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

特許第4982347号
(P4982347)

(45)発行日　平成24年7月25日(2012.7.25)　(24)登録日　平成24年4月27日(2012.4.27)

(51)Int.Cl.			F I		
G06F	9/46	(2006.01)	G06F	9/46	350
G06T	1/60	(2006.01)	G06T	1/60	450E
G09G	5/00	(2006.01)	G09G	5/00	550P

請求項の数 9　(全 27 頁)

(21)出願番号　特願2007-319351 (P2007-319351)
(22)出願日　平成19年12月11日 (2007.12.11)
(65)公開番号　特開2009-145932 (P2009-145932A)
(43)公開日　平成21年7月2日 (2009.7.2)
審査請求日　平成22年10月15日 (2010.10.15)

(73)特許権者　000003078
株式会社東芝
東京都港区芝浦一丁目１番１号
(74)代理人　100089118
弁理士　酒井　宏明
(72)発明者　峰松　美佳
東京都港区芝浦一丁目１番１号　株式会社東芝内
(72)発明者　後藤　真孝
東京都港区芝浦一丁目１番１号　株式会社東芝内
(72)発明者　西林　泰如
東京都港区芝浦一丁目１番１号　株式会社東芝内

最終頁に続く

(54)【発明の名称】画像情報の更新を検出するプログラム、方法および画像処理装置

(57)【特許請求の範囲】
【請求項１】
コンピュータに実行させる画像情報の更新部分を検出する更新検出プログラムであって、
前記コンピュータは、
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、を備え、
前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定手順と、
前記判定手順が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出手順と、
前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ手順と、
を前記コンピュータに実行させ、
前記検出手順は、前記判定手順が、前記画像情報の書き込み完了時に実行される予め定められた関数によって予め定められたレジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検

出し、
前記検出手順は、前記判定手順が、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって前記レジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する、
更新検出プログラム。
【請求項２】
コンピュータに実行させる画像情報の更新部分を検出する更新検出プログラムであって、
前記コンピュータは、
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、
前記コンピュータは、前記画像記憶部に対する前記画像情報の書き込み完了時に実行される予め定められた関数が実行されたときに例外を発生する例外発生部と、を備え、
前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定手順と、
前記判定手順が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出手順と、
前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ手順と、
を前記コンピュータに実行させ、
前記検出手順は、前記判定手順が、前記例外が発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、
前記例外発生部は、前記画像記憶部に対する前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数が実行されたときに例外を発生する、
更新検出プログラム。
【請求項３】
コンピュータに実行させる画像情報の更新部分を検出する更新検出プログラムであって、
前記コンピュータは、
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、を備え、
前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定手順と、
前記判定手順が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出手順と、
前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ手順と、
を前記コンピュータに実行させ、
前記検出手順は、前記判定手順が、前記画像情報の書き込み完了時に実行される予め定められた関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、
前記検出手順は、前記判定手順が、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって割り込みが発生したことが遷移の原因である

と判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する、
更新検出プログラム。
【請求項４】
画像処理装置で実行される画像情報の更新部分を検出する更新検出方法であって、
前記画像処理装置は、
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、を備え、
判定部が、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定ステップと、
検出部が、前記判定ステップが、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出ステップと、
エミュレータ部が、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータステップと、を含み、
前記検出ステップは、前記判定ステップが、前記画像情報の書き込み完了時に実行される予め定められた関数によって予め定められたレジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、
前記検出ステップは、前記判定ステップが、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって前記レジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する、
更新検出方法。
【請求項５】
画像処理装置で実行される画像情報の更新部分を検出する更新検出方法であって、
前記画像処理装置は、
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、
前記画像記憶部に対する前記画像情報の書き込み完了時に実行される予め定められた関数が実行されたときに例外を発生する例外発生部と、を備え、
判定部が、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定ステップと、
検出部が、前記判定ステップが、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出ステップと、
エミュレータ部が、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータステップと、を含み、
前記検出ステップは、前記判定ステップが、前記例外が発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、
前記例外発生部は、前記画像記憶部に対する前記画像情報の書き込み完了時に前記表示

用の記憶部に切り替えることを要求する関数が実行されたときに例外を発生する、
更新検出方法。
【請求項６】
画像処理装置で実行される画像情報の更新部分を検出する更新検出方法であって、
前記画像処理装置は、
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、を備え、
判定部が、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定ステップと、
検出部が、前記判定ステップが、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出ステップと、
エミュレータ部が、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータステップと、を含み、
前記検出ステップは、前記判定ステップが、前記画像情報の書き込み完了時に実行される予め定められた関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、
前記検出ステップは、前記判定ステップが、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する、
更新検出方法。
【請求項７】
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、
前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定部と、
前記判定部が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記画像情報の更新部分を検出する検出部と、
前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ部と、を備え、
前記検出部は、前記判定部が、前記画像情報の書き込み完了時に実行される予め定められた関数によって予め定められたレジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、
前記検出部は、前記判定部が、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって前記レジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する、
画像処理装置。
【請求項８】
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセ

ッサと、
前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定部と、
前記判定部が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記画像情報の更新部分を検出する検出部と、
前記画像記憶部に対する前記画像情報の書き込み完了時に実行される予め定められた関数が実行されたときに例外を発生する例外発生部と、
前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ部と、を備え、
前記検出部は、前記判定部が、前記例外が発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、
前記例外発生部は、前記画像記憶部に対する前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数が実行されたときに例外を発生する、
画像処理装置。
【請求項９】
画像情報を記憶可能な画像記憶部と、
予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、
前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定部と、
前記判定部が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記画像情報の更新部分を検出する検出部と、
前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ部と、を備え、
前記検出部は、前記判定部が、前記画像情報の書き込み完了時に実行される予め定められた関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、
前記検出部は、前記判定部が、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する、
画像処理装置。
【発明の詳細な説明】
【技術分野】
【０００１】
この発明は、仮想計算機上で動作するアプリケーション等で描画された画像情報の更新を検出するプログラム、方法および画像処理装置に関するものである。
【背景技術】
【０００２】
近年、１台のハードウェアで複数の仮想計算機を動作させる仮想計算機環境が実現されている。例えば、Ｘｅｎ（非特許文献１）、およびＶＭＷａｒｅ（非特許文献２）などが、仮想計算機の実行を制御するソフトウェアとして知られている。
【０００３】
仮想計算機環境では、仮想計算機モニタ（ＶＭＭ）上で複数のゲストＯＳが動作する。ＶＭＭは、複数のゲストＯＳのハードウェアリソースへのアクセスの調停などを行う。一般に、ＶＭＭは、ネットワークカードなどの共有リソースへのアクセスは制御するが、ゲストＯＳごとに割り当てられるグラフィックメモリなどへのアクセスには関与しない。したがって、画面を更新する際は、ゲストＯＳがグラフィックメモリに画像情報を描画し、ＶＭＭが画像情報に基づいて画面を更新する必要がある。ところが、上述のようにＶＭＭ

はゲストＯＳのグラフィックメモリへのアクセスに関与しないため、ゲストＯＳの描画イベントがＶＭＭに伝わらない。そこで、ＶＭＭは、ゲストＯＳの描画完了を検知して画面を更新する機能を備える必要がある。

【０００４】

ＶＭＭがゲストＯＳから通知を受けて画面更新を行う方法として、以下の２つが考えられる。

（１）ゲストＯＳからの明示的な通知は行なわず、ＶＭＭで画面更新を検知する。

（２）ゲストＯＳから明示的にＶＭＭに画面更新を通知する。

【０００５】

例えば、Ｘｅｎでは（１）の方法を用いている。すなわち、Ｘｅｎでは、ゲストＯＳは改変せずに、タイマーを用いてＶＭＭからゲストＯＳにポーリングをかけ、画面更新の有無を調べるという手法が用いられている。

【０００６】

一方、ＶＭＷａｒｅでは、（２）の方法を用いており、ゲストＯＳのビデオドライバを改変して、描画の区切りでゲストＯＳのビデオドライバからＶＭＭに対して画面更新通知を発行するという手法が用いられている。

【０００７】

【非特許文献１】ｈｔｔｐ：／／ｘｅｎ．ｏｒｇ／

【非特許文献２】ｈｔｔｐ：／／ｗｗｗ．ｖｍｗａｒｅ．ｃｏｍ／

【発明の開示】

【発明が解決しようとする課題】

【０００８】

しかしながら、上記（１）の方法では、グラフィックメモリに描画を終えたタイミングが分からず、描画途中の画像を表示する可能性がある。また、上記（２）の方法では、ゲストＯＳのビデオドライバを仮想計算機環境に合わせて改変する必要があり、汎用性がない。

【０００９】

本発明は、上記に鑑みてなされたものであって、ゲストＯＳを改変することなく適切なタイミングで画像情報の更新を検出することができるプログラム、方法および画像処理装置を提供することを目的とする。

【課題を解決するための手段】

【００１０】

上述した課題を解決し、目的を達成するために、本発明は、コンピュータに実行させる画像情報の更新部分を検出する更新検出プログラムであって、前記コンピュータは、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、を備え、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定手順と、前記判定手順が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出手順と、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ手順と、を前記コンピュータに実行させ、前記検出手順は、前記判定手順が、前記画像情報の書き込み完了時に実行される予め定められた関数によって予め定められたレジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記検出手順は、前記判定手順が、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって前記レジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する、更新検出プログラムである。また。本発明は、コンピュータに実行させる画像情報の更新部分を検出する更新検

出プログラムであって、前記コンピュータは、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、前記コンピュータは、前記画像記憶部に対する前記画像情報の書き込み完了時に実行される予め定められた関数が実行されたときに例外を発生する例外発生部と、を備え、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定手順と、前記判定手順が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出手順と、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ手順と、を前記コンピュータに実行させ、前記検出手順は、前記判定手順が、前記例外が発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記例外発生部は、前記画像記憶部に対する前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数が実行されたときに例外を発生する、更新検出プログラムである。また。本発明は、コンピュータに実行させる画像情報の更新部分を検出する更新検出プログラムであって、前記コンピュータは、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、を備え、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定手順と、前記判定手順が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出手順と、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ手順と、を前記コンピュータに実行させ、前記検出手順は、前記判定手順が、前記画像情報の書き込み完了時に実行される予め定められた関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記検出手順は、前記判定手順が、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する、更新検出プログラムである。

【００１１】

また、本発明は、画像処理装置で実行される画像情報の更新部分を検出する更新検出方法であって、前記画像処理装置は、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、を備え、判定部が、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定ステップと、検出部が、前記判定ステップが、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出ステップと、エミュレータ部が、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータステップと、を含み、前記検出ステップは、前記判定ステップが、前記画像情報の書き込み完了時に実行される予め定められた関数によって予め定められたレジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記検出ステップは、前記判定ステップが、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって前記レジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する。また、本発明は、

画像処理装置で実行される画像情報の更新部分を検出する更新検出方法であって、前記画像処理装置は、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、前記画像記憶部に対する前記画像情報の書き込み完了時に実行される予め定められた関数が実行されたときに例外を発生する例外発生部と、を備え、判定部が、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定ステップと、検出部が、前記判定ステップが、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出ステップと、エミュレータ部が、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータステップと、を含み、前記検出ステップは、前記判定ステップが、前記例外が発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記例外発生部は、前記画像記憶部に対する前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数が実行されたときに例外を発生する。また、本発明は、画像処理装置で実行される画像情報の更新部分を検出する更新検出方法であって、前記画像処理装置は、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、を備え、判定部が、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定ステップと、検出部が、前記判定ステップが、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する検出ステップと、エミュレータ部が、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータステップと、を含み、前記検出ステップは、前記判定ステップが、前記画像情報の書き込み完了時に実行される予め定められた関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記検出ステップは、前記判定ステップが、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する。

【００１２】

また、本発明は、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定部と、前記判定部が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記画像情報の更新部分を検出する検出部と、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ部と、を備え、前記検出部は、前記判定部が、前記画像情報の書き込み完了時に実行される予め定められた関数によって予め定められたレジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記検出部は、前記判定部が、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって前記レジスタにアクセスされたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する。また、本発明は、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる

第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定部と、前記判定部が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記画像情報の更新部分を検出する検出部と、前記画像記憶部に対する前記画像情報の書き込み完了時に実行される予め定められた関数が実行されたときに例外を発生する例外発生部と、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ部と、を備え、前記検出部は、前記判定部が、前記例外が発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記例外発生部は、前記画像記憶部に対する前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数が実行されたときに例外を発生する。また、本発明は、画像情報を記憶可能な画像記憶部と、予め定められた遷移条件が満たされたときに、ゲストＯＳを動作させる第１モードから、前記ゲストＯＳを管理する仮想計算機モニタを動作させる第２モードへ遷移するプロセッサと、前記プロセッサが前記第２モードに遷移したときに、遷移の原因を判定する判定部と、前記判定部が、前記ゲストＯＳ上で前記画像記憶部に対する前記画像情報の書き込み完了に関連する処理が実行されたことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記画像情報の更新部分を検出する検出部と、前記画像情報の書き込み用の記憶部と前記画像情報の表示用の記憶部とを切り替えるマルチバッファリング機能をエミュレートするエミュレータ部と、を備え、前記検出部は、前記判定部が、前記画像情報の書き込み完了時に実行される予め定められた関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出し、前記検出部は、前記判定部が、前記画像情報の書き込み完了時に前記表示用の記憶部に切り替えることを要求する関数によって割り込みが発生したことが遷移の原因であると判定した場合に、書込み前後で前記画像情報が一致しない部分を表す前記更新部分を検出する。

【発明の効果】

【００１３】

本発明によれば、ゲストＯＳを改変することなく適切なタイミングで画像情報の更新を検出することができるという効果を奏する。

【発明を実施するための最良の形態】

【００１４】

以下に添付図面を参照して、この発明にかかるプログラム、方法および画像処理装置の最良な実施の形態を詳細に説明する。

【００１５】

（第１の実施の形態）

第１の実施の形態にかかる画像処理装置は、ハードウェアレベルで仮想化を支援するプロセッサと、描画が完了したときに所定のレジスタにアクセスする関数を実行するビデオドライバを備える。そして、所定のレジスタにアクセスしたときにＶＭＭに制御が移るようにプロセッサを設定することにより、描画完了をＶＭＭで検知可能とし、描画完了の検知を契機として画像情報の更新部分の検出処理を実行する。

【００１６】

なお、これまでのＶＭＭは、ソフトウェアによって仮想化の処理を行ってきた。一方、ハードウェアであるプロセッサやチップセットが仮想化を支援することにより、仮想化の負荷を軽減できると考えられる。そこで、ＩｎｔｅｌのＶＴ（Virtualization Technology）やＡＭＤのＳＶＭ（Secure Virtual Machine）などの仮想化支援機能を備えたプロセッサが提供され始めている。本実施の形態では、このような、ハードウェアレベルの仮想化支援機能を実装したプロセッサの利用を想定している。

【００１７】

ここで、ハードウェアレベルの仮想化支援機能を備えるプロセッサの概要について説明する。

【００１８】

このようなプロセッサでは、１つのプロセッサで複数のＯＳを同時に稼動するための新たな動作モードとして、「ゲストモード」が用意されている。ゲストＯＳは、この「ゲストモード」で動作する。また、このようなプロセッサでは、仮想化専用命令セットが用意されている。例えば、「ゲストモード」に入る際にはＶＭ　Ｅｎｔｒｙ命令が実行される。また、「ゲストモード」から抜ける際にはＶＭ　Ｅｘｉｔ命令が実行される。なお、「ゲストモード」を抜けると、ＶＭＭが動作するモード（以下、ホストモードという）に遷移する。

【００１９】

ＶＭ　Ｅｎｔｒｙは、ゲストＯＳを起動するときなどに実行される。一方、ＶＭ　Ｅｘｉｔは、ゲストＯＳでの特権命令発行時やＩ／Ｏからの割り込み発生時など、ゲストＯＳからＶＭＭに処理を戻す条件が発生したときに実行される。

【００２０】

各ゲストＯＳを管理するために、Ｉｎｔｅｌ　ＶＴの場合はＶＭＣＳ（Virtual Machine Control Structure）、ＡＭＤ　ＳＶＭの場合はＶＭＣＢ（Virtual Machine Control Block）と呼ばれるゲストＯＳ管理構造体が利用される。ゲストＯＳ管理構造体には、例えば、ゲストＯＳからＶＭＭに処理を戻す条件が記述される。

【００２１】

次に、本実施の形態で利用するビデオドライバの概要について説明する。本実施の形態では、ゲストＯＳのビデオドライバとして、ＤｉｒｅｃｔＦＢに対応したビデオドライバを利用する。ＤｉｒｅｃｔＦＢのビデオドライバは、グラフィックカードがマルチバッファリング（後述）をサポートしている場合に、描画完了時にバッファを切り替えるためのＦｌｉｐＲｅｇｉｏｎ関数を備えている。

【００２２】

すなわち、ＦｌｉｐＲｅｇｉｏｎ関数は、グラフィックカードがマルチバッファリングをサポートしている場合に発行される。仮想計算機環境の場合は、グラフィックカードをエミュレートするエミュレータ部１４１（後述）がマルチバッファリングをサポートしている場合に、ＦｌｉｐＲｅｇｉｏｎ関数が実行される。また、ＦｌｉｐＲｅｇｉｏｎ関数が実行されると、表示用バッファ（後述）のアドレス（表示開始アドレス）を指定するレジスタがアクセスされる。

【００２３】

本実施の形態では、このレジスタへのアクセスを条件としてプロセッサがＶＭＭに処理を移すようにゲストＯＳ管理構造体を設定することにより、描画完了を契機として画像情報の更新部分の検出処理を実行するように構成する。これにより、描画が完了したタイミングで更新部分を検出して最新画像を出力できるため、描画途中の画像を出力することを防止できる。また、ＤｉｒｅｃｔＦＢのビデオドライバ自体の改変は不要であるため、汎用性を高めることができる。

【００２４】

ここで、マルチバッファリングの詳細について説明する。マルチバッファリングとは、フレームバッファを１つのみ利用するシングルバッファリングと異なり、複数のバッファ（表示用バッファおよび描画用バッファ）を利用して画像情報を描画する方法である。なお、表示開始アドレスとは、表示用バッファの先頭アドレスを意味する。

【００２５】

シングルバッファリングでは、描画用バッファと表示用バッファとを区別していないため、描画の過程が表示画面に現れて画面がちらつく場合がある。図１は、シングルバッファリングでの表示画面の一例を示す図である。図１に示すように、シングルバッファリングでは描画する過程が画面上に見える場合がある。

【００２６】

そこで、画面のちらつきを防ぐために、２つのバッファを利用するダブルバッファリングや、３つのバッファを利用するトリプルバッファリングなどのマルチバッファリングという手法が用いられる。

【００２７】

マルチバッファリングでは、画像情報を表示用バッファに直接描画せず、描画用バッファに対してすべての画像情報の描画が完了してから、表示用バッファに転送する。なお、マルチバッファリングには、表示用バッファを固定し、描画が完了した描画用バッファから表示用バッファにコピーを行う方法や、描画完了時に表示用バッファと描画用バッファのポインタを切り替える方法などが存在する。また、トリプルバッファリングの場合は、３つのバッファのうち、１つを表示用バッファ、２つを描画用バッファとして用いる。

【００２８】

図２は、ポインタを切り替える方法によるダブルバッファリングの概要を示す図である。図２に示すように、表示用バッファ２１は、描画用バッファ２２に対する描画が完了するまで画像情報は描画されない。描画用バッファ２２への描画が完了すると、ポインタを切り替えることにより、描画用バッファ２２を新たな表示用バッファ２４とし、表示用バッファ２１を新たな描画用バッファ２３とする。これにより、常に描画が完了した画像情報を画面に表示することができる。

【００２９】

次に、画像処理装置の構成について説明する。図３は、第１の実施の形態にかかる画像処理装置１００のブロック図である。図３に示すように、画像処理装置１００は、ハードウェア１１０と、ハードウェア１１０を利用して機能するソフトウェア構成として、ＶＭＭ１２０と、ゲストＯＳ１３０と、管理ＯＳ１４０と、アプリケーション１５０とを備えている。なお、同図ではゲストＯＳ１３０を１つのみ記載しているが、ＶＭＭ１２０上では複数のゲストＯＳ１３０を実行することができる。この場合、後述の、フレームバッファ領域、表示開始アドレスレジスタ領域、バックアップ記憶部、エミュレータ部、検出部、および条件記憶部に記憶されるゲストＯＳ管理構造体は、それぞれ、ゲストＯＳの台数分用意される。

【００３０】

最初に、同図を用いて仮想化技術の概要について説明する。上述のように、仮想計算機環境では、ＶＭＭ１２０上で複数のゲストＯＳ１３０が動作する。ＶＭＭ１２０は、複数のゲストＯＳ１３０からの共有ハードウェアリソースへのアクセスの調停などを行う。

【００３１】

ゲストＯＳ１３０は、仮想的な計算機環境を動作させるＯＳである。管理ＯＳ１４０は、ゲストＯＳ１３０を管理する権限を持っており、ゲストＯＳ１３０の起動や、ゲストＯＳ１３０の代わりに共有ハードウェアリソースにアクセスする処理などを行う。

【００３２】

アプリケーション１５０は、ゲストＯＳ１３０上で動作する各種処理を提供するプログラムである。１つのゲストＯＳ１３０と、このゲストＯＳ１３０上で動作するアプリケーション１５０とによって、１つの仮想計算機環境が実現される。

【００３３】

次に、各構成部の詳細について説明する。

【００３４】

ハードウェア１１０は、プロセッサ１１１と、条件記憶部１１２と、バックアップ記憶部１１３と、グラフィックカード１１４と、フレームバッファ領域と表示開始アドレスレジスタ領域とを備えている。なお、同図では省略しているが、ハードウェア１１０は、ＲＯＭおよび通信Ｉ／Ｆなどのハードウェアリソースを備えてもよい。

【００３５】

プロセッサ１１１は、ＩｎｔｅｌのＶＴやＡＭＤのＳＶＭなどのように、ハードウェアレベルの仮想化支援機能を備えるプロセッサである。条件記憶部１１２は、ゲストＯＳ１３０からＶＭＭ１２０に処理を戻す条件を含むゲストＯＳ管理構造体を記憶する。

【００３６】
ここで、条件記憶部１１２に記憶されるゲストＯＳ管理構造体の具体的な設定方法について説明する。第１の実施の形態では、以下のようにゲストＯＳ管理構造体を設定することにより、ＭＭＩＯ（Memory Mapped I/O）を介したレジスタアクセスを検知する。
【００３７】
まず、Ｉｎｔｅｌ　ＶＴのＶＭＣＳの設定例について説明する。この場合、ＶＭＣＳのＶＭ　Ｅｘｉｔ制御領域の「Primary Processor-Based VM-Execution Controls」、「I/O-Bitmap」、およびＶＭ　Ｅｘｉｔ情報領域の「Exit Reason」、「Exit Qualification」を用いる。
【００３８】
具体的には、まず、「Primary Processor-Based VM-Execution Controls」の２５ビット目の「Use I/Obitmaps」を１に設定する。また、「I/O-Bitmap」のうち、アクセスを検知すべきレジスタに対応するＩ／Ｏポート番号に対応するビットを１に設定する。
【００３９】
これにより、検知すべきレジスタがアクセスされるとＶＭ　Ｅｘｉｔが発生する。そして、この例では、ＶＭ　Ｅｘｉｔ情報領域の「Exit Reason」の０－１５ビット（ＶＭ　Ｅｘｉｔの原因）が３０となり、また、「Exit Qualification」の１６－３１ビット（ポート番号）が対応するポート番号になる。これらの情報を参照することにより、ＶＭＭ１２０は、検知すべきレジスタへのアクセスが原因でＶＭ　Ｅｘｉｔが発生したことを判定することができる。
【００４０】
次に、ＡＭＤ　ＳＶＭのＶＭＣＢの設定例について説明する。この場合、ＶＭＣＢの制御エリアの「I/O Permission Map」、「Exception Bitmap」、「EXITCODE」、および「EXITINFO1」を用いる。
【００４１】
具体的には、まず、ＶＭＣＢのバイトオフセット「040h」（「h」は１６進数表記であることを表す）の「I/O Permission Map」のうち、アクセスを検出すべきレジスタに対応するＩ／Ｏポート番号に対応するビットを１に設定する。
【００４２】
これにより、検知すべきレジスタがアクセスされるとＶＭ　Ｅｘｉｔが発生する。そして、この例では、「EXITCODE」が「7Bh」となり、また、「EXITINFO1」の１６－３１ビットが対応するポート番号になる。これらの情報を参照することにより、ＶＭＭ１２０は、検知すべきレジスタへのアクセスが原因でＶＭ　Ｅｘｉｔが発生したことを判定することができる。
【００４３】
バックアップ記憶部１１３は、フレームバッファ１１４ａ（後述）に記憶されている画像情報全体のバックアップデータを記憶可能な記憶部である。
【００４４】
グラフィックカード１１４は、画像描画に関する機能を備えた拡張カードであり、フレームバッファ１１４ａと、表示開始アドレスレジスタ１１４ｂとを備えている。
【００４５】
フレームバッファ１１４ａは、ゲストＯＳ１３０の画面の１画面分の画像情報を記憶する記憶部である。表示開始アドレスレジスタ１１４ｂは、表示開始アドレスを記憶するレジスタである。
【００４６】
なお、条件記憶部１１２、バックアップ記憶部１１３、フレームバッファ領域、および表示開始アドレスレジスタ領域は、ＲＡＭ（Random Access Memory）、ＨＤＤ（Hard Disk Drive）、光ディスク、メモリカードなどの一般的に利用されているあらゆる記憶媒体により構成することができる。フレームバッファ領域１１５ａは、グラフィックカード１１４のフレームバッファ１１４ａに対応する領域として、ゲストＯＳ１３０のそれぞれに

ついて確保される領域である。表示開始アドレスレジスタ領域１１５ｂは、グラフィックカード１１４の表示開始アドレスレジスタ１１４ｂに対応する領域として、ゲストＯＳ１３０のそれぞれについて確保される領域である。

【００４７】

ＶＭＭ１２０は、条件設定部１２１と、判定部１２２とを備えている。

【００４８】

条件設定部１２１は、画像情報の描画が完了したときにＶＭＭ１２０に処理が移るようにゲストＯＳ管理構造体に条件を事前に設定する。上述のように、ＤｉｒｅｃｔＦＢのビデオドライバでは、描画完了時にＦｌｉｐＲｅｇｉｏｎ関数が呼ばれると表示開始アドレスレジスタ１１４ｂがアクセスされる。そこで、条件設定部１２１は、表示開始アドレスレジスタ１１４ｂがアクセスされたときにＶＭ　Ｅｘｉｔが発生するように、ゲストＯＳ管理構造体を設定する。これにより、画面更新完了の際にＶＭＭ１２０に処理を移すことができる。

【００４９】

判定部１２２は、ＶＭ　ＥｘｉｔによりＶＭＭ１２０に処理が移ったときに、処理が移った原因を判定する。例えば、プロセッサ１１１がＩｎｔｅｌのＶＴに対応する場合、判定部１２２は、ゲストＯＳ管理構造体（ＶＭＣＳ）のＶＭ　Ｅｘｉｔ情報領域の「Exit Reason」を参照して原因を判定する。また、プロセッサ１１１がＡＭＤのＳＶＭに対応する場合、判定部１２２は、ゲストＯＳ管理構造体（ＶＭＣＢ）の制御エリアの「EXITCODE」および「EXITINFO1」を参照して原因を判定する。

【００５０】

なお、ＶＭＭ１２０は、判定した原因に応じて各種処理を実行する。例えば、原因が表示開始アドレスレジスタ１１４ｂへのアクセスと判定された場合は、エミュレータ部１４１のイベント処理部１４１ａ（後述）に、表示開始アドレスレジスタ１１４ｂがアクセスされたことを通知する。

【００５１】

ゲストＯＳ１３０は、画像表示に関する機能として、グラフィックライブラリ１３１と、ビデオドライバ１３２とを備えている。なお、ゲストＯＳ１３０は、画像表示に関する機能以外にも、仮想的な計算機環境を動作させるために必要なあらゆる機能を備えている。

【００５２】

グラフィックライブラリ１３１は、描画部１３１ａを備えている。

【００５３】

描画部１３１ａは、アプリケーション１５０から指定された描画命令に従い、各種画像処理を行ってその結果である画像情報を出力する。例えば、描画部１３１ａは、ある領域を拡大・縮小する描画命令を受けた場合、描画命令に含まれる座標情報などから拡大・縮小する領域を特定し、当該領域に対して拡大・縮小を行った結果である領域の座標情報などを含む画像情報を出力する。なお、座標情報とは、例えば、１０２４×７６８ピクセルの画面の場合、画面左上を（０、０）、画面右下を（１０２３、７６７）とするような座標系で表された情報をいう。また、描画部１３１ａは、画像情報の座標情報をフレームバッファ１１４ａの仮想アドレスに変換して、書き込み部１３２ａ（後述）に渡す。また、描画部１３１ａは、画像情報の描画が完了したときに、書き込み部１３２ａ（後述）に描画の完了を通知する。

【００５４】

ビデオドライバ１３２は、上述のようにＤｉｒｅｃｔＦＢに対応したドライバであり、書き込み部１３２ａを備えている。なお、適用可能なドライバはＤｉｒｅｃｔＦＢ対応のドライバに限られず、描画完了時に特定の関数や命令を実行し、特定のレジスタにアクセスするように構成されているドライバであれば、あらゆる種類のドライバを利用できる。

【００５５】

書き込み部１３２ａは、描画部１３１ａにより指定された仮想アドレスに対応する物理

メモリ領域に、書き込みが要求された画像情報を書き込む。

【００５６】

また、書き込み部１３２ａは、描画部１３１ａからの通知により画像情報の描画が完了したことを検知したときにＦｌｉｐＲｅｇｉｏｎ関数を実行する。なお、書き込み部１３２ａは、上述のようにエミュレータ部１４１がマルチバッファリングをサポートしている場合に、描画が完了したときにＦｌｉｐＲｅｇｉｏｎ関数を実行する。したがって、書き込み部１３２ａは、マルチバッファリングをサポートしていることをエミュレータ部１４１から事前に通知されている必要がある。

【００５７】

管理ＯＳ１４０は、詳細な構成として、エミュレータ部１４１と、検出部１４２とを備えている。

【００５８】

エミュレータ部１４１は、ゲストＯＳ１３０に対して、マルチバッファリングをサポートしているグラフィックカードのエミュレーションを提供する。すなわち、エミュレータ部１４１は、マルチバッファリングをサポートしていることをゲストＯＳ１３０に通知する。これにより、ゲストＯＳ１３０のビデオドライバ１３２内の書き込み部１３２ａは、描画完了時にＦｌｉｐＲｅｇｉｏｎ関数を発行するようになる。

【００５９】

なお、エミュレータ部１４１は、マルチバッファリングのエミュレーションを行う場合、実際に複数のバッファを利用して通常のマルチバッファリング処理を行っても良いし、実際には１枚のバッファのみを利用してマルチバッファリングをエミュレーションしても良い。いずれの場合も、書き込み部１３２ａは、（１）画像情報の描画、（２）更新通知（ＦｌｉｐＲｅｇｉｏｎ関数）、（３）バッファ切り替えが完了するまで待つ、（４）次の画像情報の描画、という処理を順次実行する。１枚のバッファのみを利用する場合は、表示用バッファの更新画像を生成するまで次の画像情報の描画を待たせる必要があるため、エミュレータ部１４１は、更新画像の生成が完了したときにバッファ切り替えが完了したように見せかけるエミュレーションを行う。

【００６０】

エミュレータ部１４１は、さらに詳細な構成として、イベント処理部１４１ａを備えている。

【００６１】

イベント処理部１４１ａは、表示開始アドレスレジスタ１１４ｂがアクセスされたことをＶＭＭ１２０から通知された場合に、画面更新のイベントが発生したと判断し、更新部分の検出処理を開始するために検出部１４２にイベント発生を通知する。

【００６２】

検出部１４２は、画面が更新された場合に、バックアップ記憶部１１３の画像情報と、フレームバッファ１１４ａの最新の画像情報とを比較し、相違点を含む矩形を更新部分として検出する。

【００６３】

次に、このように構成された第１の実施の形態にかかる画像処理装置１００による画像処理について説明する。画像処理は、画像情報をフレームバッファ１１４ａおよびバックアップ記憶部１１３に書き込む画像書き込み処理と、バックアップした画像情報と最新の画像情報とを比較して画像情報の更新部分を検出する更新部分検出処理に分けられる。

【００６４】

最初に、画像書き込み処理および更新部分検出処理の概要について図４を用いて説明する。

【００６５】

画像書き込み処理では、まず、予めフレームバッファ１１４ａの全体のバックアップをバックアップ記憶部１１３に保存する。フレームバッファ１１４ａのピクセルの値によって初期状態か否かを判断できる場合は、この時点でのバックアップの保持を省略するよう

に構成してもよい。例えば、初期状態ですべてのピクセルが黒の場合であれば、ピクセルがすべて黒の場合はバックアップが存在しない初期状態であることを判断できるため、バックアップの保持を省略できる。

【００６６】

この状態から、アプリケーション１５０が画面表示のためにグラフィックライブラリ１３１を介してフレームバッファ１１４ａに対する描画命令を送る（１）。フレームバッファ１１４ａに対する描画完了を検知すると、検出部１４２が、更新部分検出処理を開始する（２）。検出部１４２は、バックアップした画像情報と、フレームバッファ１１４ａの最新の画像情報とを比較し、更新部分の矩形を検出する（３）。

【００６７】

検出した更新部分の画像情報は、ゲストＯＳ１３０の画面として表示される（４）。また、バックアップ記憶部１１３の画像情報をフレームバッファ１１４ａの最新の画像情報と同じ画像情報に更新する（５）。

【００６８】

なお、画像処理装置１００を仮想計算機サーバとし、ネットワークを介して接続された端末装置にゲストＯＳ１３０の画面を送信し、端末装置上で画面を表示するように構成してもよい。この場合、上記（４）で更新部分の画像情報を表示する代わりに、更新部分の画像情報を端末装置に送信する。

【００６９】

次に、更新部分検出処理の詳細な流れについて図５を用いて説明する。図５は、更新部分検出処理の全体の流れを示すフローチャートである。

【００７０】

なお、条件設定部１２１が、表示開始アドレスレジスタ１１４ｂにアクセスするとＶＭ　Ｅｘｉｔが発生するようにゲストＯＳ管理構造体を予め設定していることを前提とする。また、書き込み部１３２ａは、マルチバッファリングをサポートしていることをエミュレータ部１４１から通知されていることを前提とする。

【００７１】

この状態から、まず、アプリケーション１５０が、実行した処理に関連する画面情報の更新などが発生した場合に、表示画面の描画を要求する描画命令を出力する（ステップＳ５０１）。次に、描画部１３１ａが、描画命令に応じて画像処理を行ない、その結果である画像情報を生成する（ステップＳ５０２）。次に、書き込み部１３２ａが、フレームバッファ１１４ａの指定された位置に、要求された画像情報を書き込む（ステップＳ５０３）。フレームバッファ１１４ａに対する描画が完了すると、書き込み部１３２ａが、マルチバッファリングのバッファ切り替えのために、ＦｌｉｐＲｅｇｉｏｎ関数を実行する（ステップＳ５０４）。そして、この関数により、表示開始アドレスレジスタ１１４ｂがアクセスされる。

【００７２】

プロセッサ１１１は、表示開始アドレスレジスタ１１４ｂがアクセスされると、ゲストＯＳ管理構造体の設定に従い、ＶＭ　Ｅｘｉｔを発生させる（ステップＳ５０５）。これにより、ゲストＯＳ１３０からＶＭＭ１２０に処理が移る。ＶＭＭ１２０の判定部１２２は、ゲストＯＳ管理構造体のＶＭ　Ｅｘｉｔ情報領域を参照し、ＶＭ　Ｅｘｉｔが発生した原因を判定する。そして、判定部１２２は、画面更新が原因であると判定した場合、画面更新をエミュレータ部１４１に通知する（ステップＳ５０６）。

【００７３】

イベント処理部１４１ａは、通知を受信すると検出部１４２に対して検出処理の開始を指示する（ステップＳ５０７）。検出部１４２は、フレームバッファ１１４ａに書き込まれた最新の画像情報と、バックアップ記憶部１１３に保存されている画像情報とを比較し、相違する部分を更新部分として検出する（ステップＳ５０８）。

【００７４】

検出部１４２は、検出した更新部分を出力して（ステップＳ５０９）、更新部分検出処

理を終了する。

【００７５】

このように、第１の実施の形態にかかる画像処理装置では、マルチバッファリングで利用される表示開始アドレスレジスタ１１４ｂにアクセスしたときにＶＭＭに制御が移るようにプロセッサを設定することにより、描画完了をＶＭＭで検知可能とし、描画完了の検知を契機として画像情報の更新部分の検出処理を実行することができる。

【００７６】

これにより、描画途中の画像を表示する可能性がなく、表示画像の見栄えが良くなる。また、プロセッサの設定を変更するのみで描画完了をＶＭＭでフックすることができるため、仮想環境に合わせてグラフィックライブラリやビデオドライバを改変する必要がない。すなわち、標準的なグラフィックライブラリやビデオドライバを用いて上記機能を実現することができる。

【００７７】

（変形例）

これまでは、ＶＭＭがハードウェア上で直接実行される例について説明した。これに対し、ＶＭＭをハードウェア上で間接的に実行するように構成しても、上記実施の形態と同様の機能を実現できる。間接的に実行するとは、ハードウェア上では、動作基盤となるホストＯＳが動作し、このホストＯＳ上でさらにＶＭＭが動作することを意味する。

【００７８】

図６は、このように構成した第１の実施の形態の変形例にかかる画像処理装置６００のブロック図である。図６に示すように、画像処理装置６００は、ハードウェア１１０を利用して機能するソフトウェア構成として、ホストＯＳ６６０と、ＶＭＭ６２０と、ゲストＯＳ１３０と、アプリケーション１５０とを備えている。

【００７９】

本変形例では、ホストＯＳ６６０が追加され、管理ＯＳ１４０が削除され、管理ＯＳ１４０内の構成であるエミュレータ部１４１および検出部１４２がＶＭＭ６２０内に含まれる点が第１の実施の形態のブロック図である図３と異なっている。その他の構成および機能は、第１の実施の形態にかかる画像処理装置１００の構成を表すブロック図である図３と同様であるので、同一符号を付し、ここでの説明は省略する。また、画像書き込み処理および更新部分検出処理は、図４および図５と同様であるのでその説明を省略する。

【００８０】

（第２の実施の形態）

第１の実施の形態では、描画完了通知の際に表示開始アドレスレジスタがアクセスされることを利用し、表示開始アドレスレジスタへのアクセスをＶＭＭでフックすることにより画面更新を検出していた。

【００８１】

しかし、ビデオドライバの種類によっては、このように描画完了時に所定のレジスタにアクセスしない場合がある。一方、ＶＥＳＡ（Video Electronics Standards Association）のビデオドライバのように、描画完了通知の際にＢＩＯＳコールを発行するビデオドライバが存在する。そこで、第２の実施の形態では、描画完了時に所定のＢＩＯＳコールを発行するビデオドライバを備える例について説明する。第２の実施の形態では、所定のＢＩＯＳコールが発行されたときにＶＭＭに制御が移るようにプロセッサを設定して描画完了をＶＭＭで検知可能とし、描画完了の検知を契機として画像情報の更新部分の検出処理を実行する。

【００８２】

最初に、本実施の形態で利用するビデオドライバの概要について説明する。本実施の形態では、ゲストＯＳのビデオドライバとしてＶＥＳＡのビデオドライバを利用する。ＶＥＳＡのビデオドライバは、グラフィックカードがマルチバッファリングをサポートしている場合に、描画完了時にバッファを切り替えるためのＳｅｔ　Ｄｉｓｐｌａｙ　Ｓｔａｒｔ　ＢＩＯＳコールを実行する。

【００８３】
このＢＩＯＳコールをＶＭＭでフックする方法は、プロセッサの動作モードによって異なる。なお、ここでのプロセッサの動作モードは、ゲストモードとホストモードとを有する上述の動作モードと異なるモードであり、リアルモードとプロテクトモードとを含む動作モードを意味する。
【００８４】
リアルモードは、アクセス可能なメモリ領域は０～１Ｍｂｙｔｅｓまでで、同時実行できるプログラムは１つが前提の動作モードである。一方、プロテクトモードは、アクセス可能なメモリ領域の拡大、マルチタスクサポート、データセキュリティ機能などが追加された動作モードである。
【００８５】
プロセッサは、電源投入時は必ずリアルモードで起動され、その後ソフトウェアによってプロテクトモードに移行する。通常は、コンピュータへの電源投入直後以外はプロテクトモードで動作する。
【００８６】
プロセッサがリアルモードで動作中の場合は、ＢＩＯＳコールを発行する際にＩＮＴ　１０ｈソフトウェア割り込みが発生する。そこで、ＩＮＴ　１０ｈソフトウェア割り込みをフックしてＶＭ　Ｅｘｉｔさせることができる。
【００８７】
一方、プロセッサがプロテクトモードで動作中の場合は、ＶＭＭでフック可能な割り込みやレジスタアクセスや例外が発生しない。そこで、この場合は、まず該当するＢＩＯＳコールのアドレスを、プロセッサ内のデバッグレジスタにブレークポイントとして設定する。これにより、該当するＢＩＯＳコールが発行されるとデバッグ例外を発生させることができる。そして、このデバッグ例外の発生をフックすることにより、ＶＭ　Ｅｘｉｔさせることができる。
【００８８】
すなわち、本実施の形態では、所定のＢＩＯＳコールの発行により発生するＩＮＴ　１０ｈソフトウェア割り込みまたはデバッグ例外を条件としてＶＭ　Ｅｘｉｔが発生するように、ゲストＯＳ管理構造体を設定する。これにより、プロセッサがリアルモードおよびプロテクトモードのいずれの動作モードで動作中の場合であっても、画面更新完了の際にＶＭＭに処理を移すことができる。
【００８９】
次に、第２の実施の形態にかかる画像処理装置の機能と構成について説明する。図７は、第２の実施の形態にかかる画像処理装置７００のブロック図である。図７に示すように、画像処理装置７００は、ハードウェア７１０と、ハードウェア７１０を利用して機能するソフトウェア構成として、ＶＭＭ７２０と、ゲストＯＳ７３０と、管理ＯＳ１４０と、アプリケーション１５０とを備えている。
【００９０】
第２の実施の形態では、ハードウェア７１０内の条件記憶部７１２が記憶するゲストＯＳ管理構造体の内容、ＶＭＭ７２０内の条件設定部７２１の機能、およびゲストＯＳ７３０の機能が第１の実施の形態と異なっている。その他の構成および機能は、第１の実施の形態にかかる画像処理装置１００の構成を表すブロック図である図３と同様であるので、同一符号を付し、ここでの説明は省略する。
【００９１】
条件記憶部７１２は、所定のＢＩＯＳコールに伴って発生する割り込みまたは例外に対応して、ゲストＯＳ７３０からＶＭＭ７２０に処理を戻す条件を含むゲストＯＳ管理構造体を記憶する。
【００９２】
ここで、条件記憶部７１２に記憶されるゲストＯＳ管理構造体の具体的な設定方法について説明する。第２の実施の形態では、以下のようにゲストＯＳ管理構造体を設定するこ

とにより、ＩＮＴ　１０ｈソフトウェア割り込み、およびデバッグ例外を検知する。
【００９３】
まず、Ｉｎｔｅｌ　ＶＴのＶＭＣＳの設定例について説明する。この場合、ＶＭＣＳのＶＭ　Ｅｘｉｔ制御領域の「Exception Bitmap」、およびＶＭ　Ｅｘｉｔ情報領域の「Exit Reason」、「Exit Qualification」、「IDT-Vectoring Information」を用いる。
【００９４】
ＩＮＴ　１０ｈソフトウェア割り込みを検知する場合は、まず、プロセッサ１１１の中に備えられているＴＳＳ（Task State Segment）のソフトウェア割り込みリダイレクションビットマップの１０ビット目を１に設定する。そして、「Exception Bitmap」のうち、一般保護例外に対応するビットを１にする。ＩＮＴ　１０ｈソフトウェア割り込みにより、一般保護例外またはタスクスイッチが発生するとＶＭ　Ｅｘｉｔが発生する。
【００９５】
一般保護例外の場合、ＶＭ　Ｅｘｉｔ情報領域の「Exit Reason」の０－１５ビットが０、「IDT-Vectoring Information」の０－７ビット（割り込みのベクタ）が１０、８－１０ビット（割り込みの種類）が４になる。また、タスクスイッチの場合、ＶＭ　Ｅｘｉｔ情報領域の「Exit Reason」の０－１５ビットが９、「Exit Qualification」の３０－３１ビット（タスクスイッチの発生原因）が３、「IDT-Vectoring Information」の０－７ビットが１０、８－１０ビットが４になる。
【００９６】
これらの情報を参照することにより、ＶＭＭ７２０は、ＩＮＴ　１０ｈソフトウェア割り込みが原因でＶＭ　Ｅｘｉｔが発生したことを判定することができる。
【００９７】
デバッグ例外を検知する場合は、「Exception Bitmap」のうち、デバッグ例外に対応するビットを１にする。これにより、デバッグ例外が発生するとＶＭ　Ｅｘｉｔが発生する。そして、デバッグ例外が原因でＶＭ　Ｅｘｉｔが発生すると、ＶＭ　Ｅｘｉｔ情報領域の「Exit Reason」の０－１５ビットが０、「Exit Qualification」の０－３ビット（Ｂ０－Ｂ３フラグ）がブレークポイント番号になる。
【００９８】
これらの情報を参照することにより、ＶＭＭ７２０は、デバッグ例外が原因でＶＭ　Ｅｘｉｔが発生したことを判定することができる。
【００９９】
次に、ＡＭＤ　ＳＶＭのＶＭＣＢの設定例について説明する。この場合、ＶＭＣＢの制御エリアの「Exception Bitmap」、「EXITCODE」、および「EXITINFO1」を用いる。
【０１００】
ＩＮＴ　１０ｈソフトウェア割り込みを検知する方法としては、すべてのソフトウェア割り込みを対象としてＶＭ　Ｅｘｉｔさせる方法と、ＩＮＴ　１０ｈソフトウェア割り込みのみを対象としてＶＭ　Ｅｘｉｔさせる方法が存在する。
【０１０１】
前者の場合、ＶＭＣＢのバイトオフセット「00Ch」の２１ビット目を１にする。これにより、ソフトウェア割り込みの発生に応じてＶＭ　Ｅｘｉｔが発生し、「EXITCODE」が「75h」、「EXITINFO1」の０－７ビットが１０になる。
【０１０２】
後者の場合、プロセッサ１１１の中に備えられているＣＲ４（Control Register 4）の０ビット目を１にして、ＴＳＳのソフトウェア割り込みリダイレクションビットマップの１０ビット目を１にして、ＶＭＣＢのバイトオフセット「008h」の「Exception Bitmap」のうち、一般保護例外に対応する１３ビット目を１にする。これにより、ＩＮＴ　１０ｈソフトウェア割り込みの発生に応じてＶＭ　Ｅｘｉｔが発生し、「EXITCODE」が「53h」になる。
【０１０３】
これらの情報を参照することにより、ＶＭＭ７２０は、ＩＮＴ　１０ｈソフトウェア割

り込みが原因でＶＭ　Ｅｘｉｔが発生したことを判定することができる。
【０１０４】
デバッグ例外を検知する場合は、ＶＭＣＢのバイトオフセット「008h」の「Exception Bitmap」のうち、デバッグ例外に対応する１ビット目を１にする。これにより、デバッグ例外の発生に応じてＶＭ　Ｅｘｉｔが発生し、「EXITCODE」が「41h」になる。
【０１０５】
これらの情報を参照することにより、ＶＭＭ７２０は、デバッグ例外が原因でＶＭ　Ｅｘｉｔが発生したことを判定することができる。
【０１０６】
条件設定部７２１は、このような設定方法に従い、画像情報の描画が完了したときにＶＭＭ７２０に処理が移るようにゲストＯＳ管理構造体に条件を事前に設定する。また、条件設定部７２１は、描画完了時に発行されるＢＩＯＳコールのアドレスを調べ、デバッグレジスタにブレークポイントとして設定するブレークポイント設定処理を実行する。ブレークポイント設定処理の詳細については後述する。
【０１０７】
ゲストＯＳ７３０は、グラフィックライブラリ１３１およびビデオドライバ７３２に加え、例外発生部７３３をさらに備えている。
【０１０８】
ビデオドライバ７３２は、上述のようにＶＥＳＡに対応したドライバであり、書き込み部７３２ａを備えている。書き込み部７３２ａは、画像情報の描画が完了したときに、表示用バッファを切り替えるためにＳｅｔ　Ｄｉｓｐｌａｙ　Ｓｔａｒｔ　ＢＩＯＳコールを実行する点が、第１の実施の形態の書き込み部１３２ａと異なっている。
【０１０９】
なお、適用可能なドライバはＶＥＳＡ対応のドライバに限られず、描画完了時に特定のＢＩＯＳコールを実行するドライバであれば、あらゆる種類のドライバを利用できる。
【０１１０】
例外発生部７３３は、デバッグレジスタに設定された仮想アドレスの示すメモリ領域がアクセスされたときにデバッグ例外を発生する。本実施の形態では、Ｓｅｔ　Ｄｉｓｐｌａｙ　Ｓｔａｒｔ　ＢＩＯＳコールのアドレスをデバッグレジスタに設定するため、例外発生部７３３は、このＢＩＯＳコールが実行されるとデバッグ例外を発生する。
【０１１１】
なお、Ｓｅｔ　Ｄｉｓｐｌａｙ　Ｓｔａｒｔ　ＢＩＯＳコールのアドレスは以下のようにして取得し、デバッグレジスタに設定することができる。図８は、第２の実施の形態におけるブレークポイント設定処理の全体の流れを示すフローチャートである。ブレークポイント設定処理とは、指定したアドレスにプログラムが来た場合にプログラムを中断するための処理である。本実施の形態では、Ｓｅｔ　Ｄｉｓｐｌａｙ　Ｓｔａｒｔ　ＢＩＯＳコールのアドレスにプログラムが来るとプログラムが中断され、ＶＭＭ７２０に処理が移る。
【０１１２】
まず、条件設定部７２１は、ブレークポイントとして設定するアドレスを調べるために、ＢＩＯＳイメージの先頭３２Ｋｂ内に存在するＰｒｏｔｅｃｔｅｄ　Ｍｏｄｅ　Ｉｎｆｏｒｍａｔｉｏｎ　Ｂｌｏｃｋ構造体を見つける。条件設定部７２１は、この構造体から、プロテクトモード時のＳｅｔ　Ｄｉｓｐｌａｙ　Ｓｔａｒｔ　ＢＩＯＳコールのアドレスを取得することができる（ステップＳ８０１）。なお、ＢＩＯＳイメージは、図示しない不揮発性メモリなどに記憶されている。
【０１１３】
Ｓｅｔ　Ｄｉｓｐｌａｙ　Ｓｔａｒｔ　ＢＩＯＳコールのアドレスを取得した後、条件設定部７２１は、取得したアドレスをブレークポイントとしてデバッグレジスタに設定する。
【０１１４】

ここで、デバッグレジスタの構成例について図９～図１１を用いて説明する。なお、デバッグレジスタは、ＤＲ０～ＤＲ７の８個のレジスタを含む。このうち、ＤＲ４およびＤＲ５は、予約済みで使用されないデバッグレジスタであるため、説明を省略する。

【０１１５】

図９は、デバッグレジスタＤＲ０～ＤＲ３の構成例を示す図である。ＤＲ０～ＤＲ３は、ブレークポイントとする仮想アドレスをそれぞれ設定するレジスタであり、デバッグアドレスレジスタと呼ばれる。

【０１１６】

図１０は、デバッグレジスタＤＲ６の構成例を示す図である。ＤＲ６は、デバッグ例外が発生したときの状況を示すレジスタであり、デバッグ状態レジスタと呼ばれる。同図で、Ｂ０～Ｂ３フラグがセットされている場合、対応するデバッグアドレスレジスタに設定されているブレークポイントのブレーク条件が満たされてデバッグ例外が発生したことを示す。

【０１１７】

図１１は、デバッグレジスタＤＲ７の構成例を示す図である。ＤＲ７は、デバッグアドレスレジスタに設定されているブレークポイントのブレーク条件を設定するレジスタであり、デバッグ制御レジスタと呼ばれる。

【０１１８】

Ｌ０～Ｌ３フラグまたはＧ０～Ｇ３フラグがセットされている場合、対応するデバッグアドレスレジスタに設定されているブレークポイントが有効になっていることを示す。Ｌ０～Ｌ３フラグは現在のタスクに対してのみ有効にする場合にセットされ、Ｇ０～Ｇ３フラグはすべてのタスクに対して有効にする場合にセットされる。

【０１１９】

Ｒ／Ｗ０～Ｒ／Ｗ３フィールドは、対応するデバッグアドレスレジスタに設定されているブレークポイントのブレーク条件を示す。命令の実行でブレークする場合は、「００」を設定する。ＬＥＮ０～ＬＥＮ３フィールドは、対応するデバッグアドレスレジスタに設定されているブレークポイントのサイズを示す。命令の実行でブレークする場合は、「００」を設定する。

【０１２０】

また、ＧＤフラグがセットされている場合、デバッグレジスタを上書きするときにデバッグ例外が発生する。このフラグをセットすることにより、条件設定部７２１により設定したブレークポイントをゲストＯＳ７３０により上書きされるのを防ぐことができる。

【０１２１】

このように構成されたデバッグレジスタの場合、取得したアドレスをブレークポイントとして設定するためには、デバッグアドレスレジスタであるＤＲ０～ＤＲ３のいずれかと、デバッグ制御レジスタであるＤＲ７の対応するビットを設定する。

【０１２２】

図８に戻り、デバッグレジスタの設定処理の詳細について説明する。条件設定部７２１は、ＤＲ０～ＤＲ３のいずれかに、取得したアドレスをブレークポイントとして設定する（ステップＳ８０２）。ここでは、ブレークポイント０（ＤＲ０）を利用する例について説明する。この場合、条件設定部７２１は、デバッグアドレスレジスタ（ＤＲ０）に取得したアドレスを設定する。

【０１２３】

次に、条件設定部７２１は、デバッグ制御レジスタ（ＤＲ７）のＧ０フラグを１に設定する（ステップＳ８０３）。また、条件設定部７２１は、ＤＲ７のＲ／Ｗ０フィールドを「００」に設定する（ステップＳ８０４）。さらに、条件設定部７２１は、ＤＲ７のＬＥＮ０フィールドを「００」に設定する（ステップＳ８０５）。

【０１２４】

なお、図８の記号「＊」は、デバッグアドレスレジスタＤＲ０～ＤＲ３のいずれにブレークポイントを設定したかによって数値が変わることを意味している。すなわち、例えば

ＤＲ１にブレークポイントを設定した場合には、ＤＲ７のＧ１フラグ、Ｒ／Ｗ１フィールド、ＬＥＮ１フィールドを設定することを意味する。

【０１２５】

このようにデバッグレジスタを設定することにより、設定したアドレスにアクセスするとデバッグ例外が発生する。なお、デバッグ状態レジスタ（ＤＲ６）のＢ０フラグを確認することにより、デバッグ例外が発生した原因がブレークポイント０へのアクセスであることを確認できる。

【０１２６】

次に、このように構成された第２の実施の形態にかかる画像処理装置７００による更新部分検出処理について図１２を用いて説明する。なお、画像処理の概要は、第１の実施の形態の図４と同様であるので説明を省略する。図１２は、第２の実施の形態における更新部分検出処理の全体の流れを示すフローチャートである。

【０１２７】

ステップＳ１２０１からステップＳ１２０３までの、画像情報書き込み処理は、第１の実施の形態にかかる画像処理装置１００におけるステップＳ５０１からステップＳ５０３までと同様の処理なので、その説明を省略する。

【０１２８】

画像情報を書き込み後、書き込み部７３２ａは、マルチバッファリングのバッファ切り替えのために、Ｓｅｔ　Ｄｉｓｐｌａｙ　Ｓｔａｒｔ　ＢＩＯＳコールを発行する（ステップＳ１２０４）。

【０１２９】

このＢＩＯＳコールが発行されると、上述のようなデバッグレジスタの設定に従い、例外発生部７３３がデバッグ例外を発生する（ステップＳ１２０５）。なお、プロセッサ１１１がリアルモードで動作中の場合は、デバッグ例外の代わりにＩＮＴ　１０ｈソフトウェア割り込みが発生する。

【０１３０】

プロセッサ１１１は、デバッグ例外またはＩＮＴ　１０ｈソフトウェア割り込みが発生すると、ゲストＯＳ管理構造体の設定に従い、ＶＭ　Ｅｘｉｔを発生させる（ステップＳ１２０６）。これにより、ゲストＯＳ７３０からＶＭＭ７２０に処理が移る。

【０１３１】

ステップＳ１２０７からステップＳ１２１０までの、原因判定処理、検出処理、更新部分出力処理は、第１の実施の形態にかかる画像処理装置１００におけるステップＳ５０６からステップＳ５０９までと同様の処理なので、その説明を省略する。

【０１３２】

このように、第２の実施の形態にかかる画像処理装置では、所定のＢＩＯＳコールの発行時にＶＭＭに制御が移るようにプロセッサを設定することにより、描画完了をＶＭＭで検知可能とし、描画完了の検知を契機として画像情報の更新部分の検出処理を実行することができる。

【０１３３】

次に、第１または第２の実施の形態にかかる画像処理装置のハードウェア構成について図１３を用いて説明する。図１３は、第１または第２の実施の形態にかかる画像処理装置のハードウェア構成を示す説明図である。

【０１３４】

第１または第２の実施の形態にかかる画像処理装置は、プロセッサ１１１などの制御装置と、ＲＯＭ（Read Only Memory）５２やＲＡＭ５３などの記憶装置と、ネットワークに接続して通信を行う通信Ｉ／Ｆ５４と、ＨＤＤ、ＣＤ（Compact Disc）ドライブ装置などの外部記憶装置と、ディスプレイ装置などの表示装置と、キーボードやマウスなどの入力装置と、各部を接続するバス６１を備えており、通常のコンピュータを利用したハードウェア構成となっている。

【０１３５】

第１または第２の実施の形態にかかる画像処理装置で実行される更新検出プログラムは、インストール可能な形式又は実行可能な形式のファイルでＣＤ－ＲＯＭ（Compact Disk Read Only Memory）、フレキシブルディスク（ＦＤ）、ＣＤ－Ｒ（Compact Disk Recordable）、ＤＶＤ（Digital Versatile Disk）等のコンピュータで読み取り可能な記録媒体に記録されて提供される。

【０１３６】

また、第１または第２の実施の形態にかかる画像処理装置で実行される更新検出プログラムを、インターネット等のネットワークに接続されたコンピュータ上に格納し、ネットワーク経由でダウンロードさせることにより提供するように構成してもよい。また、第１または第２の実施の形態にかかる画像処理装置で実行される更新検出プログラムをインターネット等のネットワーク経由で提供または配布するように構成してもよい。

【０１３７】

また、第１または第２の実施の形態の更新検出プログラムを、ＲＯＭ等に予め組み込んで提供するように構成してもよい。

【０１３８】

第１または第２の実施の形態にかかる画像処理装置で実行される更新検出プログラムは、上述した各部（ＶＭＭ、管理ＯＳ）を含むモジュール構成となっており、実際のハードウェアとしてはプロセッサ１１１が上記記憶媒体から更新検出プログラムを読み出して実行することにより上記各部が主記憶装置上にロードされ、上述した各部が主記憶装置上に生成されるようになっている。

【産業上の利用可能性】

【０１３９】

以上のように、本発明にかかるプログラム、方法および画像処理装置は、ハードウェアレベルの仮想化支援機能を実装したプロセッサを利用した仮想計算機環境に適している。

【図面の簡単な説明】

【０１４０】

【図１】シングルバッファリングでの表示画面の一例を示す図である。
【図２】ポインタを切り替える方法によるダブルバッファリングの概要を示す図である。
【図３】第１の実施の形態にかかる画像処理装置のブロック図である。
【図４】画像書き込み処理および更新部分検出処理の概要を示す図である。
【図５】更新部分検出処理の全体の流れを示すフローチャートである。
【図６】第１の実施の形態の変形例にかかる画像処理装置のブロック図である。
【図７】第２の実施の形態にかかる画像処理装置のブロック図である。
【図８】第２の実施の形態におけるブレークポイント設定処理の全体の流れを示すフローチャートである。
【図９】デバッグレジスタの構成例を示す図である。
【図１０】デバッグレジスタの構成例を示す図である。
【図１１】デバッグレジスタの構成例を示す図である。
【図１２】第２の実施の形態における更新部分検出処理の全体の流れを示すフローチャートである。
【図１３】第１または第２の実施の形態にかかる画像処理装置のハードウェア構成を示す説明図である。

【符号の説明】

【０１４１】

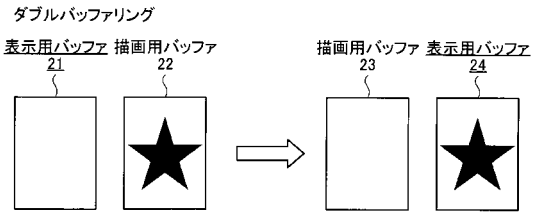
２１、２４　表示用バッファ
２２、２３　描画用バッファ
５２　ＲＯＭ
５３　ＲＡＭ
５４　通信Ｉ／Ｆ
６１　バス

１００　画像処理装置
１１０　ハードウェア
１１１　プロセッサ
１１２　条件記憶部
１１３　バックアップ記憶部
１１４　グラフィックカード
１１４ａ　フレームバッファ
１１４ｂ　表示開始アドレスレジスタ
１１５ａ　フレームバッファ領域
１１５ｂ　表示開始アドレスレジスタ領域
１２０　ＶＭＭ
１２１　条件設定部
１２２　判定部
１３０　ゲストＯＳ
１３１　グラフィックライブラリ
１３１ａ　描画部
１３２　ビデオドライバ
１３２ａ　書き込み部
１４０　管理ＯＳ
１４１　エミュレータ部
１４１ａ　イベント処理部
１４２　検出部
１５０　アプリケーション
６００　画像処理装置
６２０　ＶＭＭ
６６０　ホストＯＳ
７００　画像処理装置
７１０　ハードウェア
７１２　条件記憶部
７２０　ＶＭＭ
７２１　条件設定部
７３０　ゲストＯＳ
７３２　ビデオドライバ
７３２ａ　書き込み部
７３３　例外発生部

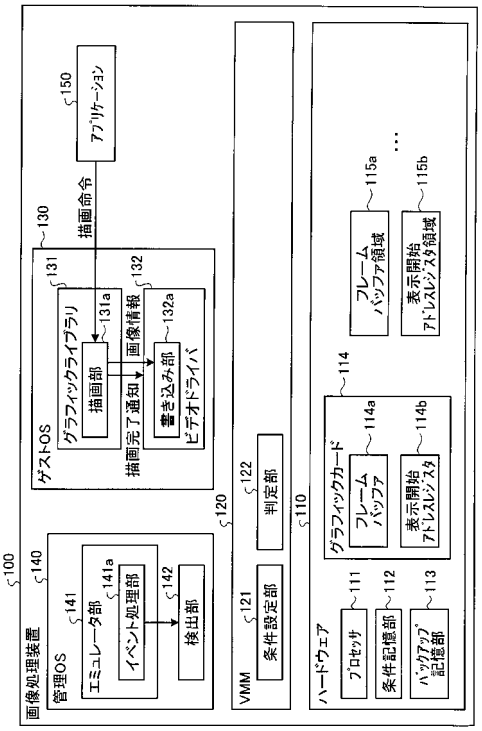
【図１】


シングルバッファリング

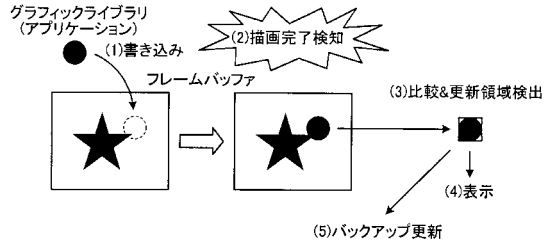
【図２】


ダブルバッファリング
表示用バッファ
21
描画用バッファ
22
描画用バッファ
23
表示用バッファ
24

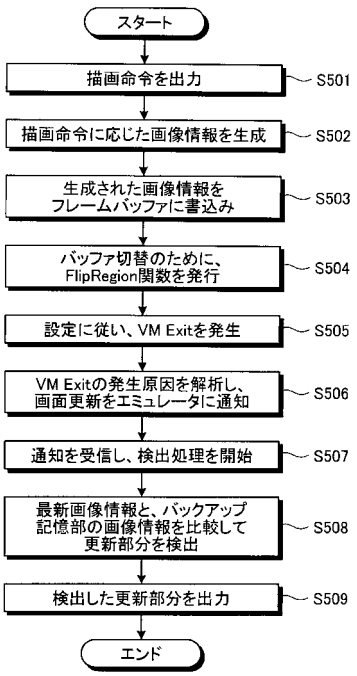
【図３】


画像処理装置
100
ゲストOS
130
アプリケーション
150
描画命令
グラフィックライブラリ
131
描画部
131a
描画完了通知
画像情報
132
書き込み部
132a
ビデオドライバ
管理OS
140
エミュレータ部
141
イベント処理部
141a
検出部
142
VMM
120
条件設定部
121
判定部
122
ハードウェア
110
プロセッサ
111
条件記憶部
112
バックアップ記憶部
113
グラフィックカード
114
フレームバッファ
114a
表示開始アドレス/ピッチ
114b
フレームバッファ領域
115a
表示開始アドレス/ピッチ外領域
115b

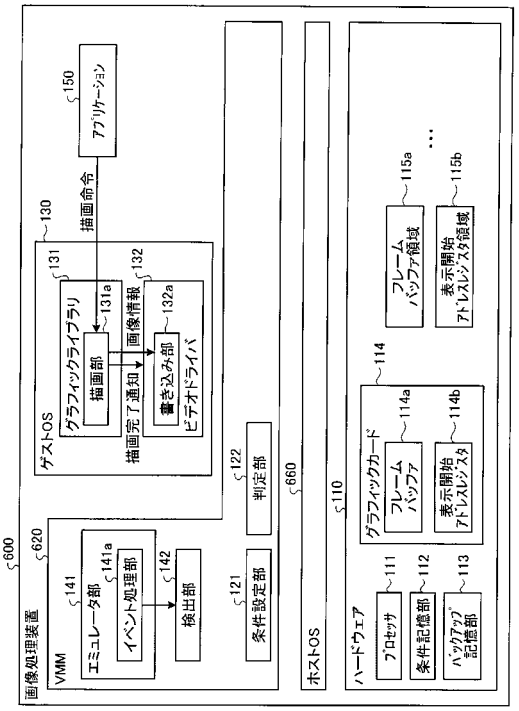
【図４】


グラフィックライブラリ
(アプリケーション)
(1)書き込み
(2)描画完了検知
フレームバッファ
(3)比較&更新領域検出
(4)表示
(5)バックアップ更新

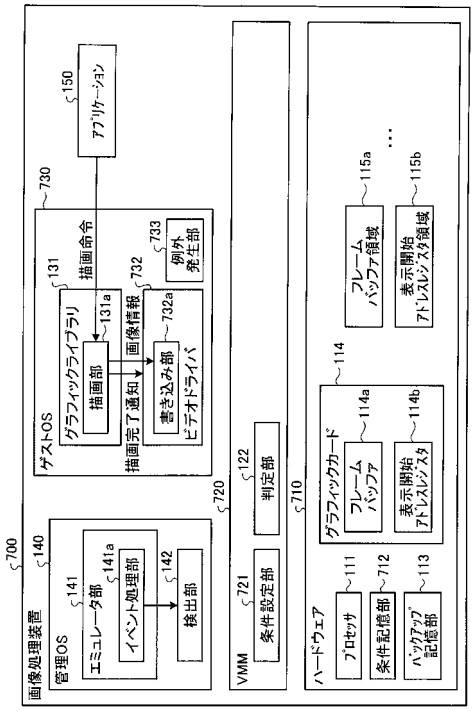
【図５】


スタート
描画命令を出力
S501
描画命令に応じた画像情報を生成
S502
生成された画像情報を
フレームバッファに書込み
S503
バッファ切替のために、
FlipRegion関数を発行
S504
設定に従い、VM Exitを発生
S505
VM Exitの発生原因を解析し、
画面更新をエミュレータに通知
S506
通知を受信し、検出処理を開始
S507
最新画像情報と、バックアップ
記憶部の画像情報を比較して
更新部分を検出
S508
検出した更新部分を出力
S509
エンド

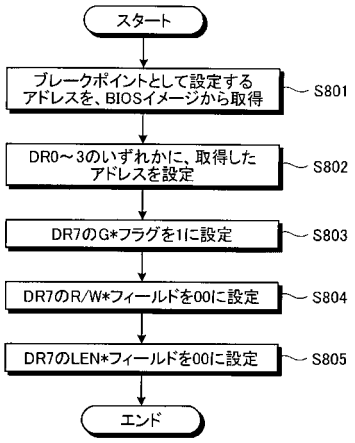
【図６】



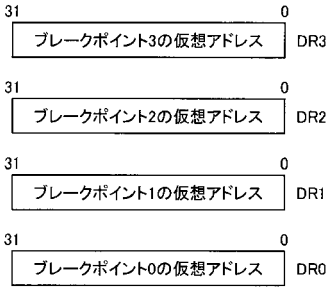
【図７】



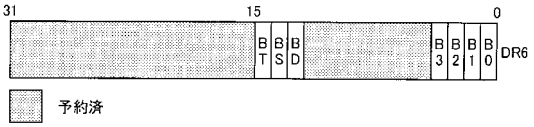
【図８】



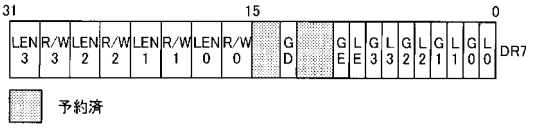
【図９】



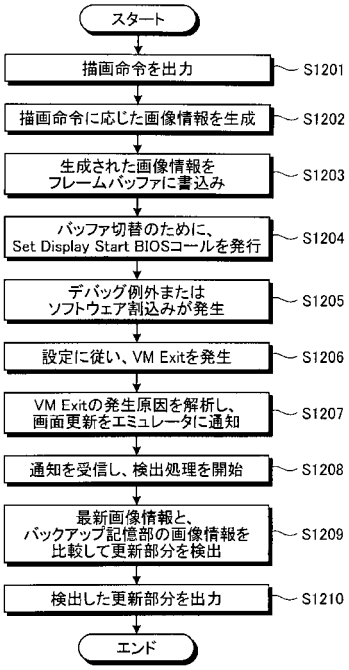
【図１０】



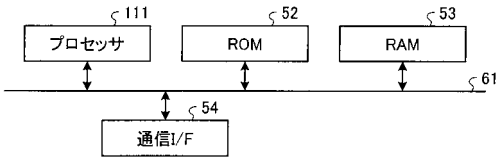
【図１１】



【図１２】


スタート
描画命令を出力
S1201
描画命令に応じた画像情報を生成
S1202
生成された画像情報を
フレームバッファに書込み
S1203
バッファ切替のために、
Set Display Start BIOSコールを発行
S1204
デバッグ例外または
ソフトウェア割込みが発生
S1205
設定に従い、VM Exitを発生
S1206
VM Exitの発生原因を解析し、
画面更新をエミュレータに通知
S1207
通知を受信し、検出処理を開始
S1208
最新画像情報と、
バックアップ記憶部の画像情報を
比較して更新部分を検出
S1209
検出した更新部分を出力
S1210
エンド

【図１３】


111
プロセッサ
52
ROM
53
RAM
61
54
通信I/F

フロントページの続き

(72)発明者　村井　信哉
東京都港区芝浦一丁目１番１号　株式会社東芝内

審査官　吉田　美彦

(56)参考文献　特開平１０－２００７５７（ＪＰ，Ａ）
特開平１０－３０７７３１（ＪＰ，Ａ）
特開２０００－３３０８０６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／４６
Ｇ０６Ｔ　　１／６０
Ｇ０９Ｇ　　５／００