(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2014-59750
(P2014-59750A)
(43)公開日　平成26年4月3日(2014.4.3)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 11/20	(2006.01)	G06F	11/20	310C	5B034
G06F 9/46	(2006.01)	G06F	9/46	350	
G06F 11/18	(2006.01)	G06F	11/18	310F	

審査請求 有　請求項の数 15　ＯＬ　（全 33 頁）

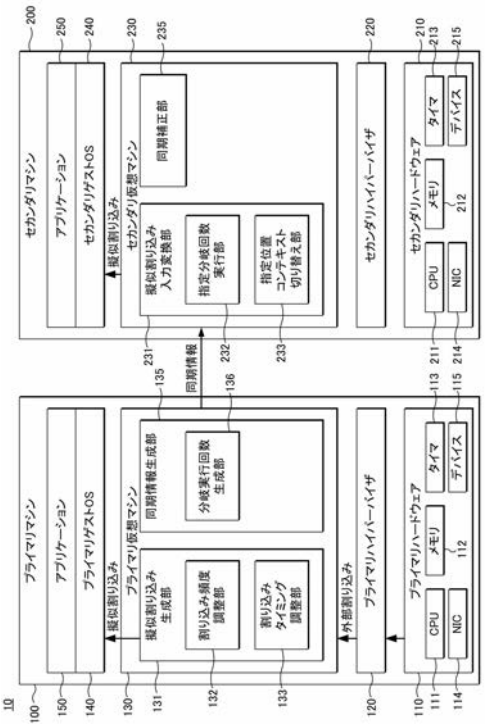
(21)出願番号　特願2012-204608 (P2012-204608)
(22)出願日　平成24年9月18日 (2012.9.18)

(71)出願人　000006507
横河電機株式会社
東京都武蔵野市中町２丁目９番３２号
(72)発明者　吉田　善貴
東京都武蔵野市中町２丁目９番３２号　横河電機株式会社内
(72)発明者　植原　正太
東京都武蔵野市中町２丁目９番３２号　横河電機株式会社内
(72)発明者　大野　毅
東京都武蔵野市中町２丁目９番３２号　横河電機株式会社内

Ｆターム(参考)　5B034 BB17　DD06

(54)【発明の名称】フォールトトレラントシステム

(57)【要約】
【課題】２台のコンピュータ上に形成された２台の仮想マシンを並列同期動作させるフォールトトレラントシステムにおいて、それぞれのコンピュータのハードウェアの差異を適切に調整する。
【解決手段】プライマリ仮想マシンと、セカンダリ仮想マシンとを備えたフォールトトレラントシステムであって、プライマリ仮想マシンは、セカンダリ仮想マシンから、動作性能情報を収集し、プライマリ仮想マシンの動作能力と、セカンダリ仮想マシンの動作能力が等しくなるように、プライマリ仮想マシンの動作能力とセカンダリ仮想マシンの動作能力とを設定する。
【選択図】図１



【特許請求の範囲】
【請求項１】
プラマリハイパーバイザが稼働するプライマリマシン上に形成され、前記プラマリハイパーバイザからの外部割り込みに基づく擬似割り込みをプラマリゲストＯＳに入力するプライマリ仮想マシンと、
セカンダリハイパーバイザが稼働するセカンダリマシン上に形成され、前記プライマリ仮想マシンから伝達された前記擬似割り込みのタイミング情報に基づいて、擬似割り込みをセカンダリゲストＯＳに入力するセカンダリ仮想マシンと、を備えたフォールトトレラントシステムであって、
前記プライマリ仮想マシンは、
前記セカンダリ仮想マシンから、動作性能情報を収集し、前記プライマリ仮想マシンの動作能力と、前記セカンダリ仮想マシンの動作能力が等しくなるように、前記プライマリ仮想マシンの動作能力と前記セカンダリ仮想マシンの動作能力とを設定することを特徴とするフォールトトレラントシステム。
【請求項２】
前記プライマリ仮想マシンは、
設定した前記セカンダリ仮想マシンの動作能力を前記セカンダリ仮想マシンに通知し、
前記セカンダリ仮想マシンは、
通知された動作能力に従って動作を行なうことを特徴とする請求項１に記載のフォールトトレラントシステム。
【請求項３】
前記動作能力は、動作周波数を含み、
前記プライマリ仮想マシンは、
前記プライマリ仮想マシンが使用可能な動作周波数と、前記セカンダリ仮想マシンが使用可能な動作周波数とを収集して、前記プライマリ仮想マシンの動作周波数と、前記セカンダリ仮想マシンの動作周波数とを設定することを特徴とする請求項１または２に記載のフォールトトレラントシステム。
【請求項４】
前記プライマリ仮想マシンは、
前記プライマリ仮想マシンが使用するデバイスの動作仕様と、前記セカンダリ仮想マシンが使用するデバイスの動作仕様とが異なる場合に、
前記プライマリ仮想マシンが使用するデバイスに対する処理が、前記セカンダリ仮想マシンが使用するデバイスに対する処理と同じになるようにエミュレートすることを特徴とする請求項１～３のいずれか１項に記載のフォールトトレラントシステム。
【請求項５】
前記プライマリ仮想マシンは、
前記デバイスの動作仕様の違いが所定の機能の有無の場合は、
前記所定の機能を無効にして、前記プライマリ仮想マシンが使用するデバイスに対する処理が、前記セカンダリ仮想マシンが使用するデバイスに対する処理と同じになるようにエミュレートし、
前記デバイスの動作仕様の違いが能力あるいは容量の場合は、動作に相違が生じる処理を前記プライマリゲストＯＳの外部で実行させることを特徴とする請求項４に記載のフォールトトレラントシステム。
【請求項６】
前記セカンダリ仮想マシンは、
前記プライマリ仮想マシンが使用するデバイスの動作仕様と、前記セカンダリ仮想マシンが使用するデバイスの動作仕様とが異なる場合に、
前記セカンダリ仮想マシンが使用するデバイスに対する処理が、前記プライマリ仮想マシンが使用するデバイスに対する処理と同じになるようにエミュレートすることを特徴とする請求項１～３のいずれか１項に記載のフォールトトレラントシステム。

【請求項７】
前記セカンダリ仮想マシンは、
前記デバイスの動作仕様の違いが所定の機能の有無の場合は、
前記所定の機能を無効にして、前記セカンダリ仮想マシンが使用するデバイスに対する処理が、前記プライマリ仮想マシンが使用するデバイスに対する処理と同じになるようにエミュレートし、
前記デバイスの動作仕様の違いが能力あるいは容量の場合は、動作に相違が生じる処理を前記セカンダリゲストＯＳの外部で実行させることを特徴とする請求項６に記載のフォールトトレラントシステム。
【請求項８】
前記セカンダリ仮想マシンは、
前記動作能力の設定後であっても、前記プライマリ仮想マシンより実行速度が遅い場合には、
遅れの度合いを取得し、取得した値が予め定められた遅延許容時間を超えると、同期初期化処理を行なうことを特徴とする請求項１～７のいずれか１項に記載のフォールトトレラントシステム。
【請求項９】
前記セカンダリ仮想マシンは、
前記取得した遅れの度合いが、想定される量より大きい場合に、故障として検出することを特徴とする請求項８に記載のフォールトトレラントシステム。
【請求項１０】
前記セカンダリ仮想マシンは、
前記動作能力の設定後であっても、前記プライマリ仮想マシンより実行速度が遅い場合には、前記プライマリゲストＯＳのアイドル時に、擬似割り込み入力タイミングを前倒しすることで、前記プライマリ仮想マシンに対する遅れを取り戻すことを特徴とする請求項１～７のいずれか１項に記載のフォールトトレラントシステム。
【請求項１１】
前記プライマリ仮想マシンは、
前記動作能力の設定後であっても、前記セカンダリ仮想マシンより実行速度が速い場合には、前記セカンダリ仮想マシンの実行速度に合わせて、ウェイトを入れながら動作を行なうことを特徴とする請求項１～７のいずれか１項に記載のフォールトトレラントシステム。
【請求項１２】
前記セカンダリ仮想マシンは、
前記動作能力の設定後であっても、前記プライマリ仮想マシンより実行速度が速い場合には、前記プライマリ仮想マシンの実行速度に合わせて、ウェイトを入れながら動作を行なうことを特徴とする請求項１～７のいずれか１項に記載のフォールトトレラントシステム。
【請求項１３】
前記プライマリ仮想マシンおよび前記セカンダリ仮想マシンは、
それぞれのストレージデバイスの内容を、それぞれの仮想マシンに割り当てられたメモリ上に展開し、ストレージデバイスアクセスをメモリアクセスに置き換えることを特徴とする請求項１～１２のいずれか１項に記載のフォールトトレラントシステム。
【請求項１４】
前記プライマリ仮想マシンおよび前記セカンダリ仮想マシンは、
前記メモリアクセスを行なうと、アクセス完了の外部割り込みを待つことなく、それぞれのゲストＯＳに擬似割り込みを入力することを特徴とする請求項１３に記載のフォールトトレラントシステム。
【請求項１５】
前記プライマリ仮想マシンおよび前記セカンダリ仮想マシンは、

それぞれのストレージデバイスの内容を、それぞれのゲストＯＳで設定されたＲＡＭディスクに展開し、ストレージデバイスアクセスをＲＡＭディスクに置き換えることを特徴とする請求項１～１２のいずれか１項に記載のフォールトトレラントシステム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、仮想マシンを利用したフォールトトレラントシステムに関する。

【背景技術】

【０００２】

フォールトトレラントシステムは、その構成の一部に問題が生じてもシステム全体が機能停止することなく動作を継続できるシステムであり、特に、連続可動性が要求されるシステム等に適用される。例えば、フォールトトレラントを適用したサーバコンピュータは、ハードウェアに障害が発生しても、外部機器のクライアントアプリケーションからのネットワークアクセスに対して、通信エラーとならずに、正しいデータを出力することができる。

【０００３】

特許文献１に記載されているように、通信可能な２台のコンピュータ上でそれぞれ稼働する仮想マシンでフォールトトレラントシステムを実現する手法が知られている。仮想マシンを利用したフォールトトレラントシステムは、２台の仮想マシンに実行状態を同期させて同じ動作をさせており、一方のコンピュータに障害が発生した場合でも、他方のコンピュータで稼働する仮想マシンが処理を引き継ぐことで、システムのサービスを途切れることなく連続して提供することができる。

【０００４】

仮想マシンを利用したフォールトトレラントシステムは、一方の仮想マシンをプライマリ仮想マシンとし、他方の仮想マシンをセカンダリ仮想マシンとする。そして、プライマリ仮想マシンを先行して動作させるとともに、外部との入出力のやり取りはプライマリ仮想マシンのみが行なうようにする。

【０００５】

一般に、同じプログラムを実行している２台のコンピュータは、外部からの入力が正確に同じタイミングで入力されると、同じ動作を行なって同じ出力を行なう。そこで、仮想マシンを利用したフォールトトレラントシステムは、プライマリ仮想マシンで、外部入力に基づく割り込みが発生すると、割り込みを行なったタイミングを同期情報としてセカンダリ仮想マシンに伝達する。そして、遅れて動作を行なっているセカンダリ仮想マシンが、同期情報で通知されたタイミングと同じタイミングで擬似的に割り込みを行なうことで、プライマリ仮想マシンとセカンダリ仮想マシンは、同期して同じ動作を行なうことになる。

【０００６】

図６は、仮想マシンを利用した従来のフォールトトレラントシステムの構成を示すブロック図である。本図に示すように、フォールトトレラントシステム６０は、ネットワーク接続されたプライマリマシン６００とセカンダリマシン７００とを備えている。

【０００７】

プライマリマシン６００は、物理的なコンピュータ環境であるプライマリハードウェア６１０上でプライマリハイパーバイザ６２０が動作しており、プライマリ仮想マシン６３０が形成されている。プライマリ仮想マシン６３０では、プライマリゲストＯＳ６４０が動作しており、プライマリゲストＯＳ６４０上で、アプリケーション６５０が実行されている。

【０００８】

プライマリハードウェア６１０は、ＣＰＵ、メモリ、ネットワークインタフェースカード（ＮＩＣ）、ストレージ等の各種デバイス等を備えている。

【０００９】

プライマリ仮想マシン６３０は、プライマリハードウェア６１０の一部が割り当てられ、外部との入出力を行なって動作する仮想的なコンピュータ環境である。プライマリ仮想マシン６３０は、プライマリハイパーバイザ６２０により管理される。

【００１０】

セカンダリマシン７００も同様に、物理的なコンピュータ環境であるセカンダリハードウェア７１０上でセカンダリハイパーバイザ７２０が動作しており、セカンダリ仮想マシン７３０が形成されている。セカンダリ仮想マシン７３０では、セカンダリゲストＯＳ７４０が動作しており、セカンダリゲストＯＳ７４０上で、アプリケーション７５０が実行されている。

【００１１】

セカンダリハードウェア７１０は、ＣＰＵ、メモリ、ネットワークインタフェースカード（ＮＩＣ）、ストレージ等の各種デバイス等を備えている。

【００１２】

セカンダリ仮想マシン７３０は、セカンダリハードウェア７１０の一部が割り当てられ、プライマリ仮想マシン６３０に同期して動作する仮想的なコンピュータ環境である。セカンダリ仮想マシン７３０は、セカンダリハイパーバイザ７２０により管理される。

【００１３】

従来のフォールトトレラントシステム６０において、プライマリ仮想マシン６３０とセカンダリ仮想マシン７３０との実行状態の同期は以下のような手順で行なわれている。

【００１４】

プライマリハイパーバイザ６２０が、プライマリハードウェア６１０から外部割り込みを受けると、プライマリ仮想マシン６３０に外部割り込みの入力を行なう。

【００１５】

次いで、プライマリ仮想マシン６３０が、プライマリゲストＯＳ６４０に擬似割り込みの入力を行なう。ここで、プライマリ仮想マシン６３０からプライマリゲストＯＳ６４０への擬似割り込みの入力について説明する。

【００１６】

プライマリゲストＯＳ６４０の処理中に、外部割り込み、特権命令、例外等の仮想マシンコンテキスト切り替えイベントが発生すると、プライマリゲストＯＳ６４０の処理を中断して、ゲストＯＳコンテキストから仮想マシンコンテキストに切り替わり、プライマリ仮想マシン６３０に処理が遷移する。

【００１７】

プライマリ仮想マシン６３０は、各種イベントに応じてそのタイミングでプライマリゲストＯＳ６４０に擬似割り込みを行なう必要があれば、擬似割り込みの設定を行なう。擬似割り込みの設定を行なった場合は、プライマリ仮想マシン６３０が処理を終えて、イベント発生時に割り込まれたプライマリゲストＯＳ６４０に処理が戻るときにプライマリゲストＯＳ６４０に擬似割り込みが入力される。

【００１８】

プライマリ仮想マシン６３０は、プライマリゲストＯＳ６４０に擬似割り込みの入力を行なうと、セカンダリ仮想マシン７３０に同期情報の伝達を行なう。同期情報には、擬似割り込みの識別情報と擬似割り込みを入力する同期タイミング情報とが含まれる。

【００１９】

同期タイミング情報は、プライマリゲストＯＳ６４０に入力された擬似割り込みと同じタイミングでセカンダリゲストＯＳ７４０に擬似割り込みを入力するための情報であり、ＣＰＵ固有の実行中断位置と実行命令数とが含まれる。

【００２０】

実行中断位置は、例えば、擬似割り込みが入力されたときに実行していた命令のアドレスを示すプログラムカウンタの値を用いることができる。また、実行命令数は、ＣＰＵが備えるパフォーマンスカウンタのＣＰＵ実行命令数カウンタで計測することができる。

【００２１】

この場合、プライマリ仮想マシン６３０がプライマリゲストＯＳ６４０に擬似割り込みの入力を行なうと、プライマリゲストＯＳ６４０の実行を再開する前に、ＣＰＵ実行命令数カウンタをゼロクリアし、ＣＰＵ実行命令数カウンタを有効化する。これにより、前回の擬似割り込み入力からプライマリゲストＯＳ６４０が実行した命令数をカウントすることができる。

【００２２】

同期タイミング情報として実行中断位置だけを用いると、実行中断位置の示す命令がループ処理に含まれる場合や条件分岐先にある場合に、ループや条件分岐の度にその命令を通過するため、擬似割り込み入力のタイミングを特定することができない。

【００２３】

また、同期タイミング情報として実行命令数だけを用いると、パイプライン機能などの高速化技術により、セカンダリゲストＯＳ７４０が指定された実行命令数で停止できず、その実行命令数を超えて停止してしまい、プライマリゲストＯＳ６４０と同じタイミングで擬似割り込みを入力することができない場合がある。

【００２４】

そこで、同期タイミング情報として実行中断位置と実行命令数とを組み合わせ、セカンダリゲストＯＳ７４０において、実行中断位置が示す命令を通る度に、実行命令数を確認することで、セカンダリゲストＯＳ７４０を、プライマリゲストＯＳ６４０で擬似割り込みが入力されたのと同じタイミングで停止させるようにしている。

【００２５】

従って、プライマリ仮想マシン６３０は、プライマリゲストＯＳ６４０の実行を中断した後、プライマリゲストＯＳ６４０に擬似割り込みの入力を行なうと、プライマリゲストＯＳ６４０が実行を中断した時点のプログラムカウンタの値と、ＣＰＵ実行命令数カウンタの値とを取得し、同期タイミング情報を生成する。そして、擬似割り込みの識別情報と同期タイミング情報とを同期情報としてセカンダリ仮想マシン７３０に伝達する。

【００２６】

同期情報を受信したセカンダリ仮想マシン７３０は、同期タイミング情報に従ってセカンダリゲストＯＳ７４０の実行を中断させる。このときのセカンダリ仮想マシン７３０の動作について、図７のフローチャートを参照して説明する。

【００２７】

同期タイミング情報の実行中断位置で指定されたプログラム位置にブレーク命令を埋め込んで（Ｓ３０１）、セカンダリゲストＯＳ７４０を再開させる（Ｓ３０２）。そして、セカンダリゲストＯＳ７４０が停止すると（Ｓ３０３：Ｙｅｓ）、ＣＰＵ実行命令数カウンタを確認して、指定された実行命令数と一致していたら（Ｓ３０４：Ｙｅｓ）、その停止位置でセカンダリゲストＯＳ７４０を中断させる（Ｓ３０５）。指定された実行命令数と一致していなければ（Ｓ３０４：Ｎｏ）、再度セカンダリゲストＯＳ７４０を再開させ（Ｓ３０２）、実行命令数の確認処理を繰り返す。

【００２８】

セカンダリ仮想マシン７３０は、セカンダリゲストＯＳ７４０を中断させると、同期情報の擬似割り込み識別情報に従って擬似割り込みの設定を行なって、（Ｓ３０６）、セカンダリゲストＯＳ７４０を再開させる（Ｓ３０７）。これにより、セカンダリゲストＯＳ７４０に、プライマリゲストＯＳ６４０と同じタイミングで擬似割り込みが入力され、プライマリ仮想マシン６３０とセカンダリ仮想マシン７３０との実行状態が同期する。

【００２９】

なお、プライマリマシン６００、セカンダリマシン７００のいずれかでハードウェアの故障が生じると、実行状態の同期が乱れる。実行状態の同期が乱れると、プライマリ仮想マシン６３０とセカンダリ仮想マシン７３０とで、出力データの値が異なってくる。そこで、従来のフォールトトレラントシステム６０は、プライマリ仮想マシン６３０の出力とセカンダリ仮想マシン７３０の出力とを照合し、出力データの値が異なる場合は、どちらかのハードウェアで故障が発生したと判断する。

【００３０】
この故障判断処理を行なうため、セカンダリ仮想マシン７３０に出力データ照合部７３１を設け、プライマリ仮想マシン６３０の出力データを収集して、セカンダリ仮想マシン７３０の出力データと照合するようにしている。
【先行技術文献】
【特許文献】
【００３１】
【特許文献１】特開２００９－８０６９５号公報
【発明の概要】
【発明が解決しようとする課題】
【００３２】
フォールトトレラントシステム６０において、プライマリマシン６００のプライマリハードウェア６１０とセカンダリマシン７００のセカンダリハードウェア７１０は、ＣＰＵ、メモリ、デバイス等、種々のハードウェアで構成されているが、双方のハードウェアの性能は必ずしも一致するとは限られない。
【００３３】
例えば、ＣＰＵのスペックが異なっていたり、メモリの容量が異なっている場合も想定される。また、フォールトトレラントシステム６０の運用過程においてデバイスが交換される等により、双方のデバイスが異なっている場合や、同じデバイスであっても、経年劣化等により性能差が生じる場合もある。
【００３４】
このため、プライマリ仮想マシン６３０とセカンダリ仮想マシン７３０とで処理速度に差が生じる場合がある。上述のように、フォールトトレラントシステム６０は、プライマリ仮想マシン６３０が先行して動作を行なうようにしているが、例えば、セカンダリ仮想マシン７３０の方が、処理速度が速いと、動作の逆転現象が生じ、プライマリ仮想マシン６３０からの同期情報に基づく実行状態の同期が行なえなくなる。プライマリ仮想マシン６３０の方が、処理速度が速いと、セカンダリ仮想マシン７３０の遅れが蓄積されていく。
【００３５】
また、デバイスが異なっていると、プライマリゲストＯＳ６４０とセカンダリゲストＯＳ７４０のドライバ処理が異なって、両者の動作に相違が生じる結果、同期ずれが生じるおそれがある。
【００３６】
このため、ハードウェアの差異を埋める仕組みが必要となる。そこで、本発明は、２台のコンピュータ上に形成された２台の仮想マシンを並列同期動作させるフォールトトレラントシステムにおいて、それぞれのコンピュータのハードウェアの差異を適切に調整することを目的とする。
【課題を解決するための手段】
【００３７】
上記課題を解決するため、本発明のフォールトトレラントシステムは、プラマリハイパーバイザが稼働するプライマリマシン上に形成され、前記プラマリハイパーバイザからの外部割り込みに基づく擬似割り込みをプラマリゲストＯＳに入力するプライマリ仮想マシンと、セカンダリハイパーバイザが稼働するセカンダリマシン上に形成され、前記プライマリ仮想マシンから伝達された前記擬似割り込みのタイミング情報に基づいて、擬似割り込みをセカンダリゲストＯＳに入力するセカンダリ仮想マシンと、を備えたフォールトトレラントシステムであって、前記プライマリ仮想マシンは、前記セカンダリ仮想マシンから、動作性能情報を収集し、前記プライマリ仮想マシンの動作能力と、前記セカンダリ仮想マシンの動作能力が等しくなるように、前記プライマリ仮想マシンの動作能力と前記セカンダリ仮想マシンの動作能力とを設定することを特徴とする。
ここで、前記プライマリ仮想マシンは、設定した前記セカンダリ仮想マシンの動作能力

を前記セカンダリ仮想マシンに通知し、前記セカンダリ仮想マシンは、通知された動作能力に従って動作を行なうことができる。

また、前記動作能力は、動作周波数を含み、前記プライマリ仮想マシンは、前記プライマリ仮想マシンが使用可能な動作周波数と、前記セカンダリ仮想マシンが使用可能な動作周波数とを収集して、前記プライマリ仮想マシンの動作周波数と、前記セカンダリ仮想マシンの動作周波数とを設定することができる。

また、前記プライマリ仮想マシンは、前記プライマリ仮想マシンが使用するデバイスの動作仕様と、前記セカンダリ仮想マシンが使用するデバイスの動作仕様とが異なる場合に、前記プライマリ仮想マシンが使用するデバイスに対する処理が、前記セカンダリ仮想マシンが使用するデバイスに対する処理と同じになるようにエミュレートすることができる。

このとき、前記プライマリ仮想マシンは、前記デバイスの動作仕様の違いが所定の機能の有無の場合は、前記所定の機能を無効にして、前記プライマリ仮想マシンが使用するデバイスに対する処理が、前記セカンダリ仮想マシンが使用するデバイスに対する処理と同じになるようにエミュレートし、前記デバイスの動作仕様の違いが能力あるいは容量の場合は、動作に相違が生じる処理を前記プライマリゲストＯＳの外部で実行させることができる。

また、前記セカンダリ仮想マシンは、前記プライマリ仮想マシンが使用するデバイスの動作仕様と、前記セカンダリ仮想マシンが使用するデバイスの動作仕様とが異なる場合に、前記セカンダリ仮想マシンが使用するデバイスに対する処理が、前記プライマリ仮想マシンが使用するデバイスに対する処理と同じになるようにエミュレートすることができる。

あるいは、前記セカンダリ仮想マシンは、前記デバイスの動作仕様の違いが所定の機能の有無の場合は、前記所定の機能を無効にして、前記セカンダリ仮想マシンが使用するデバイスに対する処理が、前記プライマリ仮想マシンが使用するデバイスに対する処理と同じになるようにエミュレートし、前記デバイスの動作仕様の違いが能力あるいは容量の場合は、動作に相違が生じる処理を前記セカンダリゲストＯＳの外部で実行させることができる。

また、前記セカンダリ仮想マシンは、前記動作能力の設定後であっても、前記プライマリ仮想マシンより実行速度が遅い場合には、遅れの度合いを取得し、取得した値が予め定められた遅延許容時間を超えると、同期初期化処理を行なうことができる。

このとき、前記セカンダリ仮想マシンは、前記取得した遅れの度合いが、想定される量より大きい場合に、故障として検出することができる。

また、前記セカンダリ仮想マシンは、前記動作能力の設定後であっても、前記プライマリ仮想マシンより実行速度が遅い場合には、前記プライマリゲストＯＳのアイドル時に、擬似割り込み入力タイミングを前倒しすることで、前記プライマリ仮想マシンに対する遅れを取り戻すことができる。

また、前記プライマリ仮想マシンは、前記動作能力の設定後であっても、前記セカンダリ仮想マシンより実行速度が速い場合には、前記セカンダリ仮想マシンの実行速度に合わせて、ウェイトを入れながら動作を行なうことができる。

また、前記セカンダリ仮想マシンは、前記動作能力の設定後であっても、前記プライマリ仮想マシンより実行速度が速い場合には、前記プライマリ仮想マシンの実行速度に合わせて、ウェイトを入れながら動作を行なうことができる。

また、前記プライマリ仮想マシンおよび前記セカンダリ仮想マシンは、それぞれのストレージデバイスの内容を、それぞれの仮想マシンに割り当てられたメモリ上に展開し、ストレージデバイスアクセスをメモリアクセスに置き換えることができる。

このとき、前記プライマリ仮想マシンおよび前記セカンダリ仮想マシンは、前記メモリアクセスを行なうと、アクセス完了の外部割り込みを待つことなく、それぞれのゲストＯＳに擬似割り込みを入力することができる。

また、前記プライマリ仮想マシンおよび前記セカンダリ仮想マシンは、それぞれのスト

レージデバイスの内容を、それぞれのゲストＯＳで設定されたＲＡＭディスクに展開し、ストレージデバイスアクセスをＲＡＭディスクに置き換えることができる。

【発明の効果】

【００３８】

本発明によれば、２台のコンピュータ上に形成された２台の仮想マシンを並列同期動作させるフォールトトレラントシステムにおいて、それぞれのコンピュータのハードウェアの差異を適切に調整することができる。

【図面の簡単な説明】

【００３９】

【図１】本実施形態に係るフォールトトレラントシステムの構成を示すブロック図である。

【図２】プライマリ仮想マシンとセカンダリ仮想マシンとの並列同期動作の概要について説明するフローチャートである。

【図３】分岐回数と実行中断位置とによる擬似割り込み入力位置特定を説明する図である。

【図４】ＣＰＵアーキテクチャの詳細が判別している場合の擬似割り込みタイミング調整動作の具体的な手順について説明するフローチャートである。

【図５】ＣＰＵアーキテクチャの詳細が不明の場合の擬似割り込みタイミング調整動作の具体的な手順について説明するフローチャートである。

【図６】仮想マシンを利用した従来のフォールトトレラントシステムの構成を示すブロック図である。

【図７】同期情報を受信したセカンダリ仮想マシンが、同期タイミング情報に従ってセカンダリゲストＯＳの実行を中断させる動作を説明するフローチャートである。

【発明を実施するための形態】

【００４０】

本発明の実施の形態について図面を参照して説明する。図１は、本実施形態に係るフォールトトレラントシステムの構成を示すブロック図である。本図に示すように、フォールトトレラントシステム１０は、ネットワーク接続されたプライマリマシン１００とセカンダリマシン２００とを備えている。プライマリマシン１００、セカンダリマシン２００は、サーバコンピュータ、パーソナルコンピュータ等の汎用的なコンピュータを用いて構成することができる。

【００４１】

プライマリマシン１００は、物理的なコンピュータ環境であるプライマリハードウェア１１０上でプライマリハイパーバイザ１２０が動作しており、プライマリハイパーバイザ１２０が動作することにより、プライマリ仮想マシン１３０が形成されている。プライマリ仮想マシン１３０では、プライマリゲストＯＳ１４０が動作しており、プライマリゲストＯＳ１４０上で、アプリケーション１５０が実行されている。

【００４２】

プライマリハードウェア１１０は、ＣＰＵ１１１、メモリ１１２、タイマ１１３、ネットワークインタフェースカード（ＮＩＣ）１１４、ストレージ等の各種デバイス１１５を備えている。

【００４３】

プライマリ仮想マシン１３０は、プライマリハードウェア１１０の一部が割り当てられ、外部との入出力を行なって動作する仮想的なコンピュータ環境である。プライマリ仮想マシン１３０は、プライマリハイパーバイザ１２０により管理され、プライマリハイパーバイザ１２０は、複数個のプライマリ仮想マシン１３０を管理することができる。

【００４４】

セカンダリマシン２００も同様に、物理的なコンピュータ環境であるセカンダリハードウェア２１０上でセカンダリハイパーバイザ２２０が動作しており、セカンダリハイパーバイザ２２０が動作することにより、セカンダリ仮想マシン２３０が形成されている。セ

カンダリ仮想マシン２３０では、セカンダリゲストＯＳ２４０が動作しており、セカンダリゲストＯＳ２４０上で、アプリケーション２５０が実行されている。

【００４５】

セカンダリハードウェア２１０は、ＣＰＵ２１１、メモリ２１２、タイマ２１３、ネットワークインタフェースカード（ＮＩＣ）２１４、ストレージ等の各種デバイス２１５を備えている。

【００４６】

セカンダリ仮想マシン２３０は、セカンダリハードウェア２１０の一部が割り当てられ、プライマリ仮想マシン１３０に同期して動作する仮想的なコンピュータ環境である。セカンダリ仮想マシン２３０は、セカンダリハイパーバイザ２２０により管理され、セカンダリハイパーバイザ２２０は、複数個のセカンダリ仮想マシン２３０を管理することができる。

【００４７】

フォールトトレラントシステム１０では、従来と同様に、外部との入出力のやり取りはプライマリ仮想マシン１３０のみが行ない、プライマリ仮想マシン１３０が先行して動作を行なうものとする。

【００４８】

本実施形態において、プライマリ仮想マシン１３０は、擬似割り込み生成部１３１と同期情報生成部１３５とを備え、セカンダリ仮想マシン２３０は、擬似割り込み入力変換部２３１と同期補正部２３５とを備えている。

【００４９】

プライマリ仮想マシン１３０の擬似割り込み生成部１３１は、プライマリハイパーバイザ１２０から実際に入力された外部割り込みに対して、割り込みの頻度やタイミングを調整した擬似割り込みを生成して、プライマリゲストＯＳ１４０に出力する。このため、擬似割り込み生成部１３１は、割り込みの頻度を調整する割り込み頻度調整部１３２と、割り込みタイミングを調整する割り込みタイミング調整部１３３とを備えている。

【００５０】

プライマリ仮想マシン１３０の同期情報生成部１３５は、タイミング調整され、プライマリゲストＯＳ１４０に入力された擬似割り込みのタイミングを、実行中断位置と分岐実行回数とで特定する。このため、同期情報生成部１３５は、擬似割り込みが入力されるまでのプライマリゲストＯＳ１４０におけるプログラムの分岐回数をカウントする分岐実行回数生成部１３６を備えている。

【００５１】

また、同期情報生成部１３５は、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０の並列同期動作に先立ち、セカンダリ仮想マシン２３０から動作性能情報を収集する。そして、プライマリハードウェア１１０とセカンダリハードウェア２１０とで、ハードウェア構成の差異がある場合等に、速度差を補正するため等の同期補正情報を生成し、セカンダリ仮想マシン２３０の同期補正部２３５に伝達する。また、必要に応じてプライマリ仮想マシン１３０の動作の調整を行なう。

【００５２】

セカンダリ仮想マシン２３０の擬似割り込み入力変換部２３１は、プライマリ仮想マシン１３０から送られた同期情報の同期タイミング情報に従って、セカンダリゲストＯＳ２４０を擬似割り込み入力位置まで進めて中断させ、擬似割り込みを入力する。擬似割り込み入力変換部２３１は、セカンダリゲストＯＳ２４０を擬似割り込み入力位置まで進めて中断させるため、指定分岐回数実行部２３２と、指定位置コンテキスト切り替え部２３３とを備えている。

【００５３】

セカンダリ仮想マシン２３０の同期補正部２３５は、並列同期動作に先立ち、プライマリ仮想マシン１３０の同期情報生成部１３５から伝達された同期補正情報に基づいて、セカンダリ仮想マシンの動作の調整を行なう。また、必要に応じて、並列同期動作時にプラ

イマリ仮想マシン１３０とセカンダリ仮想マシン２３０との速度差の調整を行なう。

【００５４】

（並列同期動作の概要）

次に、上記構成のフォールトトレラントシステム１０における、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０の実行状態を同期させる並列同期動作の概要について図２のフローチャートを参照して説明する。具体的には、プライマリハイパーバイザ１２０が、プライマリハードウェア１１０から外部割り込みを受けた場合の一連の動作について説明する。

【００５５】

プライマリハイパーバイザ１２０が、プライマリハードウェア１１０から外部割り込みを受けると（Ｓ１０１）、プライマリ仮想マシン１３０に外部割り込みの入力を行なう（Ｓ１０２）。

【００５６】

プライマリハイパーバイザ１２０からの外部割り込みが入力されたプライマリ仮想マシン１３０は、プライマリゲストＯＳ１４０に擬似割り込みの入力を行なう。このとき、入力されたすべての外部割り込みに対して擬似割り込みの入力を行なうのではなく、割り込み頻度調整部１３２が、擬似割り込みの頻度を調整する（Ｓ１０３）。

【００５７】

仮想マシンを利用したフォールトトレラントシステム１０では、プライマリゲストＯＳ１４０に擬似割り込みが入力されると、実行状態を同期させるためにセカンダリゲストＯＳ２４０にも同じタイミングで擬似割り込みを入力する処理を行なわなくてはならない。このため、プライマリ仮想マシン１３０で頻繁に擬似割り込みが発生すると実行状態同期のための処理負荷が増え、本来のサービスに割り当てるＣＰＵ時間や通信帯域等の資源が減少してしまう。

【００５８】

そこで、本実施形態のフォールトトレラントシステム１０では、割り込み頻度調整部１３２が、擬似割り込みの頻度を調整することで、実行状態同期のための処理負荷等を抑制し、本来のサービスのための処理能力低下を防止するようにしている。割り込み頻度調整部１３２が行なう割り込み頻度調整動作の詳細については後述する。

【００５９】

割り込み頻度調整部１３２が擬似割り込みの頻度を調整した結果（Ｓ１０３）、擬似割り込みを行なうと判定した場合（Ｓ１０４：Ｙｅｓ）は、プライマリゲストＯＳ１４０に擬似割り込みの入力を行なう。擬似割り込みを行なわないと判定した場合（Ｓ１０４：Ｎｏ）は、本処理を終了し、プライマリハイパーバイザ１２０からの次の外部割り込みの入力を待つ。

【００６０】

プライマリゲストＯＳ１４０に擬似割り込みの入力を行なう場合は、セカンダリゲストＯＳ２４０でも確実に同じ位置で中断できるような位置でプライマリゲストＯＳ１４０を中断させて、擬似割り込みの入力を行なう必要がある。

【００６１】

例えば、外部割り込み等の仮想マシンコンテキスト切り替えイベントにより、プライマリゲストＯＳ１４０が中断してプライマリ仮想マシン１３０に制御が移ったときに、プライマリゲストＯＳ１４０の中断位置がクリティカルセクションである場合が起こり得る。なお、クリティカルセクションは、単一のリソースに対して、複数の処理が同時期に実行されると、破綻をきたす部分であり、プログラムにより、ブレーク命令無効化等の排他制御が行なわれる部分である。

【００６２】

このような場合、その中断位置でプライマリゲストＯＳ１４０に擬似割り込みを入力すると、クリティカルセクション内であるため、セカンダリゲストＯＳ２４０では、同じ位置にブレーク命令を埋め込んだとしても、同じ位置で中断することができず、クリティカ

ルセクションを抜け出てから中断する。このため、擬似割り込みの位置がずれてしまい、実行状態の同期がずれてしまう。

【００６３】

そこで、本実施形態のフォールトトレラントシステム１０では、割り込みタイミング調整部１３３が、擬似割り込みタイミングの調整を行ない（Ｓ１０５）、クリティカルセクションを避けてプライマリゲストＯＳ１４０を中断させることで、セカンダリゲストＯＳ２４０でも確実に同じ位置で中断できるようにしている。割り込みタイミング調整部１３３が行なう擬似割り込みタイミング調整動作の詳細については後述する。

【００６４】

擬似割り込みタイミングが調整されると、プライマリ仮想マシン１３０はプライマリゲストＯＳ１４０に擬似割り込みの入力を行なう（Ｓ１０６）。そして、同期情報生成部１３５が同期情報を生成する（Ｓ１０７）。

【００６５】

従来、同期情報は、擬似割り込みの識別情報と同期タイミング情報とを含み、同期タイミング情報は、プログラムカウンタが示す実行中断位置とパフォーマンスカウンタで測定したＣＰＵ実行命令数とを含んでいた。

【００６６】

しかしながら、ＣＰＵ固有の特殊な命令によっては、パフォーマンスカウンタのカウント漏れ等が発生して、プライマリゲストＯＳ１４０が擬似割り込み入力までに実行したＣＰＵ実行命令数を正確にカウントできない場合がある。例えば、Ｉｎｔｅｌ社ＣＰＵのＰｅｎｔｉｕｍ（登録商標）では、外部割り込みの発生によりＲＥＰ命令の実行が中断された場合にＲＥＰ命令の実行がカウントされないため、ＣＰＵ実行命令数カウンタは実際の実行命令数よりも少なくなる。

【００６７】

このように、ＣＰＵ実行命令数はカウント値が不正確の場合があるため、本実施形態のフォールトトレラントシステム１０は、同期タイミング情報にＣＰＵ実行命令数を用いないようにしている。

【００６８】

しかしながら、一般に、プログラムはループ文やジャンプ文、条件分岐文を多く含んでいるため、同じ命令を何度も通過する。このため、実行中断位置だけでは、擬似割り込み入力のタイミングを特定することができない。

【００６９】

そこで、本実施形態のフォールトトレラントシステム１０は、同期タイミング情報に実行中断位置と分岐回数とを含めるようにしている。分岐回数は、前回の擬似割り込みから今回の擬似割り込みまでに実行した分岐命令の数であり、命令コードの実行順序が順次実行から変更された回数である。

【００７０】

すなわち、プライマリゲストＯＳ１４０で、分岐回数をカウントしておけば、セカンダリゲストＯＳ２４０において、その回数分の分岐の後に最初に通過する実行中断位置を、擬似割り込み入力位置と特定することができる。

【００７１】

例えば、図３（ａ）に示すように、命令アドレスａ１、ａ２、ａ３、…に命令コードＣｏｄｅＡ、ＣｏｄｅＢ、ＣｏｄｅＣ、…が割り当てられており、ＣｏｄｅＣからＣｏｄｅＧまでが３回ループされる場合に、プライマリゲストＯＳ１４０において、プログラム実行順１６番目の「ａ６」で、擬似割り込みが入力されたとする。この同期タイミングは、分岐回数のカウント値「２」、実行中断位置「ａ６」と表わすことができる。

【００７２】

この場合、セカンダリ仮想マシン２３０において、分岐回数を２回カウントし、最初の実行中断位置「ａ６」でセカンダリゲストＯＳ２４０を中断させることで、プライマリゲストＯＳ１４０と同じタイミングで擬似割り込みを入力することができる。

【００７３】
このような分岐回数のカウント処理を行なうために、同期情報生成部１３５は、分岐実行回数生成部１３６を備えている。命令コードレベルでは、ループ文やジャンプ文、条件分岐文は、すべて分岐命令に置き換えられるため、分岐実行回数生成部１３６は、実行中のアプリケーション１５０において、プライマリゲストＯＳ１４０が実行した分岐命令の数をカウントすることにより分岐回数のカウントを行なうことができる。分岐回数のカウントは、例えば、パフォーマンスカウンタを利用することができる。
【００７４】
図２の説明に戻って、プライマリ仮想マシン１３０は、擬似割り込みの識別情報と同期タイミング情報とを含んだ同期情報を生成するとセカンダリ仮想マシン２３０に伝達する（Ｓ１０８）。
【００７５】
同期情報を受信したセカンダリ仮想マシン２３０は、同期タイミング情報に従ってセカンダリゲストＯＳ２４０の実行を中断させるために、まず、指定分岐回数実行部２３２が、セカンダリゲストＯＳ２４０の実行時に、分岐回数をカウントし、同期タイミング情報の分岐回数が示す値になるとセカンダリゲストＯＳ２４０を中断させる。すなわち、指定された分岐回数までセカンダリゲストＯＳ２４０を実行させる（Ｓ１０９）。このため、指定された分岐回数に達するまでは、実行中断位置やＣＰＵ実行命令数を監視する必要がなく、処理負荷が軽減する。指定分岐回数実行部２３２が行なう指定分岐回数実行動作の詳細については後述する。
【００７６】
そして、指定位置コンテキスト切り替え部２３３が、同期タイミング情報の実行中断位置が示す位置で、セカンダリゲストＯＳ２４０の実行を中断させる（Ｓ１１０）。
【００７７】
セカンダリゲストＯＳ２４０の実行を中断させる際に、従来のようにブレーク命令を埋め込んでブレークポイントを設定する手法を用いると、仮想マシン（１３０、２３０）上で、ゲストＯＳ（１４０、２４０）やアプリケーション（１５０、２５０）のデバッグを行なっている場合に、デバッグのブレークポイントと干渉し、相互の処理に悪影響を与えてしまう。同期状態の仮想マシン（１３０、２３０）上でアプリケーション（１５０、２５０）動作の検証等を行なうニーズもあるため、ブレークポイントの干渉は避けることが望ましい。
【００７８】
また、仮想化支援機能を持っていないＣＰＵの場合は、ブレークポイントの設定でセカンダリゲストＯＳ２４０からセカンダリ仮想マシン２３０に処理を遷移することができない。
【００７９】
そこで、本実施形態のフォールトトレラントシステム１０では、指定位置コンテキスト切り替え部２３３が、ブレーク命令を用いずに、コンテキスト切り替えを行なうことにより、セカンダリゲストＯＳ２４０の実行を中断させるようにしている。指定位置コンテキスト切り替え部２３３が行なう指定位置コンテキスト切り替え動作の詳細については後述する。
【００８０】
セカンダリ仮想マシン２３０は、セカンダリゲストＯＳ２４０を中断させると、同期情報の擬似割り込み識別情報に従って擬似割り込みの設定を行なって、セカンダリゲストＯＳ２４０を再開させる。これにより、セカンダリゲストＯＳ２４０に、プライマリゲストＯＳ１４０と同じタイミングで擬似割り込みが入力され（Ｓ１１１）、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０との実行状態が同期する。
【００８１】
なお、プライマリマシン１００、セカンダリマシン２００のいずれかでハードウェアの故障が生じると、実行状態の同期が乱れる。本実施形態では、パフォーマンスカウンタの

カウント漏れ等の影響を受けずに、正確に実行状態の同期を行なうことができる。このため、ハードウェアが正常であれば、実行状態の同期が乱れることはないので、実行状態の同期の乱れを検出することでハードウェアの故障を発見することができる。

【００８２】

従来は、実行状態の同期の乱れが故障によるものか、実行命令数のカウント漏れの影響によるか判別できないため、出力データの照合が必要であったが、本実施形態のフォールトトレラントシステム１０では、実行状態の同期の乱れにより故障を検出することができるため、故障検出のための出力データの照合が不要となる。これにより、出力データの照合のためのＣＰＵ処理負荷や通信負荷を削減することができる。また、故障の発生を迅速に検出できる。

【００８３】

なお、実行状態の同期の乱れの検出は、例えば、セカンダリ仮想マシン２３０の擬似割り込み入力変換部２３１において、同期タイミング情報で指定された実行中断位置を通過する前に、分岐回数が、同期タイミング情報で指定された回数を超過した場合に同期が乱れたと判定することができる。

【００８４】

（各ブロックの動作の詳細）

次に、フォールトトレラントシステム１０の各ブロックの動作について第１実施例、第２実施例、第３実施例により詳細に説明する。

【００８５】

従来のフォールトトレラントシステムは、プライマリマシンのＣＰＵとセカンダリマシンのＣＰＵとが同じアーキテクチャであることを前提としていた。これは、従来の同期合わせの手法が、ＣＰＵアーキテクチャやＣＰＵ動作仕様に依存しており、ＣＰＵの種類が異なれば、双方の実行命令を合わせることができず、同期ずれが発生する場合があったからである。

【００８６】

しかしながら、本発明は、第１実施例、第２実施例で説明するように、プライマリマシン１００のＣＰＵ１１１とセカンダリマシン２００のＣＰＵ２１１とが同じアーキテクチャのみならず、異なるアーキテクチャの場合でも適用することができる。

【００８７】

一般に、長年に渡ってフォールトトレラントシステムを運用する場合には、故障や経年劣化による機器の交換が余儀なくされるが、この際に、従前の機器と同一仕様の機器を用意することは困難である。このため、本実施形態のフォールトトレラントシステム１０は、ＣＰＵアーキテクチャが異なる場合であっても、並列同期動作を継続することができる仕組みを備えるようにしている。

【００８８】

＜第１実施例＞

第１実施例として、プライマリマシン１００のＣＰＵ１１１とセカンダリマシン２００のＣＰＵ２１１とが同じアーキテクチャの場合の動作について説明する。ここで、ＣＰＵアーキテクチャが同じとは、ＣＰＵ命令セットに互換性があることを意味する。

【００８９】

この場合、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０とは、ＣＰＵ命令レベルの同期合わせによる並列同期動作を行なう。このため、両仮想マシン（１３０、２３０）において、ゲストＯＳ（１４０、２４０）とアプリケーション（１５０、２５０）のプログラムは、通常どおり、ＣＰＵ（１１１、２１１）に対してネイティブのＣＰＵ命令コードにコンパイルされたコード形式を使用する。この並列同期動作を、「ＣＰＵ命令同期モード」と称するものとする。

【００９０】

（割り込み頻度調整動作）

まず、割り込み頻度調整部１３２が行なう割り込み頻度調整動作（Ｓ１０３）について

説明する。プライマリ仮想マシン１３０の割り込み頻度調整部１３２は、すべての外部割り込みに対して擬似割り込みの入力を行なうのではなく、アプリケーション１５０に要求される性能や精度、許容通信ディレイ等を満たす範囲で、擬似割り込みの頻度を調整する。

【００９１】

一般に、外部割り込みは、タイマ１１３等による定周期割り込みと、ネットワークインタフェースカード（ＮＩＣ）１１４、デバイス１１５等による非定周期割り込みとに区別することができるが、割り込み頻度調整部１３２は、頻度の調整を、定周期割り込みと非定周期割り込みとで分けて行なう。

【００９２】

定周期の外部割り込みの場合は、間引きを行なって、定周期の外部割り込み数回に１回の割合で擬似割り込みをプライマリゲストＯＳ１４０に入力するようにする。

【００９３】

例えば、定周期のタイマ割り込みをプライマリゲストＯＳ１４０に入力することによって、プライマリゲストＯＳ１４０は正確な時刻を計時することができるが、アプリケーション１５０で要求される時刻の精度が高くない場合は、タイマ割り込みの入力を間引いても影響はない。

【００９４】

例えば、プライマリハイパーバイザ１２０からのタイマ割り込みの周期が１ｍｓ、アプリケーション１５０で要求される時刻の精度が１００ｍｓの場合、擬似割り込みの周期を１０ｍｓに間引いても１００ｍｓの精度を維持することができる。このとき、擬似割り込みの頻度は１／１０に減少するため、実行状態の同期処理も１／１０に減少する。

【００９５】

なお、アプリケーション１５０の要求精度に応じて、予め間引く割合を設定しておいてもよいし、パラメータで調整できるようにしてもよい。また、定周期の外部割り込みに対して一律に間引く割合を設定してもよいし、定周期の外部割り込みの要因ごとに間引く割合を設定してもよい。

【００９６】

非定周期の外部割り込みの場合は、処理途中に入力される外部割り込みに対する擬似割り込みを省き、処理が完了した最後の外部割り込みのみを擬似割り込みとしてプライマリゲストＯＳ１４０に入力する。

【００９７】

例えば、ＮＩＣ１１４からのネットワークの受信割り込みは、ＮＩＣ１１４にデータが到着する度に発生する。このため、データとして意味のある単位を受信する間に、複数回の外部割り込みが発生する。従来は、外部割り込みの発生ごとにプライマリゲストＯＳ１４０に擬似割り込みを行なって、ＮＩＣ１１４からプライマリゲストＯＳ１４０に受信データを渡していたが、割り込み頻度調整部１３２は、データ受信途中の外部割り込みに対する擬似割り込みを省き、データ受信完了後の１回の外部割り込みに対して擬似割り込みをプライマリゲストＯＳ１４０に入力する。これにより、実行状態の同期処理の回数が減少するとともに、まとまったサイズのデータを１度に処理することが可能となる。

【００９８】

なお、本手法を適用できる非定周期の外部割り込みとしては、ネットワークの送受信割り込み、ストレージの読み書き割り込み、シリアル通信の送受信割り込み、アナログ／デジタル入出力ボード等の各種Ｉ／Ｏデバイスの読み書き割り込み等が挙げられる。

【００９９】

（擬似割り込みタイミング調整動作）

次に、割り込みタイミング調整部１３３が行なう擬似割り込みタイミング調整動作（Ｓ１０５）について説明する。擬似割り込みタイミング調整は、セカンダリゲストＯＳ２４０が同じ位置で中断できるように、プライマリゲストＯＳ１４０のクリティカルセクションを避けて擬似割り込みを入力するための調整である。

【０１００】
擬似割り込みタイミング調整動作は、ＣＰＵアーキテクチャの詳細が判別している場合と、ＣＰＵアーキテクチャの詳細が不明の場合とで処理が異なる。なお、ＣＰＵアーキテクチャの詳細が判別している場合とは、中断位置のＣＰＵ命令や状態フラグ等から、中断位置がクリティカルセクションかどうかを判別できる場合をいう。
【０１０１】
まず、ＣＰＵアーキテクチャの詳細が判別している場合の具体的な動作手順について、図４のフローチャートを参照して説明する。
【０１０２】
プライマリ仮想マシン１３０は、プライマリハイパーバイザ１２０から外部割り込みの入力を受けると（Ｓ２０１）、プライマリゲストＯＳ１４０の実行の中断を待つ（Ｓ２０２）。
【０１０３】
プライマリゲストＯＳ１４０の実行が中断し、プライマリ仮想マシン１３０に処理が遷移すると、プライマリ仮想マシン１３０は、プライマリ仮想マシン１３０の状態を取得する（Ｓ２０３）。取得する状態は、クリティカルセクションかどうかを判別するための情報であり、例えば、ＣＰＵ１１１の割り込みマスクの設定情報、フラグレジスタの内容等である。
【０１０４】
そして、取得した情報を元に、中断位置がクリティカルセクションかどうかを判断する（Ｓ２０４）。例えば、割り込みマスクで割り込みの入力が禁止されていないこと、ブレーク命令の設定を無効化するフラグがフラグレジスタに設定されていないこと等で、中断位置がクリティカルセクションでないと判断することができる。
【０１０５】
中断位置がクリティカルセクションでない場合（Ｓ２０４：Ｎｏ）は、中断位置を擬似割り込みの入力タイミングに設定する（Ｓ２０５）。
【０１０６】
中断位置がクリティカルセクションの場合（Ｓ２０４：Ｙｅｓ）は、プライマリ仮想マシン１３０の状態を確認しながらプライマリゲストＯＳ１４０の実行を進める（Ｓ２０６）。このとき、例えば、ステップ実行を行なうことで、１命令ごとに状態を確認する（Ｓ２０３）。そして、クリティカルセクションかどうかを判断し（Ｓ２０４）、クリティカルセクションを抜け出すと（Ｓ２０４：Ｎｏ）、擬似割り込みの入力タイミングに設定する（Ｓ２０５）。
【０１０７】
次に、ＣＰＵアーキテクチャの詳細が不明の場合の具体的な動作手順について、図５のフローチャートを参照して説明する。ＣＰＵアーキテクチャの詳細が不明の場合、プライマリ仮想マシン１３０の状態を取得しても、クリティカルセクションかどうかを判別できないため、中断のためのイベントを設定し、実際に中断できた位置を、擬似割り込みを入力する中断位置とする。
【０１０８】
プライマリ仮想マシン１３０は、プライマリハイパーバイザ１２０から外部割り込みの入力を受けたら（Ｓ３０１）、プライマリゲストＯＳ１４０の実行の中断を待つ（Ｓ３０２）。
【０１０９】
プライマリゲストＯＳ１４０の実行が中断し、プライマリ仮想マシン１３０に処理が遷移すると、プライマリ仮想マシン１３０は、プライマリゲストＯＳ１４０に、所定命令数実行後、例えば、１命令実行後に中断する旨のイベントの設定を行なう（Ｓ３０３）。
【０１１０】
そして、ステップ実行でプライマリゲストＯＳ１４０の実行を再開し（Ｓ３０４）、１命令実行ごとに中断の要因を取得する（Ｓ３０５）。これは、中断の要因が、必ずしも設

定した中断イベントによるものとは限られないからである。
【０１１１】
中断の要因が、設定した中断イベントによるものの場合（Ｓ３０６：Ｙｅｓ）は、中断位置がクリティカルセクションではなく、セカンダリゲストＯＳ２４０でも同じ位置で中断できるものと判断し、擬似割り込みの入力タイミングに設定する（Ｓ３０７）。
【０１１２】
中断の要因が、設定した中断イベントによるものでない場合（Ｓ３０６：Ｎｏ）は、中断位置がクリティカルセクションであり、セカンダリゲストＯＳ２４０では同じ位置で中断できないものと判断し、再度、１命令実行後に中断する旨のイベントの設定を行なって（Ｓ３０３）、プライマリゲストＯＳ１４０の実行再開（Ｓ３０４）以降の処理を繰り返し、クリティカルセクションを抜け出すと（Ｓ３０６：Ｙｅｓ）、擬似割り込みの入力タイミングに設定する（Ｓ３０７）。
【０１１３】
（同期情報生成動作）
次に、同期情報生成部１３５が行なう同期情報の生成動作（Ｓ１０７）について説明する。プライマリマシン１００のＣＰＵ１１１とセカンダリマシン２００のＣＰＵ２１１とが同じアーキテクチャの場合、同期情報の同期タイミング情報における実行中断位置は、ＣＰＵ命令コードの実行停止位置で示され、プログラムカウンタ値を用いることができる。
【０１１４】
また、同期タイミング情報における分岐実行回数は、前回の同期タイミング情報生成時からプライマリゲストＯＳ１４０で実行された分岐回数で示される。分岐回数は、ＣＰＵ命令コードレベルで実行した分岐命令をカウントすることで得ることができる。
【０１１５】
（指定分岐回数実行動作）
次に、指定分岐回数実行部２３２が行なう指定分岐回数実行動作（Ｓ１０９）について説明する。指定分岐回数実行動作は、セカンダリ仮想マシン２３０が、同期タイミング情報の分岐回数で示された回数でセカンダリゲストＯＳ２４０を中断させる動作である。
【０１１６】
指定分岐回数実行部２３２は、ＣＰＵ命令コードレベルで実行した分岐命令をカウントし、カウント値が指定された分岐回数に達すると、その旨をセカンダリ仮想マシン２３０に通知する。これにより、セカンダリ仮想マシン２３０にコンテキストが切り替えられ、セカンダリゲストＯＳ２４０の実行が中断する。
【０１１７】
なお、ＣＰＵ２１１のパイプライン機能等の高速化により、指定された分岐回数の実行直後に停止できないＣＰＵアーキテクチャの場合には、指定された分岐回数より少ない値に設定し、指定された分岐回数で確実に中断できるようにする。
【０１１８】
（指定位置コンテキスト切り替え動作）
次に、指定位置コンテキスト切り替え部２３３が行なう指定位置コンテキスト切り替え動作（Ｓ１１０）について説明する。指定位置コンテキスト切り替え動作は、セカンダリゲストＯＳ２４０を指定分岐回数で中断した後、指定された実行中断位置で、ブレーク命令を用いずに、セカンダリ仮想マシン２３０にコンテキストを切り替える動作である。
【０１１９】
セカンダリ仮想マシン２３０に専用のコンテキスト切り替え命令が用意されている場合は、セカンダリゲストＯＳ２４０の実行中断位置に、コンテキスト切り替え命令を埋め込むようにする。コンテキスト切り替え命令は、例えば、Ｉｎｔｅｌ社ＣＰＵのＶＭＣＡＬＬ命令である。
【０１２０】
セカンダリゲストＯＳ２４０が、コンテキスト切り替え命令を実行すると、コンテキス

トがセカンダリゲストＯＳ２４０からセカンダリ仮想マシン２３０に切り替わり、セカンダリゲストＯＳ２４０の実行が中断される。

【０１２１】

専用のコンテキスト切り替え命令が用意されていない場合は、実行中断位置で擬似割り込みやＣＰＵ例外のイベントを発生させてコンテキスト切り替えが行なわれるように設定する。コンテキスト切り替えが行なわれると、コンテキストがセカンダリゲストＯＳ２４０からセカンダリ仮想マシン２３０に切り替わり、セカンダリゲストＯＳ２４０の実行が中断される。

【０１２２】

いずれの場合も、セカンダリゲストＯＳ２４０の実行が中断されると、セカンダリ仮想マシン２３０は、擬似割り込みの設定をして、セカンダリゲストＯＳ２４０の実行を再開させ、そのタイミングで擬似割り込みがセカンダリゲストＯＳ２４０に入力される。

【０１２３】

＜第２実施例＞

第２実施例として、プライマリマシン１００のＣＰＵ１１１とセカンダリマシン２００のＣＰＵ２１１とが異なるアーキテクチャの場合の動作について説明する。ここで、ＣＰＵアーキテクチャが同じとは、ＣＰＵ命令セットに互換性がないことを意味する。

【０１２４】

この場合、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０とは、中間言語レベルの同期合わせによる並列同期動作を行なう。ここで、「中間言語」は、「中間コード」、「バイトコード」等とも称され、高水準言語のソースコードと機械語のコードの中間にあたる中間表現の言語である。この並列同期動作を「中間言語同期モード」と称するものとする。

【０１２５】

両仮想マシン（１３０、２３０）間で、ＣＰＵ命令セットの互換性がないため、ゲストＯＳ（１４０、２４０）とアプリケーション（１５０、２５０）のプログラムは、ネイティブのＣＰＵ命令コードの代わりに、Ｊａｖａ（登録商標）やＣ＃等を変換した中間言語のプログラムを使用する。

【０１２６】

両仮想マシン（１３０、２３０）は、インタープリタやＪＩＴコンパイラ等の既存のＣＰＵ命令変換手法を用いて、中間言語を仮想マシンＣＰＵに対してネイティブのＣＰＵ命令に変換して実行する。

【０１２７】

なお、プライマリマシン１００のＣＰＵ１１１とセカンダリマシン２００のＣＰＵ２１１とが同じアーキテクチャの場合も、中間言語同期モードによる並列同期動作を行なうことができるが、ＣＰＵ命令同期モードによる並列同期動作の方が、実行速度が速くなる。

【０１２８】

次に、第２実施例における各ブロックの動作について説明するが、大部分は第１実施例のＣＰＵ命令同期モードと同様であるため、第１実施例と異なる処理についてのみ説明する。

【０１２９】

（擬似割り込みタイミング調整動作）

第１実施例のＣＰＵ命令同期モードにおいて、擬似割り込みタイミング調整動作（Ｓ１０５）は、ＣＰＵアーキテクチャの詳細が判別している場合と、ＣＰＵアーキテクチャの詳細が不明の場合とで処理が異なっていたが、第２実施例の中間言語同期モードでは、ＣＰＵアーキテクチャの詳細が判別している場合と同じ処理を行なう。

【０１３０】

すなわち、中断位置のＣＰＵ命令や状態フラグ等から、中断位置がクリティカルセクションかどうかを判別できるため、図４に示したフローチャートに従って、クリティカルセクションを避けてプライマリゲストＯＳ１４０の中断位置を設定し、擬似割り込みを入力

する。
【０１３１】
また、中間言語同期モードでは、中断位置のＣＰＵ命令や状態フラグに加え、中間言語レベルでクリティカルセクションかどうかを判断することができる。例えば、中間言語がＪａｖａ（登録商標）のバイトコードの場合は、クリティカルセクションはｓｙｎｃｈｒｏｎｉｚｅｄの排他制御を使って実装することが想定される。このため、実行済みの中間言語の命令を確認することで、プライマリ仮想マシン１３０は、中断したＪａｖａ（登録商標）の中間言語がクリティカルセクションにいるかを把握できる。
【０１３２】
（同期情報の生成動作）
第１実施例のＣＰＵ命令同期モードでは、同期情報の生成動作（Ｓ１０７）において、同期タイミング情報に含まれる実行中断位置は、ＣＰＵ命令コードの実行停止位置で示し、プログラムカウンタ値を用いた。また、同期タイミング情報に含まれる分岐実行回数は、前回の同期タイミング情報生成時から実行されたＣＰＵ命令コードの分岐回数で示した。
【０１３３】
これに対し、第２実施例の中間言語同期モードでは、実行中断位置は、擬似割り込みを入力した時点で実行した中間言語レベルの実行命令位置を用いる。また、分岐実行回数は、前回の同期タイミング情報生成時から実行された中間言語レベルの分岐回数を用いる。
【０１３４】
（指定位置コンテキスト切り替え動作）
第１実施例のＣＰＵ命令同期モードでは、指定位置コンテキスト切り替え動作（Ｓ１１０）において、セカンダリ仮想マシン２３０に専用のコンテキスト切り替え命令が用意されている場合は、セカンダリゲストＯＳ２４０の実行中断位置に、コンテキスト切り替え命令を埋め込み、専用のコンテキスト切り替え命令が用意されていない場合は、実行中断位置で擬似割り込みやＣＰＵ例外のイベントを発生させてコンテキスト切り替えが行なわれるように設定した。第２実施例の中間言語同期モードでも、中間言語レベルの実行中断位置で、ＣＰＵ命令変換により同様の設定を行なう。
【０１３５】
＜第３実施例＞
次に、第３実施例として、プライマリマシン１００とセカンダリマシン２００とのハードウェア構成に相違がある場合の並列同期動作について説明する。プライマリマシン１００のプライマリハードウェア１１０とセカンダリマシン２００のセカンダリハードウェア２１０は、ＣＰＵ（１１１、２１１）、メモリ（１１２、２１２）、ストレージ等のデバイス（１１５、２１５）等、種々のハードウェアで構成されているが、双方のハードウェアの機能や性能は必ずしも一致するわけではない。
【０１３６】
例えば、ＣＰＵ（１１１、２１１）のスペックが異なっていたり、メモリ（１１２、２１２）の容量が異なっている場合も想定される。また、フォールトトレラントシステム１０の運用過程においてデバイス２１５が交換される等により、双方のデバイス（１１５、２１５）が異なっている場合もある。
【０１３７】
このため、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０とで、動作は同じだが、処理速度に差が生じる場合がある。上述のように、フォールトトレラントシステム１０は、プライマリ仮想マシン１３０が先行して動作を行なうようにしているが、例えば、セカンダリ仮想マシン２３０の方が、処理速度が速いと、動作の逆転現象が生じ、プライマリ仮想マシン１３０からの同期情報に基づく実行状態の同期が行なえなくなる。プライマリ仮想マシン１３０の方が、処理速度が速いと、セカンダリ仮想マシン２３０の遅れが蓄積されていく。
【０１３８】

また、異なるデバイス（１１５、２１５）により、プライマリゲストＯＳ１４０とセカンダリゲストＯＳ２４０のドライバ処理が異なって、動作に相違が生じる結果、同期ずれが生じるおそれがある。

【０１３９】

そこで、フォールトトレラントシステム１０では、プライマリハードウェア１１０とセカンダリハードウェア２１０との差異に基づく処理速度差や同期ずれを補正する仕組みを設けている。具体的には、プライマリ仮想マシン１３０の同期情報生成部１３５とセカンダリ仮想マシン２３０の同期補正部２３５とが同期補正動作を行なう。同期補正動作は、並列同期動作実行前の同期補正動作と、並列同期動作実行時の同期補正動作とに分けることができる。

【０１４０】

（並列同期動作実行前の同期補正動作）

プライマリ仮想マシン１３０の同期情報生成部１３５は、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０の並列同期動作に先立ち、セカンダリ仮想マシン２３０からセカンダリ仮想マシン２３０に関する動作性能情報を収集する。

【０１４１】

動作性能情報は、セカンダリ仮想マシン２３０におけるセカンダリゲストＯＳ２４０に対するインタフェースの動作仕様や、セカンダリ仮想マシン２３０のプログラム実行速度の指標となるベンチマーク情報等を含んだ情報である。同期情報生成部１３５は、使用可能な範囲で複数項目の動作性能情報を取得するようにする。

【０１４２】

同期情報生成部１３５は、プライマリ仮想マシン１３０自身の動作性能情報も取得して、セカンダリ仮想マシン２３０の動作性能情報と対照する。そして、プライマリ仮想マシン１３０の並列同期動作における動作能力と、セカンダリ仮想マシン２３０の並列同期動作における動作能力とがほぼ等しくなるように、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０が使用する動作能力を設定する。

【０１４３】

動作能力は、例えば、動作周波数、メモリサイズ、キャッシュサイズ、ＴＬＢ（Translation Look-aside Buffer）／ＥＰＴ（Enhanced Page Table）仕様等の動作仕様、ＭＩＰＳ値やＳＰＥＣ値等のベンチマーク情報である。中間言語同期モードの場合は、中間言語のベンチマーク値を利用することができる。例えば、中間言語としてＪａｖａ（登録商標）のバイトコードを用いた場合は、ベンチマーク値としてＳＰＥＣｊｂｂ、ＣａｆｆｅｉｎｅＭａｒｋ等が挙げられる。

【０１４４】

そして、セカンダリ仮想マシン２３０が使用する動作能力を同期補正情報としてセカンダリ仮想マシン２３０の同期補正部２３５に通知する。セカンダリ仮想マシン２３０の同期補正部２３５は、通知された同期補正情報に従って、並列動作実行時におけるセカンダリ仮想マシン２３０の動作能力を設定する。

【０１４５】

動作周波数の調整による同期補正を例に説明する。なお、上述のように、プライマリマシン１００とセカンダリマシン２００とのハードウェア構成に相違があるときは、動作は同じだが、処理速度に差が生じる場合と、動作自体が異なる場合とがある。本例は、動作は同じだが、処理速度に差が生じる場合の同期補正の説明である。

【０１４６】

仮想マシン（１３０、２３０）の動作周波数の調整には、省電力のために用意された電力制御機能を利用することができる。例えば、標準の電力制御機能にはＡＣＰＩ（Advanced Configuration and Power Interface）を使用し、動作周波数を変更する機能には、Ｌｉｎｕｘ（登録商標）のｃｐｆｒｅｑサブシステム等を使用することができる。この機能を利用して、双方の仮想マシン（１３０、２３０）は、使用可能な動作周波数を複数個用意しておく。

【０１４７】
そして、プライマリ仮想マシン１３０の同期情報生成部１３５は、セカンダリ仮想マシン２３０で使用可能な動作周波数の一覧をセカンダリ仮想マシン２３０から取得するとともに、プライマリ仮想マシン１３０自身で使用可能な動作周波数の一覧を取得して、両者がほぼ一致する動作周波数を選択する。選択した動作周波数はセカンダリ仮想マシン２３０の同期補正部２３５に同期補正情報として通知する。
【０１４８】
プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０の並列同期動作時には、それぞれの仮想マシン（１３０、２３０）は、選択された動作周波数で動作するようにすることで、プライマリハードウェア１１０とセカンダリハードウェア２１０との動作周波数の差異に基づく処理速度差が解消される。
【０１４９】
なお、プライマリ仮想マシン１３０の使用可能な動作周波数と、セカンダリ仮想マシン２３０の使用可能な動作周波数とが一致しない場合は、例えば、セカンダリ仮想マシン２３０の方を速い動作周波数とし、後述するように、セカンダリ仮想マシン２３０の動作にウェイトを入れるようにすることができる。
【０１５０】
中間言語同期モードの場合は、ベンチマーク値の違いによりプライマリ仮想マシン１３０とセカンダリ仮想マシン２３０の処理時間に差が生じる。従って、取得した双方のベンチマーク値に基づいて、動作周波数と同様の調整を行なうことによって処理速度差を補正する。
【０１５１】
次に、ハードウェア構成の相違により、プライマリ仮想マシン１３０とセカンダリ仮想マシン２３０とで動作自体が異なる場合の同期補正について説明する。
【０１５２】
例えば、ＣＰＵアーキテクチャが同じでもデバイス１１５等の動作仕様が異なる場合には、双方のゲストＯＳ（１４０、２４０）が同じドライバ処理を実行するように変更することで、双方のゲストＯＳ（１４０、２４０）が同じ動作を行ない、同期ずれを防ぐことができる。具体的には、並列同期動作の開始前に、プライマリ仮想マシン１３０またはセカンダリ仮想マシン２３０のデバイスエミュレータに動作仕様の違いを吸収する機能を組み込むようにする。
【０１５３】
この機能は、補正の対象によって下記の２種類に分類できる。
【０１５４】
（１）ハードウェア機能の有無の違いに対する補正
あるハードウェア機能を一方の仮想マシン（１３０、２３０）だけが持っている場合は、機能を持っている方の仮想マシン（１３０、２３０）がその機能を利用しないようにし、双方の仮想マシン（１３０、２３０）がそのハードウェア機能を持たない状態で動作する。
【０１５５】
例えば、ＦＰＵ（浮動小数点演算装置）機能の有無が異なる場合、双方の仮想マシン（１３０、２３０）ともＦＰＵ機能を持たない状態で動作させる。このとき浮動小数点演算はゲストＯＳ（１４０、２４０）内のエミュレーションコードで実行される。具体的には、浮動小数点演算実行時に例外が発生し、ゲストＯＳ（１４０、２４０）の例外ハンドラがこの例外を捕捉してソフトウェアで演算をエミュレートする。
【０１５６】
あるいは、機能を持っていない方の仮想マシン（１３０、２３０）が、その機能をゲストＯＳ（１４０、２４０）の外部でソフトウェアの処理によってエミュレートすることにより、違いを吸収するようにしてもよい。ゲストＯＳ（１４０、２４０）の外部とは、仮想マシン（１３０、２３０）やハイパーバイザ（１２０、２２０）などである。

【０１５７】
例えば、ＦＰＵ機能を持っていない方の仮想マシン（１３０、２３０）が、仮想マシン（１３０、２３０）内のエミュレーションコードで浮動小数点演算を実行する。具体的には、浮動小数点演算実行時に発生する例外によって仮想マシンコンテキストへの切り替えを行ない、仮想マシン（１３０、２３０）内のソフトウェアで演算をエミュレートした後、ゲストＯＳコンテキストに戻る。ゲストＯＳ（１４０、２４０）の中では例外を捕捉しないため、ゲストＯＳ（１４０、２４０）内のエミュレーションコードは実行されない。
【０１５８】
（２）ハードウェア機能の能力や容量の違いに対する補正
ハードウェア機能の能力や容量が異なる場合は、動作に相違が生じる処理をゲストＯＳ（１４０、２４０）の外部で実行する。ゲストＯＳ（１４０、２４０）の外部とは、仮想マシン（１３０、２３０）やハイパーバイザ（１２０、２２０）、ハードウェア（１１０、２１０）などである。
【０１５９】
例えば、ＴＬＢのキャッシュサイズが異なる場合、一方の仮想マシン（１３０、２３０）のみがキャッシュにヒットせず、ＴＬＢミスが発生する場合がある。そこで、ＴＬＢミスに対するリカバリー処理を、ゲストＯＳ（１４０、２４０）の外部で実行する。Ｉｎｔｅｌ社ＣＰＵの場合は、ハードウェアの機能でリカバリー処理を行なうことができるが、ＣＰＵ（１１１、２１１）にハードウェアのリカバリー機能がない場合は、仮想マシン（１３０、２３０）やハイパーバイザ（１２０、２２０）のソフトウェアでリカバリー処理を行なう。
【０１６０】
中間言語同期モードの場合は、中間言語で表現されたゲストＯＳプログラムを仮想マシン（１３０、２３０）が仮想メモリ上でインタープリタなどのソフトウェア機能を使って実行する。このため、動作仕様の違いによってゲストＯＳ（１４０、２４０）の動作に相違が生じることはない。
【０１６１】
（並列同期動作実行時の同期補正動作）
並列同期動作の開始前の調整によってもプライマリ仮想マシン１３０とセカンダリ仮想マシン２３０の処理速度差が解消しきれない場合には、同期情報生成部１３５、同期補正部２３５は、並列同期動作時に、以下のような同期補正を行なう。
【０１６２】
まず、動作能力調整後のプライマリ仮想マシン１３０が動作能力調整後のセカンダリ仮想マシン２３０よりも高速である場合の同期補正について説明する。
【０１６３】
同期情報生成部１３５は、セカンダリ仮想マシン２３０に通知する同期補正情報の動作速度情報として、プライマリ仮想マシン１３０の動作能力とするか、セカンダリ仮想マシン２３０の動作能力とするかを選択する。この選択は、例えば、パラメータ設定に基づいて行なうことができる。
【０１６４】
プライマリ仮想マシン１３０の動作能力とした場合は、プライマリ仮想マシン１３０の先行動作が拡大していくため、セカンダリ仮想マシン２３０の遅れを解消するための処理を行なう。このとき、セカンダリ仮想マシン２３０の遅れが想定される量よりも大きい場合に、マシンの故障が発生したと検出するようにしてもよい。
【０１６５】
ここでは、遅れを解消するための処理として、２つの補正方法を説明する。まず、第１の補正方法である遅れをリセットして再同期する方法について説明する。
【０１６６】
セカンダリ仮想マシン２３０の同期補正部２３５は、セカンダリ仮想マシン２３０のプログラムの実行速度の実測遅延時間が、同期補正情報から得られる想定遅延時間と等しい

か確認する。
【０１６７】
ここで、実測遅延時間について説明する。セカンダリ仮想マシン２３０は、プライマリ仮想マシン１３０から、タイマによる定周期疑似割り込みの同期タイミング情報が伝達されるが、この同期タイミング情報は、プライマリ仮想マシン１３０の実行速度で実行命令数に換算した周期時間のことであり、セカンダリ仮想マシン２３０にとっては理論値である。一方、セカンダリ仮想マシン２３０にタイマによる定周期割り込みを発生させた時の周期時間は、実測値である。実測値から理論値を引いた値が実測遅延時間である。
【０１６８】
例えば、プライマリ仮想マシン１３０で実際に発生させたタイマ割り込みの周期値が１０ｍｓのとき、セカンダリ仮想マシン２３０では、プライマリ仮想マシン１３０と同じ動作を行なった後に擬似割り込みを入力すると、セカンダリ仮想マシン２３０でのタイマ割り込みの周期の実測値が１５ｍｓであったとする。この場合、理論値が１０ｍｓ、実測値が１５ｍｓとなるため、タイマ割り込みは５ｍｓの遅れと算出される。
【０１６９】
実測遅延時間が、同期補正情報から得られる想定遅延時間と同程度であれば、正常な遅れと判断し、処理を継続する。処理を継続することにより、実測遅延時間は積算されるため、許容範囲となる遅延許容時間を予め定めておき、実測遅延時間が遅延許容時間を超えた時点で、並列同期動作の開始時と同じ同期初期化処理を再度行なって、遅れをリセットする。
【０１７０】
例えば、プライマリ仮想マシン１３０のプログラム実行速度が１５００ＭＩＰＳ（Million Instructions Per Second）、セカンダリ仮想マシン２３０のプログラム実行速度が１０００ＭＩＰＳの場合、セカンダリ仮想マシン２３０では、１秒ごとに５００ＭＩ（５億命令数）を処理する時間分の遅延が発生していれば、正常の遅れと判断する。また、１０秒間継続して蓄積された５０００ＭＩ（５０億命令数）を処理する時間は、セカンダリ仮想マシン２３０では、５秒間に相当し、遅延許容時間を超えた場合には、再同期して遅れをリセットする。
【０１７１】
次に、第２の補正方法である時間を進めて遅れを取り戻す方法について説明する。これは、プライマリゲストＯＳ１４０がアイドル状態になり、次の擬似割り込みが入力されるまで動作が停止している場合、つまり、ＣＰＵ処理の負荷が１００％より小さい場合は、セカンダリゲストＯＳ２４０が再開する時間を進めることで遅れを取り戻す。
【０１７２】
プライマリゲストＯＳ１４０のアイドル状態とは、実行する処理がなく、プライマリゲストＯＳ１４０の動作が停止している状態である。プライマリゲストＯＳ１４０は、擬似割り込みが入力されて割り込みハンドラ処理を実行するまでは動作を再開しない。それまでは、プライマリ仮想マシン１３０が外部割り込みの入力を待っている状態である。
【０１７３】
プライマリ仮想マシン１３０では、実際に外部割り込みが入力されるまで待ち続け、セカンダリ仮想マシン２３０では、擬似割り込みを入力するために同期情報を受信するまで待ち続ける。
【０１７４】
セカンダリ仮想マシン２３０が遅れている場合は、同期情報が溜まっているため、次に入力する擬似割り込みを知ることができる。そのため、セカンダリ仮想マシン２３０は、プライマリ仮想マシン１３０と同じ時間待つ必要はなく、セカンダリゲストＯＳ２４０が再開する時間を進めて擬似割り込みを入力することができる。
【０１７５】
例えば、第１の補正方法と同様に、プライマリ仮想マシン１３０のプログラム実行速度を１５００ＭＩＰＳ、セカンダリ仮想マシン２３０のプログラム実行速度を１０００ＭＩ

ＰＳとする。プライマリゲストＯＳ１４０のＣＰＵ負荷が２０％の場合、１秒間に３００ＭＩの処理を行ない、１秒間の残りの時間だけプライマリ仮想マシン１３０を停止させる。プライマリ仮想マシン１３０の停止時間は、プライマリ仮想マシン１３０が、１２００ＭＩの処理を行なう時間分である。

【０１７６】

一方、セカンダリ仮想マシン２３０は、１秒間にプライマリ仮想マシン１３０と同じ３００ＭＩの処理を行なった後に、プライマリ仮想マシン１３０と同じ１２００ＭＩの処理を行なう時間分停止するのではなく、１秒間の残りの時間、つまり、セカンダリ仮想マシン２３０が７００ＭＩの処理を行なう時間分停止する。このように、セカンダリ仮想マシンの停止時間を１２００ＭＩから７００ＭＩ処理時間分に短縮することにより、セカンダリ仮想マシン２３０の遅れを取り戻すことが可能となる。

【０１７７】

次に、プライマリ仮想マシン１３０がセカンダリ仮想マシン２３０よりも高速である場合に、セカンダリ仮想マシン２３０に通知する同期補正情報の動作速度情報として、セカンダリ仮想マシン２３０の動作能力とした場合について説明する。

【０１７８】

この場合、プライマリ仮想マシン１３０がセカンダリ仮想マシン２３０の動作能力で動作するため、並列同期動作時に、プライマリ仮想マシン１３０の同期情報生成部１３５は、同期補正処理を行ない、プログラム実行速度を調整する。具体的には、セカンダリ仮想マシン２３０のプログラム実行速度に合わせて、ウェイトを入れながら動作を行なう。

【０１７９】

次に、動作能力調整後のセカンダリ仮想マシン２３０が動作能力調整後のプライマリ仮想マシン１３０よりも高速である場合の同期補正について説明する。同期情報生成部１３５は、セカンダリ仮想マシン２３０に通知する同期補正情報の動作速度情報を、プライマリ仮想マシン１３０の動作能力とする。

【０１８０】

そして、プライマリ仮想マシン１３０は、並列同期動作時には、同期補正処理を行なわず、セカンダリ仮想マシン２３０側で、プライマリ仮想マシン１３０のプログラム実行速度に合わせて、ウェイトを入れながら動作させる。

【０１８１】

例えば、プライマリ仮想マシン１３０のプログラム実行速度が１０００ＭＩＰＳ、セカンダリ仮想マシン２３０の実行速度が１５００ＭＩＰＳの場合、セカンダリ仮想マシン２３０は、定周期の擬似割り込みごとに、１０００ＭＩの処理を行なった後で、５００ＭＩの処理をする時間分のウェイトを入れる。これにより、双方の仮想マシン（１３０、２３０）は、等しく１０００ＭＩＰＳの処理を行なうことができる。

【０１８２】

（アクセス時間が不確定なストレージデバイスに対する対応）

仮想マシン（１３０、２３０）から磁気ディスク、ＳＳＤ、ＣＤ、ＤＶＤ、テープ等のストレージデバイス（１１５、２１５）にアクセスを行なう場合、ストレージデバイス（１１５、２１５）の個体差や経年劣化等によりアクセス要求に対する完了待ち時間にばらつきが生じる。このばらつきによる同期ずれを防ぐためには、遅い方の完了待ち時間に合わせる必要があり、同期待ちが発生して実行速度の低下を招くことになる。

【０１８３】

本実施形態のフォールトトレラントシステム１０では、デバイス（１１５、２１５）がストレージデバイスの場合、仮想マシン（１３０、２３０）が、デバイス（１１５、２１５）をメモリ上にエミュレーションして、デバイス（１１５、２１５）へのアクセスをメモリアクセスに置き換えることにより、デバイス（１１５、２１５）への読み書き要求に対する完了の待ち時間をゼロにすることができる。

【０１８４】

この場合、デバイス（１１５、２１５）への読み書き要求に対する完了割り込みは、実

際の完了割り込みを待つ必要はなく、要求時のメモリアクセスと同時に擬似割り込みをゲストＯＳ（２４０、２４０）に入力することによって、読み書き要求と完了擬似割り込みを共通の同期タイミングで処理することができる。これにより、完了割り込みによる擬似割り込みの同期処理が不要となり、処理負荷が軽減されるとともに、同期待ちによる実行速度の低下を防ぐことができる。

【０１８５】

メモリ上にエミュレーションしたデバイス（１１５、２１５）の内容の変更は、ゲストＯＳ（１４０、２４０）からのアクセスとは非同期にデバイス（１１５、２１５）に反映する。対応方法は、メモリを用意する場所によって処理が異なる。

【０１８６】

（１）仮想マシン（１３０、２３０）内のメモリを使用する場合

仮想マシン（１３０、２３０）は、ストレージデバイス（１１５、２１５）の内容を仮想マシン（１３０、２３０）に割り当てられたメモリ上に展開し、ストレージへのアクセスをメモリアクセスに置き換える。このメモリを「仮想ストレージ」と称する。仮想ストレージの内容が更新された場合は、ゲストＯＳ（１４０、２４０）からのアクセスとは非同期にストレージデバイス（１１５、２１５）に反映させる。

【０１８７】

具体的には、以下の手順によりデータの読み込み、書き込みを行なう。まず、仮想ストレージの初期化として、ストレージデバイス（１１５、２１５）の内容を仮想マシン（１３０、２３０）に割り当てられたメモリ上に展開しておく。

【０１８８】

データの読み込みの場合は、ゲストＯＳ（１４０、２４０）が仮想ストレージにデータ読み込みを要求すると、仮想マシン（１３０、２３０）が、仮想ストレージ上の内容をゲストＯＳ（１４０、２４０）に通知するとともに、読み込み完了の擬似割り込みをゲストＯＳ（１４０、２４０）に入力する。

【０１８９】

データの書き込みの場合は、ゲストＯＳ（１４０、２４０）が仮想ストレージにデータ書き込みを要求すると、仮想マシン（１３０、２３０）が、仮想ストレージにデータを書き込むとともに、書き込み完了の擬似割り込みをゲストＯＳ（１４０、２４０）に入力する。そして、仮想マシン（１３０、２３０）は、仮想ストレージの変更箇所を、非同期でストレージデバイス（１１５、２１５）に書き込む。ストレージデバイス（１１５、２１５）への書き込み完了後、完了の外部割り込みが仮想マシン（１３０、２３０）に入力されるが、この完了の外部割り込みに基づく擬似割り込みは行なう必要がない。

【０１９０】

（２）ゲストＯＳ（１４０、２３０）内のメモリを使用する場合

ゲストＯＳ（１４０、２３０）がＲＡＭディスクを利用できる場合は、ストレージデバイス（１１５、２１５）のエミュレーションにゲストＯＳ（１４０、２３０）で設定されたＲＡＭディスクを使用する。ＲＡＭディスクとは、読み書きに対する待ち時間がゼロであり、完了割り込みが発生しないデバイスである。例えば、Ｌｉｎｕｘ（登録商標）などのＯＳでＲＡＭディスクを利用できる。

【０１９１】

上述の仮想ストレージへの読み書きはゲストＯＳ（１４０、２３０）内のＲＡＭディスクに対して行なわれ、仮想マシン（１３０、２３０）に対して読み書きの要求は行なわれない。ゲストＯＳ（１４０、２３０）の内部で処理が完結しているため、読み書き要求や完了割り込みに対する同期処理が不要となる。

【０１９２】

具体的には、以下の手順によりＲＡＭディスクに対するデータの読み込み、書き込みを行なう。まず、ＲＡＭディスクの初期化として、ストレージデバイス（１１５、２１５）の内容をゲストＯＳ（１４０、２３０）のメモリに展開しておく。

【０１９３】

データの読み込みの場合は、ゲストＯＳ（１４０、２３０）がＲＡＭディスクにデータ読み込みを要求すると、ＲＡＭディスク上のストレージデバイス（１１５、２１５）の内容がゲストＯＳ（１４０、２３０）に通知される。読み込み完了の擬似割り込みは不要である。

【０１９４】

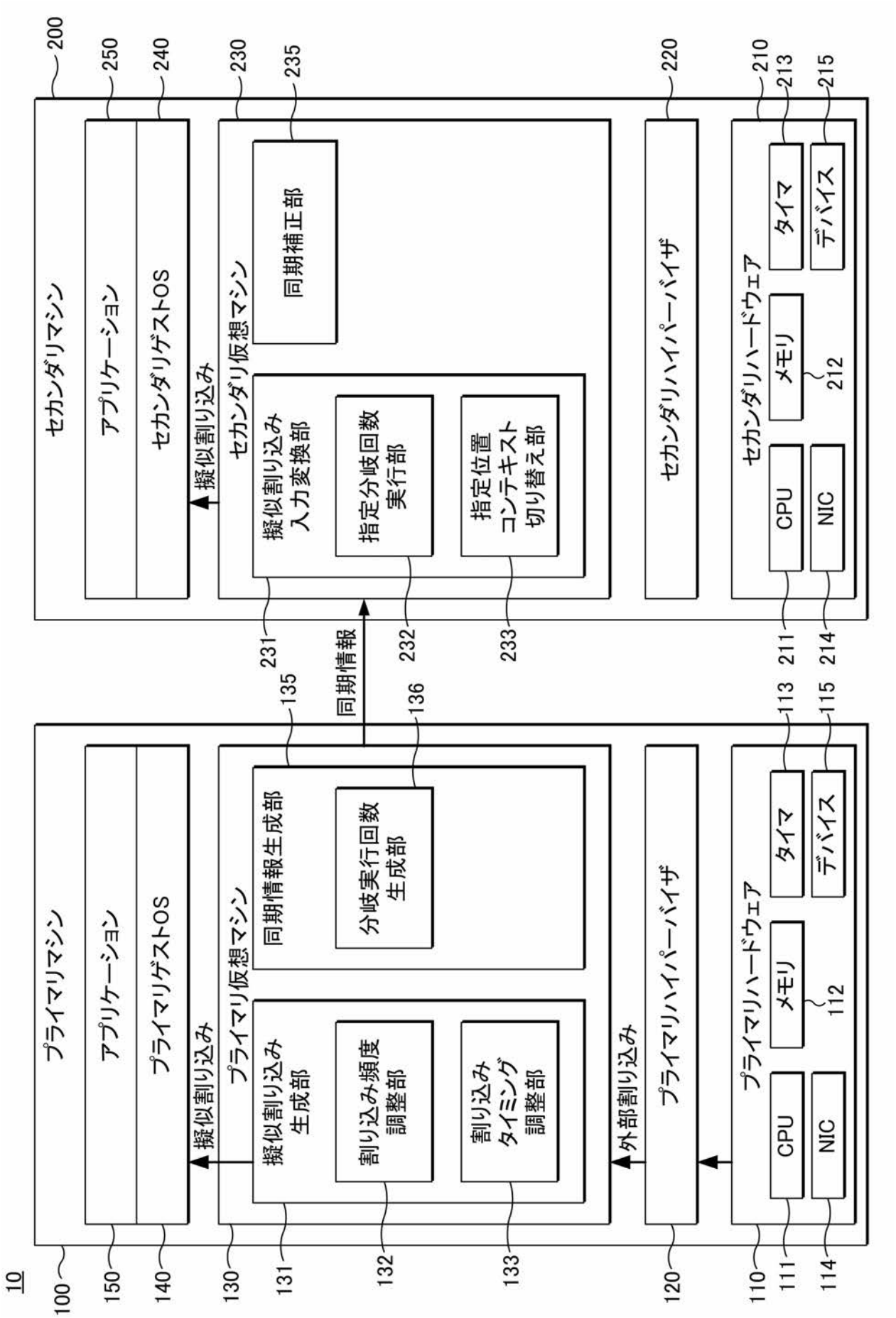
データの書き込みの場合は、ゲストＯＳ（１４０、２３０）がＲＡＭディスクにデータ書き込みを要求すると、ＲＡＭディスクにデータが書き込まれる。書き込み完了の擬似割り込みは不要である。そして、ＲＡＭディスクの内容を定期的にストレージデバイス（１１５、２１５）に保存するようにする。

【符号の説明】

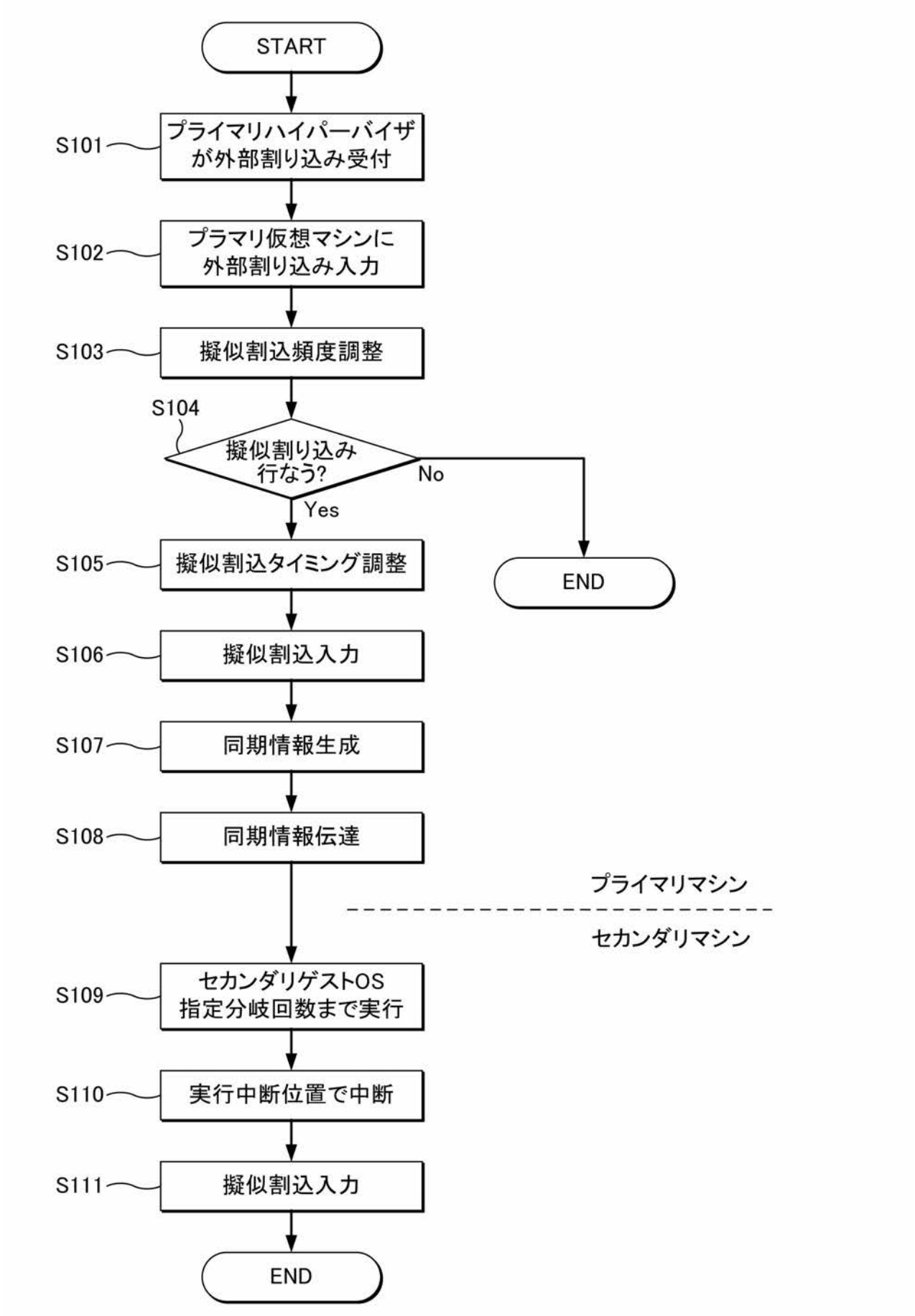
【０１９５】

１０…フォールトトレラントシステム、６０…フォールトトレラントシステム、１００…プライマリマシン、１１０…プライマリハードウェア、１１１…ＣＰＵ、１１２…メモリ、１１３…タイマ、１１４…ＮＩＣ、１１５…デバイス、１２０…プライマリハイパーバイザ、１３０…プライマリ仮想マシン、１３１…擬似割り込み生成部、１３２…割り込み頻度調整部、１３３…割り込みタイミング調整部、１３５…同期情報生成部、１３６…分岐実行回数生成部、１４０…プライマリゲストＯＳ、１５０…アプリケーション、２００…セカンダリマシン、２１０…セカンダリハードウェア、２１１…ＣＰＵ、２１２…メモリ、２１３…タイマ、２１４…ＮＩＣ、２１５…デバイス、２２０…セカンダリハイパーバイザ、２３０…セカンダリ仮想マシン、２３１…擬似割り込み入力変換部、２３２…指定分岐回数実行部、２３３…指定位置コンテキスト切り替え部、２３５…同期補正部、２４０…セカンダリゲストＯＳ、２５０…アプリケーション、６００…プライマリマシン、６１０…プライマリハードウェア、６２０…プライマリハイパーバイザ、６３０…プライマリ仮想マシン、６４０…プライマリゲストＯＳ、６５０…アプリケーション、７００…セカンダリマシン、７１０…セカンダリハードウェア、７２０…セカンダリハイパーバイザ、７３０…セカンダリ仮想マシン、７３１…出力データ照合部、７４０…セカンダリゲストＯＳ、７５０…アプリケーション

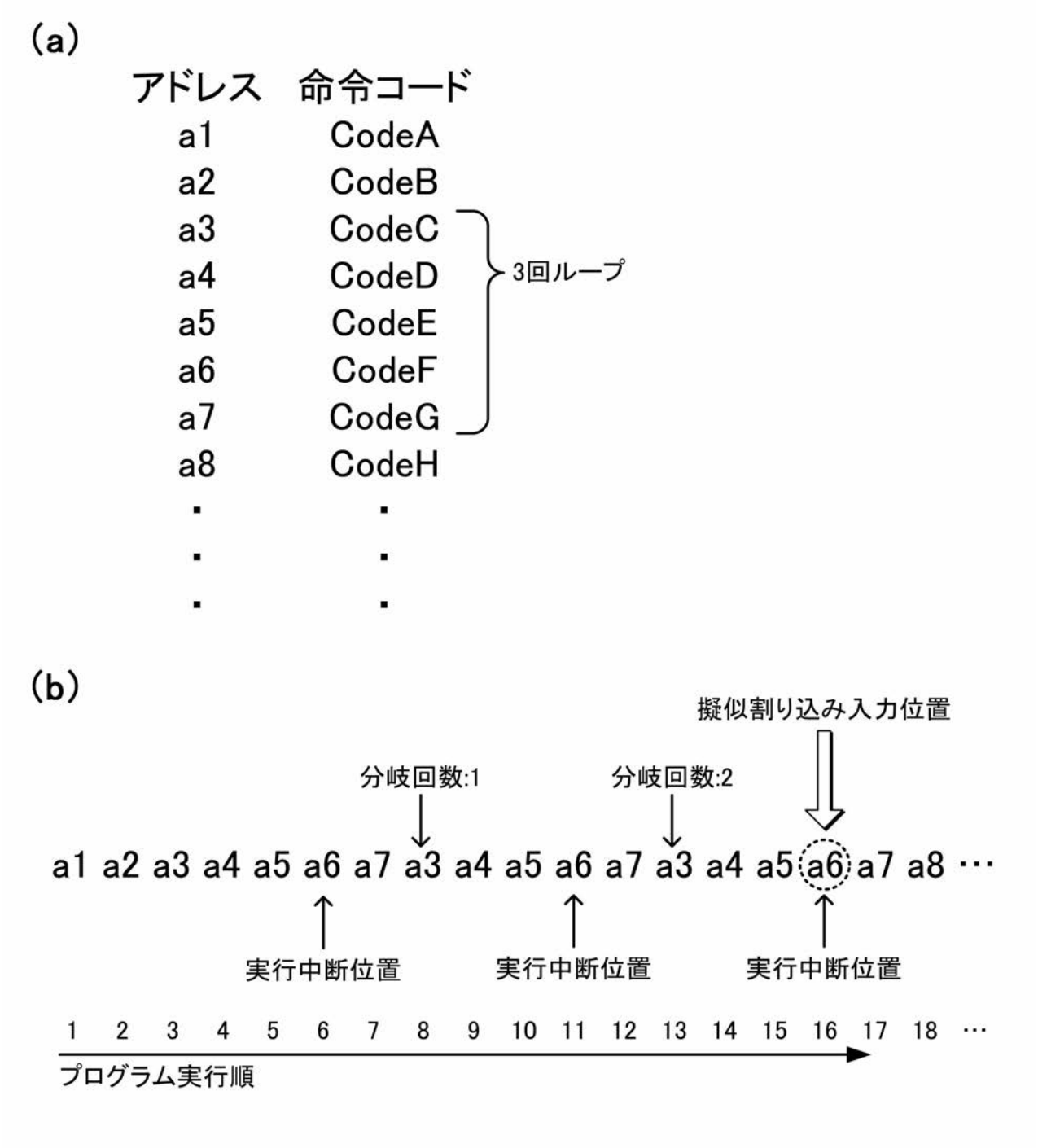
【図１】


10
プライマリマシン
100
アプリケーション
150
プライマリゲストOS
140
擬似割り込み
プライマリ仮想マシン
130
擬似割り込み
生成部
131
割り込み頻度
調整部
132
割り込み
タイミング
調整部
133
同期情報生成部
135
分岐実行回数
生成部
136
プライマリハイパーバイザ
120
外部割り込み
プライマリハードウェア
110
CPU
111
NIC
114
メモリ
112
タイマ
113
デバイス
115
同期情報
セカンダリマシン
200
アプリケーション
250
セカンダリゲストOS
240
擬似割り込み
セカンダリ仮想マシン
230
擬似割り込み
入力変換部
231
指定分岐回数
実行部
232
指定位置
コンテキスト
切り替え部
233
同期補正部
235
セカンダリハイパーバイザ
220
セカンダリハードウェア
210
CPU
211
NIC
214
メモリ
212
タイマ
213
デバイス
215

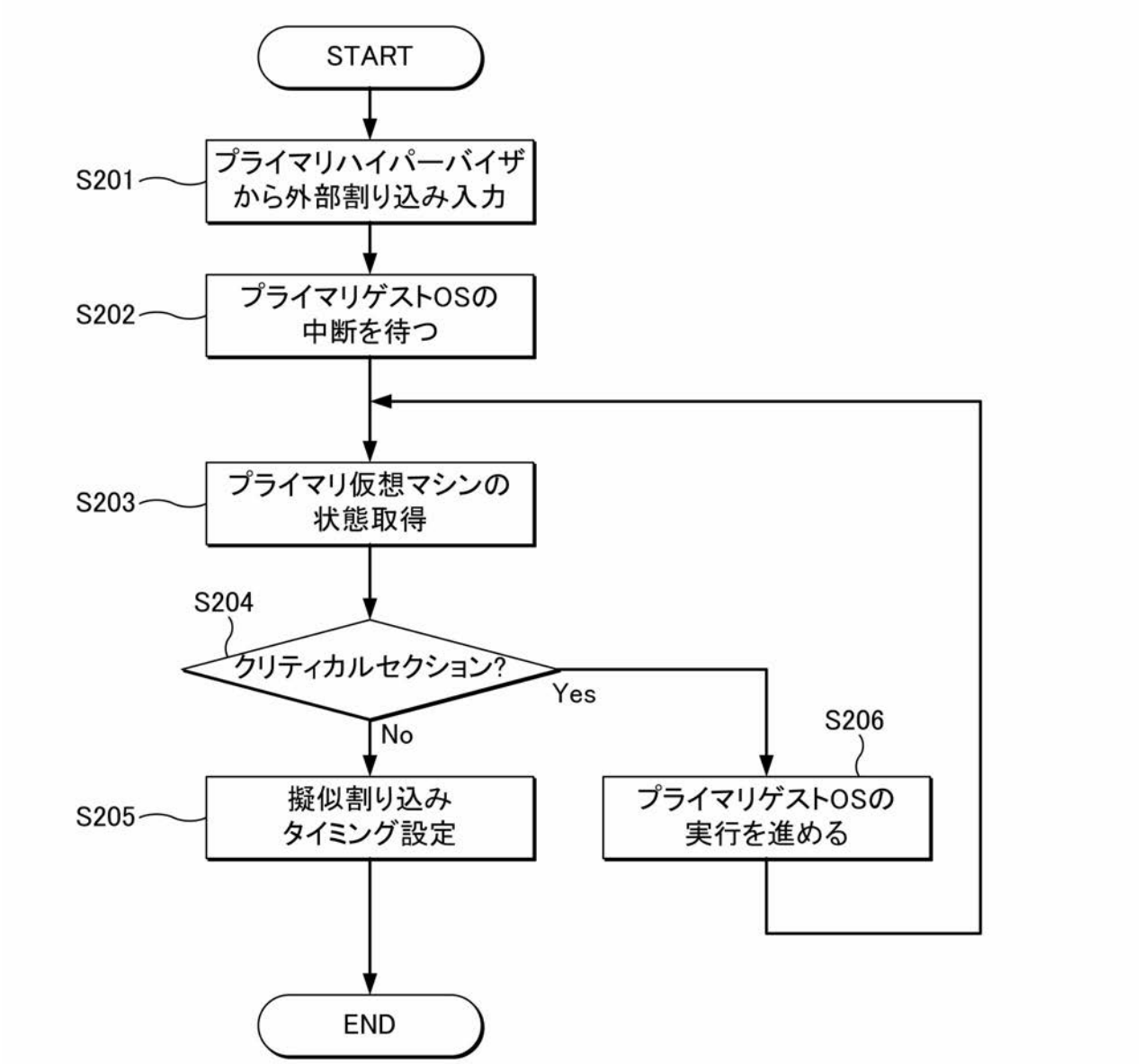
【図２】


START
S101
プライマリハイパーバイザが外部割り込み受付
S102
プラマリ仮想マシンに外部割り込み入力
S103
擬似割込頻度調整
S104
擬似割り込み行なう?
No
Yes
END
S105
擬似割込タイミング調整
S106
擬似割込入力
S107
同期情報生成
S108
同期情報伝達
プライマリマシン
セカンダリマシン
S109
セカンダリゲストOS指定分岐回数まで実行
S110
実行中断位置で中断
S111
擬似割込入力
END

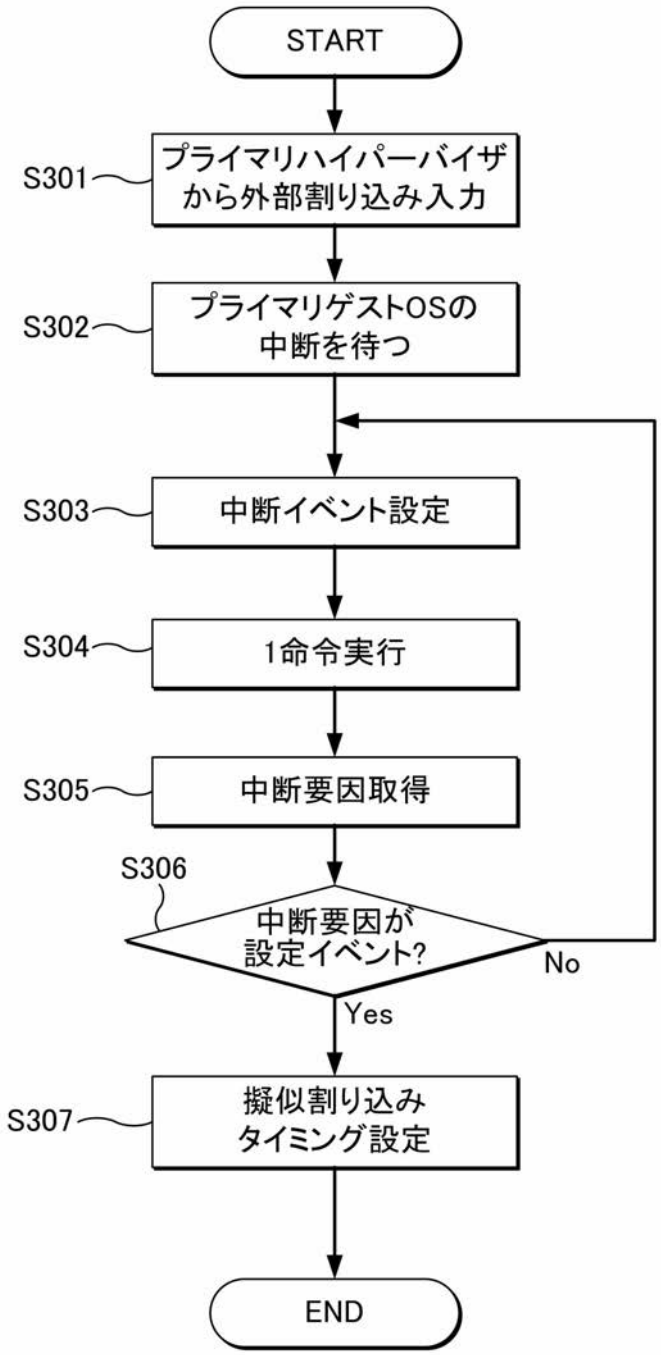
【図３】


(a)
アドレス 命令コード
a1 CodeA
a2 CodeB
a3 CodeC
a4 CodeD
a5 CodeE
a6 CodeF
a7 CodeG
a8 CodeH
3回ループ
(b)
擬似割り込み入力位置
分岐回数:1
分岐回数:2
a1 a2 a3 a4 a5 a6 a7 a3 a4 a5 a6 a7 a3 a4 a5 a6 a7 a8 …
実行中断位置
実行中断位置
実行中断位置
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 …
プログラム実行順

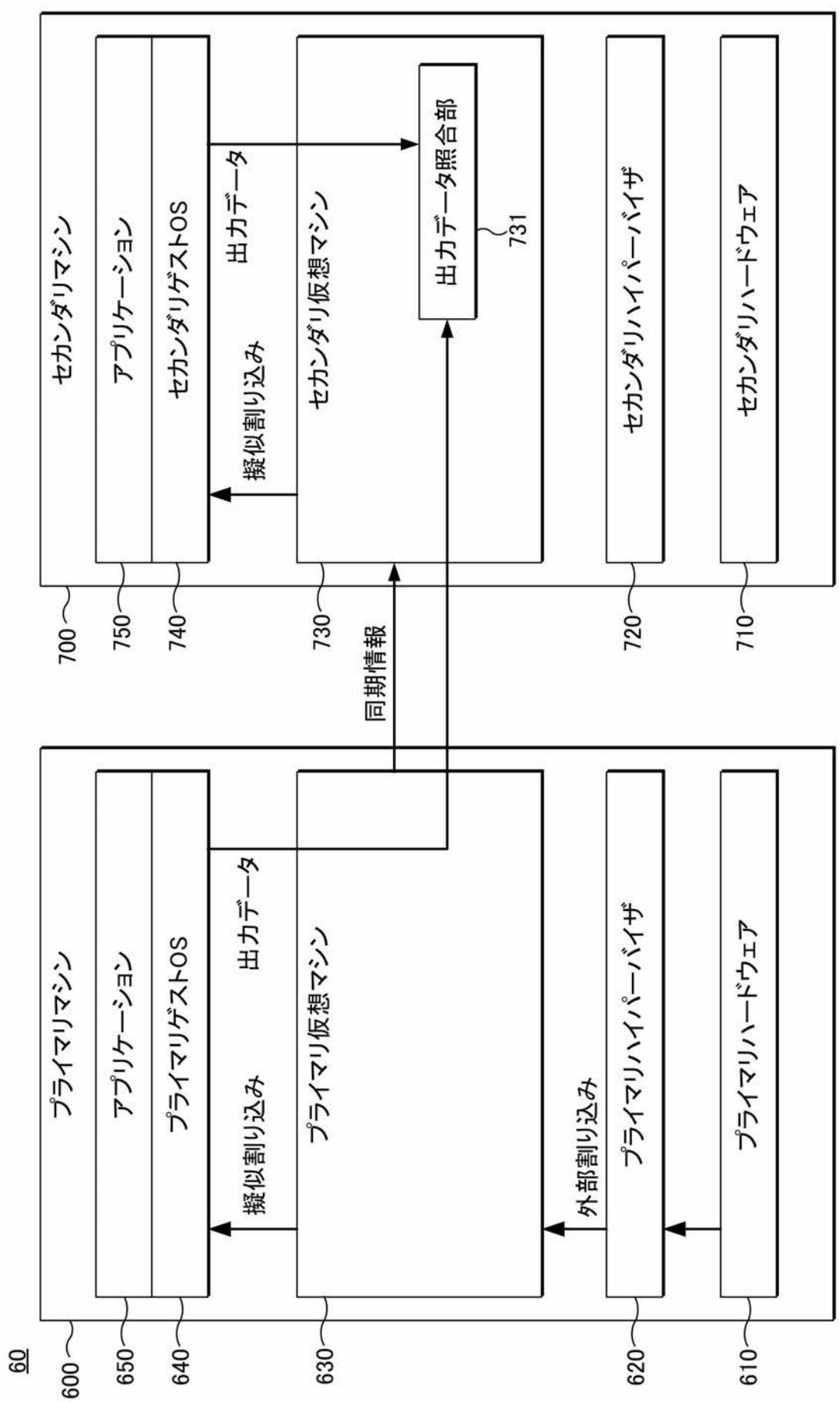
【図４】


START
S201
プライマリハイパーバイザから外部割り込み入力
S202
プライマリゲストOSの中断を待つ
S203
プライマリ仮想マシンの状態取得
S204
クリティカルセクション?
Yes
No
S205
擬似割り込みタイミング設定
S206
プライマリゲストOSの実行を進める
END

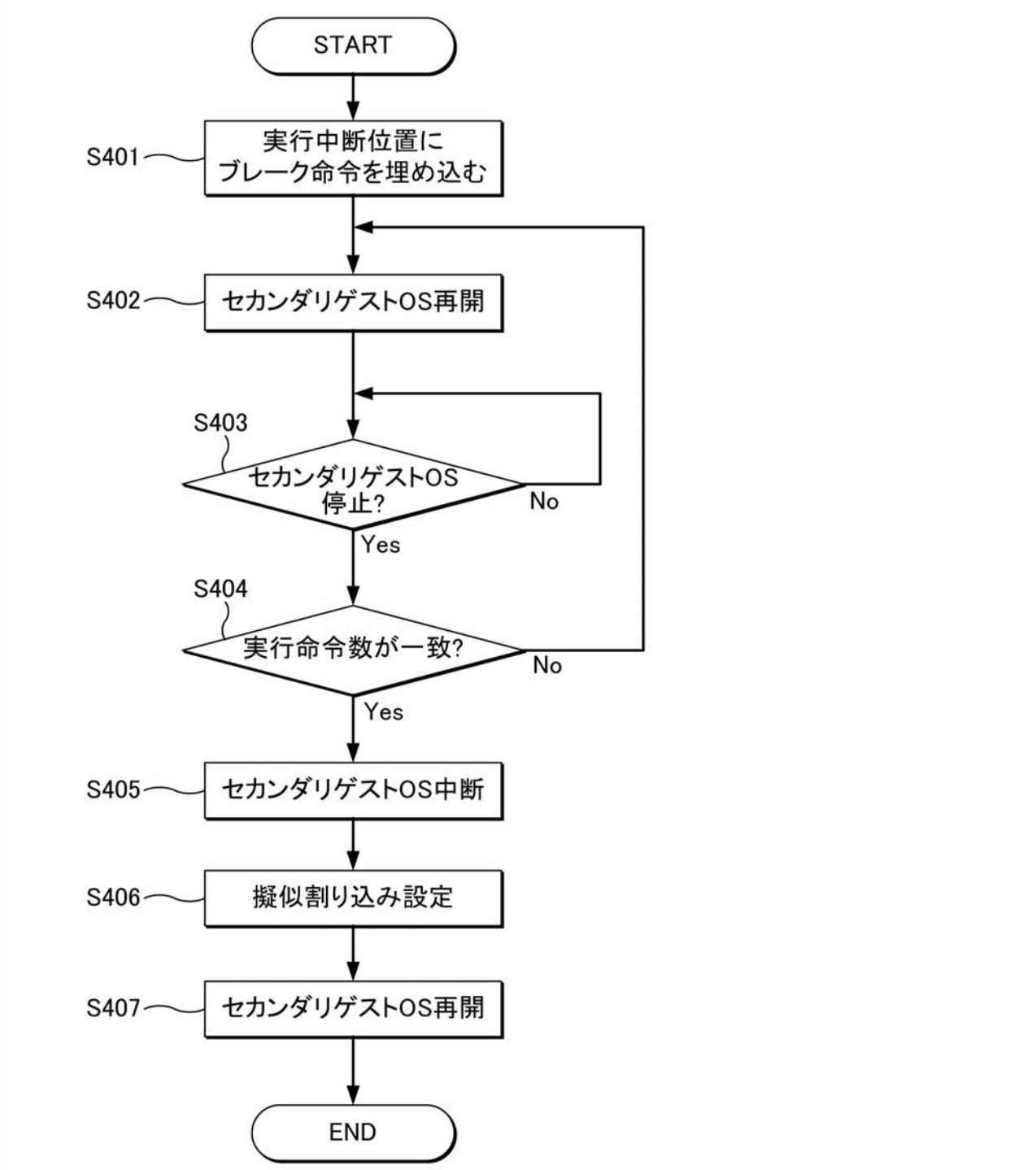
【図５】


START
S301
プライマリハイパーバイザから外部割り込み入力
S302
プライマリゲストOSの中断を待つ
S303
中断イベント設定
S304
1命令実行
S305
中断要因取得
S306
中断要因が設定イベント?
No
Yes
S307
擬似割り込みタイミング設定
END

【図６】


60
プライマリマシン
アプリケーション
プライマリゲストOS
プライマリ仮想マシン
プライマリハイパーバイザ
プライマリハードウェア
600
650
640
630
620
610
擬似割り込み
出力データ
外部割り込み
同期情報
セカンダリマシン
アプリケーション
セカンダリゲストOS
セカンダリ仮想マシン
出力データ照合部
731
セカンダリハイパーバイザ
セカンダリハードウェア
700
750
740
730
720
710
出力データ
擬似割り込み

【図７】


START
S401
実行中断位置に
ブレーク命令を埋め込む
S402
セカンダリゲストOS再開
S403
セカンダリゲストOS
停止?
No
Yes
S404
実行命令数が一致?
No
Yes
S405
セカンダリゲストOS中断
S406
擬似割り込み設定
S407
セカンダリゲストOS再開
END