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

特許第4353990号
(P4353990)

(45)発行日　平成21年10月28日(2009.10.28)　(24)登録日　平成21年8月7日(2009.8.7)

(51)Int.Cl.			F I		
G06F	9/48	(2006.01)	G06F	9/46	452F
G06F	1/32	(2006.01)	G06F	1/00	332Z
G06F	1/04	(2006.01)	G06F	1/04	301C
G06F	1/26	(2006.01)	G06F	1/00	330C

請求項の数 3　(全 17 頁)

(21)出願番号　特願2007-133131 (P2007-133131)
(22)出願日　平成19年5月18日 (2007.5.18)
(65)公開番号　特開2008-287592 (P2008-287592A)
(43)公開日　平成20年11月27日 (2008.11.27)
審査請求日　平成19年5月18日 (2007.5.18)

(73)特許権者　396023993
株式会社半導体理工学研究センター
神奈川県横浜市港北区新横浜３丁目１７番地２　友泉新横浜ビル６階
(74)代理人　100058479
弁理士　鈴江　武彦
(74)代理人　100091351
弁理士　河野　哲
(74)代理人　100088683
弁理士　中村　誠
(74)代理人　100108855
弁理士　蔵田　昌俊
(74)代理人　100075672
弁理士　峰　隆司

最終頁に続く

(54)【発明の名称】マルチプロセッサ制御装置

(57)【特許請求の範囲】
【請求項１】
複数のプロセッサで実行される各プログラムの性能制約を満たす範囲で、前記複数のプロセッサの合計消費電力と合計消費エネルギーとのうちの少なくとも一方を抑制する「前記複数のプロセッサから前記複数のプロセッサの共有リソースへ発行されるリクエストの優先度」を決定し、前記各プログラムの性能制約を満たす範囲で、前記合計消費電力と合計消費エネルギーとのうちの少なくとも一方を抑制する「前記複数のプロセッサの周波数と電源電圧とのうちの少なくとも一方」を決定する協調制御手段と、
前記協調制御手段によって決定された優先度に応じて、前記複数のプロセッサからのリクエストを前記共有リソースへ発行する第１制御手段と、
前記協調制御手段によって決定された周波数と電源電圧とのうちの少なくとも一方に応じて、前記複数のプロセッサの周波数と電源電圧とのうちの少なくとも一方を制御する第２制御手段と
を具備し、
前記協調制御手段は、緊急性の高いプログラムを実行するプロセッサからのリクエストの優先度を、緊急性の低いプログラムを実行するプロセッサからのリクエストの優先度よりも高くし、
さらに前記協調制御手段は、リクエストの優先度を高くしたプロセッサの周波数と電源電圧とのうちの少なくとも一方を、優先度制御を行わなかった場合と比べて低くすることを特徴とするマルチプロセッサ制御装置。

【請求項２】
請求項１記載のマルチプロセッサ制御装置において、
前記協調制御手段は、
前記複数のプロセッサから前記共有リソースへの実際の競合の状態を監視し、前記実際の競合の状態が、前記優先度となるようにフィードバック制御を実行し、
前記各プログラムの進行状態を監視し、前記各プログラムを実行するそれぞれのプロセッサについて、前記各プログラムの進行状態が性能予測モデルに基づいて予測された予測結果より早く終了しそうな場合に、周波数と電源電圧とのうちの少なくとも一方を低下させ、前記各プログラムの進行状態が前記予測結果に間に合わなくなりそうな場合に、周波数と電源電圧とのうちの少なくとも一方を増加させるフィードバック制御を行う
ことを特徴とするマルチプロセッサ制御装置。
【請求項３】
請求項１記載のマルチプロセッサ制御装置において、
前記協調制御手段は、
前記複数のプロセッサの周波数のいずれもが目標範囲に含まれるように前記複数のプロセッサからのリクエストの優先度を制御することを特徴とするマルチプロセッサ制御装置。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、複数のプロセッサを制御するマルチプロセッサ制御装置に関する。
【背景技術】
【０００２】
従来から、制約を持つプログラムの消費電力削減手法として、制約を満たす範囲で、できる限り低い周波数・電源電圧を用いプログラムを実行するよう、周波数・電源電圧を制御する手法が提案されている。
【０００３】
ある従来の手法では、プログラムのプロファイル情報に基づき周波数・電源電圧を決定する（第１従来例）。他の従来の手法では、プログラム実行時の情報を用いて、制約を満たしていない場合に、周波数・電源電圧を上げて制約を満たすよう制御し、逆に性能に余裕がある場合に、周波数・電源電圧を下げるなどのフィードバック制御により周波数・電源電圧を決定する（第２従来例）。
【０００４】
なお、非特許文献１では、キャッシュを分割するのではなく、各プロセッサの動作周波数及び電源電圧を動的電源電圧／周波数制御（Dynamic Voltage/Frequency Scaling：ＤＶＦＳ）手法によって制御することにより、公平さ（フェアネス）を向上し、高性能化、及び低消費電力あるいは低消費エネルギーを実現する技術が提案されている。
【非特許文献１】近藤正章、中村宏、「ＣＭＰ向け動的電源電圧・周波数制御手法の提案」、情報処理学会研究報告、IPSJ SIG Technical Reports、情報研報　Vol.2005, No.56、２００５年５月３１日発行、２５頁、社団法人　情報処理学会
【発明の開示】
【発明が解決しようとする課題】
【０００５】
ＣＭＰにおいては、一般的に、例えばメモリバスなどのように複数のプロセッサコアに共有されるリソースが存在する。
【０００６】
そのため、各プロセッサコア上で実行されるプログラムの性能は他のプロセッサコア上で同時に実行されるプログラムの性質に大きく依存し、リソース競合が生じると性能が大きく低下してしまう場合がある。
【０００７】

この場合、上述した第１従来例及び第２従来例では、性能が低下したプログラムの周波数・電源電圧を上げるなどして対処するのみであり、他のプログラムとのリソース競合の影響を考慮した制御は行われないため、エネルギー効率がよくない場合がある。

【０００８】

本発明は、上記実情に鑑みてなされたもので、各プロセッサで実行されるプログラムの共有リソース競合の影響を考慮しつつ、マルチプロセッサ全体の消費電力と消費エネルギーとのうちの少なくとも一方を抑制する制御装置を提供する。

【課題を解決するための手段】

【０００９】

上記課題は、複数のプロセッサで実行される各プログラムの性能制約を満たす範囲で、複数のプロセッサの合計消費電力と合計消費エネルギーとのうちの少なくとも一方を抑制する「複数のプロセッサから複数のプロセッサの共有リソースへ発行されるリクエストの優先度」を決定し、各プログラムの性能制約を満たす範囲で、合計消費電力と合計消費エネルギーとのうちの少なくとも一方を抑制する「複数のプロセッサの周波数と電源電圧とのうちの少なくとも一方」を決定する協調制御手段と、協調制御手段によって決定された優先度に応じて、複数のプロセッサからのリクエストを共有リソースへ発行する第１制御手段と、協調制御手段によって決定された周波数と電源電圧とのうちの少なくとも一方に応じて、複数のプロセッサの周波数と電源電圧とのうちの少なくとも一方を制御する第２制御手段、とを具備するマルチプロセッサ制御装置により解決される。

協調制御手段は、緊急性の高いプログラムを実行するプロセッサからのリクエストの優先度を、緊急性の低いプログラムを実行するプロセッサからのリクエストの優先度よりも高くする。さらに協調制御手段は、リクエストの優先度を高くしたプロセッサの周波数と電源電圧とのうちの少なくとも一方を、優先度制御を行わなかった場合と比べて低くする。

【発明の効果】

【００１１】

本発明によれば、各プロセッサで実行されるプログラムの共有リソース競合の影響を考慮しつつ、この競合による性能への影響をモデル化することにより、マルチプロセッサ全体の消費電力と消費エネルギーとのうちの少なくとも一方を抑制することができる。

【発明を実施するための最良の形態】

【００１２】

以下、本発明の実施の形態について、図面を参照して説明する。

【００１３】

（第１の実施の形態）

本実施の形態においては、例えば複数のプロセッサコアを１チップに搭載したチップマルチプロセッサ（Chip Multi Processor：ＣＭＰ）を制御する装置について説明する。しかしながら、この制御装置の制御対象は必ずしもＣＭＰである必要はなく、例えば複数のチップで構成されるマルチプロセッサシステムなどでもよい。

【００１４】

本実施の形態においては、各プロセッサコアで実行されるプログラムの共有リソース競合の影響を考慮しつつ、この競合による性能への影響をモデル化することにより、消費電力と消費エネルギーとのうちの少なくとも一方を抑制するマルチプロセッサの制御装置について説明する。なお、以下においては、消費電力を抑制する場合を例として説明するが、消費エネルギーの抑制も同様の制御手法を用いることで可能である。

【００１５】

本実施の形態においては、共有リソース競合による性能への影響を優先度制御により調整し、消費電力の抑制効果を高めることを可能とする。このために、本実施の形態に係るマルチプロセッサは、優先度制御と周波数・電源電圧制御を協調して行い、全体としての消費電力が最小となるように優先度／周波数・電源電圧を制御する。

【００１６】

図１は、本実施の形態に係るマルチプロセッサ制御装置の構成の一例を示すブロック図である。

【００１７】

マルチプロセッサ１は、複数のプロセッサコアPU_0～PU_nを１チップ上に搭載している。プロセッサコアPU_0～PU_nで実行されるプログラムは性能制約（例えば、何秒以内に終了しなければいけないなどのように実行時間の上限が決まっているなど：レイテンシ（リアルタイム）制約）を持つ。

【００１８】

プロセッサコアPU_0～PU_nと主記憶装置２とは、転送管理部３及びメモリバス４により接続されている。転送管理部３は、例えばＭＭＵ（Memory Management Unit）３ａ、バスコントローラ３ｂ、アクセスキュー３ｃを具備する。

【００１９】

本実施の形態において、マルチプロセッサ１における消費電力の抑制を行うためのマルチプロセッサ制御装置は、アクセスキュー３ｃ、ＦＶＰ協調制御部５、優先度制御部６、周波数・電源電圧制御部７を具備する。

【００２０】

マルチプロセッサ１の各プロセッサコアPU_0～PU_nに対しては、個別に周波数、電源電圧を制御可能である。各プロセッサコアPU_0～PU_nから発行された主記憶装置２へのアクセスリクエストは、どのプロセッサコアから発行されたか判別可能であり、発行元のプロセッサコアに応じた優先度にしたがって制御される。なお、本実施の形態において、優先度とは、各プロセッサコアPU_0～PU_nからのアクセスが競合したときの待ち時間を、各プロセッサコアPU_0～PU_nに分配する割合とする。

【００２１】

マルチプロセッサ制御装置は、マルチプロセッサ１について、各プロセッサコアPU_0～PU_nの周波数・電源電圧とリクエストの優先度を協調して制御する。

【００２２】

プロセッサコアPU_0～PU_nは、転送管理部３を介し、共有リソースであるメモリバス４、さらには主記憶装置２に接続されている。

【００２３】

ＦＶＰ協調制御部５は、メモリバス４の競合の状態や、各プロセッサコアPU_0～PU_nで動作しているプログラム（アプリケーション、プロセス）の情報に基づいて、性能制約を満たしつつ各プログラムが実行され、かつ全体の消費電力を抑制するために、各プロセッサコアPU_0～PU_nのリクエストの優先度と周波数・電源電圧とを協調して制御し、制御信号をそれぞれ優先度制御部６、周波数・電源電圧制御部７に与える。ＦＶＰ協調制御部５は、リクエストの優先度を高くしたプロセッサコアの周波数と電源電圧とのうちの少なくとも一方を、優先度制御を行わなかった場合と比較して低下させる。優先度を高くすることにより、優先度制御を行わなかった場合と比べて周波数と電源電圧とを下げることができ、低電力化を実現することができる。

【００２４】

優先度制御部６は、ＦＶＰ協調制御部５から受けた制御信号に基づいて、主記憶装置２に対するアクセスリクエストの優先度を制御する。

【００２５】

周波数・電源電圧制御部７は、ＦＶＰ協調制御部５から受けた制御信号に基づいて、各プロセッサコアPU_0～PU_nの周波数と電源電圧とのうち少なくとも一方（周波数のみ、電源電圧のみ、あるいは周波数と電源電圧の双方）を制御する。

【００２６】

以下に、本実施の形態に係るマルチプロセッサ制御装置によって行われる制御方法について説明する。

【００２７】

図２は、プログラムの性能制約を満たす状況における、プロセッサコアPU_0の消費電

力、プロセッサコアＰＵ$_1$の消費電力、プロセッサコアＰＵ$_0$及びＰＵ$_1$の消費電力の合計のそれぞれと、リクエストの優先度との関係の一例を示すグラフである。

【００２８】

この図２において、優先度は、プロセッサコアＰＵ$_0$からのリクエストと、プロセッサコアＰＵ$_1$からのリクエストとの間の待ち時間の配分の割合を表す。

【００２９】

優先度０は、プロセッサコアＰＵ$_0$からのリクエストが、プロセッサコアＰＵ$_1$からのリクエストの全てを追い抜くレベルを表す。

【００３０】

優先度０．５は、プロセッサコアＰＵ$_0$からのリクエストと、プロセッサコアＰＵ$_1$からのリクエストとの間で、待ち時間が同じレベルを表す。

【００３１】

優先度１は、プロセッサコアＰＵ$_1$からのリクエストが、プロセッサコアＰＵ$_0$からのリクエストの全てを追い抜くレベルを表す。

【００３２】

優先度が０に向うほど、プロセッサコアＰＵ$_0$からのリクエストが、プロセッサコアＰＵ$_1$からのリクエストより優先される。

【００３３】

逆に、優先度が１に向うほど、プロセッサコアＰＵ$_1$からのリクエストが、プロセッサコアＰＵ$_0$からのリクエストより優先される。

【００３４】

例えば、プロセッサコアＰＵ$_0$の優先度が高く、プロセッサコアＰＵ$_1$の優先度が低い場合、プロセッサコアＰＵ$_0$から共有リソースへのリクエストは、プロセッサコアＰＵ$_1$からのリクエストを追い抜く。このため、プロセッサコアＰＵ$_0$のプログラムは、プロセッサコアＰＵ$_1$のプログラムよりも効率的に実行可能であり、その分、性能制約を満たす範囲でプロセッサコアＰＵ$_0$の周波数・電源電圧を低下させることができる。したがって、プロセッサコアＰＵ$_0$の優先度が高い場合、プロセッサコアＰＵ$_0$の消費電力を低下させることができる。

【００３５】

また、プロセッサコアＰＵ$_1$の優先度がプロセッサコアＰＵ$_0$の優先度よりも低い場合、プロセッサコアＰＵ$_1$から共有リソースへのリクエストは、プロセッサコアＰＵ$_0$からのリクエストに追い抜かれる。このため、プロセッサコアＰＵ$_1$のプログラムは、プロセッサコアＰＵ$_0$のプログラムよりも効率的に実行することが困難となり、その分、性能制約を満たすために、プロセッサコアＰＵ$_1$の周波数・電源電圧を上げることが必要になる。したがって、プロセッサコアＰＵ$_1$の優先度が低い場合、プロセッサコアＰＵ$_1$の消費電力が増加する。

【００３６】

この図２においては、プロセッサコアＰＵ$_1$で実行されるプログラムの方がプロセッサコアＰＵ$_0$で実行されるプログラムよりも緊急性が高く、プロセッサコアＰＵ$_1$の方がプロセッサコアＰＵ$_0$よりも高い処理速度を必要とし、消費電力が大きい。そのため、プロセッサコアＰＵ$_1$についての優先度と消費電力の関係の変化率の方が、プロセッサコアＰＵ$_0$についての優先度と消費電力の関係の変化率よりも大きくなっている。

【００３７】

この図２より、プロセッサコアＰＵ$_1$の優先度をプロセッサコアＰＵ$_0$の優先度よりも高くすることによって、合計の消費電力を抑制することができることがわかる。

【００３８】

本実施の形態では、複数のプロセッサコアＰＵ$_0$～ＰＵ$_n$から共有リソースへのアクセスが競合した場合に、緊急性の高いプログラムからのアクセスを優先させる。これにより、優先度の高いプログラムを実行するプロセッサコアについては、共有リソースに対する待ち状態が短縮されるため、このプロセッサコアの速度（周波数・電源電圧）を低下させる

ことが可能となる。なお、周波数（＝処理速度）は電源電圧にほぼ比例する。消費電力は電源電圧の２乗にほぼ比例する。

【００３９】

上記図２の状態においては、消費電力の小さい側のプロセッサコアＰＵ$_0$の優先度を低くし、消費電力の大きい側のプロセッサコアＰＵ$_1$の優先度を高くするとともに、各プロセッサコアＰＵ$_0$，ＰＵ$_1$についてプログラムの性能制約を満たす範囲で周波数・電源電圧を抑制することにより、プロセッサコアＰＵ$_0$，ＰＵ$_1$の消費電力の合計を抑制することが可能である。

【００４０】

消費電力の合計が最小になる優先度が、「最適な優先度」として制御に用いられる。優先度制御部６は、この最適な優先度となるように、実際のリクエストの発行を制御する。

【００４１】

以下に、優先度と周波数・電源電圧とを制御する制御アルゴリズムの具体例について説明する。なお。この制御アルゴリズムは一例であり、他の手法を用いることもできる。

【００４２】

共有リソースの優先度は、例えば図３に示すように、各プロセッサコアＰＵ$_0$～ＰＵ$_n$からのメモリリクエストを保持するリクエストキュー３ｃがある場合には、優先度の高いプロセッサコアからのリクエストは先に発行された優先度の低い他のプロセッサコアからのリクエストを何個分か追い越して発行することを許可するなどの手法により制御可能である。

【００４３】

共有リソースの競合が、各プロセッサコアＰＵ$_0$～ＰＵ$_n$の性能に与える影響は、各プログラムの性能制約（例えばレイテンシ制約など）、各プログラムの共有リソースへのアクセス回数、各プログラムの命令実行数、共有リソースの性能（例えば単独実行時の共有リソースアクセスによる待ち時間など）をパラメータとした性能予測モデルを構築し、定式化することができる。これにより、マルチプロセッサ１全体の消費電力が最小となる共有リソースの優先度を求めることができる。

【００４４】

優先度及び各プロセッサコアＰＵ$_0$～ＰＵ$_n$の周波数・電源電圧は、例えばタイムインターバルを用いた手法により制御可能である。

【００４５】

図４は、周波数・電源電圧調整インターバル（DVFS_interval）ごとに周波数・電源電圧を調整し、優先度調整インターバル（優先度調整interval）ごとに優先度を調整する制御手法の一例を示す図である。

【００４６】

ＦＶＰ協調制御部５は、周波数・電源電圧調整インターバル間隔で、各プロセッサコアＰＵ$_0$～ＰＵ$_n$のプログラムの性能をチェックし、性能制約を満たすために必要な性能が達成されているかを監視し、達成されていないと判断されたプログラムを実行するプロセッサコアについて、周波数と電源電圧とのうち少なくとも一方をアップするなどのフィードバック制御を実行する。

【００４７】

すなわち、ＦＶＰ協調制御部５は、プログラムの進行状態を周波数・電源電圧調整インターバル間隔でチェックし、性能制約の時刻より早く終了しそうな場合に周波数・電源電圧の少なくとも一方を低下（例えば－１）させ、進行状態が性能制約の時刻に間に合いそうにない場合に周波数・電源電圧の少なくとも一方を増加（例えば＋１）する。なお、一度、周波数・電源電圧を変更すると、Silent_intervalの間は周波数・電源電圧を変更しないとする。

【００４８】

さらに、ＦＶＰ協調制御部５は、優先度調整インターバル間隔で、各プロセッサコアＰＵ$_0$～ＰＵ$_n$から共有リソースへのリクエストの実際の競合の状態を監視し、最適と判断さ

れた優先度を満たすように、優先度についてフィードバック制御を実行する。

【００４９】

すなわち、ＦＶＰ協調制御部５は、上記図２から定まる最適な優先度（最適な待ち合わせの分配）と実際の待ち合わせの分配とを比較し、実際の待ち合わせの分配が、最適な優先度となるように制御する。

【００５０】

一般的に、最低限満たすべき性能制約が存在するプログラムを実行する場合、この性能制約を満たす範囲であれば、低い周波数・電源電圧を用いてプログラムを実行することで消費電力を抑制することができる。

【００５１】

従来のマルチプロセッサでは、メモリバスなどの共有リソースでアクセス競合が発生すると、プログラムの性能低下が生じ、この性能低下を補うために、プロセッサコアをより高い周波数・電源電圧で動作させることが必要になり、消費電力の増大を招く場合がある。

【００５２】

これに対して、本実施の形態においては、ＦＶＰ協調制御部５が、各プロセッサコアPU_0～PU_nで実行しているプログラムの状態に応じて、共有リソースの使用率を優先度制御により適切に制御し、また、この優先度制御に併せて、各プロセッサコアPU_0～PU_nの周波数・電源電圧を調整する。これにより、プログラムの性能制約を満たしつつマルチプロセッサ１全体での消費電力が抑制される。

【００５３】

上記のようなマルチプロセッサ制御装置を用いた場合の効果について以下に説明する。

【００５４】

本実施の形態においては、消費電力の大きいプロセッサコアによる共有リソースへのアクセスリクエストの優先度を、消費電力の小さいプロセッサコアによる共有リソースへのアクセスリクエストの優先度よりも高くすることにより、共有リソースへのアクセスが競合する場合であっても消費電力の大きいプロセッサコアの性能が低下することを緩和できる。これにより、マルチプロセッサ１全体での消費電力を抑制することができる。

【００５５】

本実施の形態においては、周波数・電源電圧の制御と優先度制御を独立に行う場合と比べ、マルチプロセッサ１全体の消費電力をさらに削減することができる。

【００５６】

マルチプロセッサ１は、低消費電力化と高性能化の双方を達成可能なアーキテクチャとして期待され、高性能プロセッサや組み込み向けプロセッサにおいても今後の主流になると考えられる。このようなマルチプロセッサ１に対してプログラムの性能制約を満たしつつ、低消費電力化を一層強化できる本実施の形態に係る制御の技術的意義は極めて大きく、ビジネス的にもインパクトは大きい。

【００５７】

（第２の実施の形態）

本実施の形態においては、上記第１の実施の形態と異なる方式で優先度と周波数・電源電圧とを制御する制御アルゴリズムについて説明する。

【００５８】

性能予測モデルによると、各プロセッサコアPU_0～PU_nの周波数が等しくなる場合が、最も低電力となる。

【００５９】

そこで、本実施の形態では、ＦＶＰ協調制御部５は、各プロセッサコアPU_0～PU_nの周波数のいずれもが目標範囲に含まれるように（より好ましくは等しくなるように）、優先度を制御する。

【００６０】

例えば、２つのプロセッサコアPU_0，PU_1について、プロセッサコアPU_0の周波数

がプロセッサコアＰＵ$_1$の周波数よりも高い場合、ＦＶＰ協調制御部５は、プロセッサコアＰＵ$_0$の優先度を高くする。

【００６１】

逆に、プロセッサコアＰＵ$_1$の周波数がプロセッサコアＰＵ$_0$の周波数よりも高い場合、ＦＶＰ協調制御部５は、プロセッサコアＰＵ$_1$の優先度を高くする。

【００６２】

このような制御アルゴリズムを用いた場合にも、上記第１の実施の形態の場合と同様に、マルチプロセッサ１全体での消費電力を抑制することができる。

【００６３】

（第３の実施の形態）

本実施の形態においては、上記第１及び第２の実施の形態における性能予測モデルの一例について説明する。

【００６４】

ここでは、説明を簡略化するために、図５に示すような２個のプロセッサコアＰＵ$_0$、ＰＵ$_1$を搭載するマルチプロセッサ（チップマルチプロセッサ）１の場合について説明する。

【００６５】

各プロセッサコアＰＵ$_0$、ＰＵ$_1$は、それぞれが、キャッシュＬ1，Ｌ2を内蔵しており、メモリバス４と主記憶装置２とを共有している。各プロセッサコアＰＵ$_0$，ＰＵ$_1$は、それぞれ独立なプログラムＴ$_0$，Ｔ$_1$を実行する。各プロセッサコアＰＵ$_0$、ＰＵ$_1$は、性能制約としてレイテンシ制約を持つ。

【００６６】

以下においては、メモリバス４のアクセス競合について説明する。

【００６７】

プログラム実行中のプロセッサコアは、図６に示すように、命令の実行を行っている状態（稼動）、キャッシュＬ2へのアクセスがミスし、そのデータを待ってストールしている状態（ストール）の２つの状態をとる。

【００６８】

プロセッサコアの周波数・電源電圧を変化させると、図７に示すように、命令を実行完了するために必要な稼動時間が変化する。ここで、共有のメモリバス４の周波数・電源電圧は一定であるため、ストール時間は不変である。

【００６９】

図８は、レイテンシ制約と、プログラムの実行開始から実行終了までの時間との関係を表すタイミングチャートである。レイテンシ制約を持つプログラムを実行する場合には、レイテンシ制約内に、プログラムの実行開始から実行終了までの時間が収まる必要がある。

【００７０】

図９に示すように、プログラムの実行開始から実行完了までにおける稼働時間が（レイテンシ制約－ストール時間）と等しくなるように選んだ周波数・電源電圧が、性能制約を満たす範囲で最も低い（消費電力を最小にする）周波数・電源電圧である。

【００７１】

稼動時のプロセッサコアの性能は、周波数に比例するので、周波数選択について（１）式が成り立つ。

【数１】

$$\text{周波数} \propto \frac{\text{プログラムの命令数}}{(\text{レイテンシ制約}-\text{ストール時間})} \quad \cdots(1)$$

【００７２】

以下において、Ｌ$_i$はプログラムＴ$_i$のレイテンシ制約、ｍ$_i$はプログラムＴ$_i$の実行中に発生するキャッシュＬ2に対するキャッシュミスの回数（＝共有リソースへのアクセス回

数）、s_iはプログラムT_iの実行中のストール時間の長さ（＝単独実行時の共有リソースアクセスによる待ち時間）、l_BはキャッシュＬ2のキャッシュミス１回分のデータを転送するのに共有リソースが必要とする時間（＝キャッシュＬ2のキャッシュミス１回あたりの共有リソース占有時間）、I_iはプログラムT_iの命令実行数、とする。

【００７３】

まず、プロセッサコアPU_iが単独で動作するとき、すなわちアクセス競合がない場合について説明する。この場合、プロセッサコアPU_iの実効的な稼働時間T_iは（２）式で与えられる。

【数２】

$$t_i = L_i - s_i \qquad \cdots(2)$$

【００７４】

すなわち、時間t_iの間にプログラムT_iの命令をちょうど全て処理できるように、プロセッサコアPU_iの周波数・電源電圧を設定すればよい。

【００７５】

よって、プロセッサコアPU_iに設定すべき周波数f_iは（３）式で表される。なお、ｃは定数である。

【数３】

$$f_i = c\frac{I_i}{t_i} \qquad \cdots(3)$$

【００７６】

プロセッサコアPU_iを周波数f_iで動かすのに必要な電源電圧をV_iとすると、プロセッサコアPU_iが１命令を実行する際に消費するエネルギーe_iは（４）式で与えられる。なお、ｋは定数である。

【数４】

$$e_i = kV_i^2 \qquad \cdots(4)$$

【００７７】

以上より、プロセッサコアPU_iの平均消費電力P_i（エネルギー／時間）は（５）式より求められる。

【数５】

$$P_i = \frac{I_i}{L_i}e_i = \frac{kI_iV_i^2}{L_i} \qquad \cdots(5)$$

【００７８】

続いて、各プロセッサコアPU_0、PU_1を同時に動作させた場合について説明する。

【００７９】

プロセッサコアPU_0上でキャッシュＬ2のキャッシュミスが発生し、主記憶装置からデータを取得しようとする場合、プロセッサコアPU_1が共有リソースを占有している確率（コンフリクトが発生する確率）は、（６）式で表される。

【数６】

$$P_0 = \frac{m_1 l_B}{L_1} \qquad \cdots(6)$$

【００８０】

また、プロセッサコアPU_1の共有リソース利用の時間分布が一様であると仮定すると、コンフリクトが発生したときにプロセッサコアPU_0の転送が待たされる時間の期待値

は、（７）式で表される。

【数７】

$$E_w = \frac{1}{2} l_B \qquad \cdots(7)$$

【００８１】

したがって、プロセッサコアPU_1との競合によって増加するプロセッサコアPU_0のストール時間の、プロセッサコアPU_0のキャッシュL2のキャッシュミス１回あたりの期待値は（８）式で表される。

【数８】

$$P_0 E_w = \frac{m_1 l_B^2}{2L_1} \qquad \cdots(8)$$

【００８２】

このとき、プロセッサコアPU_0の実効稼動時間はt_0から（９）式のt_0’に変化する。

【数９】

$$t_0' = t_0 - m_0 \frac{m_1 l_B^2}{2L_1} \qquad \cdots(9)$$

【００８３】

このt_0’より単独動作時と同様に周波数f_0’と電源電圧V_0’が決まる。

【００８４】

プロセッサコアPU_1についても同様に、t_1’、f_0’、V_0’が求まるため、競合がある場合の各プロセッサコアの消費電力P_i’は、（１０）式で与えられる。

【数１０】

$$P_i' = \frac{I_i}{L_i} e_i' = \frac{k I_i V_i'^2}{L_i} \qquad \cdots(10)$$

【００８５】

次に、共有リソースの優先度制御を行うことによる電力の変化について説明する。

【００８６】

プロセッサコアPU_0，PU_1を同時に動作させるとき、競合によって増加するストール時間の総和は、単位時間あたりの値として（１１）式で表される。

【数１１】

$$l_{total} = \frac{m_0}{L_0} \frac{m_1 l_B^2}{2L_1} + \frac{m_1}{L_1} \frac{m_0 l_B^2}{2L_0} = \frac{m_0 m_1 l_B^2}{L_0 L_1} \qquad \cdots(11)$$

【００８７】

l_{total}は、優先度制御を行っても変わることはない。

【００８８】

しかし、以下に説明するように、各プロセッサコアが競合により被るストールの増分（性能ペナルティ）の比率を変えることはできる。

【００８９】

優先度制御を行う場合において、２つのプロセッサコアPU_0，PU_1が同時に共有リソースを使おうとした場合、プロセッサコアPU_0を優先すると、図１０に示す状態となる。

【００９０】
一方、プロセッサコアPU_1を優先すると、図１１に示す状態となる。
【００９１】
この図１０及び図１１に示したように、待ち時間の総和は不変だが、その配分を変えることは可能である。
【００９２】
そこで、理想的な優先度制御部があると仮定し、この理想的な優先度制御部が、各プロセッサコアPU_0，PU_1が受ける性能ペナルティの比率を（１２）式となるように制御するとする（ｒ＝０ならプロセッサコアPU_0の転送を必ず先に行う。ｒ＝１ならその逆）。
【数１２】

$$PU_0 : PU_1 = r : (1-r) \text{ ただし } (0 \leqq r \leqq 1) \qquad \cdots(12)$$

【００９３】
この場合、各プロセッサコアの実効稼働時間t_i’は、ｒの関数として（１３）式及び（１４）式のように表される。
【数１３】

$$t'_0 = t_0 - L_0 r l_{total} \qquad \cdots(13)$$

【００９４】
【数１４】

$$t'_1 = t_1 - L_1 (1-r) l_{total} \qquad \cdots(14)$$

【００９５】
したがって、マルチプロセッサ１の合計消費電力P_{total}（＝P_0‘＋P_1‘）もｒの関数であり、優先度制御によって変化する。
【００９６】
以下において、上記のような性能予測モデルにおいて、マルチプロセッサ１の合計消費電力P_{total}を最小にするｒの値は一般的にただ一つに決まり、以下に示すｄP_{total}／ｄｒが０になるときに最小となる。
【数１５】

$$\frac{d}{dr} P_{total}$$

$$= \frac{I_0}{L_0} \frac{d}{dr} e'_0 + \frac{I_1}{L_1} \frac{d}{dr} e'_1$$

$$= l_{total} \left\{ n \left(\frac{I_0}{t'_0} \right)^{n+1} + (n-1) \left(\frac{I_0}{t'_0} \right)^{n} + \cdots + \left(\frac{I_0}{t'_0} \right)^{2} \right.$$

$$\left. - n \left(\frac{I_1}{t'_1} \right)^{n+1} - (n-1) \left(\frac{I_1}{t'_1} \right)^{n} - \cdots - \left(\frac{I_1}{t'_1} \right)^{2} \right\}$$

$$\cdots(15)$$

【００９７】
これにより、電力最小の条件としてI_0／t_0’＝I_1／t_1’、すなわちf_0’＝f_1’が導かれる。各プロセッサコアが制約を満たすために必要な周波数を等しくすると電力最小になる。この場合のｒの値は（１６）式で与えられる。

【数１６】

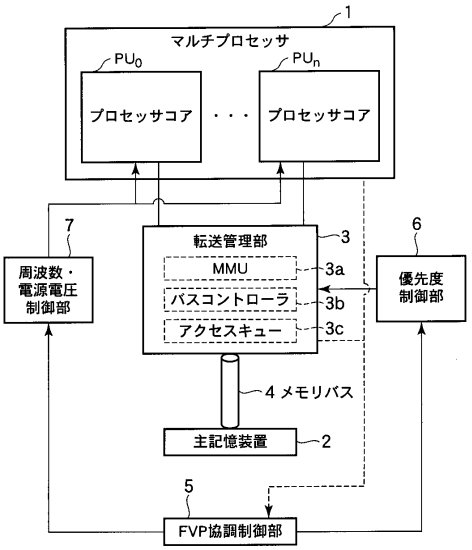
$$r_{min} = \frac{I_1 t_0 - I_0 t_1 + I_0 L_1 l_{total}}{(I_0 L_1 + I_1 L_0) l_{total}} \quad \cdots(16)$$

【００９８】
このｒ$_{min}$より最適な優先度＝最適な待ち合わせ時間の分配比が得られる。
【００９９】
（第４の実施の形態）
本実施の形態においては、上記第１の実施の形態で説明した優先度と周波数・電源電圧の制御アルゴリズム（以下、第１制御アルゴリズム）を用いた場合と、上記第２の実施の形態で説明した優先度と周波数・電源電圧の制御アルゴリズム（以下、第２制御アルゴリズム）を用いた場合の評価について説明する。
【０１００】
図１３は、第１制御アルゴリズム及び第２制御アルゴリズムを使用しない場合、第１制御アルゴリズムを用いた場合、第２制御アルゴリズムを用いた場合の消費電力の状態の一例を示す図である。
【０１０１】
この図１３では、２つのプロセッサコアＰＵ$_0$，ＰＵ$_1$のうち、プロセッサコアＰＵ$_0$でＨ２６４デコーダが実行され、プロセッサコアＰＵ$_1$で他のプログラム「art」又はプログラム「bzip2」が実行された場合の例を示している。
【０１０２】
この図１３からも分かるように、第１制御アルゴリズム及び第２制御アルゴリズムを使用しない場合よりも、第１制御アルゴリズムを用いた場合、第２制御アルゴリズムを用いた場合の方が、約１０％程度消費電力を抑制できている。
【０１０３】
なお、この評価値は、プログラムの制約の厳しさ、各プロセッサコアＰＵ$_0$，ＰＵ$_1$の負荷のばらつきなどにより変化する。
【０１０４】
上記各実施の形態は、その要旨を変更しない範囲において、種々変形可能である。
【０１０５】
例えば、共有リソースは、各種メモリ、バンク、ＰＣＩバス、ディスプレイ、各種インタフェースなど、各プロセッサコアＰＵ$_0$～ＰＵ$_n$に共有される各種アクセス先であってもよい。
【０１０６】
ＦＶＰ協調制御部５は、ソフトウェアにより実現されるとしてもよい。少なくとも一つのプロセッサコアにより、ＦＶＰ協調制御部５の機能が実現されるとしてもよい。
【０１０７】
上記各実施の形態においては、アクセスキュー３ｃ内のリクエストの追い抜きにより優先度を調整しているが、その他の手法により優先度を調整してもよい。例えば、プロセッサコアごとにリクエスト記憶部を設け、優先度制御部６の制御タイミングにそってリクエスト記憶部から共有リソースにリクエストが発行されるとしてもよい。
【０１０８】
優先度と周波数・電源電圧の制御アルゴリズムとしては、他のアルゴリズムを用いることもできる。
【０１０９】
上記各実施の形態において、各種の構成要素は自由に組み合わせ又は分離することができる。例えばＦＶＰ協調制御部５、優先度制御部６、周波数・電源電圧制御部７は、任意に組み合わせることができる。ＦＶＰ協調制御部５、優先度制御部６、周波数・電源電圧制御部７のすべてを組み合わせて１ユニットとしてもよい。

【図面の簡単な説明】
【０１１０】
【図１】本発明の第１の実施の形態に係るマルチプロセッサ制御装置の構成の一例を示すブロック図。
【図２】プログラムの性能制約を満たす状況における、プロセッサコアＰＵ$_0$の消費電力、プロセッサコアＰＵ$_1$の消費電力、プロセッサコアＰＵ$_0$及びＰＵ$_1$の消費電力の合計のそれぞれと、リクエストの優先度との関係の一例を示すグラフ。
【図３】リクエストキューの一例を示すブロック図。
【図４】周波数・電源電圧調整インターバルごとに周波数・電源電圧を調整し、優先度調整インターバルごとに優先度を調整する制御手法の一例を示す図。
【図５】本発明の第３の実施の形態に係るマルチプロセッサと共有リソースとの関係の一例を示すブロック図。
【図６】プロセッサコアの稼動状態とストール状態との関係の一例を示すタイミングチャート。
【図７】周波数が異なる場合のプログラムの実行開始から実行完了までの時間の変化の一例を示すタイミングチャート。
【図８】レイテンシ制約と、プログラムの実行開始から実行終了までの時間との関係を表すタイミングチャート。
【図９】プロセッサコアの稼働時間の上限とストール時間の総和との関係の一例を示す図。
【図１０】プロセッサコアＰＵ$_1$よりもプロセッサコアＰＵ$_0$を優先させる状態の一例を示す図。
【図１１】プロセッサコアＰＵ$_0$よりもプロセッサコアＰＵ$_1$を優先させる状態の一例を示す図。
【図１２】市販のプロセッサの電圧・周波数の関係を線形近似した結果の一例を示す図。
【図１３】第１の実施の形態に係るマルチプロセッサ制御装置を適用した場合と第２の実施の形態に係るマルチプロセッサ制御装置を適用した場合の評価の一例を示す図。
【符号の説明】
【０１１１】
１…マルチプロセッサ、ＰＵ$_0$～ＰＵ$_n$…プロセッサコア、２…主記憶装置、３…転送管理部、３ａ…ＭＭＵ、３ｂ…バスコントローラ、３ｃ…アクセスキュー、４…メモリバス、５…ＦＶＰ協調制御部、６…優先度制御部、７…周波数・電源電圧制御部

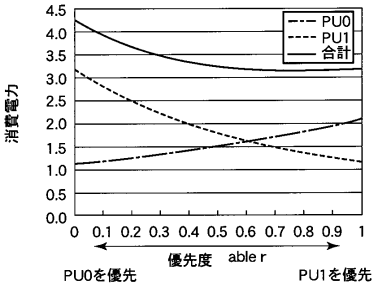
【図１】

図１



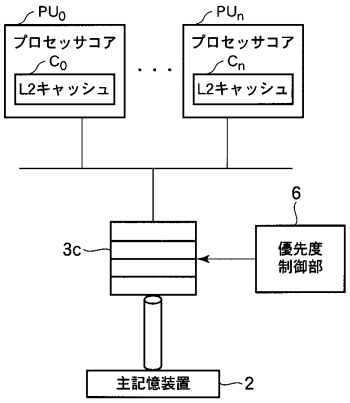
【図２】

図２



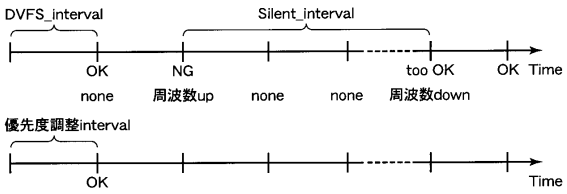
【図３】

図３



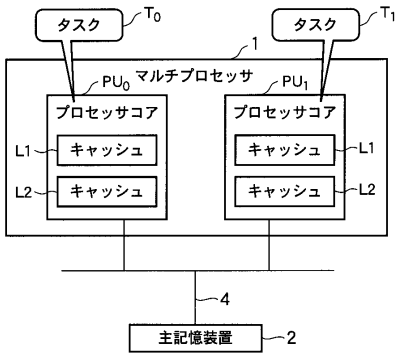
【図４】

図４



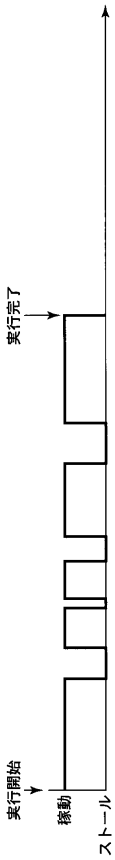
【図５】

図５



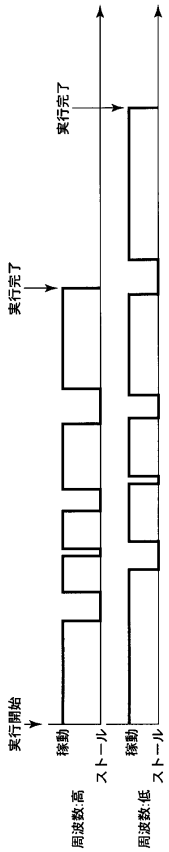
【図６】

図６



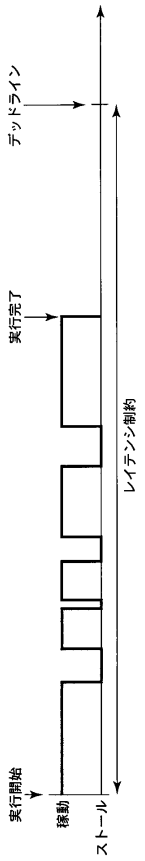
【図７】

図 7



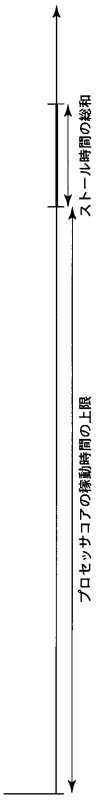
【図８】

図 8



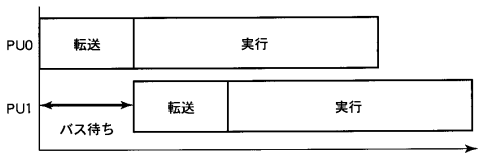
【図９】

図 9



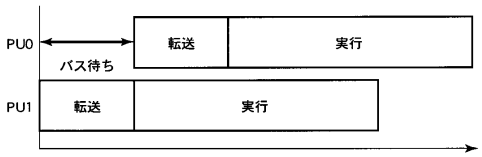
【図１０】

図 10



【図１１】

図 11



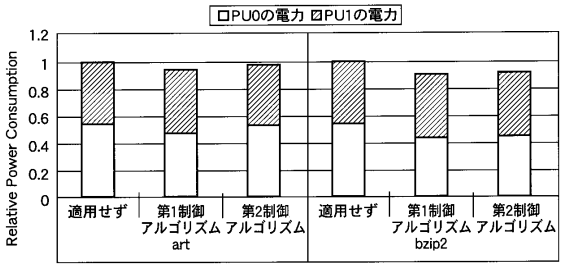
【図１２】

図 12

Processor Clock	1600MHz	1400MHz	1200MHz	1000MHz	800MHz	600MHz	400MHz	200MHz
Bus Clock	200MHz	200MHz	200MHz	200MHz	200MHz	200MHz	200MHz	200MHz
Processor Core Vdd	1.502V	1.390V	1.278V	1.167V	1.055V	0.944V	0.832V	0.721V

【図１３】

図 13



フロントページの続き

(74)代理人 100109830
弁理士 福原 淑弘
(74)代理人 100084618
弁理士 村松 貞男
(74)代理人 100092196
弁理士 橋本 良郎
(72)発明者 中村 宏
東京都世田谷区奥沢１－４６－９
(72)発明者 近藤 正章
東京都杉並区方南１－２２－１０ ＶＩＬＬＡ－HONANCHO１０１号室
(72)発明者 南谷 崇
神奈川県鎌倉市極楽寺４－９－１
(72)発明者 渡辺 亮
東京都文京区向丘２－３４－５ ホワイトマンション２０１

審査官 大塚 良平

(56)参考文献 特開２００７－１１４８５６（ＪＰ，Ａ）
特開２００３－３３７７１３（ＪＰ，Ａ）
国際公開第２００７／１４１８４９（WO，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／４６ － ９／５４
Ｇ０６Ｆ １／０４ － １／１４
Ｇ０６Ｆ １／２６ － １／３２