(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2013-161299
(P2013-161299A)
(43)公開日　平成25年8月19日(2013.8.19)

| (51)Int.Cl. | | F I | | テーマコード（参考） |
|---|---|---|---|---|
| G06F 13/10 | (2006.01) | G06F 13/10 | 330B | 5B014 |

審査請求 未請求　請求項の数 8　ＯＬ　（全 17 頁）

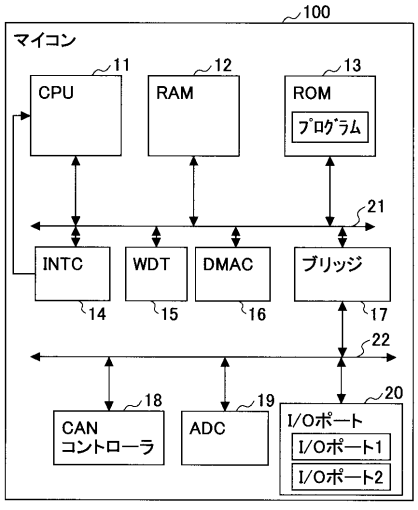
(21)出願番号　特願2012-23447(P2012-23447)
(22)出願日　平成24年2月6日(2012.2.6)

(71)出願人　000003207
トヨタ自動車株式会社
愛知県豊田市トヨタ町１番地
(74)代理人　100070150
弁理士　伊東　忠彦
(72)発明者　神田　渉
愛知県豊田市トヨタ町１番地　トヨタ自動車株式会社内

Ｆターム(参考)　5B014 FB03　FB04

(54)【発明の名称】情報処理装置、インタフェースアクセス方法

(57)【要約】
【課題】特別なハードウェアを用いることなく適切な量の管理情報を使用することでＩ／Ｏポートを保護できる情報処理装置を提供すること。
【解決手段】複数のプログラムが記憶されたプログラム記憶手段１３と、前記プログラムを実行する１つ以上の演算手段１１と、外部の回路と通信可能な複数のインタフェース２０と、を有する情報処理装置１００であって、前記プログラムがアクセス可能な前記インタフェースが登録されたインタフェース登録テーブル３４と、前記インタフェースにアクセス要求した前記プログラムが、前記インタフェースにアクセスすることが許可されている場合、前記プログラムの代わりに前記インタフェースにアクセスするアクセス制御手段３３と、を有する。
【選択図】図２



【特許請求の範囲】
【請求項１】
複数のプログラムが記憶されたプログラム記憶手段と、
前記プログラムを実行する１つ以上の演算手段と、
外部の回路と通信可能な複数のインタフェースと、を有する情報処理装置であって、
前記プログラムがアクセス可能な前記インタフェースが登録されたインタフェース登録テーブルと、
前記インタフェースにアクセス要求した前記プログラムが、前記インタフェースにアクセスすることが許可されている場合、前記プログラムの代わりに前記インタフェースにアクセスするアクセス制御手段と、
を有する情報処理装置。
【請求項２】
複数の前記プログラムのそれぞれが前記インタフェースにアクセスすることを禁止するアクセス禁止手段を有し、
前記アクセス制御手段は、前記アクセス禁止手段から前記プログラムが前記インタフェースにアクセス要求したことを検出する、
ことを特徴とする請求項１記載の情報処理装置。
【請求項３】
前記インタフェースにアクセスした前記プログラム及び前記インタフェースを識別するためのプログラム識別情報を有し、
前記アクセス制御手段は、前記プログラムが前記インタフェースにアクセス要求した際、前記プログラム識別情報に基づき、前記プログラム及び前記インタフェースを特定する、
ことを特徴とする請求項１又は２記載の情報処理装置。
【請求項４】
前記プログラム識別情報は、前記アクセス禁止手段が、前記プログラムが前記インタフェースにアクセス要求したことを検出した際に前記演算手段が実行していた前記プログラムのアドレスである、
ことを特徴とする請求項３記載の情報処理装置。
【請求項５】
複数の前記プログラムが同じ前記インタフェースにアクセスする場合、
前記アクセス制御手段は、第一のプログラムによる前記インタフェースへのアクセスが、前記第一のプログラムよりも優先度が低い第二のプログラムにより待機状態となることを予測した場合、前記第二のプログラムの前記インタフェースへのアクセスを行う前に、前記第一のプログラムの前記インタフェースへのアクセスを行い、前記第一のプログラムの前記インタフェースへのアクセスが完了した後、前記第二のプログラムの前記インタフェースへのアクセスを行う、
ことを特徴とする請求項１～４いずれか１項記載の情報処理装置。
【請求項６】
前記アクセス制御手段は、前記インタフェースにアクセス要求した前記プログラム及びアクセスが許可されたか否かを記録しておき、アクセスが許可されていないという記録が多い方の前記プログラムの優先度が、アクセスが許可されていないという記録が少ない前記プログラムよりも低いと決定する、
ことを特徴とする請求項５記載の情報処理装置。
【請求項７】
前記アクセス制御手段は、前記プログラムが動作していた個々のハードウェアをエミュレートするハイパーバイザが有する機能であり、
前記ハイパーバイザは、複数の前記プログラムの各命令を解析した結果に基づき、前記ハードウェア上で前記プログラムが実行された場合の前記ハードウェアのメモリ、ＣＰＵのレジスタ、及び、入出力インタフェースのエミュレート状態を更新することで、複数の

前記プログラムを実行する、
ことを特徴とする請求項１～６いずれか１項記載の情報処理装置。
【請求項８】
複数のプログラムが記憶されたプログラム記憶手段と、
前記プログラムを実行する１つ以上の演算手段と、
外部の回路と通信可能な複数のインタフェースと、を有する情報処理装置のインタフェースアクセス方法であって、
アクセス制御手段が、前記プログラムがアクセス可能な前記インタフェースが登録されたインタフェース登録テーブルを参照して、前記インタフェースにアクセス要求した前記プログラムが、前記インタフェースにアクセスすることが許可されている場合、前記プログラムの代わりに前記インタフェースにアクセスする、
インタフェースアクセス方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、Ｉ／Ｏポートを有する情報処理装置に関し、システムやアプリ毎にＩ／Ｏポートへのアクセスを制御可能な情報処理装置に関する。
【背景技術】
【０００２】
マイコンなどの情報処理装置では、複数のタスクがそれぞれ独立に実行される場合がある。複数のタスクがそれぞれ専用のＩ／Ｏにアクセスすべき場合に、あるタスクが別のタスク専用のＩ／Ｏにアクセスするようなことが生じることは好ましくない。このため、従来から、タスクによるＩ／Ｏへのアクセスを保護する技術が知られている。
【０００３】
従来のマイコンでは、ハードウェアのメモリアクセス保護資源（例えばメモリ管理ユニット）によりタスクのＩ／Ｏポートへのアクセスを制限している。しかしながら、マイコンが提供するメモリ保護資源はページ又はブロックと呼ばれる単位（例えば５１２バイト～４ｋバイト）でメモリ保護を提供している。これに対し、Ｉ／Ｏポートのレジスタは数バイト（例えば１～４バイト）で構成されているため、従来の方法ではＩ／Ｏポート単位でアクセス制御することができない（１ページ又は１ブロックのメモリ保護で複数のＩ／Ｏポートにアクセス制限してしまう）。
【０００４】
また、新たにハードウェアを追加することでタスクのＩ／Ｏポートへのアクセスを制限する技術も考えられている（例えば、特許文献１参照。）。図１２は、特許文献１に開示されたアクセス保護を説明する図の一例である。特許文献１では、ＯＳ（Operating System）が各アプリのプロセッサバスアドレス空間へのアクセス権をページ単位で管理している。また、プロセッサバスアドレス空間とＩ／Ｏアドレス空間を変換器が変換し、Ｉ／Ｏアドレス空間の各Ｉ／Ｏポートを、プロセッサバスアドレス空間にマッピングする。ＯＳはアプリがＩ／Ｏへのアクセス権を有する場合に、変換器にＩ／Ｏへのアクセスを許可するので、アプリの不正なアクセスを防止できる。
【先行技術文献】
【特許文献】
【０００５】
【特許文献１】特開平７－１９１９０４号公報
【発明の概要】
【発明が解決しようとする課題】
【０００６】
しかしながら、特許文献１に開示された技術又は従来の技術では以下のような問題がある。
１．変換器を追加する必要がある。

サーバやワークステーション向けのプロセッサと異なり車両等の組み込み用のプロセッサでは、変換器を有していないことが多いため、変換器を追加する必要が生じてしまう。または、変換器を搭載したマイコンへ変更しなければならない。
２．変換器を追加すると、アプリ側から見えるＩ／Ｏポートのアドレスが変更される。すなわち、変換器の追加前までアプリがプロセッサバスアドレスに出力したＩ／Ｏのアドレスが変換器により変わってしまう。このため、アプリを修正する必要が生じてしまう。１つのアプリだけなら修正も可能だが、例えば、仮想化技術で複数のアプリを１つの仮想環境に実装した場合、仮想化による統合のメリットが失われてしまう。
３．メモリ保護の単位を細かくするとＯＳが保持するアクセス権に関する情報が増加する上記のハードウェアのメモリアクセス保護資源を用いる場合、ハードウェアの制約によりシステム全体で同じ単位（例えば４バイト）でメモリ保護する必要がある。このため、メモリアクセス保護資源を用いてＩ／Ｏポートに最適な粒度（例えば４バイト）でメモリ保護する場合、保護単位毎にアクセス権の設定が必要になってしまう。例えば、１ページを４ｋバイトとした場合、１ページだけで１０００個のアクセス権の設定が必要になる。したがって、保護単位に反比例してＯＳの管理情報が増加してしまう。

【０００７】

本発明は、上記課題に鑑み、特別なハードウェアを用いることなく適切な量の管理情報を使用することでＩ／Ｏポートを保護できる情報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】

【０００８】

本発明は、複数のプログラムが記憶されたプログラム記憶手段と、前記プログラムを実行する１つ以上の演算手段と、外部の回路と通信可能な複数のインタフェースと、を有する情報処理装置であって、前記プログラムがアクセス可能な前記インタフェースが登録されたインタフェース登録テーブルと、前記インタフェースにアクセス要求した前記プログラムが、前記インタフェースにアクセスすることが許可されている場合、前記プログラムの代わりに前記インタフェースにアクセスするアクセス制御手段と、を有する。

【発明の効果】

【０００９】

特別なハードウェアを用いることなく適切な量の管理情報を使用することでＩ／Ｏポートを保護できる情報処理装置を提供することができる。

【図面の簡単な説明】

【００１０】

【図１】Ｉ／Ｏポートへのアクセス制御を説明する図の一例である。
【図２】マイコンのハードウェア構成図の一例である。
【図３】仮想化技術による２つのシステムの統合を説明する図の一例である。
【図４】メモリ保護機構の機能を模式的に説明する図の一例である。
【図５】システム１，２によるＩ／Ｏポートへのアクセスを禁止する際のＰＴＥの設定例を示す図である。
【図６】ハイパーバイザの機能ブロック図の一例である。
【図７】システム実行情報等の一例を示す図である。
【図８】アクセス履歴テーブルの一例を示す図である。
【図９】マイコンの動作手順を示すフローチャート図の一例である。
【図１０】アクセスタイミングの予測を説明する図の一例である。
【図１１】マイコンの動作手順を示すフローチャート図の一例である。
【図１２】特許文献１に開示されたアクセス保護を説明する図の一例である。

【発明を実施するための形態】

【００１１】

以下、本発明を実施するための形態について図面を参照しながら説明する。

【実施例１】

【００１２】

図１は、本実施例のＩ／Ｏポートへのアクセス制御を説明する図の一例である。

【００１３】

１．まず、システムの起動時に、例えばハイパーバイザ３１は、メモリ保護機構３２に対しアプリ１、アプリ２がＩ／Ｏポートにアクセスすることを禁止する。なお、ハイパーバイザ３１は全てのＩ／Ｏポートにアクセスすることができる。

【００１４】

２．アプリ１又はアプリ２がＩ／Ｏポートにアクセスしようとすると、メモリ保護機構３２がアクセス違反（例外を発生させる）を検出する。

【００１５】

３．ハイパーバイザ３１は、メモリ保護機構３２からアクセスしたアプリとＩ／Ｏポートの情報を取得して、アプリがＩ／Ｏポートにアクセスする権限を有しているか否かを判定する。

【００１６】

４．アプリにＩ／Ｏポートへのアクセス権がある場合、ハイパーバイザ３１が該アプリの代わりにＩ／Ｏポートへアクセスする。

【００１７】

５．アプリにＩ／Ｏポートへのアクセス権がない場合、ハイパーバイザ３１はアクセスしようとしたアプリに対しアクセス違反を通知する。

【００１８】

したがって、変換器などの新たなハードウェアを追加することなく、各アプリのＩ／Ｏアクセスを保護できる。論理アドレスから物理アドレスの変換はメモリ保護機構３２が行うので、アプリが指定するＩ／Ｏポートのアドレスには変更が必要ない。また、メモリ保護機構３２に対し、全てのＩ／Ｏポートのアクセスを禁止すればよいので、ハイパーバイザ３１の管理情報が増大することもない。

【００１９】

〔構成例〕

図２は、マイコン１００のハードウェア構成図の一例である。マイコン１００は、メインバス２１に接続された、ＣＰＵ１１、ＲＡＭ１２、ＲＯＭ１３、ＩＮＴＣ１４、ＷＤＴ１５、及び、ＤＭＡＣ１６を有し、周辺バス２２に接続されたＣＡＮ（Controller Area Network）コントローラ１８、ＡＤＣ１９及びＩ／Ｏポート２０を有する。メインバス２１と周辺バス２２はブリッジ１７を介して接続されている。

【００２０】

ＣＰＵ１１は、好ましくは複数のコアを有し、ＲＯＭ１３に記憶されたプログラムを実行することでマイコン１００の全体を制御する。ＣＰＵはメモリ保護機構３２を有している。ＲＡＭ１２は、ＣＰＵ１１がプログラムを実行する際の作業メモリである。また、フラッシュＲＯＭ１３にはハイパーバイザ３１や各システム（ＯＳ、デバイスドライバ、ミドルウェア及びアプリ）が記憶されている。

【００２１】

ＩＮＴＣ１４は割込みレジスタを監視して、周辺機器からの割り込み要求を割込みの優先順位に基づき調停してＣＰＵ１１に通知する。これによりＣＰＵ１１は、例えばＩＳＲ（Interrupt Service Routine）を実行して、割込みした周辺機器に応じて定められているタスクを起床させる。

【００２２】

ＷＤＴ１５は、動作クロックをカウントして計測した時間が予め定められたリセット時間に達すると（オーバーフローすると）、異常検出（マイコンリセット、ＣＰＵ割込み等）する回路である。

【００２３】

ＤＭＡＣ１６は、ＲＡＭ１２と周辺回路の間や、ＲＡＭ１２内で、ＣＰＵ１１を介することなくデータを移動する。ＣＰＵ１１がＩ／Ｏポート２０から外部にデータを送信する場合、ＣＰＵ１１はＲＡＭ１２に記憶されているデータのアドレスと送信先の周辺回路の

アドレスを指示する。ＤＭＡＣ１６は、ＲＡＭ１２から読み出したデータを周辺回路のアドレスに対応するＩ／Ｏポート２０に書き込む。また、ＣＰＵ１１が外部からデータを受信する場合、Ｉ／Ｏポート２０にデータが到達すると、ＤＭＡＣ１６がそれを検出してＲＡＭ１２にデータを転送する。受信が終了するとＤＭＡＣ１６はＩＮＴＣ１４に割込み要求することで、ＣＰＵ１１はＲＡＭ１２のデータを読み出すことができる。

【００２４】

ブリッジ１７は、メインバス２１と周辺バス２２の間の周波数の違いを吸収し、メインバス２１に接続された回路と周辺バス２２に接続された回路とを通信可能に接続する。ＣＡＮコントローラ１８は、マイコン１００がＣＡＮバス４３に接続された他のＥＣＵ（Electronic Control Unit）と通信するための通信回路である。ＣＡＮコントローラ１８はＣＡＮフレームの受信をＤＭＡＣ１６に通知すると、ＤＭＡＣ１６がＣＡＮコントローラ１８の受信バッファからＣＡＮフレームを読み出しＲＡＭ１２に記憶する。受信が終了するとＤＭＡＣ１６はＩＮＴＣ１４に割込み要求することで、ＣＰＵ１１にＣＡＮフレームの受信を通知する。

【００２５】

ＡＤＣ（Ａ／Ｄコントーラ）１９は、マイコン１００に接続されたセンサのアナログ信号をデジタル信号に変換して、アクチュエータ等に制御信号を出力する。Ｉ／Ｏポート２０は周辺回路と入出力のインタフェースであり、周辺回路毎に複数用意されている。Ｉ／Ｏポート２０には、例えばセンサ、アクチュエータ、スイッチ等が接続されている。本実施例では、Ｉ／Ｏポート１をシステム１がＩ／Ｏポート２をシステム２がそれぞれ使用する。

【００２６】

マイコン１００は各種の電子制御装置に搭載される。本実施例では、異なる電子制御装置に搭載されていたシステム１，２を１つの電子制御装置に搭載することも可能ため、電子制御装置が単一の機能のみを有しているとは限らない。しかし、電子制御装置の機能としては、例えば、ＨＶ－ＥＣＵ、エンジンＥＣＵ、ブレーキＥＣＵ、ナビＥＣＵ等、種々のものがある。本実施例のマイコン１００はこれらの機能の１つ以上を提供する。

【００２７】

〔システムの統合〕

図３は、仮想化技術による２つのシステムの統合を説明する図の一例である。ＥＣＵ１に搭載されたアプリ１とＯＳ１をシステム１、ＥＣＵ２に搭載されたアプリ２とＯＳ２をシステム２と称する。ＥＣＵ１とＥＣＵ２のマイコンは同じものである必要はなく、ＯＳ１とＯＳ２も同じものである必要はない。

【００２８】

仮想化技術を利用することで、２つのシステム１，２をそのまま１つのマイコン１００に搭載することができる。システムの変更が不要なので低コスト、短期間にシステム統合が可能となる。ところで、車両の機能安全にはＡＳＩＬ（Automotive Safety Integrity Level）という指標の安全性レベルがあり、ＡＳＩＬに応じた機能安全が達成されることが求められる。システム１，２の統合の際、システム１と２のＡＳＩＬが異なる場合があると考えられるが、ＡＳＩＬが異なる場合、ＡＳＩＬが高いシステムのＩ／Ｏポート２０をＡＳＩＬが低いシステムから保護することが要求される。本実施例ではこのＩ／Ｏポート２０の保護をハイパーバイザ３１とメモリ保護機構３２を利用して実現する

ハイパーバイザ３１は、マイコン１，２のハードウェアをエミュレートするソフトウェアである。マイコン１，２は一般的なハードウェア（ＣＰＵ、メモリ、Ｉ／Ｏポート等）を有している。例えば、マイコン１，２のメモリをエミュレートする場合、ハイパーバイザ３２にはマイコン１，２に搭載されているＲＯＭ、ＲＡＭの領域を確保する処理（一連の命令）が記述されている。マイコン１００がマイコン１，２のプログラムを実行する際は、この確保した領域をマイコン１，２のメモリとしてアクセスすることで、マイコン１，２の動作をエミュレートする。すなわち、確保された領域には、マイコン１，２がシステム１，２を動作させた場合と同じデータが記憶される。

【００２９】
ＣＰＵをエミュレートする場合、ハイパーバイザ３２には、ＣＰＵが持つレジスタ（汎用レジスタ、プログラムカウンタ、フラグレジスタ、スタックポインタ）に対応した領域をマイコン１００のＲＡＭ１２に確保する処理が記述されている。また、ハイパーバイザ３２には、エミュレートしたメモリから命令を１つずつ読み出す、解読する、実行する、という処理の繰り返し、及び、実行結果に応じてＲＡＭ１２に確保したレジスタの値を更新する処理が記述されている。解読する際には、マイコン１，２のＣＰＵが実行可能な全ての命令のうちどの命令に相当するかを特定する。そして解読結果に応じて、確保されたレジスタを操作する処理が記述されている。実行の内容としては、メモリからレジスタへのデータのコピー（例えばＬｏａｄ命令の場合）、レジスタからメモリへのデータのコピー（例えばＳｔｏｒｅ命令の場合）、レジスタの値の加減乗除、論理演算、等がある。したがって、ＲＡＭ１２に確保されたレジスタには、マイコン１，２がシステム１，２を動作させた場合にマイコン１，２のＣＰＵのレジスタに格納されるデータと同じデータが記憶される。

【００３０】
Ｉ／Ｏポートをエミュレートする場合、ハイパーバイザ３２には、入出力が指示されたＩ／Ｏポートに応じた処理が記述されている。すなわち、解読の結果、マイコン１，２のＩ／Ｏポートを操作する命令である場合、マイコン１００のＩ／Ｏポートにアクセスする処理が記述されている。例えば、入力ポートの指定に対し、その入力ポートに対応するＩ／Ｏポート２０からデータを読み出す処理が記述されている。よって、例えばマイコン１，２でセンサが接続されていた入力ポートが指定された場合、その入力ポートに接続されていたセンサが接続されているマイコン１００のＩ／Ｏポート２０からデータを読み出す、処理が記述されている。

【００３１】
出力ポートの指定に対し、その出力ポートにデータを書き込む、処理が記述されている。よって、例えばマイコン１，２でアクチュエータが接続されていた出力ポートが指定された場合、その出力ポートに接続されていたアクチュエータが接続されているマイコン１００のＩ／Ｏポート２０にデータを書き出す、処理が記述されている。

【００３２】
なお、本実施例ではマイコン１，２がメモリマップドＩ／Ｏ（Ｉ／Ｏポートをアドレスに割り当ててアクセスするＩ／Ｏ指定方式）によりＩ／Ｏポート２０にアクセスしている。したがって、上記した入力ポート又は出力ポートへのアクセス命令を検出することは、アクセス先のＩ／Ｏポート２０に対応するアドレス（解読した命令のアクセス先のアドレス（オペランド）を検出することで行われる。

【００３３】
システム１，２のようにＯＳも含めて１つのマイコン１００に搭載する場合、ＯＳ１，２が実行する命令がＣＰＵ１１の特権モードでのみ実行が許可される命令の場合がある。特権モードとはＣＰＵ１１の動作モードであり、対比されるモードにユーザモードがある。一般的なアプリはシステムの安定性やセキュリティを維持するためユーザモードでのみ実行が許可される。

【００３４】
しかし、マイコン１００のハイパーバイザ３１上で実行されるＯＳ１，２はユーザモードで実行されているので、ハイパーバイザ３１がＯＳ１、２の実行をエミュレートできないおそれがある（そのまま実行すると例外が発生してしまう）。このため、ハイパーバイザ３１はＯＳ１，２の特定の命令を、ユーザモードのまま実行できるように別の命令に変換する。

【００３５】
特権モードからユーザモードへは、マイコン１００に用意されている特定の命令を実行することで変更される。逆に、ユーザモードから特権モードへは、ハイパーバイザ３１が特殊な命令を実行した場合に変更されるので、システム１，２が特権モードに切り替える

ことはできない。

【００３６】

この他、マイコン１，２が搭載するＣＡＮコントローラによる通信も、ハイパーバイザ３１はマイコン１，２をエミュレートすることができる。すなわち、ハイパーバイザ３１には、命令の解読によりＣＡＮコントローラによる通信を検出し、マイコン１００のＣＡＮコントローラで通信する処理が記述されている。

【００３７】

ハイパーバイザ３１は、システム１，２をアプリケーションとして実行するので、システム１，２を１つのタスクやスレッドなどの実行単位として、実行管理する。したがって、少なくともＣＰＵ１１がどちらのシステムを実行しているかを検出している。ＣＰＵ１１がシングルコアであれば、ハイパーバイザ３１は例えば時分割的にシステム１と２を切り替えてＣＰＵ１１に割り当てる。ＣＰＵ１１がマルチコアであれば、ハイパーバイザ３１は例えばコア１とコア２にそれぞれシステム１と２を割り当てる。

【００３８】

〔メモリ保護〕

このように、仮想化技術によりシステム１，２を変更することなくハイパーバイザ３１上でシステム１，２が動作する。マイコン１００は例えばベースアドレス指定によりシステム１，２を再配置可能にＲＡＭにロードする。再配置可能なプログラムが指定するアドレスを論理アドレスという。これに対しＲＡＭ上におけるシステム１，２のアドレスを物理アドレスという。システム１，２のＲＡＭのアドレスは不定であるが、システム１，２の命令に含まれるアドレスは固定なので、マイコン１００は、メモリ保護機構３２を用いて、システム１，２が出力する論理アドレスとＲＡＭの物理アドレスを対応づける。

【００３９】

図４は、メモリ保護機構３２の機能を模式的に説明する図の一例である。ハイパーバイザ３１はシステム１、２の起動を特定の命令の実行などから検出して、システム毎にページテーブルエントリ（ＰＴＥ）とページテーブルを作成する。メモリ保護機構３２は、ＲＡＭなどの物理アドレスをページ単位に分割し、論理アドレスのアドレス空間を同じサイズのページ単位に分割する。そして、物理ページのページ番号と論理ページのページ番号を対応付ける。

【００４０】

ＰＴＥには、論理ページ番号に対応づけてページテーブルのアドレス、無効フラグ、Ｒ／Ｗフラグ、及び、Ｕ／Ｈフラグなどが登録されている。ページテーブルのアドレスは、システム別に用意されたページテーブルの先頭アドレスである。無効フラグは対応する物理ページの有無を示す。Ｒ／Ｗフラグはリード／ライトの許可を示す。Ｕ／Ｈフラグは、ユーザモードと特権モード（ハイパーバイザモード）のどちらでアクセスが許可されているかを示す。

【００４１】

論理アドレスの先頭の所定数のビットが論理ページ番号で、残りがページ内のオフセットである。メモリ保護機構３２は、ＰＴＥを参照してページテーブルを参照し、システム１，２の命令に含まれる論理ページ番号に対応づけられた物理ページ番号を特定する。この物理ページ番号とオフセットを統合したアドレスが物理アドレスになる。システム毎にページテーブルが異なるので、論理アドレスが同じでも、物理アドレスが重複することがない。

【００４２】

システム１，２が論理アドレスをメモリ保護機構３２に出力すると、メモリ保護機構３２は実行中のシステムを特定して、システムに固有のページテーブルを参照する。よって、システム１，２はＲＡＭのシステム１，２の領域にアクセスすることができる。Ｉ／Ｏポート２０もアドレス空間の一部に割り当てられるので、ＲＡＭと同様にアクセスできる。

【００４３】

また、システム１がＲＡＭにマッピングされていない論理アドレスを指定した場合、メモリ保護機構３２は例外を発生する。マッピングされているかどうかは無効フラグにより判定される。また、リード／ライトの許可がない物理アドレスに変換される論理アドレスを指定した場合、特権モードでのみアクセスが許可される物理アドレスに変換される論理アドレスを指定した場合も同様である。このように、メモリ保護機構３２によりシステム１、２がアクセスするメモリが保護される。

【００４４】

そして、システム１，２がＩ／Ｏポート２０にアクセスする場合、Ｉ／Ｏポート２０に割り当てられた論理アドレスを指定する。従来であればメモリ保護機構３２はこの論理アドレスも物理アドレスに変換するが、本実施例では、システム１，２がＩ／Ｏポート２０に直接、アクセスすることが禁止される。

【００４５】

図５は、システム１，２によるＩ／Ｏポート２０へのアクセスを禁止する際のＰＴＥの設定例を示す図である。フラグ＝“１”で許可であるとする。

・Ｒ／Ｗフラグ＝１／１

・Ｕ／Ｈフラグ＝０／１

Ｒ／Ｗフラグによれば、いずれも“１”なので、このＰＴＥが指示する論理ページに対応するＩ／Ｏポート２０に対しリードすることもライトすることも可能である。しかしながら、Ｕ／Ｈフラグが“０／１”なので、ユーザモードではリード／ライトが許可されない。すなわち、システム１，２はＩ／Ｏポート２０にアクセスすることはできない。一方、特権モードではリード／ライトが許可されるので、ハイパーバイザ３１はＩ／Ｏポート２０にアクセスすることができる。

【００４６】

したがって、例えばハイパーバイザ３１がマイコン１００の起動時に、論理アドレスがＩ／Ｏポート２０を含む論理ページのＰＴＥについて、図５のように設定することで、システム１，２がＩ／Ｏポート２０にアクセスすることを禁止できる。なお、Ｉ／Ｏポート２０の論理アドレスはシステム１，２の構成から明らかであるし、開発者にとって既知である。

【００４７】

システム１，２がＩ／Ｏポート２０にアクセスする命令を実行した場合、メモリ保護機構３２は例外を発生する（この例外の発生をアクセス違反という場合がある。）。これにより、ハイパーバイザ３１はシステム１，２によるＩ／Ｏポート２０へのアクセスを検出する。

【００４８】

このように本実施例ではページ単位でＩ／Ｏポート２０にアクセス制限すればよいので（全てのＩ／Ｏポートをアクセス禁止にすればよいので）、ページテーブルエントリの数が多くなりすぎることがない。

【００４９】

〔機能構成〕

図６は、ハイパーバイザ３１の機能ブロック図の一例を示す。ハイパーバイザ３１は主にアクセス制御部３３、Ｉ／Ｏアクセス管理情報３４、及び、システム実行情報３５を有している。

【００５０】

システム実行情報３５は、アクセス違反が通知された場合に、実行中のシステム（ＯＳ又はアプリ）を特定するための情報である。システム１，２のどちらもＩ／Ｏポート２０へのアクセスが禁止されているので、どちらがアクセスしたか特定する必要があるためである。

【００５１】

図７（ａ）はシステム実行情報３５の一例を示す図である。システム実行情報３５としては、例えば、マイコン１００で実行中のシステムの識別情報、例外を発生させたプログ

ラムのアドレスが挙げられる。マイコン１００で実行中のシステムの識別情報は、ハイパーバイザ３１が定常的に管理している。ハイパーバイザ３１は、ＣＰＵ１１に割り当てているシステムをテーブルなどに登録してあるので、このテーブルを参照することでシステムを特定できる。

【００５２】

例外を発生させたプログラムのアドレスは、エミュレートされたプログラムカウンタに設定されている。プログラムカウンタの値は論理アドレスなので、ページテーブルを参照して物理アドレスを特定すれば、それがシステム１の領域かシステム２の領域かを判別できる。プログラムカウンタに設定されているアドレスは、ハイパーバイザ３１がエミュレートにより取得しているので容易に特定できる。そして、プログラムカウンタに設定されているアドレスから、システム１，２が実行した命令が分かるので、システム１，２がＩ／Ｏポート１又はＩ／Ｏポート２のどちらにアクセスしたかが判明する。

【００５３】

なお、メモリ保護機構は、論理アドレスから物理アドレスを出力するので（アドレスに関する変換を行うので）、例外を発生させたプログラムの論理アドレス、アクセス違反が発生した時のＩ／Ｏポート２０の物理アドレス、システムが書き込む予定のデータ（又はそのアドレス）が設定される。したがって、メモリ保護機構からシステム実行情報と同等の情報を取得することもできる。

【００５４】

Ｉ／Ｏアクセス管理情報３４は、各Ｉ／Ｏポート１，２に対する各システムのアクセス権が登録された情報である。すなわち、メモリ保護機構３２では禁止されているが、統合前のシステムではアクセスが許可されていたＩ／Ｏポート１，２がシステムに対応づけて登録されている。

【００５５】

図７（ｂ）はＩ／Ｏアクセス管理情報３４の一例を示す図である。図示するように、例えば、Ｉ／Ｏポート１にはシステム１が対応づけられ、Ｉ／Ｏポート２にはシステム２が対応づけられている。Ｉ／Ｏアクセス管理情報３４は静的な情報なので、開発者等がハイパーバイザ３１のビルド時に設定しておくことができる。また、マイコン１００の起動時に、ハイパーバイザ３１が動的に設定してもよい。この方法は、システム１，２がＩ／Ｏポート１，２のどちらを使用してもよい場合に採用できる。例えばシステム１，２の起動順にＩ／Ｏポート１，２を割り当てる。

【００５６】

アクセス制御部３３は、ハイパーバイザ３１内部の機能である。すなわち、特権モードで動作する。アクセス制御部３３は、メモリ保護機構３２の例外の発生により起動される。アクセス制御部３３は、例外発生時のプログラムカウンタの値により実行された命令を特定し、命令がＩ／Ｏポート２０に対応するアドレスへのアクセスによるものであるか否かを判定する。

【００５７】

例外がＩ／Ｏポート２０へのアクセスにより発生した場合、アクセス制御部３３はアクセス制御を行う。すなわち、システム実行情報３５に基づき例外が発生した時のシステムを特定し、Ｉ／Ｏアクセス管理情報３４において、該システムのアクセスが許可されているか否かを判定する。アクセスが許可されている場合は、システムの代わりにＩ／Ｏポート２０にアクセスし、アクセスが許可されていない場合は、システムに対しアクセス違反を通知する。

【００５８】

統合前のシステム１、２ではＩ／Ｏポートのアクセス違反は生じないとしてよい。しかし、Ｉ／Ｏポートに限定しなければ、アクセス違反が生じた場合、マイコン１，２に搭載されているメモリ保護機構３２が例外の発生を割込みなどでＯＳに通知する。システム１，２はＯＳ１，２が備える例えば、例外ハンドラ等を実行することで、例外発生に対応した処理を行う。この処理はＯＳ１，２により統合前から決まっている。よって、アクセス

制御部３３としては、システム１，２に例外ハンドラを実行させることで例外の発生をＯＳ１，２に通知できる。具体的には、例外ハンドラが記述されたシステム１，２のアドレスからシステム１，２の命令の解読（エミュレート）を再開する。

【００５９】

また、アクセス制御部３３は、統合の影響によりアクセス違反がどの程度生じているかを管理するため、アクセスの履歴を記録する。

【００６０】

図８は、アクセス履歴テーブルの一例を示す図である。アクセス制御部３３は、例えば、Ｉ／Ｏポート２０にアクセスしたシステム、アクセス時刻、アクセス時に実行された命令の物理アドレス、アクセス先のＩ／Ｏポート１，２、例外発生の有無等を記録する。発生時刻は例えば起動時からの経過時間であるが、絶対時刻でもよい。

【００６１】

このような記録により、開発者等は、システム１，２がどのくらいの頻度でどのＩ／Ｏポート１，２にアクセスするか、及び、周期性などを検証できる。また、システム１とシステム２のアクセスのパターン（例えば、システム１が２回アクセスするとシステム１が１回アクセスする、システム１とシステム２が連続してアクセスする）の有無を検証できる。また、例外が発生した場合には、例えば特定の物理アドレスの命令で例外が起こりやすいこと、例外発生の前後のアクセスのパターン（例えば、システム１又は２が連続してアクセスすると例外が発生する）の有無を検証できる。

【００６２】

〔例外発生によりアクセス制御部３３がＩ／Ｏポートへのアクセスを代替することのメリット〕

ハイパーバイザ３１はシステム１，２が実行する命令を解析しているので、Ｉ／Ｏポート２０へのアクセスを検出した場合に、例外発生を待つことなくＩ／Ｏポート２０にアクセスすることも可能である。この場合、メモリ保護機構３２によりユーザモードのＩ／Ｏポート２０へのアクセスを禁止しない方法と、禁止する方法が考えられる。Ｉ／Ｏポート２０へのアクセスを禁止しない方法では、想定していないプログラムが追加されることなどでＩ／Ｏポート２０へアクセスされるおそれがあるため好ましくない。

【００６３】

Ｉ／Ｏポート２０へのアクセスを禁止する方法では、Ｉ／Ｏポート２０へのアクセスを検出した場合、特権モードに切り替える処理等をハイパーバイザ３２に記述する必要が生じ、ハイパーバイザ３１の修正が必要になってしまう（ハイパーバイザ３１の汎用性が低下してしまう。）。これに対し、例外発生によりアクセス制御部３３がＩ／Ｏポート２０にアクセスする場合、ハイパーバイザ３２に例外処理用のモジュールを追加すればよいので、ハイパーバイザ３３の汎用性が低下しにくいというメリットがある。

【００６４】

したがって、本実施例の手法が好適であるが、Ｉ／Ｏポート２０へのアクセスを検出した場合に、例外発生を待つことなくＩ／Ｏポート２０にアクセスすることも可能である。

【００６５】

〔動作手順〕

図９（ａ）はマイコン１００の動作手順を示すフローチャート図の一例を、図９（ｂ）はマイコン１００の動作を矢印で示す図の一例である。なお、図９（ａ）と図９（ｂ）の括弧付きの番号は互いに対応している。

【００６６】

まず、ハイパーバイザ３１は例えば起動時にユーザモードにおけるＩ／Ｏポート２０へのアクセスを禁止する（Ｓ１０）。この後、システム１，２がハイパーバイザ３１上で実行される。

【００６７】

例えばシステム１がＩ／Ｏポート２０にアクセスする命令を実行したものとする（Ｓ２０）。

【００６８】
メモリ保護機構３２はアクセス違反を検出して例外を発生させる（Ｓ３０）。例外の発生は、ＣＰＵ１１が例外発生時に実行するプログラムの実行によりハイパーバイザ３１に通知される。
【００６９】
ハイパーバイザ３１のアクセス制御部３３は、メモリ保護機構３２からアクセスに関する以下の情報を取得する（Ｓ４０）。
・アクセス違反が発生したＩ／Ｏポートの物理アドレス
・システムが書き込む予定のデータ（又はそのアドレス）
・例外を発生させたプログラムのアドレス
そして、システム実行情報３５又はステップ４０で取得した情報に基づき実行中のシステム及びＩ／Ｏポート２０を特定する（Ｓ５０）。
【００７０】
なお、例外の発生により、ハイパーバイザ３１は少なくとも例外を発生させたシステム１のエミュレーションを停止し、システム１のコンテキスト（エミュレート内容）を退避しておく。これにより、システム１はＩ／Ｏポート２０へのアクセスを行った後は命令を実行しない。
【００７１】
アクセス制御部３３は、特定したシステム１がＩ／Ｏポート２０へのアクセス権を有するか否かを、Ｉ／Ｏアクセス管理情報３４を参照することで判定する（Ｓ６０）。
【００７２】
システム１がＩ／Ｏポート２０へのアクセス権を有している場合（Ｓ６０のＹｅｓ）、アクセス制御部３３はシステム１の代わりにＩ／Ｏポート２０へアクセスする（Ｓ７０）。アクセス制御部３３は、プログラムカウンタに設定されているアドレスから、システムが実行した命令を特定し、その命令を実行する。システムが実行した命令そのものを実行するのではなく、データの記憶元を該命令から特定し、そのデータをＩ／Ｏポート２０に書き込む命令を実行してもよい。これにより、システム１がＩ／Ｏポート２０にアクセスしようとした命令をハイパーバイザ３１が実行できる。
【００７３】
Ｉ／Ｏポート２０へのアクセスが書き込みだった場合、ハイパーバイザ３１はシステム１の代わりにＩ／Ｏポート２０にアクセスした後、コンテキストを復帰させる。これにより、システム１は、例外の発生後の命令から実行を再開できる。
【００７４】
Ｉ／Ｏポート２０へのアクセスが読み出しだった場合、ハイパーバイザ３１はシステム１の代わりにＩ／Ｏポート２０にアクセスした後、ＤＭＡＣからの割込みを待つ。ＤＭＡＣはメモリ保護機構３２が出力した物理アドレスのＩ／Ｏポート２０にアクセスして、Ｉ／Ｏポート２０の受信バッファからデータを読み出しＲＡＭに書き込む。ＤＭＡＣは、Ｉ／Ｏポート２０からＲＡＭへの転送が終了するとＣＰＵ１１に受信割り込みするので、その割り込みによりハイパーバイザ３１は中断したシステム１のコンテキストを復帰させる。システム１は、例外の発生後の命令から実行を再開し、Ｉ／Ｏポート２０から読み出されたデータをＲＡＭから読み出すことができる。
【００７５】
システム１がＩ／Ｏポート２０へのアクセス権を有していない場合（Ｓ６０のＮｏ）、アクセス制御部３３はシステム１にアクセス違反を通知する（Ｓ８０）。すなわち、例外ハンドラを実行させ、実行後、ハイパーバイザ３１は中断したシステム１のコンテキストを復帰させる。これにより、システム１は統合前と同様にアクセス違反を検出できる。
【００７６】
以上のように、本実施例のマイコン１００は、統合前にアクセスするＩ／Ｏポート２０が定められている複数のシステム１，２を１つのマイコン１００に統合した場合、アクセス制御部３３がＩ／Ｏポート２０へのアクセスを制御する。よって、システム１，２のア

クセス違反を防止できる。システム１，２には変更が必要なく、新たなハードウェアの追加も必要ない。また、統合対象のシステムが増加しても、アクセス制御部３３やメモリ保護機構３２の設定を修正するだけで対応することができる。

【実施例２】

【００７７】

本実施例ではシステムの優先度に基づきシステム１，２のアクセスを制御するマイコン１００について説明する。

【００７８】

機能ブロック図は実施例１の図６と同様である。しかし、本実施例のアクセス制御部３３はシステム１，２の優先度に基づき、優先度の高い方のシステムのＩ／Ｏポート２０へのアクセスを優先する。このような優先制御は、異なるシステム１，２が同じＩ／Ｏポート２０にアクセスする場合に有効である。

【００７９】

システム１、２の優先度は統合時に開発者等に既知の場合と、アクセス制御部３３が動的に決定する場合がある。ＡＳＩＬに基づき優先度を決定する場合、ＡＳＩＬ　ＱＭ＜Ａ＜Ｂ＜Ｃ＜Ｄの順に優先度が高い。したがって、開発者等がアクセス制御部３３にシステム１，２の優先度を予め設定しておくことができる。

【００８０】

優先度を動的に決定する場合、例えば、アクセス違反の頻度から決定する。ＡＳＩＬが高いシステムは高度に設計・検証が行われているので、アクセス違反することが少ない。したがって、アクセス違反が少ないシステムは優先度が高いシステムであることが推定される。アクセス制御部３３は、図８のアクセス履歴テーブルを参照して、過去の全て又は所定時間のアクセス違反の頻度に基づき各システムの優先度を動的に決定する。

【００８１】

そして、アクセス制御部３３は、システム１，２の過去のアクセス履歴に基づき、次のアクセスタイミングを予測する。

【００８２】

図１０は、アクセスタイミングの予測を説明する図の一例である。システムが同じＩ／Ｏポート２０にアクセスする周期をアクセス間隔Ｔａ、Ｉ／Ｏポート２０を占有する時間をアクセス時間Ｔｂとする。図示するように、システム１のアクセス間隔はＴａ１、システム２のアクセス間隔はＴａ２，システム１アクセス時間はＴｂ１、システム２のアクセス時間はＴｂ２、である。本実施例では優先度の高いシステム１を優先すればよいので、システム１のアクセス間隔Ｔａ１と、システム２のアクセス時間Ｔｂ２とが特定できていればよい。よって、システム２のアクセス間隔Ｔｂは不定期でもよい。

【００８３】

システム１のアクセス間隔Ｔａ１に規則性があるので、アクセス制御部３３はシステム１のアクセスタイミングを予測できる。すなわち、時刻ｔ１にシステム１がアクセスした場合、次回のアクセスタイミングは「時刻ｔ２＝時刻ｔ１＋アクセス間隔Ｔａ１」、時刻ｔ２にシステム１がアクセスした場合、次回のアクセスタイミングは「時刻ｔ４＝時刻ｔ２＋アクセス間隔Ｔａ１」である。

【００８４】

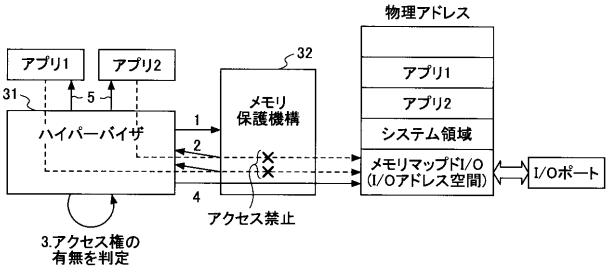
例えば、時刻ｔ３に、優先度が低いシステム２がＩ／Ｏポート２０にアクセスした場合、実施例１にて説明したように、アクセス制御部３３は例外の発生によりアクセス要求を検出する。アクセス制御部３３はシステム２によるＩ／Ｏポート２０の占有が、システム１がアクセスするまでに完了するか否かを判定する。具体的には以下を判定する。

時刻ｔ４＞現在時刻＋アクセス時間Ｔｂ２

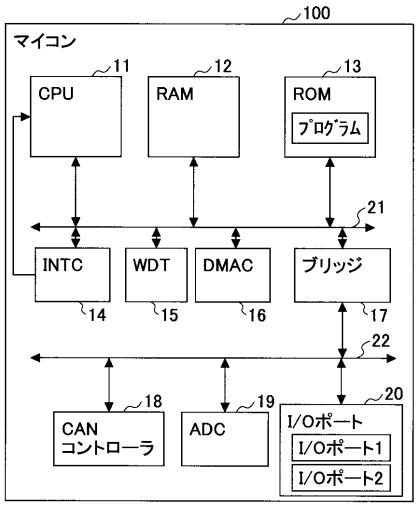
時刻ｔ４の方が、「現在時刻＋アクセス時間Ｔｂ２」よりも大きい場合、システム２によるＩ／Ｏポート２０のアクセスは、システム１がＩ／Ｏポート２０にアクセスを開始するまでに完了する。この場合、アクセス制御部３３はシステム２のアクセス要求を、待機することなく実行する。

【００８５】
時刻ｔ４の方が、「現在時刻＋アクセス時間Ｔｂ２」よりも大きくない場合、システム２によるＩ／Ｏポート２０のアクセスは、システム１がＩ／Ｏポート２０にアクセスを開始するまでに完了しない。この場合、アクセス制御部３３は、システム１のＩ／Ｏポート２０のアクセスを優先するため、システム２のアクセス要求を、システム１のアクセスが完了するまで待機させる。こうすることで、統合によりシステム１，２が同じＩ／Ｏポート２０にアクセスする状況が生じても、優先が高いシステム１を優先できる。
【００８６】
図１１（ａ）はマイコン１００の動作手順を示すフローチャート図の一例を、図１１（ｂ）はマイコン１００の動作を矢印で示す図の一例である。なお、図１１（ａ）と図１１（ｂ）の括弧付きの番号は互いに対応している。
【００８７】
図１１の手順のうちステップＳ１０～Ｓ６０までの処理は図９と同様であるが、本実施例では優先度が低いシステム２がアクセスするものとする。特定したシステム２がＩ／Ｏポート２０へのアクセス権を有していない場合、実施例１と同様にアクセス制御部３３はアクセス違反を通知する（Ｓ８０）。
【００８８】
特定したシステム２がＩ／Ｏポート２０へのアクセス権を有している場合（Ｓ６０のＹｅｓ）、アクセス制御部３３は、アクセス履歴テーブルにより、システム１のアクセス間隔Ｔａ１と、システム２のアクセス時間Ｔｂ２を特定する（Ｓ６２）。アクセス間隔やアクセス時間は、予め更新されていることが好ましいが、アクセス要求がある度に算出してもよい。
【００８９】
アクセス制御部３３は、システム２よりも優先度が高いシステム１が、システム２のアクセス完了までにＩ／Ｏポート２０にアクセスするか否かを判定する（Ｓ６４）。
【００９０】
システム２のアクセス完了までにシステム１がＩ／Ｏポート２０にアクセスしない場合（Ｓ６４のＮｏ）、アクセス制御部３３はシステム２の代わりにＩ／Ｏポート２０へアクセスする（Ｓ８２）。
【００９１】
システム２のアクセス完了までにシステム１がＩ／Ｏポート２０にアクセスする場合（Ｓ６４のＹｅｓ）、アクセス制御部３３はシステム２のアクセスよりもシステム１のアクセスを先に行い、その後、システム２の代わりにＩ／Ｏポート２０へアクセスする（Ｓ８４）。
【００９２】
こうすることで、優先度の異なるシステムが同じＩ／Ｏポート２０を共有する場合、優先度の高いシステムを優先することができ、システム１は統合前と同程度のタイミングでＩ／Ｏポート２０にアクセスすることが可能になる。
【００９３】
以上説明したように、本実施例のマイコン１００によれば、実施例１の効果に加え、優先度が高いシステム１のＩ／Ｏポート２０のアクセスを優先度が低いシステム２よりも優先できる。
【符号の説明】
【００９４】
３１　ハイパーバイザ
３２　メモリ保護機構
３３　アクセス制御部
３４　Ｉ／Ｏアクセス管理情報
３５　システム実行情報
１００　マイコン

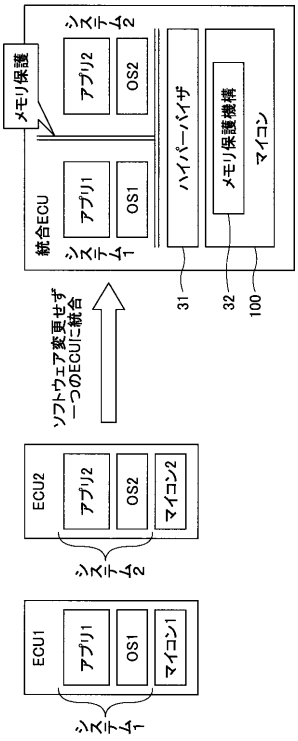
【図１】


物理アドレス
アプリ1
アプリ2
32
メモリ
保護機構
31
5
ハイパーバイザ
1
2
4
アプリ1
アプリ2
システム領域
メモリマップドI/O
(I/Oアドレス空間)
I/Oポート
アクセス禁止
3.アクセス権の
有無を判定

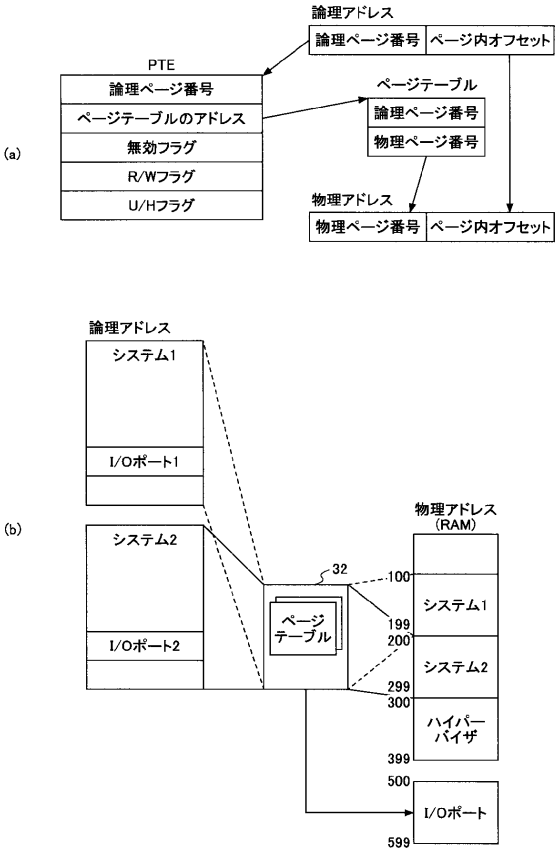
【図２】


100
マイコン
11
CPU
12
RAM
13
ROM
プログラム
21
INTC
WDT
DMAC
ブリッジ
14
15
16
17
22
18
19
20
CAN
コントローラ
ADC
I/Oポート
I/Oポート1
I/Oポート2

【図３】


システム2
メモリ保護
アプリ2
OS2
ハイパーバイザ
メモリ保護機構
マイコン
統合ECU
アプリ1
OS1
システム1
31
32
100
ソフトウェア変更せず
一つのECUに統合
ECU2
アプリ2
OS2
マイコン2
システム2
ECU1
アプリ1
OS1
マイコン1
システム1

【図４】

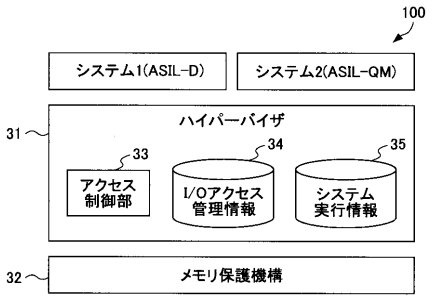

論理アドレス
論理ページ番号
ページ内オフセット
PTE
論理ページ番号
ページテーブルのアドレス
無効フラグ
R/Wフラグ
U/Hフラグ
ページテーブル
論理ページ番号
物理ページ番号
物理アドレス
物理ページ番号
ページ内オフセット
(a)
論理アドレス
システム1
I/Oポート1
システム2
I/Oポート2
(b)
32
ページ
テーブル
物理アドレス
(RAM)
100
199
200
299
300
399
500
599
システム1
システム2
ハイパー
バイザ
I/Oポート

【図５】

論理アドレスがI/Oポートを含む論理ページのPTE

| 論理ページ番号 |
|---|
| ページテーブルのアドレス |
| 無効フラグ |
| R/W=1/1 |
| U/H=0/1 |

【図６】




【図７】

(a)

システム実行情報
・実行中のシステムの識別情報
・例外を発生させたプログラムのアドレス

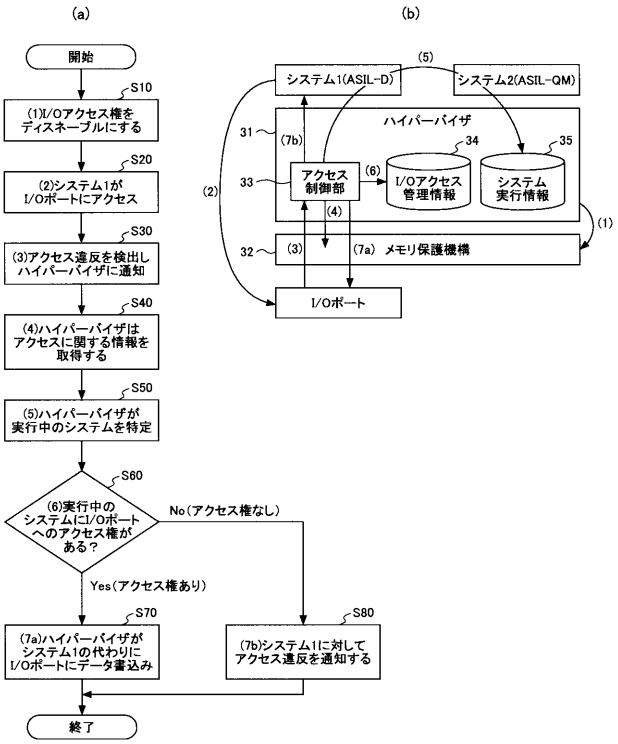
(b)

| I／Oポート | 許可 |
|---|---|
| I／Oポート1の論理アドレス | システム1 |
| I／Oポート2の論理アドレス | システム2 |

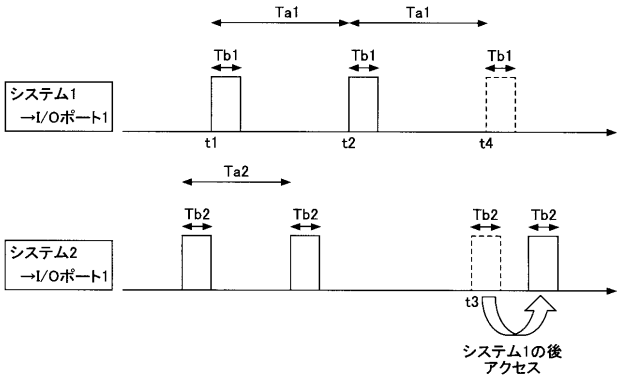
【図８】

| システム | アクセス時刻 | 物理アドレス | I／Oポート | 例外発生 |
|---|---|---|---|---|
| システム1 | 01:13 | 110 | I／Oポート1 | 無 |
| システム2 | 03:17 | 210 | I／Oポート2 | 無 |
| システム1 | 05:55 | 120 | I／Oポート1 | 無 |

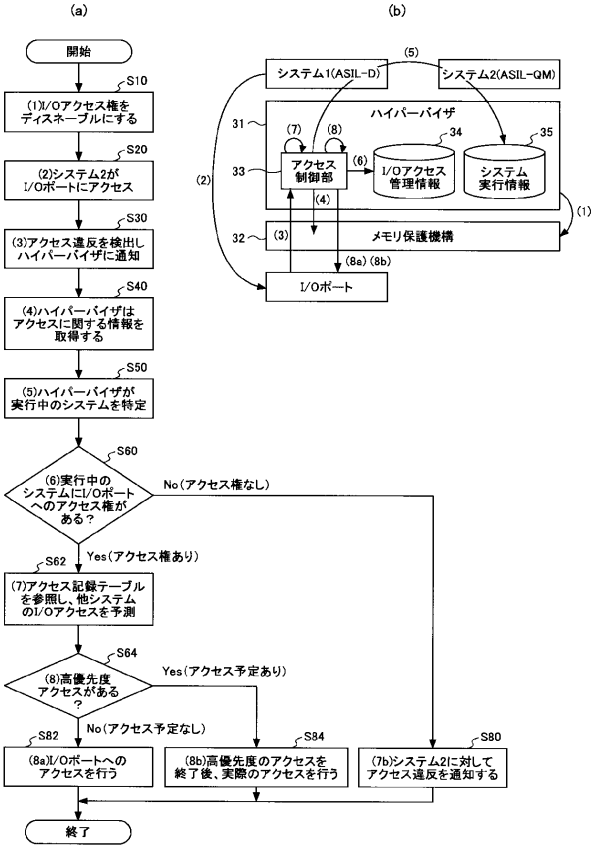
【図９】



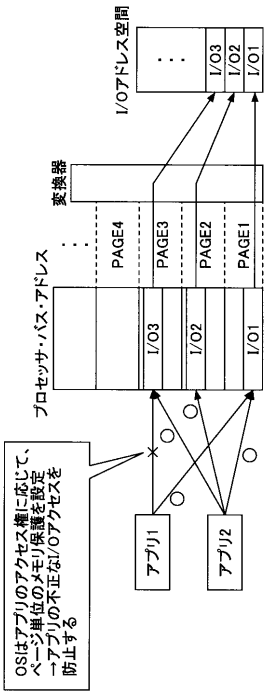

【図１０】



【図１１】


(a)
(b)
開始
S10
(1)I/Oアクセス権をディスエーブルにする
S20
(2)システム2がI/Oポートにアクセス
S30
(3)アクセス違反を検出しハイパーバイザに通知
S40
(4)ハイパーバイザはアクセスに関する情報を取得する
S50
(5)ハイパーバイザが実行中のシステムを特定
S60
(6)実行中のシステムにI/Oポートへのアクセス権がある？
No(アクセス権なし)
Yes(アクセス権あり)
S62
(7)アクセス記録テーブルを参照し、他システムのI/Oアクセスを予測
S64
(8)高優先度アクセスがある？
Yes(アクセス予定あり)
No(アクセス予定なし)
S82
(8a)I/Oポートへのアクセスを行う
S84
(8b)高優先度のアクセスを終了後、実際のアクセスを行う
S80
(7b)システム2に対してアクセス違反を通知する
終了
システム1(ASIL-D)
システム2(ASIL-QM)
ハイパーバイザ
31
(7)
(8)
34
35
33
アクセス制御部
(6)
I/Oアクセス管理情報
システム実行情報
(2)
(4)
(1)
32
(3)
メモリ保護機構
(8a) (8b)
I/Oポート
(5)

【図１２】


I/Oアドレス空間
I/O3
I/O2
I/O1
変換器
PAGE4
PAGE3
PAGE2
PAGE1
プロセッサ・バス・アドレス
I/O3
I/O2
I/O1
アプリ1
アプリ2
OSはアプリのアクセス権に応じて、ページ単位のメモリ保護を設定→アプリの不正なI/Oアクセスを防止する