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

特許第4837247号
(P4837247)

(45)発行日　平成23年12月14日(2011.12.14)　(24)登録日　平成23年10月7日(2011.10.7)

(51)Int.Cl.　F I
G06F 12/08 (2006.01)　G06F 12/08 505B
G06F 9/50 (2006.01)　G06F 12/08 501D
G06F 12/08 565
G06F 9/46 462A
G06F 9/06 640H

請求項の数 1　(全 24 頁)

(21)出願番号　特願2003-331964 (P2003-331964)
(22)出願日　平成15年9月24日 (2003.9.24)
(65)公開番号　特開2005-100034 (P2005-100034A)
(43)公開日　平成17年4月14日 (2005.4.14)
審査請求日　平成18年9月15日 (2006.9.15)

(73)特許権者　000005821
パナソニック株式会社
大阪府門真市大字門真１００６番地
(74)代理人　100077931
弁理士　前田　弘
(74)代理人　100094134
弁理士　小山　廣毅
(74)代理人　100110939
弁理士　竹内　宏
(74)代理人　100113262
弁理士　竹内　祐二
(74)代理人　100115059
弁理士　今江　克実
(74)代理人　100117710
弁理士　原田　智雄

最終頁に続く

(54)【発明の名称】プロセッサ

(57)【特許請求の範囲】
【請求項１】
３つ以上の命令からなる第１のプログラム、第２のプログラムおよび第３のプログラムを実行するプロセッサであって、
前記第１のプログラム、前記第２のプログラムおよび第３のプログラムを格納するメインメモリと、
前記第１のプログラム、前記第２のプログラムおよび第３のプログラムを格納するキャッシュメモリと、
前記第２のプログラムのプリロードを開始する前記第１のプログラム中の第１のアドレスと、前記第３のプログラムのプリロードを開始する前記第２のプログラム中の第２のアドレスとを格納する第１のレジスタと、
前記第１のプログラムを実行中に、実行アドレスが前記第１のレジスタ中の前記第１のアドレスと一致したら、前記メインメモリから前記キャッシュメモリへの前記第２のプログラムのプリロードを起動し、前記第１のプログラムの残りを実行しながら、前記第２のプログラムの実行の前に前記第２のプログラムのプリロードを完了し、前記第２のプログラムを実行中に、実行アドレスが前記第１のレジスタ中の前記第２のアドレスと一致したら、前記メインメモリから前記キャッシュメモリへの前記第３のプログラムのプリロードを起動し、前記第２のプログラムの残りを実行しながら、前記第３のプログラムの実行の前に前記第３のプログラムのプリロードを完了するプリロード手段とを備えたプロセッサ。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は情報処理制御システムに関し、特に、キャッシュメモリに格納する命令を入れ替えながらリアルタイム処理を実現することが必要なシステムを支援するものである。
【背景技術】
【０００２】
図１５に一般的なキャッシュを用いた情報処理制御システムの構成を示す。このシステムは、プログラムの全命令を格納する主記憶メモリ４００Ｈと、必要な命令のみを格納するキャッシュメモリ４０１Ｈと、主記憶メモリおよびキャッシュメモリの制御を行うキャッシュコントローラ４０２Ｈと、キャッシュメモリに格納された命令を実行するプロセッサ４０３Ｈとを備える。
【０００３】
上記情報処理制御システムでは、キャッシュメモリ４０１Ｈ上に必要な命令がある場合は、キャッシュメモリ４０１Ｈに格納されている命令をプロセッサ４０３Ｈが解読して実行する。キャッシュメモリ４０１Ｈ上に必要な命令が無い場合はミスヒットが発生し、プロセッサを止めるかカーネルプログラムでポーリング（待つこと）し、その間に、キャッシュコントローラ４０２Ｈが主記憶メモリ４００Ｈからキャッシュメモリ４０１Ｈへ必要な命令を転送し、キャッシュメモリ４０１Ｈ内の命令が有効になった時点でプロセッサ４０３Ｈがミスヒットの発生した時点の命令を再開する。
【０００４】
上記情報処理制御システムにおいて、キャッシュメモリに格納する命令を入れ替えながらプログラムを実行する場合のプログラムの流れを図１６に示す(例えば、非特許文献１参照)。図１６において、４２０は時間の流れを示し、４２１はKernel programの処理を示し、４２２はApplication programの処理を示す。４２３はKernel programとApplication programとの間を行き来するプログラムの流れを示し、４２４は主記憶メモリからキャッシュメモリへ必要な命令のDown load起動を示し、４２５はキャッシュメモリに必要な命令がなかった場合に発生するMiss hit割り込みを示す。４２０ｔ，４２３ｔ，４２６ｔはプログラムがKernel programからApplication programへ処理が移行する時刻を示し、４２１ｔ，４２４ｔはプログラムがApplication programからKernel programへ処理が移行する時刻を示し、４２２ｔ，４２５ｔは主記憶メモリからキャッシュメモリへ必要な命令のDown load起動時刻を示す。
【０００５】
以下、Kernel programとApplication programとの間を行き来するプログラムの流れ４２３について説明する。Kernel programから起動されるプログラムは時刻４２０ｔでApplication programへ処理が移行する。その後、Application programはMiss hit割り込み４２５が発生するまで実行される。時刻４２１ｔにMiss hit割り込み４２５が発生すると、プログラムはApplication programからKernel programへ処理が移行する。Kernel programでポーリングしている間の時刻４２２ｔに主記憶メモリからキャッシュメモリへ必要な命令のDown loadが起動される。時刻４２３ｔでDown loadが完了すると、Kernel programからApplication programへ処理が移行し、Miss hit割り込み４２５が発生した時点（４２１ｔ）の命令からプログラムが再開される。なお、Miss hit割り込み４２５が発生してからDown loadが完了するまでの処理としては、Kernel programでポーリングしても、Application programで命令を停止していてもどちらでもかまわない。
【０００６】
上記情報処理制御システムにおけるキャッシュコントローラ４０２Hの内部構成を図１７に示す(例えば、非特許文献１参照)。図１７において、４４０はプログラム上の論理アドレスと物理アドレスとの対応を示すCache Table Registerを示し、４４２はTag Addressを示し、４４３は物理Addressを示す。４４１は、Tag Address４４２と物理Address４４３の対応が有効か無効かを示すValid flagを示す。４４４はプログラムの論理Addressの

上位を示し、４４５はプログラムの論理Addressの下位を示す。４４６はCache Addressの上位を示し、４４７はCache Addressの下位を示す。４４７は４４５と同じ信号である。４４９は比較回路を示し、４５０はMiss hit割り込みを発生させるMiss hit flagを示す。

【０００７】

以下、上記回路の動作を説明する。この回路では、プログラムの論理Addressの上位４４４とTag Address４４２を比較回路４４９で比較した結果が一致しておりかつValid flag４４１が有効であれば、物理Address４４３の上位をCache Addressの上位４４６として、さらに、プログラムの論理アドレスの下位４４５をCache Addressの下位４４７として、キャッシュメモリ４０１HにCache Addressとして送る。プログラムの論理Addressの上位４４４とTag Address４４２を比較回路４４９で比較した結果が一致かつValid flag４４１が有効であるものがCache Table Register４４０内に存在しない場合は、Miss hit flag４５０がActiveになり、Miss hit割り込みが発生する。

【非特許文献１】David A.Patterson/John L.Hennessy著「コンピュータの構成と設計 下」日経BP社、1996年4月19日

【発明の開示】

【発明が解決しようとする課題】

【０００８】

以上説明した従来の情報処理制御システムでは、プログラムを実行するために必要な命令がキャッシュメモリ上にない場合、ミスヒット(以降、Miss hitという表現も出てくるが意味は同じ)が発生し、必要な命令がキャッシュメモリ上にDown loadされるまでApplication programが停止するということが頻繁に発生する。そのため、命令を入れ替えながらリアルタイム処理を保証することができないという問題点があった。

【０００９】

さらに、従来の情報処理制御システムでは、あるApplication programでMiss hitが発生した場合、別のApplication programを起動し、その間に、Miss hitが発生したApplication programに必要な命令を主記憶メモリからキャッシュメモリへDown loadするという機構がないため、プロセッサの稼働率を下げてしまうという問題点があった。

【００１０】

さらに、従来の情報処理制御システムでは、Down loadしているDMAの状況やMiss hitの状況を示すflagがCache Table Registerにないため、必要な命令を主記憶メモリからキャッシュメモリへプロセッサが実行する前にあらかじめダウンロードし、さらに、Miss hit割り込み発生時にApplication programを切り替えることにより、命令入れ替えを行いながらリアルタイム処理を実現する情報処理制御システムを構築することができないという問題点があった。

【００１１】

さらに、従来の情報処理制御システムでは、Down loadしている命令のDMA完了を示すflagとしてValid flagしかないので、DMAが完了する度に割り込みが発生してしまい、割り込みによるプログラムの切り替えによるロスが頻繁に発生し、プロセッサの処理効率を下げてしまうという問題点があった。

【００１２】

本発明は以上のような問題点を解消するためになされたもので、命令を入れ替えながらリアルタイム処理を保証する情報処理制御システム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】

【００１３】

本発明の請求項１記載の情報処理制御システムは、オペレーティングシステムを制御するカーネルプログラムおよびアプリケーションを実行するアプリケーションプログラムと、プログラムの全命令を格納する主記憶メモリと、必要な命令のみを格納するキャッシュメモリと、キャッシュメモリに格納された命令を実行するプロセッサからなる情報処理制御システムにおいて、必要な命令を主記憶メモリからキャッシュメモリにプロセッサが実

行する前にあらかじめダウンロードするように制御する仕組みを持つものである。

【００１４】

本発明の請求項２記載の情報処理制御システムは、請求項1記載の情報処理制御システムにおいて、カーネルプログラムおよびアプリケーションプログラムをあるまとまった単位で細分化し、細分化した単位毎に主記憶メモリからキャッシュメモリにプロセッサが実行する前にあらかじめダウンロードするように制御する仕組みを持つものである。

【００１５】

本発明の請求項３記載の情報処理制御システムは、請求項2記載の情報処理制御システムにおいて、さらに、キャッシュ内に必要な命令があるかないかを判定し、必要な命令を主記憶メモリからキャッシュメモリに転送するキャッシュコントローラを備えたシステムにおいて、キャッシュコントローラの状態を監視することにより、プログラム内部に必要な命令を主記憶メモリからキャッシュメモリにプロセッサが実行する前にあらかじめダウンロードする制御手段を持つものである。

【００１６】

本発明の請求項４記載の情報処理制御システムは、オペレーティングシステムを制御するカーネルプログラムおよびアプリケーションを実行するアプリケーションプログラムと、プログラムの全命令を格納する主記憶メモリと、必要な命令のみを格納するキャッシュメモリと、キャッシュ内に必要な命令があるかないかを判定し、必要な命令を主記憶メモリからキャッシュメモリに転送するキャッシュコントローラと、キャッシュメモリに格納された命令を実行するプロセッサからなる情報処理制御システムにおいて、プロセッサへの割り込み手段として、Miss hit割り込みおよびDMA完了割り込みを備え、Miss hit割り込みにより、カーネルプログラムがアプリケーションプログラムを切り替え、DMA完了割り込みにより、カーネルプログラムがMiss hit割り込みが発生したアプリケーションプログラムに復帰させることが出来る仕組みを持つものである。

【００１７】

本発明の請求項５記載の情報処理制御システムは、請求項4記載の情報処理制御システムにおいて、キャッシュコントローラの状態を監視することにより、プログラム内部にアプリケーションプログラム切り替え制御手段を持つものである。

【００１８】

本発明の請求項６記載の情報処理制御システムは、請求項2記載の情報処理制御システム、または、請求項4記載の情報処理制御システムにおいて、細分化された単位に対応した、命令の論理アドレスと物理アドレス対応表と、対応表に有効/無効flag、Miss hit管理flag、DMA管理flagを持つものである。

【００１９】

本発明の請求項７記載の情報処理制御システムは、請求項6記載の情報処理制御システムにおいて、請求項6記載の情報処理制御システムと、Miss hit発生時の命令フェッチプログラムカウンタ値とを用いて、Miss hit割り込み処理を実現するミスヒット割り込み機構を備え、Miss hit割り込み処理により、カーネルプログラムがアプリケーションプログラムを切り替えることが出来る手段を持つ、ものである。

【００２０】

本発明の請求項８記載の情報処理制御システムは、請求項6記載の情報処理制御システムにおいて、Miss hit管理flagとDMA管理flagを用いてDMA完了割り込みを発生するDMA Done割り込み機構を備え、それにより、カーネルプログラムがMiss hit割り込みが発生したアプリケーションプログラムへの復帰を実現できる手段を持つものである。

【００２１】

本発明の請求項９記載の情報処理制御システムは、請求項6記載の情報処理制御システムにおいて、カーネルプログラムおよびアプリケーションプログラムをあるまとまった単位で細分化し、細分化した単位毎に主記憶メモリからキャッシュメモリにプロセッサが実行する前にあらかじめダウンロードするように制御する仕組みを持つものである。

【００２２】

本発明の請求項１０記載の情報処理制御システムは、請求項6記載の情報処理制御システムにおいて、プロセッサへの割り込み手段として、Miss hit割り込みおよびDMA完了割り込みを備え、Miss hit割り込みにより、カーネルプログラムがアプリケーションプログラムを切り替え、DMA完了割り込みにより、カーネルプログラムがMiss hit割り込みが発生したアプリケーションプログラムに復帰させることが出来る仕組みを持つものである。

【００２３】

本発明の請求項１１記載の情報処理制御システムは、請求項6記載の情報処理制御システムにおいて、Miss hit発生時にアプリケーションプログラムを切り替え、その後、発生するDMA完了割り込みをマスクする機構を持つものである。

【００２４】

本発明の請求項１２記載の情報処理制御システムは、請求項6記載の情報処理制御システムにおいて、キャッシュメモリにある命令を解読する命令解読装置と、プロセッサの状態を管理する状態管理装置を備え、プロセッサへの割り込みが発生すると、命令解読装置の出力を無効にし、さらに、状態管理装置をリセットする機構を持つものである。

【発明の効果】

【００２５】

本発明の請求項１にかかる情報処理制御システムによれば、Miss hitの発生頻度を下げることが出来、命令を入れ替えながらリアルタイム処理を実現することができるという効果がある。

【００２６】

本発明の請求項２にかかる情報処理制御システムによれば、Miss hitが発生する頻度を下げることが出来、ミスヒットペナルティーを低減することが出来、それにより、命令を入れ替えながらさらに高速なリアルタイム処理を実現することが出来るという効果がある。

【００２７】

本発明の請求項３にかかる情報処理制御システムによれば、module毎のPre loadを適切な時に適切な数だけ実行することが出来るという効果がある。なお、ここでは、必要な命令を主記憶メモリからキャッシュメモリへプロセッサが実行する前にあらかじめダウンロードすることをPre loadと呼び、Kernel programおよびApplication programをあるまとまった単位に細分化することをmodule化と呼ぶこととする。

【００２８】

本発明の請求項４にかかる情報処理制御システムによれば、命令を主記憶メモリからキャッシュメモリへdown loadすることで発生するミスヒットペナルティを隠蔽し、プロセッサとしての稼働率を上げることが出来るという効果がある。

【００２９】

本発明の請求項５にかかる情報処理制御システムによれば、Pre loadもしくはDown load実行によるミスヒットペナルティを隠蔽することが出来るという効果がある。

【００３０】

本発明の請求項６にかかる情報処理制御システムによれば、module単位でのPre loadや、Miss hit割り込み発生時にApplication programを切り替えながらプログラムを実行することが出来、命令入れ替えを行いながらリアルタイム処理を実現する情報処理制御システムを構築することが出来るという効果がある。

【００３１】

本発明の請求項７にかかる情報処理制御システムによれば、Pre load実行している状態でのMiss hit割り込みとPre load実行していない状態でのMiss hit割り込みのどちらの割り込みも処理することが出来るという効果がある。

【００３２】

本発明の請求項８にかかる情報処理制御システムによれば、DMA完了ごとに発生していたプログラムの切り替えロスを削減でき、プロセッサの処理効率の低下を抑えることが出来、DMA Done割り込みの発生を抑えることが出来、DMA Done割り込みの発生をコントロー

ルすることが出来るという効果がある。

【００３３】

本発明の請求項９にかかる情報処理制御システムによれば、DMA Process flagからPre load実行状態を判定し、module毎のPre loadを適切な時に適切な数だけ実行することが出来るという効果がある。

【００３４】

本発明の請求項１０にかかる情報処理制御システムによれば、Pre loadもしくはDown load実行によるミスヒットペナルティーを隠蔽することが出来、さらに、Miss hit割り込みが発生したApplication programへの復帰を早く実現できるという効果がある。

【００３５】

本発明の請求項１１にかかる情報処理制御システムによれば、別Application programの処理を保護することが出来、また、割り込みマスクを解除した時点で、再び、DMA Done割り込みが発生するので、Miss hit割り込みが発生したApplication programに復帰することが出来るという効果がある。

【００３６】

本発明の請求項１２にかかる情報処理制御システムによれば、無効な命令により状態管理装置が間違った状態に陥ることを防ぐことが出来、割り込み復帰後も命令解読を正常に行うことが出来るという効果がある。

【発明を実施するための最良の形態】

【００３７】

以下、本発明の実施の形態を図面を参照して詳しく説明する。なお、図面において同一部分または相当部分には同一の符号を付してその説明は繰り返さない。

【００３８】

（第１の実施形態）

第１の実施形態による情報処理制御システムの構成を図１に示す。このシステムは、プログラムの全命令を格納する主記憶メモリ４００Hと、必要な命令のみを格納するキャッシュメモリ４０１Hと、主記憶メモリ４００Hおよびキャッシュメモリ４０１Hの制御を行うキャッシュコントローラ４０２Hと、キャッシュメモリ４０１Hに格納された命令を実行するプロセッサ４０３Hとを備える。

【００３９】

以下、図１に示した情報処理制御システムの動作について説明する。ここでは、必要な命令を主記憶メモリ４００Hからキャッシュメモリ４０１Hへプロセッサ４０３Hが実行する前にあらかじめダウンロードすることをPre loadと呼ぶことにする。Pre loadを実現するにはPre loadを実現する仕組みをプログラムに組み込む必要がある。Application program内でのPre loadについては、Application program起動時にPre loadを起動するタイミング(プログラムカウンタ値)をBreak pointとしてレジスタに設定し、プログラムがBreak pointにヒットしたらBreak point割り込みが発生することによりPre loadが起動する方法と、Application program内部にPre load起動プログラムを直接記述することによりPre loadを起動する方法の２種類の方法がある。異なるApplication programに対するPre loadについてはそれぞれのApplication programがお互いのPre loadを起動するタイミングを知らないので、Kernel programがそれぞれのApplication programのPre loadを起動するタイミングをBreak pointとしてレジスタに設定し、プログラムがBreak pointにヒットしたらBreak point割り込みが発生し、Pre loadを起動する方法がある。

【００４０】

図１に示した情報処理制御システムにおいてPre loadを実現する場合のプログラムの流れを図２に示す。図２において、１００は時間の流れを示し、１０１はKernel programの処理を示し、１０２はApplication program Aの処理を示し、１０３はApplication program Bの処理を示し、１０４はKernel programとApplication programを行き来するプログラムの流れを示し、１０５はBreak point割り込みを示し、１０６はPre load起動を示す。１００ｔはPre load起動タイミングをBreak pointとしてレジスタに設定する時刻を示

し、１０３ｔはPre load起動時刻を示し、１０１ｔ，１０４ｔはKernel programからApplication program Aへ処理が移行する時刻を示し、１０２ｔ，１０５ｔはApplication program AからKernel programへ処理が移行する時刻を示し、１０６ｔはKernel programからApplication program Bへ処理が移行する時刻を示す。

【００４１】

以下、Pre loadを実現する際のプログラムの流れを説明する。Kernel program起動後、時刻１００ｔでPre load起動タイミングをBreak pointとしてレジスタに設定する。設定後、時刻１０１ｔでKernel programからApplication program Aへ処理が移行し、Application program Aの処理が始まる。Application program AがBreak pointで設定したプログラムカウンタに到達した時点でBreak point割り込み１０５が発生し、時刻１０２ｔでApplication program AからKernel programへ処理が移行する。Kernel programはBreak point割り込み１０５を受けて、時刻１０３ｔにApplication program Bに対するPre loadを起動する。Pre loadを起動するとKernel programの処理は終わり、時刻１０４ｔにKernel programからApplication program Aに処理が移行する。時刻１０５ｔでApplication program Aの処理がApplication program切り替えタイミングに到達すると、再びKernel programへ処理が移行する。Kernel programは時刻１０６ｔにApplication program Bを起動する。この時点では、Application program Bの命令がキャッシュメモリ４０１H上にPre loadされているので、Miss hit割り込みを発生させることなしに、Application program Bの処理を実行できる。

【００４２】

図２では異なるApplication programに対するPre loadについて説明したが、Application program内でのPre loadにおいても、Pre loadタイミングの設定方法が異なるだけで処理の流れとしては基本的に同じである。また、図２ではApplication programをA/Bの２種類しか記載していないが、Application programの数に制約があるわけではないということはいうまでもない。

【００４３】

このように第１の実施形態によれば、Application program内でのPre loadについては、Application program起動時にPre loadを起動するタイミング(プログラムカウンタ値)をBreak pointとしてレジスタに設定し、Break point割り込みによりPre loadが起動する方法と、Application program内部にPre load起動プログラムを直接記述することによりPre loadを起動する方法を取ることにより、Pre loadを実現し、異なるApplication programに対するPre loadについてはKernel programがそれぞれのApplication programのPre loadを起動するタイミングをBreak pointとしてレジスタに設定しBreak point割り込みにより、Pre loadを起動する方法を取ることによりPre loadを実現する。このようにしてPre loadを実現することにより、Miss hitの発生頻度を下げることができるので、命令を入れ替えながらリアルタイム処理を実現することができる。

【００４４】

(第２の実施形態)

第２の実施形態による情報処理制御システムの構成は図１に示したものと同様である。このシステムは、Pre load処理の単位をmodule化し、module化した単位毎にPre loadを実現する仕組みを持つことを特徴とする。

【００４５】

次に本発明の実施の形態２にかかる情報処理制御システムについて説明する。まず、Kernel programおよびApplication programをあるまとまった単位に細分化することをmodule化と呼ぶこととする。図３は、Kernel programやApplication program等のソフトウェアをmodule化した図である。ソフト構成としてはKernel programの上でApplication programが動作する構成となっている。１２０はmodule化したソフトウェアの全体構成を示し、１２１はKernel program、１２２はApplication program Aを示し、１２３はApplication program Bを示し、１２４はApplication program Cを示し、１２５はApplication program Dを示す。ここで示したApplication programはA/B/C/Dの4種類であるが、Application

programの数に制約があるわけではないということは言うまでもない。Kernel program１２１、Application program１２２～１２５はそれぞれmodule化され、module化された単位毎にPre loadを実現する仕組みを持っている。

【００４６】

図４は本実施の形態２により、module化したプログラムをPre loadする場合のプログラムの流れを示す。Pre load起動方法としては、Application program内でのPre loadについては、Break pointを設定することによりPre loadを起動する方法と、Application program内部にPre load起動タイミングをプログラムに直接記述することによりPre loadを起動する方法の2種類がある。図４では後者を用いて説明することとする。１４０は時間の流れを示し、１４１はKernel programの処理を示し、１４２はApplication program Aの処理を示し、１４３はApplication program Bの処理を示す。１４４はKernel programとApplication programを行き来するプログラムの流れを示し、１４５はPre load起動を示し、１４６はBreak point割り込みを示す。１４７，１４８，１４９はそれぞれApplication program Aのmodule A，module B，module Cの処理区間を示し、１５０，１５１はそれぞれApplication program Bのmodule A，module Bの処理区間を示す。１４０ｔはPre load起動タイミングをBreak pointとしてレジスタに設定する時刻を示し、１４３ｔ，１４７ｔ，１５１ｔ，１５６ｔはPre load起動時刻を示し、１４１ｔ，１４４ｔ，１４８ｔ，１５２ｔはKernel programからApplication program Aへ処理が移行する時刻を示し、１４２ｔ，１４６ｔ，１５０ｔ，１５３ｔはApplication program AからKernel programへ処理が移行する時刻を示し、１５４ｔ，１５７ｔはKernel programからApplication program Bへ処理が移行するタイミングを示し、１５５はApplication program BからKernel programへ処理が移行する時刻を示す。また、１４５ｔはApplication program A内部でmodule Aからmodule Bへ処理が移行する時刻を示し、１４９ｔはApplication program A内部でmodule Bからmodule Cへ処理が移行する時刻を示し、１５８ｔはApplication program B内部でmodule Aからmodule Bへ処理が移行する時刻を示す。

【００４７】

以下、module単位でPre loadを実現する際のプログラムの流れを説明する。Kernel program起動後、時刻１４０ｔでApplication program Bに対するPre load起動タイミングをBreak pointとしてレジスタに設定する。設定後、時刻１４１ｔでKernel programからApplication program Aへ処理が移行し、Application program Aの処理が始まる。Application program Aのmodule A内部に記述しているmodule Bに対するPre load起動命令を発行した時点でApplication program AからKernel programへ処理が移り（時刻１４２ｔ）、時刻１４３ｔにおいてmodule Bに対するPre loadを起動する。Pre loadを起動するとKernel programの処理は終わり、時刻１４４ｔにKernel programからApplication program Aのmodule Aに処理が移行する。時刻１４５ｔに到達した時点で、Application program Aのmodule Aの処理が終わり、module Bの処理が開始される。Application program Aのmodule B内部に記述しているmodule Cに対するPre load起動命令を発行した時点でApplication program AからKernel programへ処理が移り（時刻１４６ｔ）、時刻１４７ｔにmodule Cに対するPre loadを起動する。Pre loadを起動するとKernel programの処理は終わり、時刻１４８ｔにKernel programからApplication program Aのmodule Bに処理が移行する。時刻１４９ｔに到達した時点で、Application program Aのmodule Bの処理が終わり、module Cの処理が開始される。Application program Aのmodule CがBreak pointで設定したプログラムカウンタに到達した時点でBreak point割り込み１４６が発生し、時刻１５０ｔでApplication program AからKernel programへ処理が移行する。Kernel programはBreak point割り込み１４６を受けて、時刻１５１ｔにPre loadを起動する。Pre loadを起動するとKernel programの処理は終わり、時刻１５２ｔにKernel programからApplication program Aに処理が移行する。時刻１５３ｔでApplication program Aの処理がApplication program切り替えタイミングに到達すると、再びKernel programへ処理が移行する。Kernel programは時刻１５４ｔにApplication program Bを起動し、以降、Application program Bの処理が同様に行われる。Application program Aのmodule B，module CおよびApplic

ation program Bのmodule Aの処理が始まる時点では、必要な命令がキャッシュメモリ上にPre loadされているので、Miss hit割り込みを発生させることなしにApplication programの処理を実行できる。

【００４８】

上記説明では述べていないがPre loadについては、moduleによっては、多数のmoduleに対してPre loadの起動を行うものや全くPre loadの起動を行わないものがあってもかまわない。

【００４９】

図４ではApplication programをA/Bの2種類、module をA/B/Cの3種類しか記載していないが、Application programやmoduleの数に制約があるわけではないということはいうまでもない。

【００５０】

このように実施の形態２によれば、Pre load処理の単位をmodule化し、module化した単位毎にPre loadを実現する仕組みを持つことにより、module化する前に比べて、Pre load起動の予測精度があがり、そのことにより、Miss hitの発生頻度を下げることができる。さらに、一回のPre loadにかかるDMA時間も短縮することができるため、仮にMiss hitが発生したとしてもミスヒットペナルティーを低減することができる。このように実施の形態２では命令を入れ替えながらリアルタイム処理を実現することができ、さらに、実施の形態１以上に高速なリアルタイム処理にも対応できる。

【００５１】

(第３の実施形態)

第３の実施形態による情報処理制御システムの構成は図１に示したものと同様である。このシステムは、キャッシュコントローラ４０２H内部のCache Table Registerの情報と、Pre load実行状態の情報を利用することにより、module毎のPre loadを適切な時に適切な数だけ実行することを特徴とする。

【００５２】

次に本発明の実施の形態３にかかる情報処理制御システムについて説明する。図５は本発明の実施の形態３において、Application programのmodule毎にPre loadを実行する際のフローチャートである。このフローではキャッシュコントローラ４０２Hの状態(特にPre load実行状態)からPre loadを起動するかどうかを判定する。ステップ１６０において、Pre loadを起動する場合、ステップ１６１を通って、Application program のmodule AからKernel programのステップ１６２に処理を進める。module内部にPre load起動がない場合は、ステップ１６６のmodule Aの処理を行い、その後、Pre loadを実行しないままステップ１６７のmodule Bの処理へ移行する。ステップ１６２においては、Cache Table Register内部にPre loadする予定のプログラムカウンタ(PC)と一致するTag Addressが存在するかどうかを判定する。ない場合はステップ１６４に処理を進め、Pre load起動を行う。ある場合は、ステップ１６３に処理を進め、ステップ１６２の条件を満たしているものがPre load中かどうかを判定する。Pre load中でない場合は、ステップ１６４に処理を進め、Pre load起動を行う。Pre load中である場合は、なにも実行せず、ステップ１６５に処理を進める。また、ステップ１６４でPre load起動を行った後も、ステップ１６５に処理を進める。ステップ１６５においては全Pre load起動が完了したかどうかを判定する。完了していない場合は、ステップ１６２に戻り、上記処理を繰りかえす。完了した場合は、ステップ１６６に処理を進め、module Aの残りの処理を行い、その後、ステップ１６７のmodule Bの処理を行う。

【００５３】

このように実施の形態３によれば、キャッシュコントローラ４０２H内部のCache Table Registerの情報と、Pre load実行状態の情報を利用することにより、module毎のPre loadを適切な時に適切な数だけ実行することができる。

【００５４】

(第４の実施形態)

第４の実施形態による情報処理制御システムの構成は図１に示したものと同様である。このシステムは、プロセッサ４０３Ｈへの割り込み手段として、Miss hit割り込みおよびDMA Done割り込みを備えることにより、命令がMiss hitした場合にアプリケーションプログラムを切り替えながら実行できる点を特徴とする。

【００５５】

次に本発明の実施の形態４にかかる情報処理制御システムについて説明する。図６は本実施の形態４により、Miss hit割り込みとDMA完了割り込み(以降、DMA Done割り込みと呼ぶ)を備えることにより、命令のMiss hitが発生した場合、Application programを切り替えながら、プログラムを実行する際のプログラムの流れを示す。１８０は時間の流れを示し、１８１はKernel programの処理を示し、１８２はApplication program Aの処理を示し、１８３はApplication program Bの処理を示し、１８４はKernel programとApplication programを行き来するプログラムの流れを示す。１８５はdown load起動を示し、１８６はMiss hit割り込みを示し、１８７はDMA Done割り込みを示す。DMA Done割り込みはdown loadが完了した際に発生するものである。１８０ｔ，１８５ｔ，１９０ｔはKernel programからApplication program Aへ処理が移行する時刻を示し、１８１ｔ，１８６ｔはApplication program AからKernel programへ処理が移行する時刻を示し、１８３ｔ，１８８ｔはKernel programからApplication program Bへ処理が移行する時刻を示し、１８４ｔ，１８９ｔはApplication program BからKernel programへ処理が移行する時刻を示す。

【００５６】

以下、命令のMiss hitが発生した場合、Application programを切り替えながらプログラムを実行する際のプログラムの流れを説明する。Kernel program起動後、時刻１８０ｔでKernel programからApplication program Aへ処理が移行し、Application program Aの処理が始まる。キャッシュメモリ４０１Ｈ上に必要な命令がなくなるとMiss hit割り込みが発生する。時刻１８１ｔでMiss hit割り込み１８６が入った場合、Application program AからKernel programへ処理が移行する。時刻１８２ｔにMiss hit割り込みを受けて、Application program Aに対する命令のdown loadを起動する。その後、Kernel programが待ち状態にあるApplication programを検出すると、Kernel programはApplication programの切り替えを行う。Application program Bが待ち状態にあった場合、Kernel programがApplication program Bを起動する(時刻１８３ｔ)。Application program B実行中もdown loadは実行されており、down loadが完了した時点でDMA Done割り込み１８７を発生し、Application program BからKernel programへ処理が移行する(時刻１８４ｔ)。DMA Done割り込み１８７がApplication program Aに対するものであった場合は、Kernel programがApplication program Aを起動し、Miss hit割り込み１８６が発生した時点の命令から再開される(時刻１８５ｔ)。DMA Done割り込み１８７については各Application program毎、または、module毎に割り込み要因が存在するものとする。

【００５７】

図６では２種類のApplication programの切り替えについて説明したが、Application programが複数有り、Miss hit割り込み後、Application programを切り替え、さらに、切り替え先のApplication programでMiss hit割り込みが発生したとしても基本的に同じ処理となる。また、図６ではApplication programをA/Bの2種類しか記載していないが、Application programの数に制約があるわけではないということはいうまでもない。

【００５８】

このように実施の形態４によれば、プロセッサ４０３Ｈへの割り込み手段として、Miss hit割り込み１８６およびDMA Done割り込み１８７を備えることにより、命令がMiss hitした場合にアプリケーションプログラムを切り替えながら実行できる。特にDMA Done割り込み１８７についてはApplication program毎、module毎に割り込み要因を持つことで、Miss hit割り込み１８６が発生した後、Application programが切り替わっても、DMA Done割り込み１８７が発生した際に、元のApplication programに戻ることができる。Miss hit発生した際にApplication programの切り替えを行うことで命令を主記憶メモリ４００Ｈ

からキャッシュメモリ４０１Ｈへdown loadすることで発生するミスヒットペナルティを隠蔽し、プロセッサとしての稼働率を上げることができる。

【００５９】

(第５の実施形態)

第５の実施形態による情報処理制御システムの構成は図１に示したものと同様である。このシステムは、キャッシュコントローラ４０２Ｈ内部のCache Table Registerの情報と、Pre load実行状態の情報を利用することにより、Down loadを起動すべきかどうかを判定することを特徴とする。

【００６０】

次に本発明の実施の形態５にかかる情報処理制御システムについて説明する。図７は本発明の実施の形態５において、Miss hitが発生した場合にApplication programを切り替えながら実行する際のフローチャートである。このフローではキャッシュコントローラ４０２Ｈの状態(特にPre load実行状態)からdown loadを起動するかどうかを判定する。また、Miss hitが発生した際に待ち状態にあるApplication programがあるかどうかを判定したう上で、Application programの切り替えを実行する。図７において、２００Ｈ，２０１Ｈはハードウェアの処理である。例えば、Application program Aのmodule A処理中に２００Ｈでハードウェアが Miss hit割り込みを検出すると、ステップ２００のApplication program Aのmodule A処理からKernel programへ処理が移行し、ステップ２０１へ進む。ステップ２０１においては、Cache Table Register内部にDown loadする予定のプログラムカウンタ(PC)と一致するTag Addressが存在するかどうかを判定する。ない場合はステップ２０３に処理を進め、Down load起動を行う。ある場合は、ステップ２０２に処理を進め、ステップ２０１の条件を満たしているものがPre load中かどうかを判定する。Pre load中でない場合は、ステップ２０３に処理を進め、Down load起動を行う。Pre load中である場合は、なにも実行せず、ステップ２０４に処理を進める。また、ステップ２０３でDown load起動を行った後も、ステップ２０４に処理を進める。ステップ２０４においては待ち状態にあるApplication programがあるかどうかを判定し、ある場合はApplication programの切り替えを行い、ステップ２０６に進み、別のApplication programを実行し、ない場合はKernel program内のステップ２０５でDown load完了までポーリングする。Down loadが完了し、２０１Ｈでハードウェアが DMA Done割り込みを検出すると、ステップ２０５、ステップ２０６からKernel program内のステップ２０７のDMA Done割り込み処理へ移行する。ステップ２０７でDMA Done割り込み処理を行った後、元のApplication program Aのmodule Aの処理に戻り、Miss hit割り込みが発生した時点の命令からプログラムを再開する。

【００６１】

このように実施の形態５によれば、キャッシュコントローラ４０２Ｈ内部のCache Table Registerの情報と、Pre load実行状態の情報を利用することにより、Down loadを起動すべきかどうかを判定することができる。また、Pre loadもしくはDown load起動後に、待ち状態にあるApplication programがあるかどうかを判定し、Application program切り替えを実行することで、Pre loadもしくはDown load実行によるミスヒットペナルティを隠蔽することができる。

【００６２】

(第６の実施形態)

第６の実施形態による情報処理制御システムの構成は図１に示したものと同様である。このシステムは、module単位でDMA実行状況やMiss hit発生状況がわかる仕組みになっている点を特徴とする。

【００６３】

次に本発明の実施の形態６にかかる情報処理制御システムについて説明する。図８は本実施の形態６においてキャッシュコントローラ４０２Ｈの内部構成を示すブロック図である。図８において、２２０はプログラム上の論理Addressと物理Addressとの対応を示すCache Table Registerを示し、２２４はTag Addressを示し、２２５は物理Addressを示す。

２２１はTag Address２２４と物理Address２２５の対応が有効か無効かを示すValid flagを示す。Cache Table Register２２０はmodule単位での対応表として構成されている。２２２はTag Address２２４と物理Address２２５に対応したmoduleにおいて、Miss hitが発生しているかどうかを示すMiss hit History flagを示し、２２３はTag Address２２４と物理Address２２５に対応したmoduleにおいて、Pre loadもしくはDown loadのDMAが実行中であるかどうかを示すDMA Process flagを示す（今後、Tag Address、物理Address、Valid flag、Miss hit History flag、DMA Process flagのペアをブロックと呼ぶ。Cache Table Register２２０は複数のブロックから構成されていることとなる。）。２２６はプログラムの論理Addressの上位を示し、２２７はプログラムの論理Addressの下位を示す。２２８はCache Addressの上位を示し、２２９はCache Addressの下位を示す。２２９は２２７と同じ信号である。２３１は比較回路を示し、２３２はMiss hit割り込みを発生させるMiss hit flagを示す。Cache Table Register２２０はプログラムから書き込みも読み出しも可能な構成となっている。さらに、Valid flag２２１はDMAが完了した時点でセットされ、DMA Process flag２２３はDMAが完了した時点でリセットされる仕組みにしている。

【００６４】

以下、図８に示した回路の動作を説明する。この回路ではプログラムの論理Addressの上位２２６とTag Address２２４を比較回路２３１で比較した結果が一致しておりかつValid flag２２１が有効であれば、物理Addressの上位２２５をCache Addressの上位２２８として、さらに、プログラムの論理Addressの下位２２７をCache Addressの下位２２９として、キャッシュメモリ４０１HにCache Addressとして送る。Cache Table Register２２０内に、プログラムの論理Addressの上位２２６とTag Address２２４を比較回路２３１で比較した結果が一致かつValid flag２２１が有効であるものが存在しない場合は、Miss hit flag２３２がActiveになり、Miss hit割り込みが発生する。また、Miss hit割り込みが発生した際にMiss hit History flag２２２をセットすれば、Cache Table Registerのどこで Miss hit割り込みが発生しているかがわかる仕組みになっている。さらに、DMA Process flag２２３はDMA完了時点で自動的にリセットされる仕組みになっているため、Pre loadやDown loadのDMA起動時にセットしておけば、module単位でのDMA実行状況がわかる仕組みになっている。

【００６５】

このように実施の形態６によれば、module単位でDMA実行状況やMiss hit発生状況がわかる仕組みになっているため、module単位でのPre loadや、Miss hit割り込み発生時にApplication programを切り替えながらプログラムを実行することができ、命令入れ替えを行いながらリアルタイム処理を実現する情報処理制御システムを構築することができる。

【００６６】

(第７の実施形態)

第７の実施形態による情報処理制御システムは第６の実施形態を発展させたものである。このシステムは、Miss hit割り込み発生時にMiss hit割り込み発生時の命令フェッチプログラムカウンタ値(以降、EIPCと呼ぶ)と、命令デコードプログラムカウンタ値(以降、EPCと呼ぶ)を自動的に退避する仕組みを持っている。図９は図８に示した本発明の実施の形態６にかかる情報処理制御システムと、EIPCと、DMA process flagを用いて、Miss hit割り込み処理を実現する際のフローチャートである。図９において、２４０Hはハードウェア（プロセッサ４０３Hまたはキャッシュコントローラ４０２H）の処理である。Application program Aのmodule A実行中に、２４０Hにおいてハードウェアが Miss hit割り込みを検出すると、自動的にEIPCとEPCが退避され、ステップ２４０のApplication program Aのmodule Aの処理からKernel programへ処理が移行し、ステップ２４１に進む。ステップ２４１においてはCache Table Register内部にEIPCと一致するTag Addressが存在するかどうかを判定する。ない場合はステップ２４３に処理を進め、Cache Table Registerを任意に選択する。ある場合は、ステップ２４２に処理を進め、ステップ２４１の条件が成立しているブロックにおいて、DMA Process flagが1かどうかを判定する。DMA Process

flagが１でない場合はPre load実行中でないと判断し、ステップ２４３に処理を進め、Cache Table Registerを任意に選択する。DMA Process flagが１である場合は、Pre load実行中と判断し、なにも実行せず、ステップ２４５に処理を進める。また、ステップ２４３でCache Table Registerを任意に選択する処理に移った場合は、その後、ステップ２４４へ進む。ステップ２４４では、選択されたブロックのValid flagを0に設定し、DMA Process flagを１に設定し、Tag Addressと物理Addressを設定し、Down load起動を行う。その後、ステップ２４５に進む。ステップ２４５では、ステップ２４２からステップ２４５に移行してきた場合は、ステップ２４２の条件が成立しているブロックのMiss hit History を１に設定し、ステップ２４４からステップ２４５に移行してきた場合は、ステップ２４３で選択されたブロックのMiss hit History を１に設定する。ステップ２４５の処理が終わると、ステップ２４６に進む。ステップ２４６においては待ち状態にあるApplication programがあるかどうかを判定する。ある場合はApplication programを切り替え、ステップ２４８の別のApplication programを実行する、ない場合はKernel program内のステップ２４７でDown load完了までポーリングする。

【００６７】

このように実施の形態７によれば、本発明の実施の形態６にかかる情報処理制御システムと、EIPCと、DMA process flagを用いて、Miss hit割り込み処理を実現することにより、Pre load実行している状態でのMiss hit割り込みとPre load実行していない状態でのMiss hit割り込みのどちらの割り込みも処理することができる。

【００６８】

(第８の実施形態)

第８の実施形態による情報処理制御システムの構成は図１に示したものと同様である。図１０は本実施の形態８におけるキャッシュコントローラ４０２Ｈの内部構成を示すブロック図である。図１０において、図８と同一符号は同一または相当部分を示し、２３３はDMA Done割り込みflagを示す。DMA Done割り込みflag２３３はMiss hit History flag２２２を１に設定しているブロックに対してDMA完了した場合のみ発生する構成にしている。まず、DMA Process flag２２３を１に設定し、次にPre loadまたは、Down loadのDMAを起動し、その後、Miss hit History flag２２２を１に設定すると、DMA完了時点で自動的にDMA Process flag２２３がリセットされ0になるので、DMA Done割り込みが発生する仕組みにしている。

【００６９】

このように実施の形態８によれば、Pre loadによるDMA完了時など、DMAが完了する度にDMA Done割り込みがすることはなくなるので、DMA完了ごとに発生していたプログラムの切り替えロスを削減でき、プロセッサの処理効率の低下を抑えることが出来る。また、仮にMiss hit割り込みが発生した場合でも、DMA Done割り込みを発生させたくない場合などは、Miss hit History flagを０のままにしておくことで、DMA Done割り込みの発生を抑えることが出来、DMA Done割り込みの発生をコントロールすることが出来る。

【００７０】

(第９の実施形態)

第９の実施形態による情報処理制御システムは第８の実施形態の応用例である。図１１は図１０に示した本発明の実施の形態８にかかる情報処理制御システムを用いて、Application programのmodule毎にPre loadを実行する際のフローチャートである。ステップ２８０において、Pre loadを起動する場合、ステップ２８１を通って、Application program のmodule AからKernel programのステップ２８２に処理を進める。module内部にPre load起動がない場合は、ステップ２８７のmodule Aの処理を行い、その後、Pre loadを実行しないままステップ２８８のmodule Bの処理へ移行する。ステップ２８２においては、Cache Table Register２２０内部にPre loadする予定のプログラムカウンタ(PC)と一致するTag Address２２４が存在するかどうかを判定する。ない場合はステップ２８４に処理を進め、Cache Table Register２２０を任意に選択する。ある場合は、ステップ２８３に処理を進め、ステップ２８２の条件が成立しているブロックにおいて、DMA Process flag２

２３が1かどうかを判定する。DMA Process flag２２３が１でない場合はPre load実行中でないと判断し、ステップ２８４に処理を進め、Cache Table Register２２０を任意に選択する。DMA Process flag２２３が１である場合は、Pre load実行中と判断し、なにも実行せず、ステップ２８６に処理を進める。また、ステップ２８４でCache Table Register２２０を任意に選択する処理に移った場合は、その後、ステップ２８５へ進む。ステップ２８５では、選択されたブロックのValid flag２２１を0に設定し、DMA Process flag２２３を１に設定し、Tag Address２２４と物理Address２２５を設定し、Pre load起動を行う。その後、ステップ２８６に進む。ステップ２８６においては全Pre load起動が完了したかどうかを判定し、完了していない場合は、ステップ２８２に戻り、上記処理を繰りかえす。完了した場合は、ステップ２８７に処理を進め、module Aの残りの処理を行い、その後、ステップ２８８のmodule Bの処理を行う。また、２８５の処理を行った時点で、ハードウェアは２８０Ｈにおいて、Valid flag２２１が０に設定され、DMA Process flag２２３が１に設定され、２８１ＨにおいてDMA起動がかかる。２８２ＨでDMAが完了した時点で、自動的にValid flag２２１は１に設定され、DMA Process flag２２３は０に設定される。

【００７１】

このように実施の形態９によれば、本発明の実施の形態８にかかる情報処理制御システムを用いて、DMA Process flag２２３からPre load実行状態を判定し、module毎のPre loadを適切な時に適切な数だけ実行することが出来る。

【００７２】

(第１０の実施形態)

次に本発明の実施の形態１０にかかる情報処理制御システムは、図９に示した本発明の実施の形態７にかかる情報処理制御システムと、図１０に示した本発明の実施の形態８にかかる情報処理制御システムを用いて、Miss hit割り込みが発生した場合にApplication programを切り替えながら実行するものである。この処理のフローチャートを図１２に示す。図１２において、３００Ｈ，３０１Ｈ，３０２Ｈ，３０３Ｈ，３０４Ｈ，３０５Ｈ，３０６Ｈはハードウェアの処理である。Application program Aのmodule A実行中に、３００ＨにおいてハードウェアがMiss hit割り込みを検出すると、自動的にEIPCとEPCが退避され、ステップ３００のApplication program Aのmodule Aの処理からKernel programへ処理が移行し、ステップ３０１に進む。ステップ３０１においてはCache Table Register２２０内部にEIPCと一致するTag Address２２４が存在するかどうかを判定する。ない場合はステップ３０３に処理を進め、Cache Table Register２２０を任意に選択する。ある場合は、ステップ３０２に処理を進め、ステップ３０１の条件が成立しているブロックにおいて、DMA Process flag２２３が1かどうかを判定する。DMA Process flag２２３が１でない場合はPre load実行中でないと判断し、ステップ３０３に処理を進め、Cache Table Register２２０を任意に選択する。DMA Process flag２２３が１である場合は、Pre load実行中と判断し、なにも実行せず、ステップ３０５に処理を進める。また、ステップ３０３でCache Table Register２２０を任意に選択する処理に移った場合は、その後、ステップ３０４へ進む。ステップ３０４では、選択されたブロックのValid flag２２１を0に設定し、DMA Process flag２２３を１に設定し、Tag Address２２４と物理Address２２５を設定し、Down load起動を行う。その後、ステップ３０５に進む。ステップ３０５では、ステップ３０２からステップ３０５に移行してきた場合は、ステップ３０２の条件が成立しているブロックのMiss hit History flag２２２を１に設定し、ステップ３０４からステップ３０５に移行してきた場合は、ステップ３０３で選択されたブロックのMiss hit History flag２２２を１に設定する。ステップ３０５の処理が終わると、ステップ３０６に進む。ステップ３０６においては待ち状態にあるApplication programがあるかどうかを判定する。ある場合はApplication programを切り替え、ステップ３０８の別のApplication programを実行する、ない場合はKernel program内のステップ３０７でDown load完了までポーリングする。また、３０４の処理を行った時点で、ハードウェアは３０１Ｈにおいて、Valid flag２２１が０に設定され、DMA Process flag２２３が１に設定され

、３０２ＨにおいてDMA起動がかかる。

【００７３】

その後、ステップ３０５の処理を行った時点で、３０３ＨにおいてMiss hit History flag２２２が１に設定され、３０５ＨにおいてDMAが完了した時点で、３０４Ｈにおいて自動的にValid flag２２１が１に設定され、DMA Process flag２２３が０に設定される。Miss hit History flag２２２が１に設定されている状態でDMA Process flag２２３が０になれば、DMA Done割り込みが検出され、ステップ３０７とステップ３０８の処理からKernel programのステップ３０９のDMA Done割り込みルーチンに処理が移行する。その後、ステップ３１０で切り替え先のApplication programのmodule判定を行い、ステップ３１１においてMiss hit History flag２２２を０に設定し、ステップ３１２において、DMA Done割り込みを発生させたApplication programのmoduleへ処理が移行する。

【００７４】

このように実施の形態１０によれば、本発明の実施の形態７にかかる情報処理制御システムと、本発明の実施の形態８にかかる情報処理制御システムを用いてプログラムを制御することにより、Application program切り替えを行いながらプログラムの実行が出来るため、Pre loadもしくはDown load実行によるミスヒットペナルティーを隠蔽することが出来る。また、DMA Done割り込みがmodule単位で必要な命令がキャッシュメモリ上にそろった時点で発生するため、Miss hit割り込みが発生したApplication programへの復帰を早く実現できる。

【００７５】

(第１１の実施形態)

第１１の実施形態にかかる情報処理制御システムは、第６の実施形態にかかる情報処理制御システムと、DMA Done割り込みマスク機構を用いて、DMA Done割り込みをマスクするものである。この情報処理制御システムでは、module毎に発生するDMA Done割り込みをマスクする機構を持ち、この機構については割り込み要因そのものを消してしまうのではなく、割り込み要因自体は保持できるようになっており、割り込みマスクを解除した時点で再び割り込み要因が発生し、割り込みが入る仕組みになっている。図１３は、本発明の実施の形態６にかかる情報処理制御システムと、DMA Done割り込みマスク機構を用いて、DMA Done割り込みをマスクする際のフローチャートを示す。このフローチャートはMiss hit割り込みが発生した後の処理を表す。３２０Ｈ，３２１Ｈ，３２２Ｈはハードウェアの処理である。Miss hit割り込み発生後、Down load起動等を行った後、ステップ３２０において、Miss hit History flagを１に設定する。ステップ３２０の処理が終わると、ステップ３２１に進む。ステップ３２１においては待ち状態にあるApplication programがあるかどうかを判定する。ある場合はApplication programを切り替え、ステップ３２３へ進み、ない場合はKernel program内のステップ３２２でDown load完了までポーリングする。ステップ３２３においては割り込みマスクが必要かどうかを判定する。必要であれば、ステップ３２４において割り込みマスクを設定し、その後、ステップ３２５で別Application programを実行する。必要でない場合は、ステップ３２５に直接進み、別Application programを実行する。また、３２０の処理を行った時点で、ハードウェアは３２０Ｈにおいて、Miss hit History flagが１に設定され、３２４の処理を行った時点で、３２１Ｈにおいて、DMA Done割り込みマスクが設定される。その後、３２２Ｈにおいて、DMA Done割り込みが検出される。ステップ３２２においては、DMA Done割り込みが検出された時点でDMA Done割り込みが発生し、ステップ３２９のDMA Done割り込み処理へ進む。また、ステップ３２５において、DMA Done割り込みマスクが設定されていない場合は、DMA Done割り込みが検出された時点でDMA Done割り込みが発生し、ステップ３２９のDMA Done割り込み処理へ進む。ステップ３２５において、DMA Done割り込みマスクが設定されている場合は、DMA Done割り込みが検出された時点でもDMA Done割り込みが発生せず、ステップ３２７に進み、別Application programの処理が継続される。その後、ステップ３２８で割り込みマスクを解除した時点で、DMA Done割り込みが発生し、ステップ３２９のDMA Done割り込み処理へ進む。

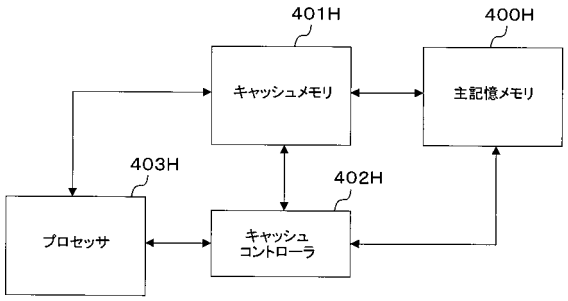
【００７６】
このように実施の形態１１によれば、Miss hit割り込み後のDMA Done割り込みに対して割り込み要因を残したまま割り込みマスクすることで、別Application programの処理を保護することが出来る。また、割り込み要因を残した状態でDMA Done割り込みをマスクするため、割り込みマスクを解除した時点で、再び、DMA Done割り込みを発生し、Miss hit割り込みが発生したApplication programに復帰することが出来る。
【００７７】
(第１２の実施形態)
次に本発明の実施の形態１２にかかる情報処理制御システムについて説明する。
【００７８】
図１４は本発明の実施の形態１２にかかる情報処理制御システムのブロック図である。３４０Ｈは割り込み管理機構を示し、３４１Ｈはプログラムを格納しているメモリを示し、３４２Ｈは命令解読装置、３４３Ｈは状態管理装置を示す。３４０Ｓは命令を示す信号線で、３４１Ｓは３４３Ｈに示した状態管理装置が３４２Ｈに示した命令解読装置を制御する制御線である。３４２Ｓは３４０Ｈに示した割り込み管理機構が３４２Ｈに示した命令解読装置と３４３Ｈに示した状態管理装置を制御するための制御線である。通常のプロセッサにおいては３４３Ｈに示した状態管理装置は存在しない。命令ビットフィールド上に命令以外のもの、たとえば、暗号化情報等を含むような場合は、通常の命令の解読を行う命令解読装置と暗号化情報により状態が変化する状態管理装置が存在することがある。この場合、３４３Ｈの状態管理装置から出力される制御線３４１Ｈにより、３４２Ｈに示した命令解読装置の命令解読方法が変わる。
【００７９】
図１４において、３４０Ｈ割り込み管理機構が割り込みを検出すると３４２Ｈに示した命令解読装置と３４３Ｈに示した状態管理装置に接続されている３４２Ｓに示した制御線がActiveになる。３４２ＳがActiveにすることで、３４２Ｈに示した命令解読装置の出力を無効化し、さらに、３４３Ｈに示した状態管理装置をリセットする。
【００８０】
このように実施の形態１２によれば、割り込み発生時に３４３Ｈに示した状態管理機構をリセットすることにより、無効な命令により状態管理装置が間違った状態に陥ることを防ぐことが出来、割り込み復帰後も命令解読を正常に行うことが出来る。
【産業上の利用可能性】
【００８１】
本発明は、キャッシュメモリに格納する命令を入れ替えながらリアルタイム処理を実現することが必要なシステム、たとえばＭＰＥＧ　ＡＶデコーダやデジタルＴＶ関係のＬＳＩなどに適用可能である。
【図面の簡単な説明】
【００８２】
【図１】第１の実施形態による情報処理制御システムの全体構成を示すブロック図である。
【図２】本発明の第１の実施形態にかかる情報処理制御システムにおいて、Pre load処理をする場合のプログラムの流れを示す図である。
【図３】本発明の第２の実施形態にかかる情報処理制御システムにおいて、module化したソフトウェアの構成図である。
【図４】本発明の第２の実施形態にかかる情報処理制御システムにおいて、module単位でPre loadする場合のプログラムの流れを示す図である。
【図５】本発明の第３の実施形態にかかる情報処理制御システムにおいて、module単位でPre loadする場合のフローチャートを示す図である。
【図６】本発明の第４の実施形態にかかる情報処理制御システムにおいて、Miss hit発生時Application programを切り替えながら実行する場合のプログラムの流れを示す図である。

【図７】本発明の第５の実施形態にかかる情報処理制御システムにおいて、Miss hit発生時Application programを切り替えながら実行する場合のフローチャートを示す図である。
【図８】本発明の第６の実施形態にかかる情報処理制御システムにおいて、Cache Table Registerとキャッシュメモリとの構成を示すブロック図である。
【図９】本発明の第７の実施形態にかかる情報処理制御システムにおいて、Miss hit割り込み処理を実現する場合のフローチャートを示す図である。
【図１０】本発明の第８の実施形態にかかる情報処理制御システムにおいて、DMA Done割り込みを発生させる際のCache Table Registerとキャッシュメモリとの構成を示すブロック図である。
【図１１】本発明の第９の実施形態にかかる情報処理制御システムにおいて、module単位でPre loadする場合のフローチャートを示す図である。
【図１２】本発明の第１０の実施形態にかかる情報処理制御システムにおいて、Miss hit発生時Application programを切り替えながら実行する場合のフローチャートを示す図である。
【図１３】本発明の第１１の実施形態にかかる情報処理制御システムにおいて、DMA Done割り込みマスクを実現する場合のフローチャートを示す図である。
【図１４】本発明の第１２の実施形態にかかる情報処理制御システムにおいて、メモリと、命令解読装置と、状態管理装置と、割り込み管理機構との関係を示すブロック図である。
【図１５】一般的なキャッシュを用いた情報処理制御システムの構成図である。
【図１６】キャッシュメモリに格納する命令を入れ替えながらプログラムを実行する場合の一般的なプログラムの流れを示す図である。
【図１７】キャッシュコントローラ内のCache Table Registerと、キャッシュメモリの構成を示す一般的なブロック図である。
【符号の説明】
【００８３】

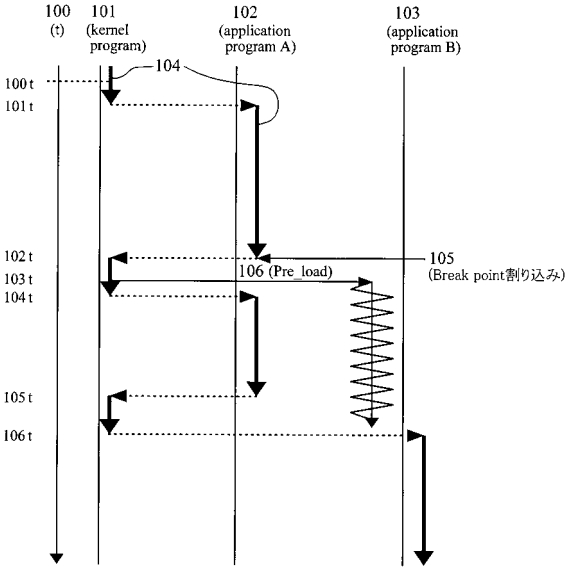
１００	時間の流れ
１０１	Kernel Program処理
１０２	Application program A処理
１０３	Application program B処理
１０４	プログラムの流れ
１０５	Break point割り込み
１００ｔ～１０６ｔ	時刻
１２０	ソフトウェア構成
１２１	Kernel program
１２２	Application program A
１２３	Application program B
１２４	Application program C
１２５	Application program D
１４０	時間の流れ
１４１	Kernel Program処理
１４２	Application program A処理
１４３	Application program B処理
１４４	プログラムの流れ
１４５	Pre load起動
１４６	Break point割り込み
１４７	Application program A module A処理区間
１４８	Application program A module B処理区間
１４９	Application program A module C処理区間

１５０	Application program B module A処理区間
１５１	Application program B module B処理区間
１４０ｔ～１５８ｔ	時刻
１６０～１６７	フローチャート内処理
１８０	時間の流れ
１８１	Kernel Program処理
１８２	Application program A処理
１８３	Application program B処理
１８４	プログラムの流れ
１８５	Down load起動
１８６	Miss hit割り込み
１８７	Break point割り込み
１８０ｔ～１９０ｔ	時刻
２００～２０８	フローチャート内処理
２０１Ｈ、２０２Ｈ	ハード処理
２２０	Cache Table Register
２２１～２２５	レジスタ
２２６～２２９	制御信号
２３０	メモリ
２３１	比較回路
２３２	制御信号
２３３	制御信号
２４０～２４８	フローチャート内処理
２４０Ｈ	ハード処理
２８０～２８８Ｈ	フローチャート内処理
２８０Ｈ～２８３Ｈ	ハード処理
３００～３１２	フローチャート内処理
３００Ｈ～３０６Ｈ	ハード処理
３２０～３２９	フローチャート内処理
３２０Ｈ～３２２Ｈ	ハード処理
３４０Ｈ	割り込み管理機構
３４１Ｈ	メモリ
３４２Ｈ	命令解読装置
３４３Ｈ	状態管理装置
３４０Ｓ～３４２Ｓ	制御信号
４００Ｈ	主記憶メモリ
４０１Ｈ	キャッシュメモリ
４０２Ｈ	キャッシュコントローラ
４０３Ｈ	プロセッサ
４２０	時間の流れ
４２１	Kernel Program処理
４２２	Application program処理
４２３	プログラムの流れ
４２４	Down load処理
４２５	Miss hit割り込み
４２０ｔ～４２３ｔ	時刻

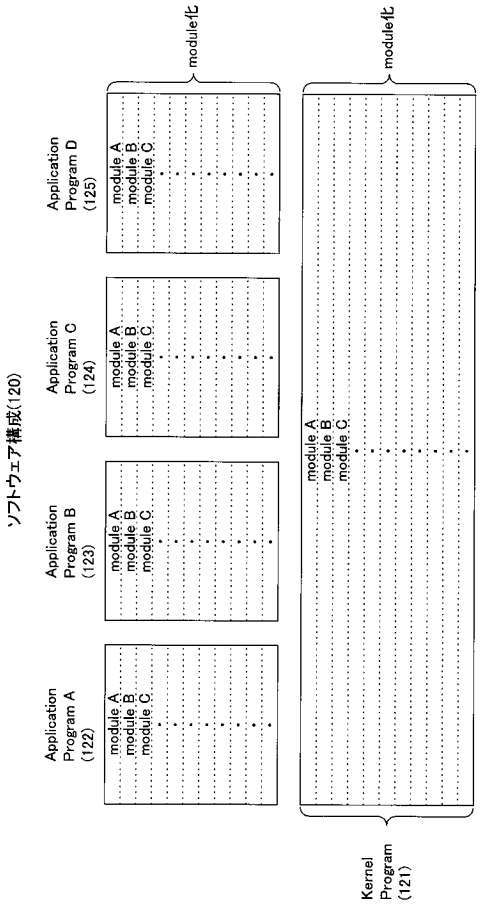
【図１】


401H
400H
キャッシュメモリ
主記憶メモリ
403H
402H
プロセッサ
キャッシュ
コントローラ

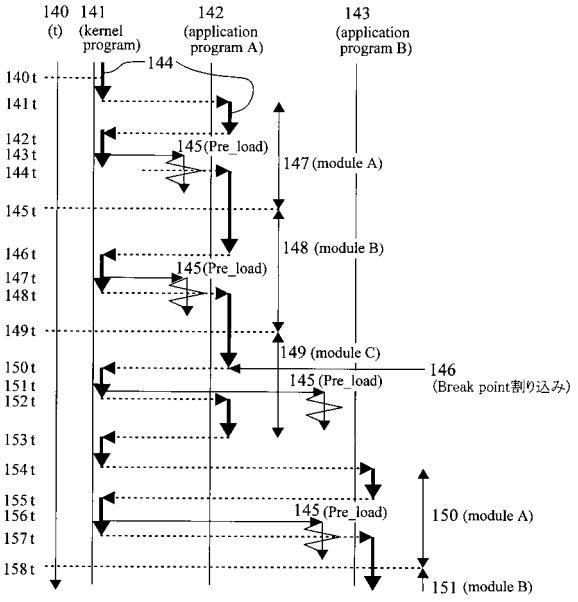
【図２】


100
(t)
101
(kernel program)
102
(application program A)
103
(application program B)
104
100t
101t
102t
103t
104t
105t
106t
105
(Break point割り込み)
106 (Pre_load)

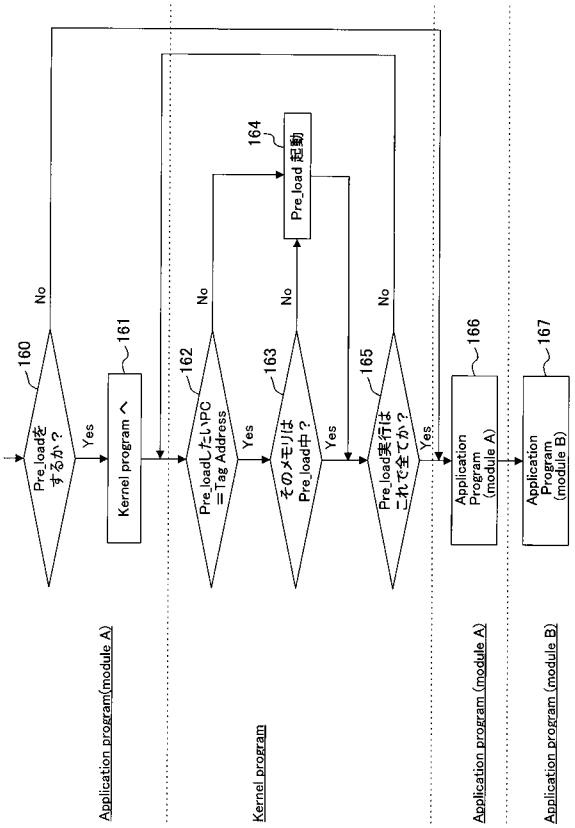
【図３】


ソフトウェア構成(120)
module化
module化
Application Program D (125)
Application Program C (124)
Application Program B (123)
Application Program A (122)
module A
module B
module C
Kernel Program (121)

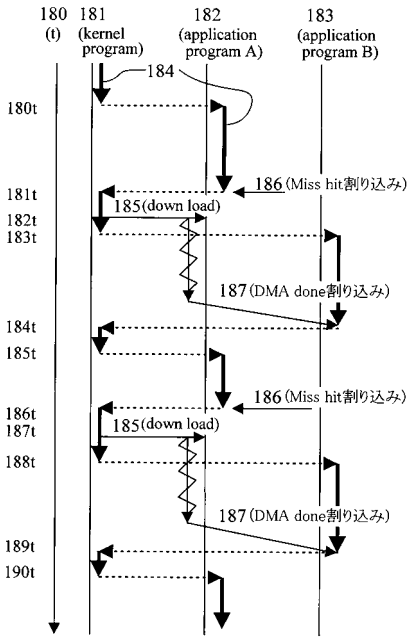
【図４】


140
(t)
141
(kernel program)
142
(application program A)
143
(application program B)
144
140t
141t
142t
143t
144t
145t
146t
147t
148t
149t
150t
151t
152t
153t
154t
155t
156t
157t
158t
145(Pre_load)
147 (module A)
148 (module B)
149 (module C)
146
(Break point割り込み)
150 (module A)
151 (module B)

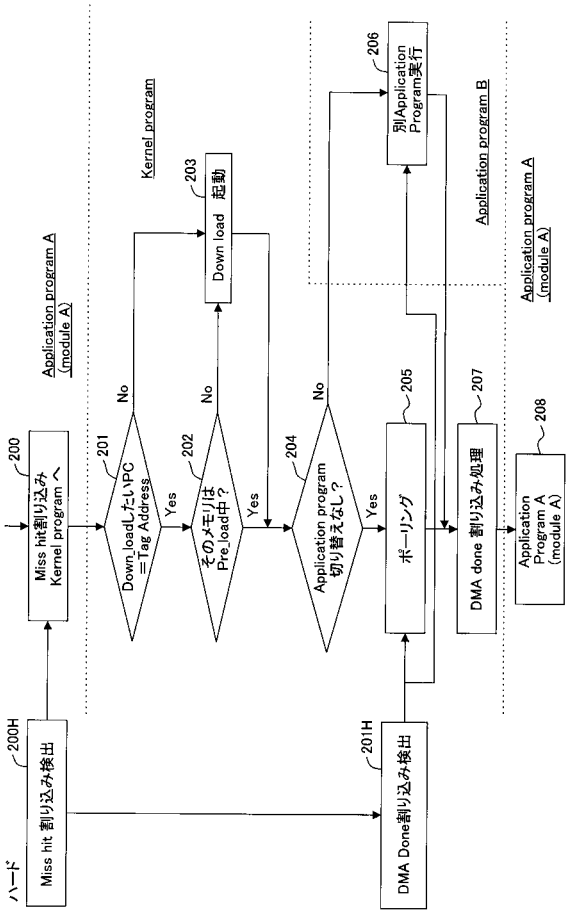
【図５】


160
Pre_loadをするか？
No
Yes
161
Kernel program へ
162
Pre_loadしたいPC=Tag Address
163
そのメモリはPre_load中？
164
Pre_load 起動
165
Pre_load実行はこれで全てか？
166
Application Program (module A)
167
Application Program (module B)
Application program(module A)
Kernel program
Application program (module A)
Application program (module B)

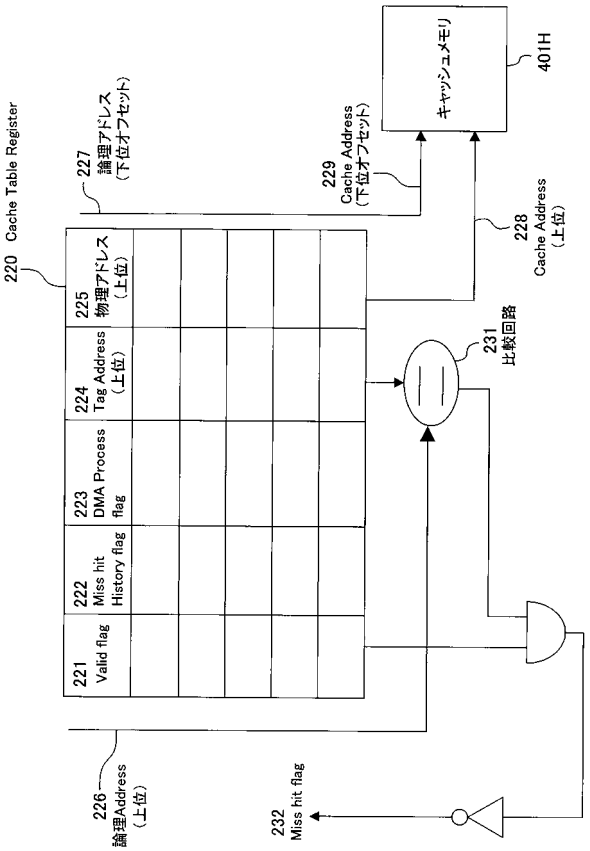
【図６】


180 (t)
181 (kernel program)
182 (application program A)
183 (application program B)
184
186 (Miss hit割り込み)
185(down load)
187 (DMA done割り込み)
186 (Miss hit割り込み)
185(down load)
187 (DMA done割り込み)
180t
181t
182t
183t
184t
185t
186t
187t
188t
189t
190t

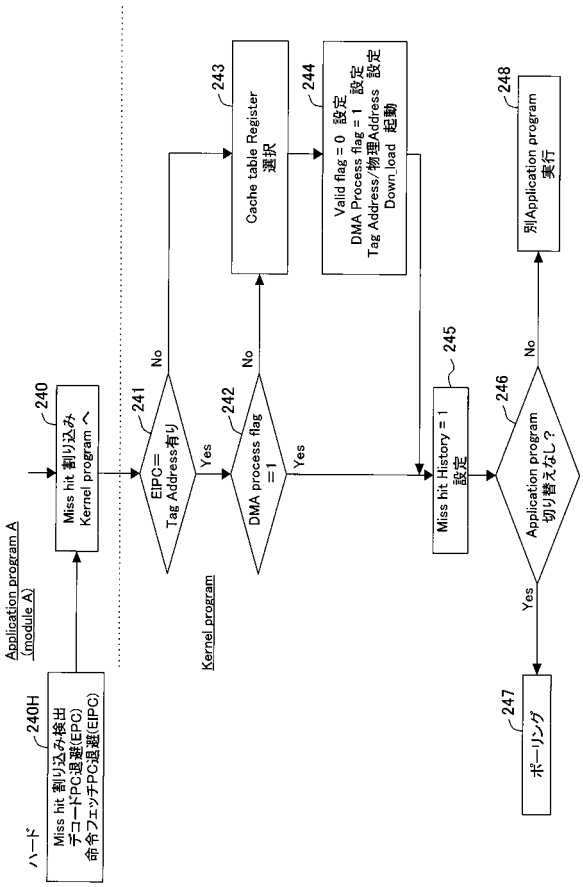
【図７】


200
Miss hit割り込み Kernel program へ
201
Down_loadしたいPC=Tag Address
202
そのメモリはPre_load中？
203
Down load 起動
204
Application program 切り替えなし？
205
ポーリング
206
別Application Program実行
207
DMA done 割り込み処理
208
Application Program A (module A)
Application program A (module A)
Kernel program
Application program B
200H
Miss hit 割り込み検出
201H
DMA Done割り込み検出
ハード

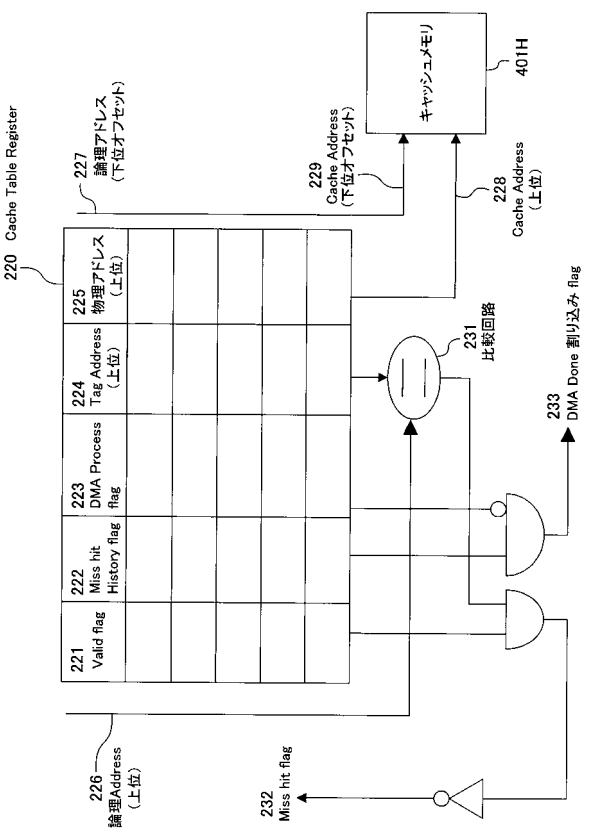
【図８】


220 Cache Table Register
221 Valid flag
222 Miss hit History flag
223 DMA Process flag
224 Tag Address (上位)
225 物理アドレス (上位)
227 論理アドレス (下位オフセット)
229 Cache Address (下位オフセット)
228 Cache Address (上位)
401H キャッシュメモリ
231 比較回路
226 論理Address (上位)
232 Miss hit flag

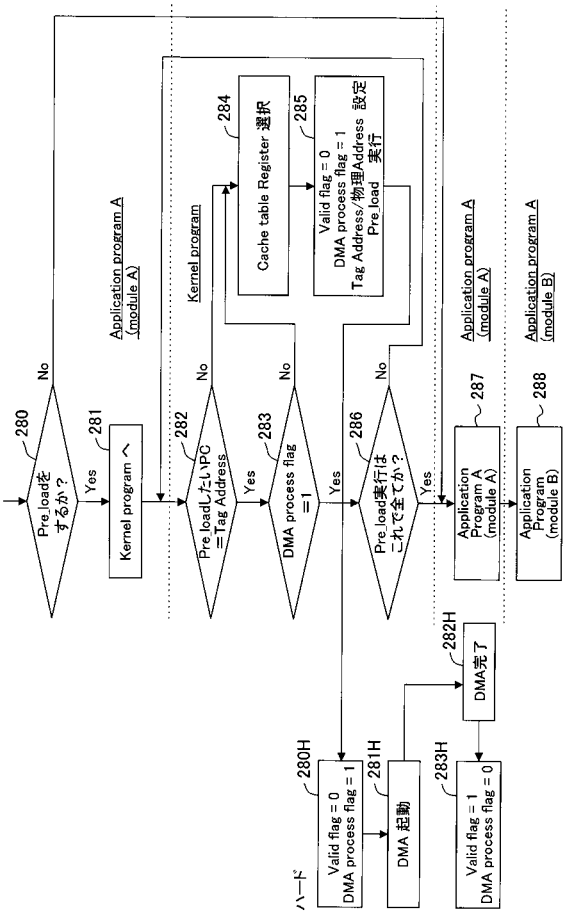
【図９】



【図１０】

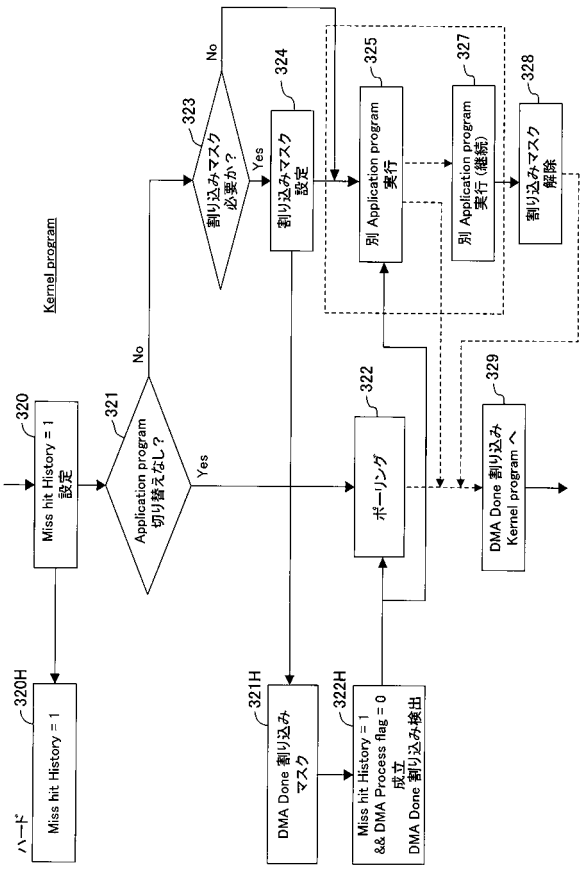


【図１１】

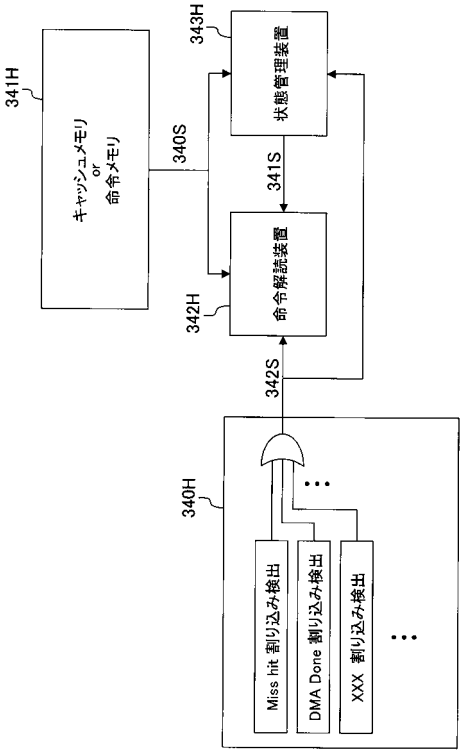


【図１２】

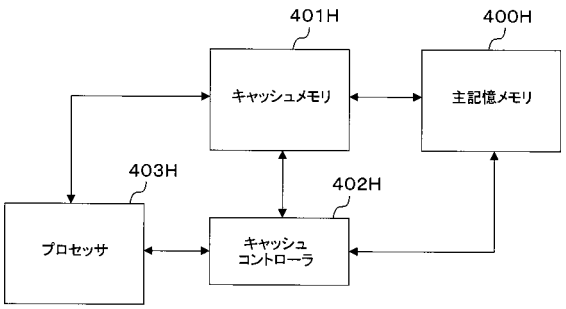
【図１３】



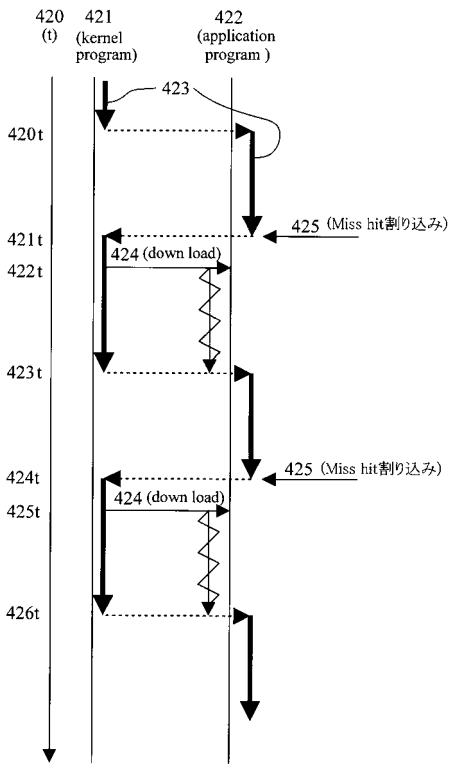
【図１４】



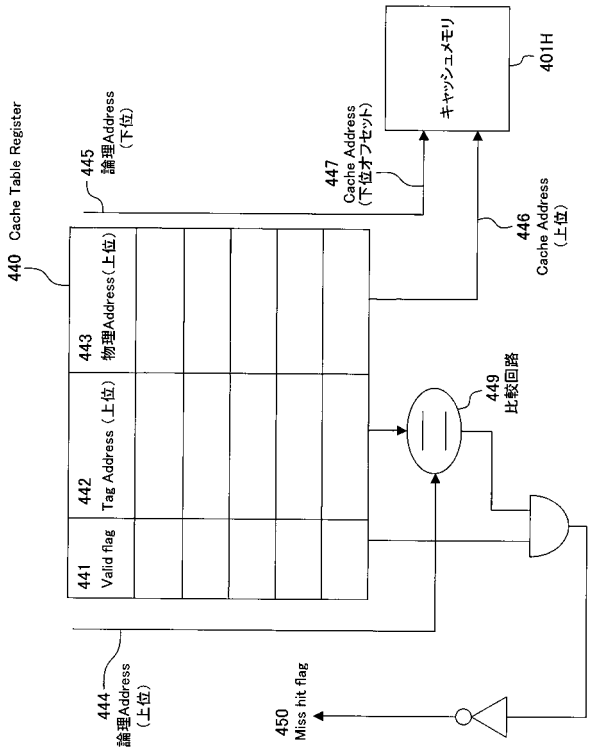
【図１５】



【図１６】



【図１７】


キャッシュメモリ
401H
Cache Table Register
440
445
論理Address
(下位)
447
Cache Address
(下位オフセット)
446
Cache Address
(上位)
443
物理Address(上位)
442
Tag Address (上位)
441
Valid flag
449
比較回路
444
論理Address
(上位)
450
Miss hit flag

フロントページの続き

(72)発明者　法貴　光典
大阪府門真市大字門真１００６番地　　松下電器産業株式会社内

審査官　▲高▼橋　正▲徳▼

(56)参考文献　特開平１１－３２８１２３（ＪＰ，Ａ）
特開平１０－０４０１０６（ＪＰ，Ａ）
特開平１１－０３９１５３（ＪＰ，Ａ）
特開平０３－１５４１３９（ＪＰ，Ａ）
特開昭６２－１７９０３３（ＪＰ，Ａ）
特開２０００－０３５８９４（ＪＰ，Ａ）
特開２００３－０７６５５９（ＪＰ，Ａ）
国際公開第０３／０７５１５４（ＷＯ，Ａ１）
特開昭６０－２６３２３８（ＪＰ，Ａ）
特開平０３－２１４３３７（ＪＰ，Ａ）
特開昭５６－１５９８８６（ＪＰ，Ａ）
特開平０９－２１８８２３（ＪＰ，Ａ）
特開昭５６－１１７３８４（ＪＰ，Ａ）
特開平１０－３４０１９７（ＪＰ，Ａ）
特開平０２－１０９１２８（ＪＰ，Ａ）
特開平０７－０５６７５５（ＪＰ，Ａ）
特開２００１－０５６７８１（ＪＰ，Ａ）
特開２００２－２５９２０９（ＪＰ，Ａ）
国際公開第０３／０２１４３９（ＷＯ，Ａ１）
特開平０３－０７１２４８（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１２／０８－１２／１２，
Ｇ０６Ｆ　１２／０６，
Ｇ０６Ｆ　　９／５０，
Ｇ０６Ｆ　　９／０６，
Ｇ０６Ｆ　　９／３８