻りがインターモーダル関数戻りではない場合、それはドメイン間関数戻りである（対応する関数呼び出しがインターモーダル呼び出しでもドメイン間呼び出しでもない場合、リンクレジスタは単に関数戻りアドレスに設定され、ダミー関数戻りアドレスを使用する必要はなかったはずであるため）。

10

【0182】

ドメイン間関数戻りのためのステップ442において、スレッドモード遷移無効化フラグTMID34のチェックが実行される。現在のモードがスレッドモードであり、ドメイン遷移がスレッドモードで無効にされていることを示すためにTMIDフラグ34が1である場合、ステップ444において、INV TMI Usage Faultは、図12のステップ308又は図14のステップ408と同様に再びトリガされる。しかしながら、ステップ444において、INV TMI障害は、分岐が試行された時点で現在のドメインがいずれのドメインであっても処理される（すなわち、ステップ308及び408の場合のようなセキュアドメインではなく、ドメイン間関数戻りが試行されているソースドメイン）。これにより、分岐を要求したスレッドの管理を担当するプロセスが障害を処理し、必要に応じて宛先ドメインに必要なリソースを構成できることが保証される。

20

【0183】

ステップ442において、現在のモードがハンドラモードであると判定されたか、又は、遷移が有効にされていることを示すTMIDフラグ34がゼロであると判定された場合、ステップ446において、分岐を続行することを許可することができる。分岐が行われる関数戻りアドレス174は、現在の動作モード、遷移されたセキュリティドメインに基づいて、そしてスレッドモードである場合には、スタックポインタ選択値30に基づいて決定された関連スタックから読み出される。ドメイン間遷移が許可されているかどうかを検証するために必要な完全性チェックが実行される。これらの完全性チェックは、例えば、関数戻りスタックフレーム172内の例外番号179が、スタックフレームが現在の動作モードと同じモードで作成されたことを示すかどうか（すなわち、例外番号179は、現在の動作モードがスレッドモードである場合には0であり、現在の動作モードがハンドラモードである場合には非ゼロである）、並びにHDTフラグ178がドメイン間関数戻りについて予想されるようにゼロであるかどうかのチェックを含むことができる。また、許可されるドメイン遷移を制限する任意の他の要件をチェックすることができる。完全性チェックに合格した場合、関数戻りアドレス174への分岐は許可されて成功し、その後、後続の処理は、関数戻りアドレス174からフェッチされた命令から継続する（これは、図14のステップ400で遭遇した試行された分岐のターゲットアドレスとして実際に指定されたダミー関数戻りアドレス140ではなく、関数戻りスタックフレーム172からロードされた実関数戻りアドレスである）。関数戻りスタックフレーム172を指し示すスタックポインタは、スタックデータ構造からの関数戻りスタックフレームのアンスタック及び除去を考慮するように調整することができる。必要な完全性チェックのいずれかが失敗した場合、障害がシグナリングされる（これは、セキュアドメインで処理され得るセキュリティ障害であり得る）。

30

40

【0184】

ステップ440において、試行された関数戻りがインターモーダル関数戻りであると判定された場合、ステップ450において、処理回路は、現在のドメインが低セキュアドメインであり、ダミー関数戻りアドレス140内のセキュリティドメインインジケータ148が、セキュアドメインへの戻りが要求されていることを示すかどうかを判定する。これが該当する場合、ステップ452において、現在のドメイン（低セキュアドメイン）で

50

処理されるべきである I N V P C 使用障害がトリガされる。この障害は、上記の図 9 に示すように、低セキュアスレッドモードからセキュアハンドラモードへの禁止された組み合わせドメイン／モード遷移を防止するためにトリガされ、そうでなければセキュリティの脆弱性を引き起こす可能性がある。

【0185】

試行されたインターモーダル関数戻りが、低セキュアドメインからセキュアドメインへのものではない（同じドメイン内に留まるか、又はセキュアドメインから低セキュアドメインへのものである）場合、方法はステップ 454 に進み、構成されている場合、図 12 のステップ 328 と同様にエラー同期バリア動作がトリガされる。ここでも、インターモーダル関数戻りでエラー同期バリアをトリガすることによって、これは、エラーがインターモーダル関数戻りの前に実行されている低特権処理に起因する可能性がある場合、侵入的なシステムリセット操作の必要性を低減するためにエラーの特定を助けることができる。

10

【0186】

ステップ 456 で、インターモーダル関数戻り命令に応答して、処理回路は、戻りが行われている宛先セキュリティドメインに関連付けられた主スタックから関数戻りスタックフレーム 172 をロードする。ステップ 458 において、処理回路は、完全性クロスチェック動作を実行し、その完全性クロスチェックの結果が成功したかどうか判定する。この完全性クロスチェックは、関数戻りスタックフレーム上のいくつかの項目をチェックし得る。例えば、処理回路は、関数戻りアドレス 174 がアドレスの予約済み範囲 130 のうちの 1 つであるかどうかをチェックすることができる。関数戻りアドレスが予約済みアドレス範囲のうちの 1 つである場合、これは、実際には関数戻りスタックフレーム 172 内の関数戻りアドレス 174 と同じスタック上の相対位置に所定の署名 190 を含む例外スタックフレームであるスタックフレームに基づいて関数戻りが試行されるというインジケーションであり得、例外及び関数エントリと戻りとのこの不一致は、攻撃のための手段であり得る。従って、スタックフレーム上の関数戻りアドレス 174 が予約済みアドレス 130 である場合、完全性クロスチェックは失敗する。

20

【0187】

また、完全性クロスチェックの一部として、処理回路は、制御レジスタ 20 内の例外番号値 36 に基づいて、現在のモードがスレッドモードであるかどうかを確認する。現在のモードがハンドラモードである場合、完全性クロスチェックは失敗する。

30

【0188】

また、完全性クロスチェックは、関数戻りスタックフレーム内の例外番号フィールド 179 をチェックすることを含み、ゼロの場合、インターモーダル関数戻りがスレッドモードからハンドラモードに戻ると予想されるため、完全性クロスチェックが失敗するようにトリガする。

【0189】

また、完全性クロスチェックは、HDT フラグ 178 を使用して、インターモーダル呼び出し分岐命令に応答して関数戻りスタックフレームが保存されたかどうかを検出する。HDT フラグが 0 である場合、これは、インターモーダル関数呼び出しではなくドメイン間関数呼び出しに応答して関数戻りスタックフレームがセットアップされたことを示し（図 12 のステップ 312 を参照）、そのため、完全性クロスチェックは失敗する。ここでも、これは、関数呼び出し及び戻りの不一致シーケンスを防止して、スタックフレームの不一致に基づく攻撃の機会を制限し、また、一般にアクセス可能なリンクレジスタに保存されている間にダミー関数戻りアドレス 140 内のインターモーダルフラグ 146 の潜在的な修正に対して保護し、HDT フラグ 178 は、インターモーダルフラグが正しいことをチェックするためのクロスチェック値として機能する。

40

【0190】

完全性クロスチェックのいずれかが失敗した場合、ステップ 460 において、セキュリティチェックの失敗が発生し、このタイプの障害が分岐を試行するコードに関連付けら

50

れた宛先セキュリティドメインで処理されるべきであることを示すINVP C使用障害がトリガされる。

【0191】

あるいは、完全性クロスチェックに全てパスした場合には、ステップ462において、インターモダル関数戻りを作用させることができる。インターモダル関数戻りが実行されると、ステップ462において、いくつかのアクションが実行される。セキュリティ状態は、ダミー関数戻りアドレス140内のセキュリティドメインインジケータ148によって示されるセキュリティ状態に更新される。これにより、セキュリティドメインが、対応するインターモダル関数呼び出しが行われる前に実行されていたドメインに復元されることが保証される。

10

【0192】

また、ステップ462において、インターモダル関数戻りの前に、TMIDフラグ34が0であったか1であったかを記録するためのインジケーションが保存される。このインジケーションは、例外エントリに保存されるTMIEフラグ158とは別個のものである。TMIDの以前の値のインジケーションが記憶される位置は、実施形態ごとに異なり得る。例えば、インジケーションは、汎用レジスタ16に保存することができ、又は後続の条件付き命令によって直接テストすることができるフラグレジスタ内の条件フラグなどの制御レジスタ20に保存することができ。

【0193】

また、ステップ462において、制御レジスタ32内のTMID値34は、スレッドモードドメイン遷移が再び有効にされることを示すために0に設定されてもよい。これにより、インターモダル関数戻りの前に実行された例外ハンドラコードの非特権セクションがセキュリティドメインを切り替えようと試行したかどうかにかかわらず、より特権的なラップコードに戻ると、ドメイン遷移無効化構成値34が一貫した状態になり得ることが保証される。

20

【0194】

また、インターモダル関数戻りのステップ462において、関数戻りスタックフレームがロードされた宛先スタックポインタは、スタックから除去されたアンスタックされたデータを考慮するように調整されてもよい。また、宛先ドメインのスタックポインタ選択値30は、主スタックポインタが選択されるべきであることを示すように更新され、現在の動作モードは、関数戻りスタックフレーム172内の例外番号フィールド179から例外番号レジスタ36を復元することによってハンドラモードに切り替えられる（ステップ458における以前の完全性チェックは、例外番号フィールド179が非ゼロであること、従って例外番号レジスタ36を復元すると現在のモードがハンドラモードに切り替わることをチェックしていてもよい）。ステップ464において、関数戻りスタックフレーム172から取得された実関数戻りアドレス174への分岐が実行され、その結果、処理される次の命令は関数戻りアドレスにおける命令である。

30

【0195】

従って、後続の関数戻りがインターモダル関数戻りであるべきかどうかをシグナリングするために関数（ISR201の本体）を呼び出す際に保存されるダミー関数戻りアドレス140を使用することによって、これは、関数戻りをトリガする命令自体がモードを切り替えているかどうかを認識する必要がないことを意味し、これは、インターモダル関数呼び出し及び戻りのサポートを全く認識せずに書き込まれたレガシー例外ハンドラコードが依然として使用され得ることを意味する。これは、上記の図7及び図8に関して説明した割り込み禁止シナリオに特に有用である。すなわち、関数戻り分岐命令は、それがインターモダル分岐であるかどうかにかかわらず同じ符号化を有することができる。

40

【0196】

図14及び図15では、所与のターゲットアドレスへの分岐をトリガする命令は、プログラムカウンタレジスタR15を分岐ターゲットアドレスに修正させる様々な命令のいずれかであり得ることに留意されたい。場合によっては、これは専用分岐命令であっても

50

よいが、分岐ターゲットアドレスをプログラムカウンタレジスタに移動させるレジスタ移動命令、又はスタックデータ構造から情報をポップしてプログラムカウンタレジスタにロードするスタックポップ命令であってもよい。特に、分岐がダミー関数戻りアドレス140又はダミー例外戻りアドレス142のうちの一方に対するものであり、対応する関数／例外戻りの前に更なる関数呼び出しが行われた場合にスタックに保存されることになり得る場合、コンパイラが、戻りアドレスを最初にリンクレジスタにロードし、次いでリンクレジスタを指定する戻り分岐を実行するのではなく、スタックフレームからプログラムカウンタレジスタに直接ポップするコードを生成する方が効率的であり得る。従って、実関数戻りアドレス、ダミー関数戻り又はダミー例外戻りアドレスは、関数の呼び出し時又は例外を取る時にリンクレジスタに最初に保存され得るが、対応する戻りが実行される時点までに、それはもはやリンクレジスタ内になくてもよく、図14及び図15の分岐動作をトリガする命令は、実際には専用分岐命令でなくともよいことが理解されよう。

【0197】

図16A～図16Dは、上述のチェックを使用して防止することができる可能な攻撃の例を示している。これらの攻撃は、例外又は関数呼び出し及び戻りの不一致シーケンスに基づくことができ、呼び出しによってセットアップされ、戻り時に予想される不一致スタックフレームを利用しようと試みることができる。

【0198】

図16Aの例では、セキュアハンドラモードでの処理中にステップ1で例外が発生し、例外は低セキュアドメインで取られるため、呼び出し側及び被呼び出し側スタックフレーム180、182の両方がスタックに保存される。図16Aのステップ2において、一連の例外戻り及び／又は関数呼び出しが行われ、最終的にセキュアスレッドモードで処理が実行される。セキュアスレッドモードになると、ステップ3で、ステップ1で設定された例外スタックフレーム180、182に基づいてインターモーダル関数戻りを実行する試行が行われる。例えば、これは、ダミー関数戻り値140のうちの1つに分岐することによって行うことができる。このタイプの攻撃は検出することができ、なぜなら、関数戻りに対する図15のステップ458の完全性クロスチェック検査が、関数戻りアドレス174がアドレス130の予約済みアドレス範囲のうちの1つであることを検出するためである（所定の署名190は、予想される関数戻りスタックフレーム172ではなく、ステップ3でロードされた例外スタックフレームの下部にあるため）。

【0199】

図16Bは、ステップ1でセキュアドメインから低セキュアドメインにドメイン間関数呼び出しが行われ、関数戻りスタックフレーム172がセキュア主スタックに保存される第2の例を示す。再び、ステップ2では、一連の他の遷移が実行され、次いでステップ3で、ステップ1で設定されたドメイン間スタックフレームに基づいて、インターモーダル関数戻りが試行される。このインターモーダル関数戻りは、対応する呼び出しがインターモーダル呼び出しではなくドメイン間呼び出しであったため、成功することを許可されるべきではない。これは、関数戻りスタックフレーム内のHDT値178が、ステップ1におけるドメイン間関数呼び出しにおいて0に設定され、ステップ3におけるインターモーダル関数戻りについては1でなければならない、従って、図15のステップ458における完全性クロスチェックは失敗するため、検出される。

【0200】

図16Cは、最初にステップ1でインターモーダル関数呼び出しが行われ、これはハンドラモードからスレッドモードへ非特権化し、次いでステップ2で一連の他の遷移が行われ、その後、ステップ3でドメイン間関数戻りを実行しようとする試行が行われ、ステップ1で設定されたインターモーダル関数戻りスタックフレームに基づいて、セキュアドメインから低セキュアドメインへの遷移が行われる別の潜在的な攻撃を示す。図15のステップ446において、関数戻りスタックフレーム172内のHDT値178が0に等しいかどうかの完全性チェックが実行され、HDT値が1に等しい場合、障害がトリガされ得る。従って、インターモーダル関数呼び出しが行われてドメイン間関数戻りが試行され

10

20

30

40

50

る図16Cのシナリオでは、HDTチェックは不一致を検出し、ステップ1で作成されたインターモーダルスタックフレームがステップ3でドメイン間スタックフレームと見なされ得ないことを保証する。ステップ446及び458におけるHDTチェックは、ダミー関数戻りアドレス内のインターモーダルフラグM 146と関数戻りスタックフレーム内のHDTフラグ178との間の比較と同等であると考えることができ、この比較で識別された不一致は、インターモーダル戻りの場合（図16B）又はドメイン間戻りの場合（図16C）のいずれかで障害をトリガすることになることに留意されたい。

【0201】

引き続き図16Cを参照すると、ステップ3で（インターモーダルフラグM=0で）ドメイン間戻りを実行する代わりに、攻撃者は（インターモーダルフラグM=1で）インターモーダル関数戻りを試行することができた。対応する関数呼び出しは実際にはインターモーダル呼び出しであったが、ステップ3の前の現在のモードはスレッドモードではなくハンドラモードであるため、試行されたインターモーダル関数戻りと一致しない。この場合、インターモーダル呼び出し値146は、ステップ1でインターモーダル呼び出しに設定された関数戻りスタックフレーム内のHDT値178と一致し得るが、現在の状態がハンドラモードであるときにインターモーダル関数戻りは許可されず、これは、現在のモードがスレッドモードであるかどうかをチェックする図15のステップ458の完全性チェックによって捕捉されるため、攻撃は防止される。

【0202】

図16Dは、セキユアドメインから低セキユアドメインへのドメイン間分岐命令に起因して、ステップ1でドメイン間スタックフレームがセットアップされ、続いてステップ2で一連の遷移が行われてセキユアドメインに戻り、次いでステップ3で、ステップ1でドメイン間分岐によって設定されたスタックフレームを使用してインターモーダル関数戻りを実行する試行が行われる別の例を示す。関数戻りスタックフレーム172内の例外番号フィールド179は、ステップ1で設定されたスタックフレームが、インターモーダル関数戻りがエントリを要求するハンドラモードではなくスレッドモードに設定されたことを示すので、このタイプの攻撃は成功できない。あるいは、ステップ2'に示すように、ドメイン及びモードの非セキユアスレッドモードからセキユアハンドラモードへの組み合わせの遷移が試行された場合、ステップ450で許可されていない組み合わせの遷移が防止されるため、これは再び失敗する。

【0203】

一部のチェックが重複する可能性があり、同じ種類の攻撃が2つ以上の種類のチェックの失敗をトリガする可能性があることが理解されよう。いくつかのアーキテクチャ実装形態では、これらのチェックの一部が冗長であると思われる場合、それらはアーキテクチャ要件から除外されてもよく、従って、保護するために重要であると思われるものに依拠して、異なるタイプの攻撃防止機構の全てを実行することは必ずしも必須ではない。しかしながら、これらのチェックのうちの任意の1つ以上が提供されてもよい。図16に示す攻撃は、攻撃者が元の遷移、従って作成されたスタックフレームのタイプと一致しない戻り動作（関数又は例外戻りのいずれか）を試行することによってシステムの状態を破壊しようとする、同様のミスマッチ型攻撃の膨大な数の異なる組み合わせのほんの一部であることが理解されよう。

【0204】

図12、図14、及び図15のステップ307、406及び442は、ドメイン遷移無効化構成パラメータ（TMID）34を、関数呼び出し分岐、非関数呼び出し分岐、及び関数戻りについてそれぞれチェックする具体例を示す。しかしながら、一部のアーキテクチャでは、ドメイン間の遷移をトリガすることが可能な他の有効な手段もあり得、図17は、TMID値のチェックを処理するためのより一般的な手法を示す。ステップ500において、現在のモード内のセキユアから低セキユアドメイン遷移、又は現在のモード内の低セキユアからセキユアドメイン遷移のいずれかを実行する可能な試行が検出される。この試行は、ドメイン遷移を実行する実際の試行であってもよいし、他のセキュリティ又

10

20

30

40

50

は完全性チェックに合格した場合にそのようなドメイン遷移をトリガすることがあるが、T M I Dチェックが、ドメイン遷移が実際に許可されるかどうか、又は別のチェックが失敗して障害が発生するかどうかをまだ検出することができないほど十分に早く実行されている場合には、常にドメイン遷移をトリガするとは限らない動作であってもよい。

【0205】

ステップ502において、ドメイン遷移をトリガする可能性のある試行が検出された場合、現在のモードがスレッドモードであるかどうか判定される。現在のモードがスレッドモードである場合、ステップ504において、処理回路は、T M I D値34がスレッドモードにおいてドメイン遷移が無効にされていることを示すかどうかを判定する。現在のモードがスレッドモードであり、ドメイン遷移がスレッドモードで無効にされている場合、ステップ506において、無効化ドメイン遷移障害（例えば、上記のI N V T M I U s a g e F a u l t）がシグナリングされる。障害は、試行されたドメイン遷移が作用される前に現在のセキュリティドメインで処理されるため、試行されたドメイン遷移がセキュアから低セキュアの場合、障害はセキュアドメインで処理され、試行された遷移が低セキュアからセキュアへの場合、障害は低セキュアドメインで処理される。また、場合によっては、いくつかのシンドローム情報を、障害が発生した試行された遷移に関する情報を示す制御レジスタに記憶することができる。いくつかの実装形態では、シンドローム情報を設定することは、U s a g e F a u l tステータスレジスタ（U F S R）にI N V T M Iビットを設定することを含むことができる。

【0206】

ステップ502で、現在のモードがハンドラモードであると判定された場合、又はステップ504で、T M I D値34が、スレッドモードでドメイン遷移が可能であることを示すことが検出された場合、方法はステップ508に進み、少なくとも、試行された遷移が低セキュアドメインからセキュアドメインへのものである場合、任意の他の必要なセキュリティチェックに合格したかどうかチェックされ、合格した場合、ステップ510で遷移が許可される。セキュリティチェックが必要とされなかった場合（例えば、低セキュアコードを攻撃しないようにセキュアコードを信頼することができるため、いくつかのタイプのセキュアドメインから低セキュアドメイン遷移のためのチェックは必要とされない場合がある）、セキュリティチェックを実行することなく（すなわち、そのような遷移のためにステップ508を省略することができる）、ステップ510で遷移を許可することができる。あるいは、セキュリティチェックが失敗した場合、ステップ512において、障害がトリガされてもよく、これは、許可されていないセキュア状態からのエントリ又は退出に関連するため、セキュアドメインにおいて処理されてもよい。ステップ512でトリガされる障害のタイプ及び設定される任意のシンドロームは、失敗したセキュリティチェックのタイプに依存し得る。

【0207】

図17は、連続して実行されるステップ502及び504を示しているが、いくつかの例では、逆の順序又は並行して実行することもできる。

【0208】

図18～図20は、T M I D値34を含むユースケースのいくつかの例を示している。図18は、例外ハンドラがスレッドモードで処理される本体201を含むように非特権化されるように、インターモーダル呼び出しを含む図8の例に本質的に対応する。この例では、例外は非セキュア（低セキュアとしても知られる）例外であるため、ラップ203及び本体201を含む例外ハンドラ処理は全て非セキュアドメイン内にある。セキュア／低セキュアドメイン境界を横断する任意のスレッドは、（M P U構成及びメモリに割り当てられた関連するスタック構造のスタックポインタなどの）セキュアリソースと低セキュアリソースの両方を確立する必要があるため、一方のドメインでのみ動作する図18に示すT2などのスレッドの場合、メモリの浪費及びコンテキスト切り替え時間の増加を回避するために、他方のドメインにリソースを割り当てることを回避することが望ましい場合がある。従って、好ましくは、システムは、追加のリソースが使用されるかどうかにかか

わらず、全てのスレッドに追加のリソースを割り当てるコストを回避することができる。全てのスレッドに対してセキュアリソース及び低セキュアリソースの両方を構成しないこの能力はまた、図18に示すようにスレッドモードに戻るために非特権化される必要がある割り込みの割り込み待ち時間を改善するのに役立つことができ、これは、セキュアMPU及び非セキュアMPUの一方並びにセキュアスタックポインタ及び非セキュアスタックポインタの一方を構成することを省略できるため、ラップコード203が実行するのにそれほど時間を要さないことを意味する。

【0209】

これを処理するための1つの手法は、スレッドに単一のセキュリティ状態（例えば、図18のスレッド2の例における低セキュアドメイン）のスタック及びMPU構成を割り当てることができ、次いでそのスレッドが他のセキュリティドメインに切り替えを試行する場合にのみ、他のドメインのリソースを割り当てることができることであり得る。しかしながら、TMIDパラメータ34がなければ、これを実施し、遅延割り当てをトリガするのに必要な障害を生成する唯一の方法は、MPU26を再構成することであり、その結果、プロセスがアクセスする許可を持たないか、又は領域許可データがMPU26に定義されていないアドレスにアクセスしようと試行すると、アドレス障害が生成される。これは、MPU構成がプロセスごとではなくスレッドごとに処理される必要があり、また、両方のセキュリティドメインのMPU26-S、26-NSを含むため、かなり多いセキュア／非セキュア対話が必要になり、セキュリティドメインの1つのMPU構成操作をスキップできるという利点がなくなるため、欠点となる。例えば、セキュアスレッドを実行する前に、低セキュアオペレーティングシステムは、低セキュア状態に分岐しようと試行すると障害が発生するように、低セキュアMPU26-NSを再構成する必要があり得る。従って、障害を生成するためにMPU26を使用することは困難であり、性能集約的であり、セキュア及び低セキュアオペレーティングシステムコードを提供する異なる当事者のために符号化することは困難であり得る。あるいは、どのスレッドが1つのセキュリティドメインのみを使用し、どちらも使用するかが事前に分かっている場合、この情報は遅延割り当てを回避するために使用することができるが、ソフトウェアがクラッシュして他の状態にアクセスしようと試行する可能性があるため、これに依拠することはできず、又は攻撃者がこれを利用しようと試行する可能性があるため、リソースが1つのドメインに対してのみ設定されているときに他のドメインへの遷移の検出に障害を生成する機構を提供することが依然として望ましい場合がある。

【0210】

代わりに、スレッドモードでドメイン遷移を選択的に無効にするTMIDパラメータ34を提供することにより、これは、（図18のT2の場合のように）低セキュアドメインにのみあると予想されるスレッドが、ドメイン遷移を無効にするように設定されたそのTMIDパラメータ34を最初に有することができ、次いで、ドメインを切り替えることを試行した場合にのみ、適切なリソースを構成できるように障害を生成することができることを意味する。

【0211】

従って、図18は、（インターモーダル呼び出し分岐命令のBLXT変形が例外ハンドラ本体201を呼び出すために使用されるので）非特権化された例外ハンドラ本体201に入るとスレッドモード遷移が無効にされる例を示す。この場合、例外ハンドラの本体201は、低セキュアドメイン内に留まり、セキュアドメインに切り替えることを試行せず、その結果、本体201が後で関数をラップコード203に戻す時点までに、処理リソース又はメモリ割り当ては、スレッドT2用のセキュアドメインのMPU又はスタックリソースを構成する際に浪費されていない。従って、これにより性能が向上する。本体201の処理が完了すると、ステップ604において、インターモーダルダミー関数戻りアドレス140への分岐が実行される。これはハンドラモードへの戻りをトリガし、いくつかの実装形態では、図15のステップ462で説明したようにクリアされる前に、制御レジスタ32内のTMIDフラグ34のインジケーションを（例えば汎用レジスタに）保存さ

せることができる。プログラムフローはその後、ラッパ203に戻り、そこでラッパは、TMIDフラグ34の保存されたインジケーションをチェックし、ステップ604におけるインターモーダル関数戻りの前に、TMIDフラグ34が1であったと判定する。従って、ラッパ203は、セキュア状態に関連付けられたリソースが、バックグラウンドスレッドT2に対して依然として構成されていると決定し、これらのリソースは、例外戻りを実行する前に再構成される必要がないことを判定して、T2のバックグラウンド処理に戻る。

【0212】

図19は同様のシナリオを示しているが、この場合、BLXT命令を使用して非特権例外ハンドラ本体201を呼び出し、従ってスレッドモードにある間にドメイン間遷移を無効化した後、本体は今度はセキュアドメインへの切り替えを試行し、これは前述したように使用障害をトリガする。この障害により例外が発生し、例外はハンドラモードのハンドラコード608によって処理され、その後、例外は、セキュアMPU26-S及びセキュアスタックポインタレジスタ内のスレッドT2のためのリソースの割り当て又は構成を要求することができ、その後、例外ハンドラ本体201に戻り、図19のポイント600において、低セキュアからセキュアドメイン遷移を再試行することができる（処理が非セキュアT2に戻るポイントとドメイン間遷移600との間のギャップは、明確にするために図19では誇張されており、ドメイン間遷移600は、T2に戻った後の最初の命令で実際に発生し得ることが理解されよう）。セキュアドメイン内にあることが必要とされる処理のセクションを終了した後、ポイント602でドメイン間戻りが実行されてもよく、その後の処理は図18と同じである。

【0213】

図19は、図19の部分604でインターモーダル戻りを実行すると、非特権例外ハンドラ本体201からラッパコード203に戻る遷移のために、そのインターモーダル戻りの前に、スレッドT2に対して実行されるスレッドモードの非セキュア処理のTMID値34が0であったか1であったかを示すインジケーションが記憶されるシナリオを示す。これは、ラッパコード203のドメイン遷移を再び有効にするために、インターモーダル戻りのTMIDフラグをクリアする前に行われる。図15のステップ462で上述したように、TMID34がゼロであったか、図19のステップ604の前のものであったかのこのインジケーションは、制御レジスタ、汎用レジスタ、又は条件コードのうちの1つなどの任意の便利な場所に保存することができる。これは、ハンドラモードで処理されるように非セキュアラッパコード203に処理が戻されると、610における後続のコードが図18に示される状況を図19に示される状況から区別することを可能にするため、有用である。図18に示すシナリオのように、セキュアドメインには処理がなかったため、セキュアスタックポインタ及びMPU構成は、セキュアドメインにおけるバックグラウンドコードの以前の処理中にポイント606で以前に設定されたままであり得る。対照的に、図19では、ステップ601で試行されたスレッドT2のセキュアドメインへのエントリ（ステップ600で再試行されたときに続いて成功した）は、図19のポイント608でスレッドT2のMPU／スタックリソースの構成が関連するMPU内容及びスレッドモード用のスタックポインタレジスタを変更したことを意味する。従って、図18及び図19の部分610で実行された残りのラッパコード203は、スレッドT1のセキュア部分に処理を戻す前に、部分606に存在していた情報を復元するためにセキュアスタックポインタ及びセキュアMPU構成を更新する必要があるかどうかを判定する必要がある。この決定を行うために、インターモーダル関数戻りにおけるステップ604の前にTMIDが0であるか1であるかのインジケーションを使用することができる。

【0214】

他の実装形態では、ドメイン遷移が保存されるべきインターモーダル戻りの前にスレッドモードで無効又は有効にされたかどうかのインジケーションは必須ではなく、この場合、図18及び図19のステップ610において、ラッパコード203は、ステップ606において存在していたセキュアリソースを復元するための動作を単に常に行うこと

10

20

30

40

50

ができる。しかしながら、スレッドモード遷移が無効にされたかどうかのインジケーションを保存する図 19 に示す例は、必要でない場合に冗長スタックポインタ更新動作を回避することによって、性能を向上させることができる。

【0215】

比較のために、図 20 は、IRQ 例外ハンドラが非特権化されると、今度は例外ハンドラがセキュアドメインと低セキュアドメインの両方を使用することが予想される別のシナリオを示しており、そのため、ポイント 620 で実行されるラップコード 203 の間に、両方のドメインのためにセキュア及び低セキュアリソースを構成するための適切な動作が実行され、スレッドモードで非特権状態で実行される例外ハンドラの本体 201 が、図 18 及び図 19 のような BLXT の代わりに、インターモーダル関数呼び出し分岐の BLXTI 変形で呼び出される。従って、ハンドラモードからスレッドモードへのインターモーダル関数呼び出しの実行時に、TMID 値 34 は、ドメイン遷移がスレッドモードで有効にされていることを示すために 0 に設定され、従って、割り込みハンドラ本体がセキュアドメインに遷移しようと試行するステップ 622 において、これは許可され、図 19 の例のように障害は必要とされない。ここでも、インターモーダル関数戻り 64 において、インターモーダル戻りの前に TMID が 0 であったというインジケーションを保存することは、626 において実行されたラップコード 203 によって、スレッド T1 に関連付けられたセキュア側スタックポインタ又は MPU 構成を、割り込みが発生する前にポイント 628 にあった状態に復元する必要があると判定するために使用することができる。ラップコードのいくつかの実装形態では、ラップは、本体 201 を実行する前に両方のドメインの構成を実行したため、ラップは、バックグラウンドスレッド T1 に戻る前に両方のドメインのリソースを復元する必要があることを事前に知ることになるため、ラップコード 203 は、図 20 の例では、インターモーダル関数戻り中に保存された TMID フラグのインジケーションをチェックしない場合がある。これは、インターモーダル関数戻り中に保存された TMID のインジケーションを冗長にチェックするよりも単純な手法であると思われるが、異なる例外ハンドラのための異なるラップ間でコードを共有することがより困難であることを意味し得る。従って、両方のアプローチが有用であり得る。

【0216】

図 21 は、ドメインを切り替える試行において TMID 値 34 をチェックすることが有用であり得る、より具体的なシナリオの別の例を示す。これは、図 17 の一般的なアプローチに含まれる別の特定の例である。ドメイン遷移が生じ得る別の方法は、低セキュアドメインにおいて、SAU24 によって定義されたセキュアアドレス領域のうちの 1 つのアドレスから命令をフェッチしようと試行する場合である。ステップ 700 において、命令フェッチ回路 8 は、現在のドメインが低セキュアドメインであり、SAU24 によって定義されたセキュア領域のうちの 1 つにアドレスを有する命令をフェッチする試行があったかどうかを判定する。そうでない場合、命令フェッチは通常通り継続する。現在のドメインが低セキュアドメインであり、命令がセキュアアドレス領域からフェッチされようと試行している場合には、ステップ 702 において、現在のモードがスレッドモードであるかどうかチェックされ、遷移無効化フラグ (TMID34) は、スレッドモードでのドメイン遷移が無効されていることを示し、そうである場合には、ステップ 704 において、ドメイン遷移が行われるのを防ぐために INV TMI Usage Fault がトリガされる。この障害は、現在のドメイン、すなわち低セキュアドメインで処理される。セキュアアドレス領域からフェッチされた命令は、実際にはドメイン遷移をトリガしていない可能性があるが、それにもかかわらず、後続のセキュリティチェックに合格した場合に遷移が発生する可能性があるとして、障害が控えめに生成されることに留意されたい。フェッチ段 8 で障害を生成することによって、これは、無効にされたドメイン遷移によってトリガされる Usage Fault が、実行される必要があり得る他のタイプのセキュリティチェックと比較して比較的早期に評価されることを保証し、その結果、それらの他のセキュリティチェックは、単にセキュアリソースがまだ構成されていないために失敗しない。

10

20

30

40

50

【0217】

一方、ステップ702において、フェッチ段8において、現在のモードがハンドラモードであると判定された場合、又は、TMID34が、スレッドモードドメイン遷移が有効にされていることを示す場合、ステップ706において、命令フェッチは続行することが可能にされる。セキュアアドレス領域からフェッチされた命令がデコード段10でドメインの変更をトリガする命令であると識別された場合（例えば、セキュアドメインへの有効なエントリポイントをマークするセキュアゲートウェイ命令）、命令が実行段12に到達すると、追加のセキュリティチェックが実行され、これらのチェックに合格した場合、セキュアドメインへの遷移が許可される。

【0218】

従って、図21は、TMID値34のチェックが実行段12で必ずしも行われず、命令フェッチに対して実行され得る例である。

【0219】

図22は、使用され得るシミュレータの実装形態を示す。上記の実施形態は、技術をサポートする特有の処理ハードウェアを動作させる装置及び方法の点において本発明を実装するが、コンピュータプログラムを使用して実装される本明細書に記載の実施形態による命令実行環境を提供することも可能である。このようなコンピュータプログラムは、ハードウェアアーキテクチャのソフトウェアベースの実装形態を提供する限りにおいて、シミュレータと呼ばれることが多い。様々なシミュレータコンピュータプログラムは、エミュレータ、仮想マシン、モデル、及び動的バイナリトランスレータを含むバイナリトランスレータを含む。典型的には、シミュレータの実装形態は、シミュレータプログラム810をサポートするホストオペレーティングシステム820を任意選択で実行して、ホストプロセッサ830で実行してもよい。いくつかの配置では、ハードウェアと提供された命令実行環境との間に複数の層のシミュレーションがあってもよく、及び／又は、同じホストプロセッサで提供された複数の異なる命令実行環境があってもよい。歴史的に、合理的な速度で実行するシミュレータの実装形態を提供するためには、強力なプロセッサが必要とされてきたが、互換性又は再利用の理由から別のプロセッサにネイティブなコードを実行したい場合など、特定の状況では、そのようなアプローチが正当化される場合がある。例えば、シミュレータの実装形態では、ホストプロセッサのハードウェアではサポートされていない追加機能を備えた命令実行環境を提供すること、又は異なるハードウェアアーキテクチャに典型的に関連する命令実行環境を提供することができる。シミュレーションの概要は、「Some Efficient Architecture Simulation Techniques」、Robert Bedichek、1990年冬 US ENIX Conference、53～63頁に記載されている。

【0220】

これまで、特定のハードウェア構成又は機能を参照して実施形態を説明してきたが、シミュレーションされた実施形態では、同等の機能を適切なソフトウェア構成又は機能によって提供することができる。例えば、特定の回路は、シミュレーションされた実施形態において、コンピュータプログラムロジックとして実装されてもよい。同様に、レジスタ又はキャッシュなどのメモリハードウェアは、シミュレーションされた実施形態でソフトウェアのデータ構造として実装されてもよい。前述の実施形態で参照されているハードウェア要素のうちの1つ以上がホストハードウェア（例えば、ホストプロセッサ830）に存在する配置では、いくつかのシミュレートされた実施形態は、適する場合、ホストハードウェアを使用してもよい。

【0221】

シミュレータプログラム810は、コンピュータ可読記憶媒体（これは、非一時的媒体であってもよい）に記憶されてもよく、プログラムインタフェース（命令実行環境）をターゲットコード800（これは、アプリケーション、オペレーティングシステム、及びハイパーバイザを含んでもよい）に提供し、プログラムインタフェースは、シミュレータプログラム810によってモデル化されているハードウェアアーキテクチャのアプリケー

シジョンプログラムインタフェースと同じである。従って、上述したインターモダル呼び出し分岐命令を含むターゲットコード800のプログラム命令は、シミュレータプログラム810を使用する命令実行環境内から実行してもよく、上述した装置2のハードウェア機能を実際には有していないホストコンピュータ830がこれらの機能をエミュレートすることができる。

【0222】

図22に示すように、シミュレータプログラム810は、命令デコーダ10、実行段12、例外制御回路28、及び上述のメモリアクセスチェック回路22と同等の機能を実行する、命令復号プログラムロジック811、処理プログラムロジック812、例外制御プログラムロジック813、及びメモリアクセスチェックプログラムロジック814を含む。命令復号プログラムロジック811は、ターゲットコード800の命令を復号し、符号化された命令タイプに応じて、ターゲットコード800の復号された命令についてシミュレートされた命令セットアーキテクチャで定義されたものと同等の動作を実行するようにホストハードウェア830を制御する処理プログラムロジック812内のいくつかのプログラムコードシーケンスのうちの1つを選択する。シミュレートされた例外が発生した場合、例外制御プログラムロジック813は、上述の例外制御回路28と同様の方法で例外処理を制御する。シミュレータプログラム810はまた、ターゲットコード800の命令によって必要とされるレジスタ参照をホスト装置によって使用されるホストメモリアドレス空間への同等のメモリアクセスにマッピングするレジスタエミュレートプログラムロジック815を含み、その結果、シミュレータプログラムによってシミュレートされているターゲットデータ処理装置のレジスタ14の内容に対応するホストメモリアドレス空間内にデータ構造を維持することができる。同様に、メモリアクセスプログラムロジック817は、シミュレートされたアドレス空間816からのアドレスを使用して定義される、ターゲットコード800によって必要とされるメモリアクセス及び命令フェッチを、ホストデータ処理ハードウェア830によって使用されるホストメモリアドレス空間にマッピングする。例えば、メモリアクセスプログラムロジック817は、ホストメモリアドレス空間に実際に記憶されているスタックデータ構造818へのアクセスを制御することができるが、ターゲットコード800の観点からは、シミュレートされたアドレス空間816に記憶されているように見える。

【0223】

従って、レジスタエミュレートプログラムロジック815によってエミュレートされる制御レジスタは、上述のTMIDフラグ34を含むことができ、処理プログラムロジック812は、TMIDフラグ34に基づいて、スレッドモードにおけるドメイン間遷移が許可されるかどうかをシミュレートすることができる。レジスタエミュレートプログラムロジック815は、インターモダル関数呼び出し及び戻りの実行、例外エントリ若しくは戻り、又はTMIDフラグ34に書き込むための他の命令などのイベントに応答してTMIDフラグ34を設定する構成パラメータ設定プログラムロジック819を含むことができる。また、命令復号プログラムロジック811は、上述したように、インターモダル呼び出し分岐命令に応答して、処理プログラムロジック812によって表されるシミュレートされたプロセッサの現在のモードをハンドラモードからスレッドモードに切り替えることができる。

【0224】

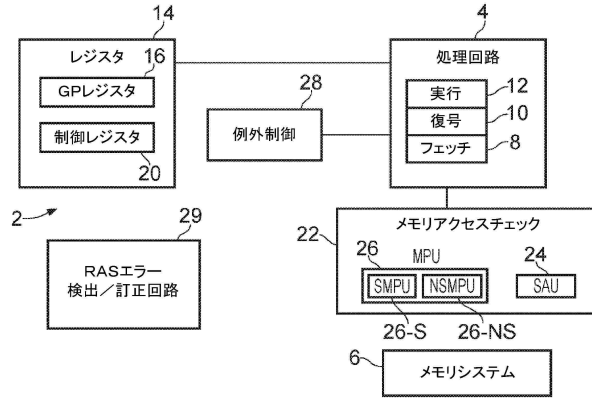
本出願において、「～ように構成された (configured to...)」という用語は、装置の要素が、定義された動作を実施することができる構成を有することを意味するために使用される。この文脈において、「構成」とは、ハードウェア又はソフトウェアの配置又は相互接続の方法を意味する。例えば、装置は、定義された動作を提供する専用ハードウェアを有してもよく、又はプロセッサ若しくは他の処理デバイスが、機能を行うようにプログラムされてもよい。「ように構成された」は、装置要素が、定義された動作を提供するためにある方法で変更される必要があることを意味しない。

【0225】

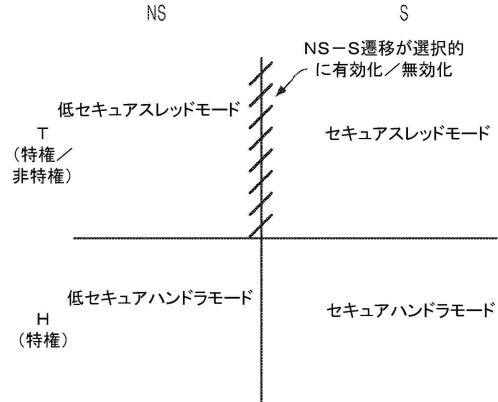
本発明の例示的な実施形態が添付の図面を参照して本明細書で詳細に説明されてきたが、本発明はそれらの正確な実施形態に限定されず、添付の特許請求の範囲によって定義される本発明の範囲及び精神から逸脱することなく、当業者によって様々な変更及び修正を行うことができることを理解されたい。

【図面】

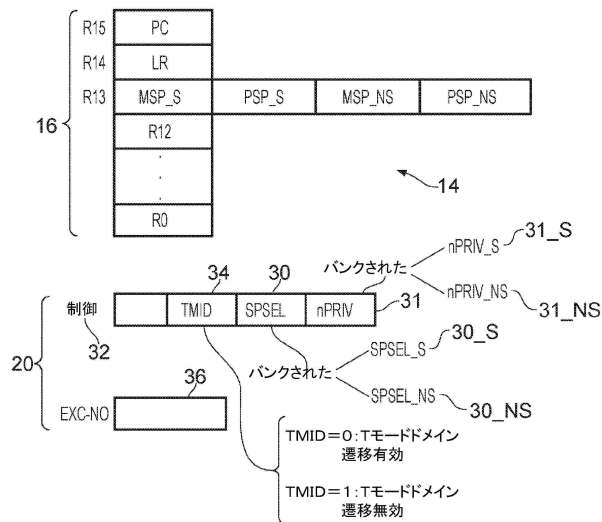
【図 1】



【図 2】

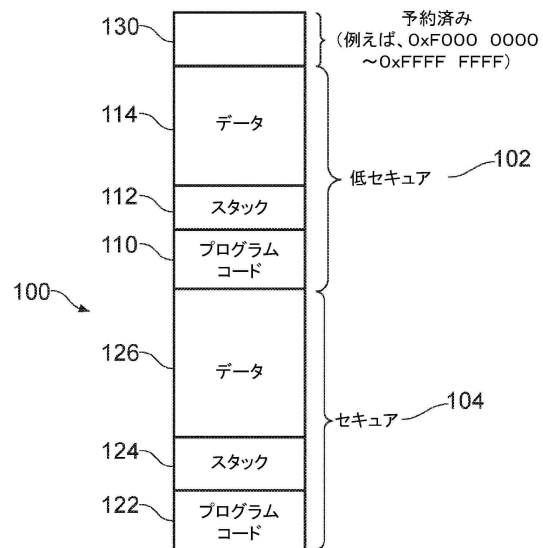


【図 3】



【図 4】

メモリアドレス空間



10

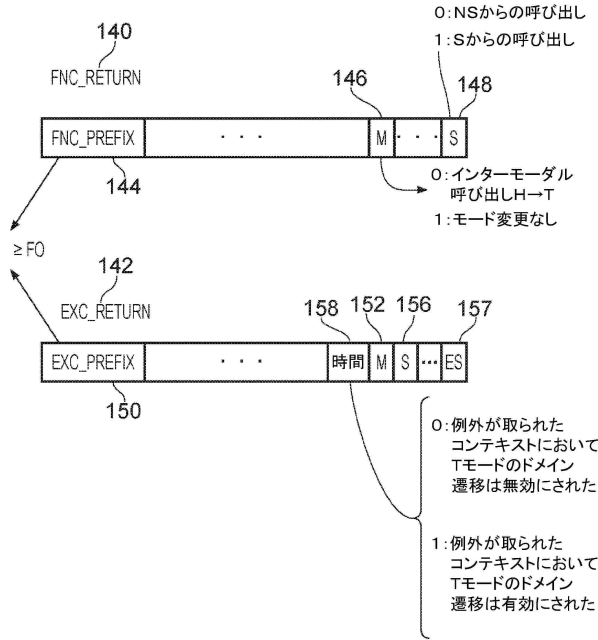
20

30

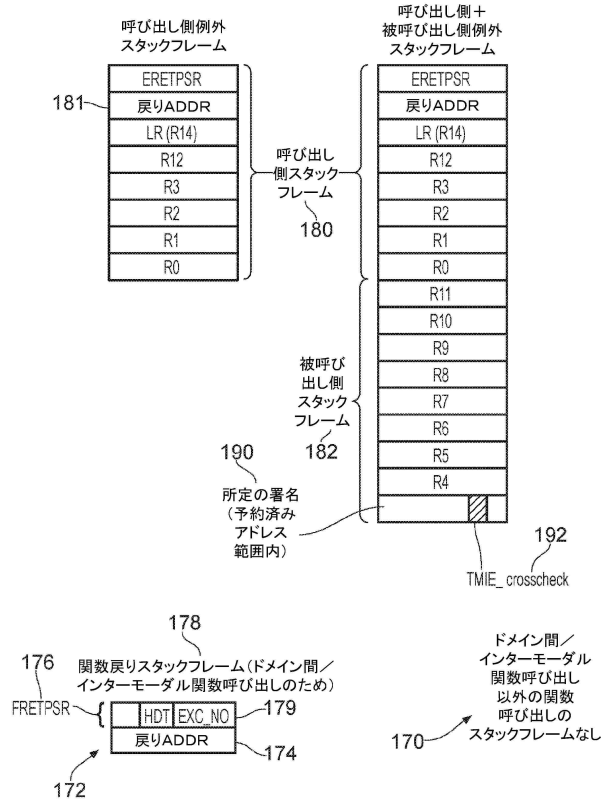
40

50

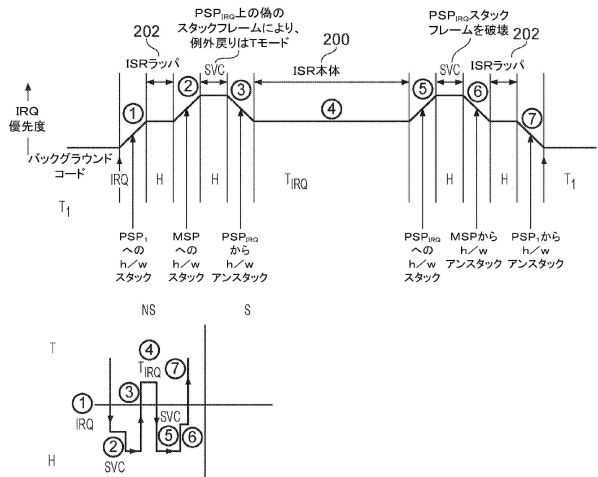
【図 5】



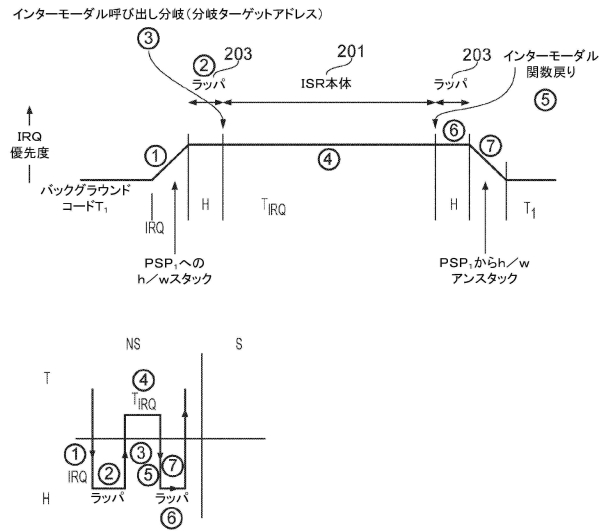
【図 6】



【図 7】



【図 8】



10

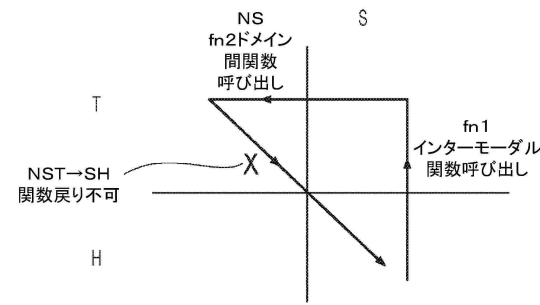
20

30

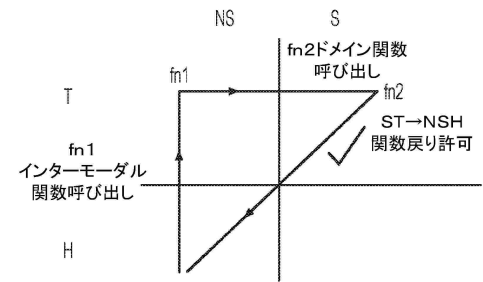
40

50

【図 9】

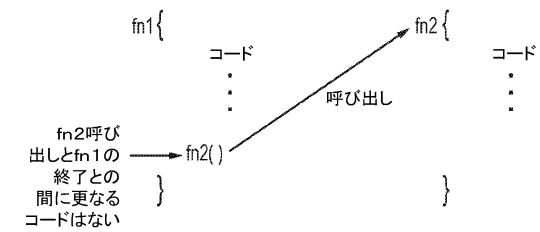


【図 10】

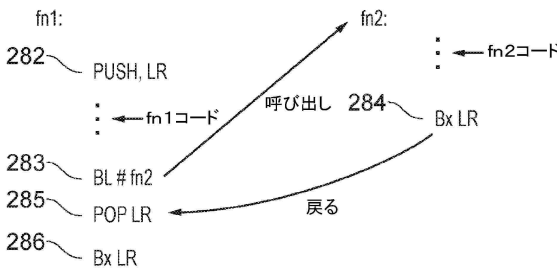


10

【図 11 A】



【図 11 B】



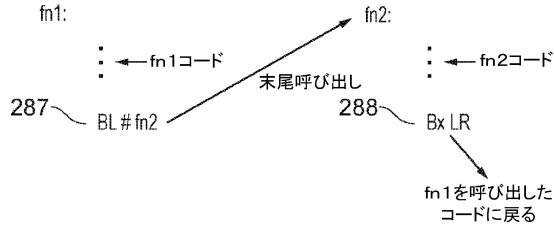
20

30

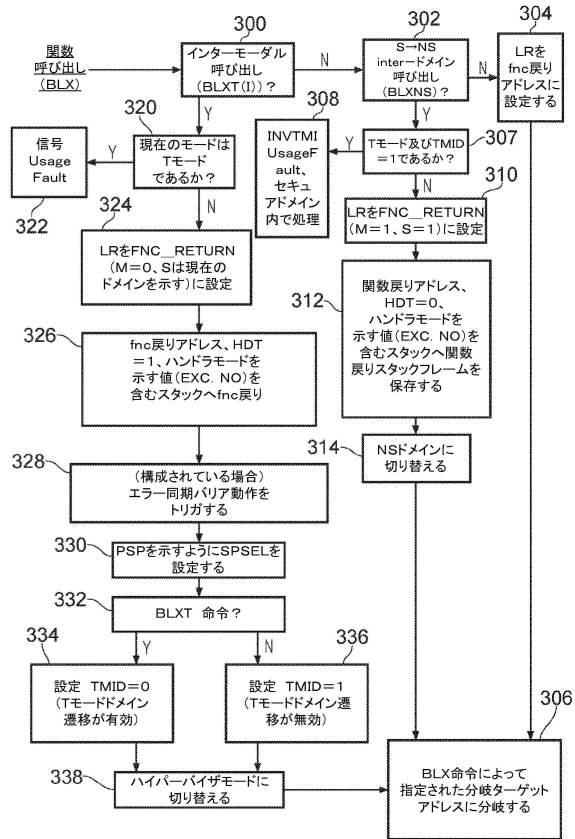
40

50

【図 1 1 C】

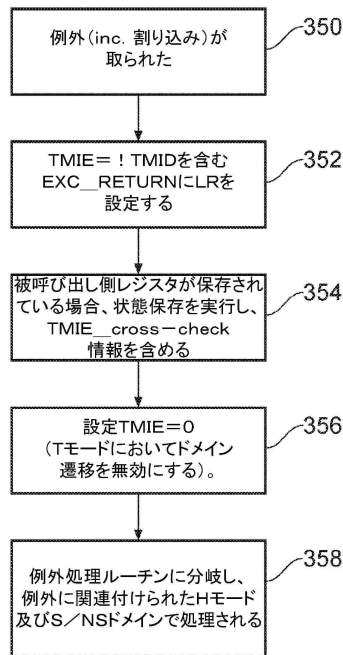


【図 1 2】

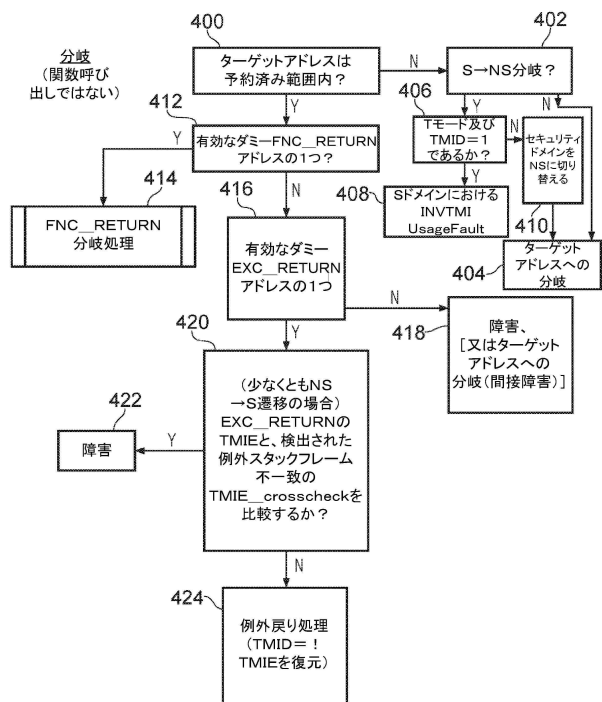


【図 1 3】

例外エントリ



【図 1 4】



10

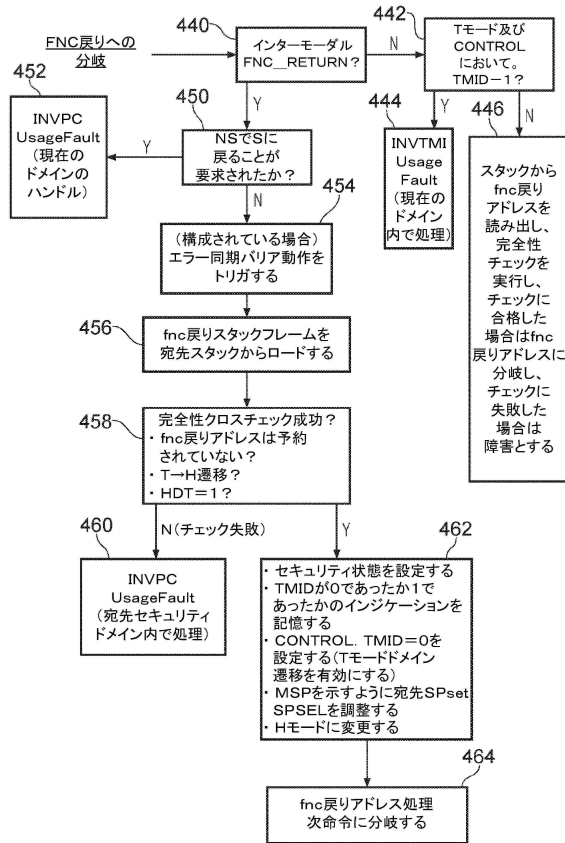
20

30

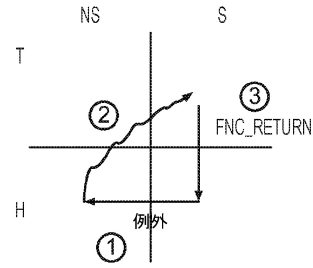
40

50

【図 15】



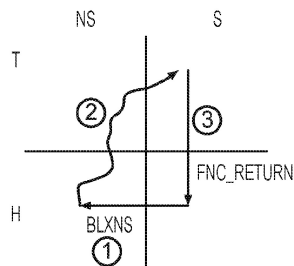
【図 16 A】



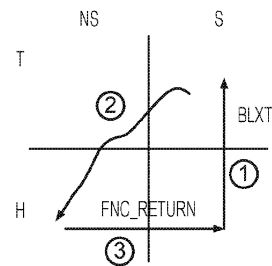
10

20

【図 16 B】



【図 16 C】

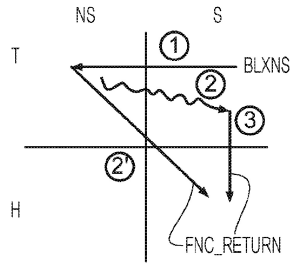


30

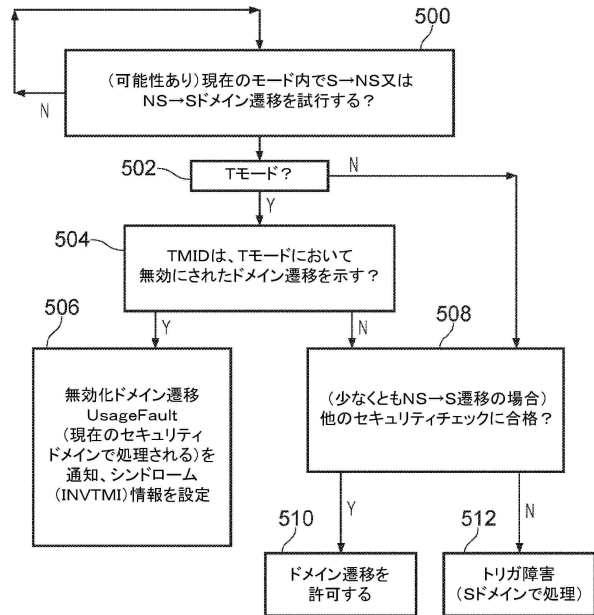
40

50

【図 16 D】

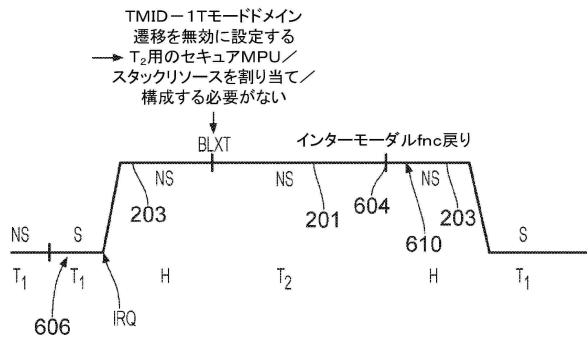


【図 17】

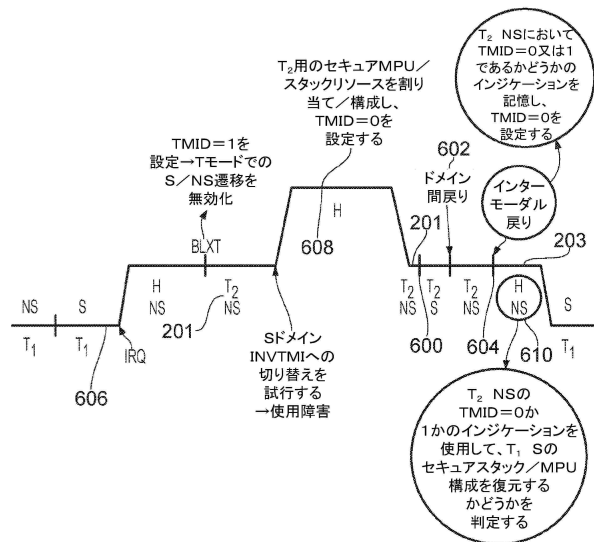


10

【図 18】



【図 19】



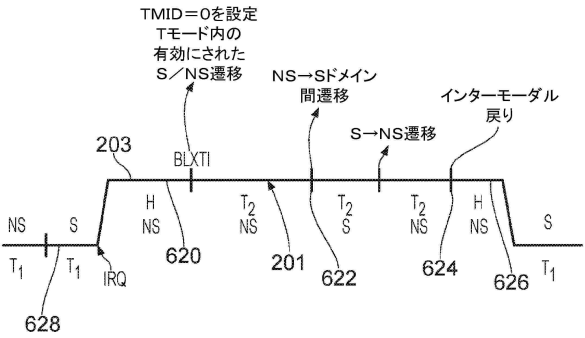
20

30

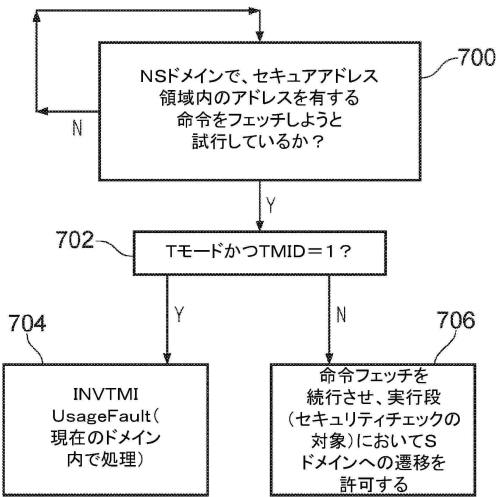
40

50

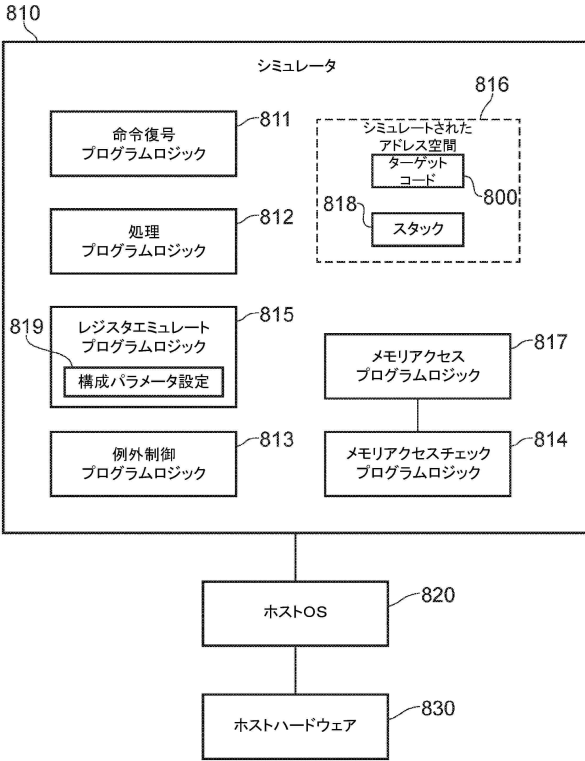
【図 2 0】



【図 2 1】



【図 2 2】



フロントページの続き

- (56)参考文献 特開 2008-257734 (JP, A)
特開 2013-161479 (JP, A)
米国特許出願公開第 2018/0218150 (US, A1)
特開 2017-010347 (JP, A)
- (58)調査した分野 (Int.Cl., DB名)
G06F 21/74
G06F 9/30
G06F 9/48