(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2006-285350
(P2006-285350A)
(43)公開日　平成18年10月19日(2006.10.19)

(51)Int.Cl.		F I		テーマコード（参考）
G06F 1/00	(2006.01)	G06F 1/00	370F	5B042
G06F 11/34	(2006.01)	G06F 11/34	M	
G06Q 30/00	(2006.01)	G06F 17/60	332	
G06F 9/50	(2006.01)	G06F 9/46	465Z	

審査請求 有　請求項の数 10　ＯＬ　（全 23 頁）

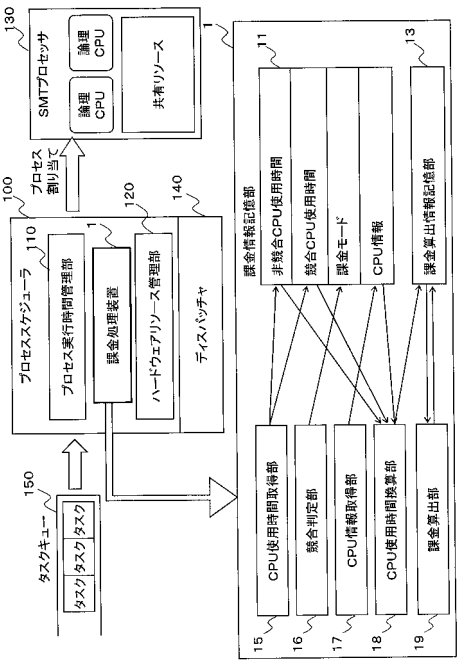
(21)出願番号　特願2005-100951 (P2005-100951)
(22)出願日　平成17年3月31日 (2005.3.31)

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100087848
弁理士　小笠原　吉義
(74)代理人　100083297
弁理士　山谷　晧榮
(72)発明者　山村　周史
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(72)発明者　久門　耕一
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

Ｆターム(参考)　5B042 GA23　HH20　MA08　MA14　MB01
MC21　MC23　MC28　MC31

(54)【発明の名称】ＳＭＴプロセッサ用課金処理装置，課金処理方法，および課金処理プログラム

(57)【要約】
【課題】　ＳＭＴプロセッサで実行されるユーザプログラムに対し公平な課金を行えるようにする。
【解決手段】　競合判定部１６は，ＳＭＴプロセッサ１３０の論理ＣＰＵでユーザのプロセス実行を開始すると，他の論理ＣＰＵでプロセスを実行中である競合状態か否かを判定し，課金情報記憶部１１に格納する。そしてＣＰＵ使用時間取得部１５は，当該プロセスの競合状態または非競合状態でのＣＰＵ使用時間を区別して収集し，課金情報記憶部１１に格納する。その後，ＣＰＵ使用時間換算部１８は，プロセス実行終了後に，競合状態および非競合状態のＣＰＵ使用時間にもとづいて，競合状態のＣＰＵ使用時間に所定の重み付けを行って換算し，課金算出部１９は，実効使用時間からプロセスへの課金額を算出する。
【選択図】　図１



【特許請求の範囲】
【請求項１】
ＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行う装置であって，
論理プロセッサでユーザプログラムのプロセスの実行が開始される時に，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記判定結果として競合状態または非競合状態のいずれかの状態を競合状態記憶部に設定する競合判定部と，
前記ユーザプログラムのプロセスが論理プロセッサを使用した時間を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ使用時間と，前記非競合状態での実行における非競合ＣＰＵ使用時間とを区別してＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得部と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間および前記非競合ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して課金額を算出する課金算出部とを備える
ことを特徴とするＳＭＴプロセッサ用課金処理装置。
【請求項２】
請求項１記載のＳＭＴプロセッサ用課金処理装置において，
所定の重み付け値を用いて，前記競合ＣＰＵ使用時間をもとに，前記ユーザプログラムのプロセス実行時の実質的な使用時間であるＣＰＵ換算使用時間を算出する使用時間換算部を備えるとともに，
前記課金算出部は，前記ＣＰＵ換算使用時間および前記ＣＰＵ使用時間記憶部から読み出した前記非競合ＣＰＵ使用時間をもとにＣＰＵ実効使用時間を算出し，所定の課金単価によって前記ＣＰＵ実効使用時間から課金額を算出する
ことを特徴とするＳＭＴプロセッサ用課金処理装置。
【請求項３】
請求項２記載のＳＭＴプロセッサ用課金処理装置において，
前記ユーザプログラムのプロセスの実行の開始から，前記プロセスを実行する論理プロセッサの性能指標であるＣＰＵ情報を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ情報と前記非競合状態での実行における非競合ＣＰＵ情報とを区別してＣＰＵ情報記憶部に格納するＣＰＵ使用情報取得部を備えるとともに，
前記使用時間換算部は，前記ＣＰＵ情報記憶部から読み出した前記競合ＣＰＵ情報および前記非競合ＣＰＵ情報の比率を用いて，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間をもとにＣＰＵ換算使用時間を算出する
ことを特徴とするＳＭＴプロセッサ用課金処理装置。
【請求項４】
ＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行う装置であって，
論理プロセッサでユーザプログラムの各プロセスを実行中のＳＭＴプロセッサのハードウェアリソースの利用状態に関するリソース状態情報を収集し，リソース状態情報記憶部に格納するリソース状態情報取得部と，
前記ユーザプログラムのプロセスが前記論理プロセッサを使用したＣＰＵ使用時間を収集しＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得部と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記リソース状態情報取得部から読み出した前記各プロセスのリソース状態情報を用いて，前記ユーザプログラムの各プロセスが，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間を前記競合状態におけるＣＰＵ使用時間と前記非競合状態におけるＣＰＵ使用時間とに区別し，前記競合状態および前記非競合状態のＣＰＵ使用時間を区別して課金

額を算出する課金算出部とを備える
ことを特徴とするＳＭＴプロセッサ用課金処理装置。
【請求項５】
請求項４記載のＳＭＴプロセッサ用課金処理装置において，
前記リソース状態情報取得部は，論理プロセッサでユーザプログラムのプロセスが実行される場合に，前記プロセスの命令が，前記ハードウェアリソースの少なくとも演算ユニットまたは処理ユニットのいずれかのユニットに対して投入されるまでの待ち時間を収集し，前記待ち時間をリソース状態情報として前記リソース状態情報記憶部に格納し，
前記課金算出部は，前記リソース状態情報記憶部から読み出した前記各プロセスの待ち時間が，所定時間以上である場合に前記プロセスの実行が競合状態であると判定し，所定の重み付け値を用いて前記競合状態のＣＰＵ使用時間をもとに，前記ユーザプログラムのプロセス実行時の実質的な使用時間であるＣＰＵ換算使用時間を算出する
ことを特徴とするＳＭＴプロセッサ用課金処理装置。
【請求項６】
請求項４または請求項５のいずれか一項に記載のＳＭＴプロセッサ用課金処理装置において，
前記課金算出部は，所定の重み付け値を用いて，前記競合状態のＣＰＵ使用時間をもとにＣＰＵ換算使用時間を算出し，前記ＣＰＵ換算使用時間および前記非競合状態のＣＰＵ使用時間をもとに，ＣＰＵ実効使用時間を算出し，所定の課金単価によって前記ＣＰＵ実効使用時間から課金額を算出する
ことを特徴とするＳＭＴプロセッサ用課金処理装置。
【請求項７】
請求項１または請求項４のいずれか一項に記載のＳＭＴプロセッサ用課金処理装置において，
前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して前記ユーザプログラムの利用課金額を算出するプログラム利用課金算出部を備える
ことを特徴とするＳＭＴプロセッサ用課金処理装置。
【請求項８】
コンピュータのＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行うための処理方法であって，
前記コンピュータが，
論理プロセッサでユーザプログラムのプロセスの実行が開始される時に，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記判定結果として競合状態または非競合状態のいずれかの状態を競合状態記憶部に設定する競合判定処理過程と，
前記ユーザプログラムのプロセスが論理プロセッサを使用した時間を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ使用時間と前記非競合状態での実行における非競合ＣＰＵ使用時間とを区別してＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得処理過程と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間および前記非競合ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して課金額を算出する課金算出処理過程とを有する
ことを特徴とするＳＭＴプロセッサ用課金処理方法。
【請求項９】
コンピュータのＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行うための処理方法であって，
前記コンピュータが，

論理プロセッサでユーザプログラムの各プロセスを実行中のＳＭＴプロセッサのハードウェアリソースの利用状態に関するリソース状態情報を収集し，リソース状態情報記憶部に格納するリソース状態情報取得処理過程と，
前記ユーザプログラムのプロセスが前記論理プロセッサを使用したＣＰＵ使用時間を収集しＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得処理過程と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記リソース状態情報取得部から読み出した前記各プロセスのリソース状態情報を用いて，前記ユーザプログラムの各プロセスが他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間を前記競合状態におけるＣＰＵ使用時間と前記非競合状態におけるＣＰＵ使用時間とに区別し，前記競合状態および前記非競合状態のＣＰＵ使用時間を区別して課金額を算出する課金算出処理過程とを有する
ことを特徴とするＳＭＴプロセッサ用課金処理方法。

【請求項１０】
コンピュータのＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行うための処理方法をコンピュータに実行させるためのプログラムであって，
前記コンピュータに，
論理プロセッサでユーザプログラムのプロセスの実行が開始される時に，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記判定結果として競合状態または非競合状態のいずれかの状態を競合状態記憶部に設定する競合判定処理過程と，
前記ユーザプログラムのプロセスが論理プロセッサを使用した時間を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ使用時間と前記非競合状態での実行における非競合ＣＰＵ使用時間とを区別してＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得処理過程と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間および前記非競合ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して課金額を算出する課金算出処理過程とを実行させるための
ＳＭＴプロセッサ用課金処理プログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は，ＳＭＴ（Ｓｉｍｕｌｔａｎｅｏｕｓ　Ｍｕｌｔｉ－Ｔｈｒｅａｄｉｎｇ）プロセッサ用課金技術に関し，さらに詳しくは，ＳＭＴプロセッサで実行されるユーザプログラムの計算資源の実質的な使用を示す情報にもとづいた課金処理技術に関する。
【背景技術】
【０００２】
ＳＭＴプロセッサとは，演算器，キャッシュメモリ，バスユニット等の１組の計算資源（ハードウェアリソース）を，命令フェッチユニット，命令デコードユニット，命令発行ユニットなどで構成される命令シーケンサとして実現される論理プロセッサ間で共有し，各論理プロセッサが複数のプログラムすなわち命令流を同時に実行して，物理的ＣＰＵ１個あたりのスループットの向上を図るものである。
【０００３】
近年，サーバ分野においてＳＭＴプロセッサの利用が開始されつつあり，ＳＭＴプロセッサにおいてユーザプログラムに対する課金システムが要求されている。
【０００４】
プロセッサの命令レベルで並列処理を行わないＣＰＵ（以下，非ＳＭＴプロセッサと呼ぶ）においては，割り当てられたクォンタムの間プロセスはＣＰＵを占有する。そのため

，ユーザプログラムに対する課金を行う場合に，ＯＳ（Ｏｐｅｒａｔｉｎｇ　Ｓｙｓｔｅｍ）がプロセスに割り当てたＣＰＵ時間を計測することによって，どのプロセスに対しても平等に課金を行うことができる（例えば，非特許文献１）。

【非特許文献１】Ｍ．Ｊ．Ｂａｃｈ著　「ＵＮＩＸカーネルの設計」共立出版　Ｐ．２２８　１９９１年

【発明の開示】

【発明が解決しようとする課題】

【０００５】

しかし，ＳＭＴプロセッサでは，複数の論理プロセッサが構築され，各論理プロセッサは別個独立にプロセスが割り当てられ，ハードウェアリソースの利用状況が変化するため，ＯＳによって割り当てられたＣＰＵ時間を用いる従来の課金システムを適用することができない。

【０００６】

図１０を用いて，ＳＭＴプロセッサおよび非ＳＭＴプロセッサのプロセス実行の相違を説明する。図１０（Ａ）に，非ＳＭＴプロセッサにおけるプロセス実行の例を示し，図１０（Ｂ）に，２つの論理プロセッサが構築されているＳＭＴプロセッサにおけるプロセス実行の例を示す。

【０００７】

図１０（Ａ）に示すように，非ＳＭＴプロセッサの場合には，１クォンタムに割り当てられるプロセスは１つである。プロセッサは，ＯＳによって割り当てられた順に，プロセスｐ１，プロセスｐ２，プロセスｐ３，・・・の各プロセスの命令を実行する。

【０００８】

これに対して，図１０（Ｂ）に示すように，１つのＳＭＴプロセッサ内で２つの論理プロセッサ（Ｌ０，Ｌ１）のうち，２つの論理プロセッサがそれぞれプロセスを実行し，共有リソースを競合して使用する状態（競合状態）と，一方の論理プロセッサのみがプロセスを実行して共有リソースを専用する状態（非競合状態）とが生じる。すなわち，あるクォンタムでは，論理プロセッサ（Ｌ０）でプロセスｐ１が実行中であり，論理プロセッサ（Ｌ１）がアイドル状態である場合に，論理プロセッサＬ０のプロセスｐ１は，共有リソースを専用する。しかし，次のクォンタムで，論理プロセッサ（Ｌ０）でプロセスｐ２を実行する場合に，論理プロセッサ（Ｌ１）でプロセス３が同時に実行されることになると，共有リソースは２つの論理プロセッサによって共用される。

【０００９】

このような，複数の論理プロセッサがプロセスを同時に実行して１つのハードウェアリソースを同時に使用するという競合状態では，各論理プロセッサが使用できるリソースの性能は，競合状態にある論理プロセッサ数に応じて低くなってしまう。図１０（Ｂ）に示すＳＭＴプロセッサでは，使用できる共有リソースは，最悪の場合で１／２となってしまい，論理プロセッサＬ０，Ｌ１は，非競合状態におけるプロセス実行時に比べて，１クォンタム当たりで半分の命令数しか処理できない。すなわち，プロセスｐ２が実行される時間は，非競合状態での実行時に比べて，２倍の時間が必要となる。これは，あるプロセスからみて，競合状態ではプロセッサの実質的性能が低下し，プロセス実行時間が延びてしまうことを意味する。そのため，従来のようにＯＳによって割り当てられるＣＰＵ時間をもとに課金すると，プロセスｐ２およびプロセスｐ３は不当に高く課金されることになり問題である。

【００１０】

本発明は，ＳＭＴプロセッサで実行されるユーザプログラムに対する公平な課金システムの実現を図るものである。本発明の目的は，ＳＭＴプロセッサ上で実行されるユーザプログラムが，物理的なプロセッサの性能や共有リソースなどを実質的にどの程度使用してＣＰＵの性能を享受したかにもとづいて課金を行うことができる課金処理技術を提供することである。

【課題を解決するための手段】

【００１１】
前記目的を達成するために，本発明は，ＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行う装置であって，１）論理プロセッサでユーザプログラムのプロセスの実行が開始される時に，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記判定結果として競合状態または非競合状態のいずれかの状態を競合状態記憶部に設定する競合判定部と，２）前記ユーザプログラムのプロセスが論理プロセッサを使用した時間を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ使用時間と，前記非競合状態での実行における非競合ＣＰＵ使用時間とを区別してＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得部と，３）前記ユーザプログラムのプロセスの実行が終了した後に，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間および前記非競合ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して課金額を算出する課金算出部とを備えることを特徴とする。

【００１２】
本発明では，ＳＭＴプロセッサ上の論理プロセッサでユーザプログラムのプロセスの実行が開始される時に，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，その判定結果を競合状態記憶部に設定する。そして，ユーザプログラムのプロセスが論理プロセッサを使用した時間を収集し，競合状態記憶部を参照して，そのプロセスが競合状態での実行であれば収集したＣＰＵ使用時間を競合ＣＰＵ使用時間としてＣＰＵ使用時間記憶部に格納する。一方，そのプロセスが非競合状態での実行であれば収集したＣＰＵ使用時間を非競合ＣＰＵ使用時間としてＣＰＵ使用時間記憶部に格納する。そして，ユーザプログラムのプロセスの実行が終了した後に，ＣＰＵ使用時間記憶部から読み出した競合ＣＰＵ使用時間および非競合ＣＰＵ使用時間にもとづいて，競合状態におけるユーザプログラムの実行と非競合状態におけるユーザプログラムの実行とを区別して課金額を算出する。

【００１３】
これにより，論理プロセッサが競合状態か非競合状態かによって，ユーザプログラムのプロセス実行時間が大きく変化するようなＳＭＴプロセッサにおいても，プロセス実行中の競合状態を考慮した，より公平な課金処理を実現することができる。

【００１４】
また，本発明は，前記の構成をとる場合に，さらに，所定の重み付け値を用いて，前記競合ＣＰＵ使用時間をもとに，前記ユーザプログラムのプロセス実行時の実質的な使用時間であるＣＰＵ換算使用時間を算出する使用時間換算部を備えるとともに，前記課金算出部は，前記ＣＰＵ換算使用時間および前記ＣＰＵ使用時間記憶部から読み出した前記非競合ＣＰＵ使用時間をもとにＣＰＵ実効使用時間を算出し，所定の課金単価によって前記ＣＰＵ実効使用時間から課金額を算出することを特徴とする。

【００１５】
これにより，競合状態でのＣＰＵ使用時間を非競合状態でのＣＰＵ使用時間に換算してユーザプログラムの課金を行うことができるため，より公平な課金処理を実現することができる。

【００１６】
また，本発明は，前記ユーザプログラムのプロセスの実行の開始から，前記プロセスを実行する論理プロセッサの性能指標であるＣＰＵ情報を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ情報と，前記非競合状態での実行における非競合ＣＰＵ情報とを区別してＣＰＵ情報記憶部に格納するＣＰＵ使用情報取得部を備えるとともに，前記使用時間換算部は，前記ＣＰＵ情報記憶部から読み出した前記競合ＣＰＵ情報および前記非競合ＣＰＵ情報との比率を用いて，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間をもとに前記ユーザプログラムのプロセス実行時の実質的な使用時間であるＣＰＵ換算使用時間を算出することを特徴とする。

【００１７】
これにより，例えばＣＰＩ情報（１命令当たりのクロックサイクル数），単位期間当たりの命令実行数などのＣＰＵの性能指標にもとづいて競合状態でのＣＰＵ使用時間を非競合状態でのＣＰＵ使用時間に換算してユーザプログラムの課金を行うことができるため，より公平な課金処理を実現することができる。
【００１８】
さらに，本発明は，１）論理プロセッサでユーザプログラムの各プロセスを実行中のＳＭＴプロセッサのハードウェアリソースの利用状態に関するリソース状態情報を収集し，リソース状態情報記憶部に格納するリソース状態情報取得部と，２）前記ユーザプログラムのプロセスが前記論理プロセッサを使用したＣＰＵ使用時間を収集しＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得部と，３）前記ユーザプログラムのプロセスの実行が終了した後に，前記リソース状態情報取得部から読み出した前記各プロセスのリソース状態情報を用いて，前記ユーザプログラムの各プロセスが，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間を前記競合状態におけるＣＰＵ使用時間と前記非競合状態におけるＣＰＵ使用時間とに区別し，前記競合状態および前記非競合状態のＣＰＵ使用時間とを区別して課金額を算出する課金算出部とを備えることを特徴とする。
【００１９】
本発明では，論理プロセッサでユーザプログラムの各プロセスを実行中のＳＭＴプロセッサのハードウェアリソースの利用状態に関するリソース状態情報を収集し，リソース状態情報記憶部に格納する。さらに，ユーザプログラムのプロセスが論理プロセッサを使用したＣＰＵ使用時間を収集しＣＰＵ使用時間記憶部に格納する。
【００２０】
そして，ユーザプログラムのプロセスの実行が終了した後に，リソース状態情報取得部から読み出した各プロセスのリソース状態情報を用いて，ユーザプログラムの各プロセスが，競合状態または非競合状態であるかを判定する。そして，ＣＰＵ使用時間記憶部から読み出したＣＰＵ使用時間を，判定結果にもとづいて，競合状態におけるＣＰＵ使用時間と非競合状態におけるＣＰＵ使用時間とに区別し，競合状態および非競合状態のＣＰＵ使用時間とを区別して課金額を算出する。
【００２１】
複数の論理プロセッサでプロセスが同時に実行されている場合でも，それぞれのプロセスの命令が共有リソースのどのユニットに投入されているかによって，競合状態による性能低下を招かない場合もある。例えば，競合する一方のプロセスがＡＬＵを使用し，他方のプロセスがロード／ストアユニットを使用しているような状態は，実質的な競合の影響は少ないと考えられる。そこで，ＳＭＴプロセッサの共有リソースの状態情報を利用し，プロセス実行時に性能低下を生じるような実質的な競合状態であるかどうかを考慮した課金を行う。これにより，物理的な共有リソースの利用状態を反映させた，より精度の高い課金処理を実現することができる。
【００２２】
さらに，前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して前記ユーザプログラムの利用課金額を算出するプログラム利用課金算出部を備えることを特徴とする。これにより，ＳＭＴプロセッサにおけるアプリケーションプログラムなどの利用料金についても課金を行うことができる。
【００２３】
また，本発明は，上記の課金処理装置の要素および処理手段によって実行される処理を行う処理方法として実施することができる。
【００２４】
また，本発明は，コンピュータにより読み取られ実行されるプログラムとして実施する

ことができる。本発明を実現するプログラムは，コンピュータが読み取り可能な，可搬媒体メモリ，半導体メモリ，ハードディスクなどの適当な記録媒体に格納することができ，これらの記録媒体に記録して提供され，または，通信インタフェースを介して種々の通信網を利用した送受信により提供されるものである。

【発明の効果】

【００２５】

本発明によれば，論理プロセッサが競合状態か非競合状態かによって，ユーザプログラムのプロセス実行時間が大きく変化するようなＳＭＴプロセッサにおいても，プロセス実行中の競合状態を考慮した，より公平な課金処理を実現することができる。

【００２６】

特に，競合状態でのＣＰＵ使用時間を非競合状態でのＣＰＵ使用時間に換算した換算使用時間をもとにユーザプログラムの課金を行うことができるため，より公平な課金処理を実現することができる。

【００２７】

また，本発明によれば，例えばＣＰＩ情報（１命令当たりのクロックサイクル数），単位期間当たりの命令実行数などのＣＰＵの性能指標を示す情報を用いて競合状態でのＣＰＵ使用時間を非競合状態でのＣＰＵ使用時間に換算し，この換算使用時間をもとにユーザプログラムの課金を行うことができるため，より公平な課金処理を実現することができる。

【００２８】

また，本発明によれば，ＳＭＴプロセッサの共有リソースの状態情報を利用し，プロセス実行時に性能低下を生じるような実質的な競合状態であるかどうかを考慮して，物理的な共有リソースの利用状態を反映させた，より精度の高い課金処理を実現することができる。

【００２９】

また，本発明によれば，ＳＭＴプロセッサにおいて，競合状態におけるＣＰＵ使用時間から換算されたＣＰＵ換算使用時間は，ＳＭＴプロセッサの性能向上率の算出に利用することができるため，より精度の高い性能向上率を計算したり，性能向上率を動的に判定したりすることができる。

【発明を実施するための最良の形態】

【００３０】

以下，本発明の実施の形態を説明する。

【００３１】

図１は，本発明の第１の実施例における構成例を示す図である。本発明にかかる課金処理装置１は，コンピュータのＯＳ（Ｏｐｅｒａｔｉｎｇ　Ｓｙｓｔｅｍ）のプロセススケジューラ１００内に組み込まれるプログラムとして実施される。

【００３２】

プロセススケジューラ１００は，一般的なＴＳＳ（Ｔｉｍｅ　Ｓｃｈｅｄｕｌｉｎｇ　Ｓｙｓｔｅｍ）によって，クォンタムごとにどのプロセスを，１つの物理的なＳＭＴプロセッサ１３０に構築される仮想的な論理プロセッサ（以下，論理ＣＰＵと呼ぶ）へ割り当てるかを管理する手段である。クォンタムは，ＯＳが管理する論理ＣＰＵへの割り当て時間単位であり，例えば１クォンタム＝１０ｍｓとする。

【００３３】

プロセススケジューラ１００は，プロセスに何クォンタムを割り当てたかを管理するプロセス実行時間管理部１１０，ＳＭＴプロセッサ１３０の論理ＣＰＵが共用する計算資源（共有リソース）を管理するハードウェアリソース管理部１２０などを備える。

【００３４】

ディスパッチャ１４０は，タスクキュー１５０に格納される実行待機中のタスクの中からプロセススケジューラ１００の判断に従って論理ＣＰＵへ投入するタスク（プロセス）を切り替える手段である。

【００３５】
課金処理装置１は，課金情報記憶部１１，課金算出情報記憶部１３，ＣＰＵ使用時間取得部１５，競合判定部１６，ＣＰＵ情報取得部１７，ＣＰＵ使用時間換算部１８，および課金算出部１９を備える。
【００３６】
課金情報記憶部１１は，非競合ＣＰＵ使用時間，競合ＣＰＵ使用時間，課金モード，ＣＰＵ情報などの課金に使用する情報を，プロセスごとに記憶する手段である。
【００３７】
課金モードは，ＳＭＴプロセッサ１３０の複数の論理ＣＰＵがプロセスを実行し共有リソースを共用している競合状態，１つの論理ＣＰＵのみがプロセスを実行し共有リソースを専用している非競合状態のいずれかの状態を記録する項目である。
【００３８】
非競合ＣＰＵ使用時間は，ユーザプログラムのプロセスの実行において，課金モードが非競合状態で論理ＣＰＵを使用した時間を記録する項目である。競合ＣＰＵ使用時間は，ユーザプログラムのプロセスの実行において，課金モードが競合状態で論理ＣＰＵを使用した時間を記録する項目である。
【００３９】
ＣＰＵ情報は，論理ＣＰＵの性能指標であるＣＰＵ情報を記録する項目である。ＣＰＵ情報は，課金モードが非競合状態でのＣＰＵ情報（非競合ＣＰＵ情報）と競合状態でのＣＰＵ情報（競合ＣＰＵ情報）とに区別して記録される。
【００４０】
課金算出情報記憶部１３は，課金算出部１９において使用される課金処理に必要な情報および課金処理の結果である課金算出情報を記憶する手段である。課金算出情報は，プロセスごとに，競合ＣＰＵ使用時間，非競合ＣＰＵ使用時間，競合ＣＰＵ使用時間と非競合ＣＰＵ使用時間の合計，競合状態でのＣＰＵ使用時間の調整処理を行った換算時間，課金対象とする実効的なＣＰＵ使用時間である実効時間，実効時間をもとに算出した課金額などである。
【００４１】
ＣＰＵ使用時間取得部１５は，プロセス実行時間管理部１１０から，ユーザプログラムのプロセスの実行について論理ＣＰＵを使用した時間を収集し，課金モードが競合状態である場合の競合ＣＰＵ使用時間と，課金モードが非競合状態である場合の非競合ＣＰＵ使用時間とを区別して課金情報記憶部１１に格納する処理手段である。
【００４２】
競合判定部１６は，論理ＣＰＵにおいてユーザプログラムのプロセスの実行が開始される時に，他の論理ＣＰＵでプロセスが実行中である状態（競合状態）か否か（非競合状態）を判定し，判定結果（競合状態／非競合状態）を課金情報記憶部１１の課金モードに格納する処理手段である。
【００４３】
ＣＰＵ情報取得部１７は，例えば，１命令当たりのクロックサイクル数（ＣＰＩ），時間当たりの命令実行数などのＣＰＵ情報を収集し，競合状態でのＣＰＵ情報（競合ＣＰＵ情報）と非競合状態でのＣＰＵ情報（非競合ＣＰＵ情報）とに区別して課金情報記憶部１１のＣＰＵ情報に格納する処理手段である。
【００４４】
ＣＰＵ使用時間換算部１８は，ユーザプログラムが終了した後に，課金情報記憶部１１から競合ＣＰＵ使用時間を読み出し，所定の重み付け値を用いて，競合状態での実質的なＣＰＵ使用時間であるＣＰＵ換算使用時間を算出する手段である。所定の重み付け値として，例えば，競合状態においてプロセス実行が可能な論理ＣＰＵ数を用いる。
【００４５】
また，ＣＰＵ使用時間換算部１８は，課金情報記憶部１１から，ＣＰＵ情報（競合ＣＰＵ情報と非競合ＣＰＵ情報）および競合ＣＰＵ使用時間とを読み出し，競合ＣＰＵ情報と

非競合ＣＰＵ情報との比率を用いて，競合ＣＰＵ使用時間からＣＰＵ換算使用時間を算出する。また，ＣＰＵ使用時間換算部１８は，課金情報記憶部１１から読み出した情報および処理結果を課金算出情報記憶部１３へ格納する。

【００４６】

課金算出部１９は，ユーザプログラムが終了した後に，課金算出情報記憶部１３の非競合ＣＰＵ使用時間と競合ＣＰＵ使用時間から換算されたＣＰＵ換算使用時間とを合計した実効時間をもとに，所定の課金単位により課金額を算出し，課金額を課金算出情報記憶部１３へ格納する手段である。

【００４７】

以下に，第１の実施例における本発明の処理例を説明する。本例において，図１に示すＳＭＴプロセッサ１３０の２つの論理ＣＰＵを，論理ＣＰＵ＿Ｌ０および論理ＣＰＵ＿Ｌ１とする。

【００４８】

図２に，論理ＣＰＵ＿Ｌ０，ＣＰＵ＿Ｌ１のスレッドでのプロセスの実行と競合状態の例を示す。論理ＣＰＵ＿Ｌ０および論理ＣＰＵ＿Ｌ１では，所定のクォンタムでプロセスが割り当てられ実行されていく。図２中，斜線で示す部分は，プロセスが実行されているクォンタムを，白色の部分は，プロセスが実行されていないアイドル中のクォンタムを示す。論理ＣＰＵ＿Ｌ０のスレッドは，ずっとプロセスが実行中である。一方，論理ＣＰＵ＿Ｌ１のスレッドは，第1番目のクォンタムではアイドル中，第２および３番目のクォンタムでプロセス実行中，第４番目のクォンタムでは再びアイドル中であるように，アイドル状態とプロセス実行中とが繰り返されている。ＳＭＴプロセッサ１３０の２つの論理ＣＰＵでは競合状態と非競合状態が繰り返されていることになる。

【００４９】

図３に，第１の実施例における本発明の処理の流れを示す。

【００５０】

課金処理装置１の競合判定部１６は，ディスパッチャ１４０のディスパッチ時またはタイマ割り込み処理時などの所定の契機に，あるプロセスが実行可能状態となったことを検出すると（ステップＳ１），現在プロセスを実行していない論理ＣＰＵ（空きＣＰＵ）があるかどうかを調べる（ステップＳ２）。さらに，空きＣＰＵ（ここでは論理ＣＰＵ＿Ｌ１）があれば（ステップＳ２のＹＥＳ），プロセススケジューラ１００を介して，他の論理ＣＰＵ（ここでは論理ＣＰＵ＿Ｌ０）でプロセスが実行中であるか否かを調べる（ステップＳ３）。他の論理ＣＰＵ＿Ｌ０でプロセスが実行中であれば（ステップＳ３のＹＥＳ），競合判定部１６は，課金情報記憶部１１の当該プロセスの課金情報の課金モードを競合状態に設定する（ステップＳ４）。

【００５１】

そして，ＣＰＵ使用時間取得部１５はプロセス実行時間管理部１１０からのＣＰＵ使用時間（クロックサイクル数）の収集を開始し，ＳＭＴプロセッサ１３０の論理ＣＰＵ＿Ｌ１がプロセスを実行する（ステップＳ５）。ここでＣＰＵ使用時間取得部１５が収集するＣＰＵ使用時間は，課金情報記憶部１１の競合ＣＰＵ使用時間に格納される。

【００５２】

一方，他の論理ＣＰＵ＿Ｌ０でプロセスが実行中でなければ（ステップＳ３のＮＯ），ＣＰＵ使用時間取得部１５は，プロセス実行時間管理部１１０からのＣＰＵ使用時間（クロックサイクル数）の収集を開始し，論理ＣＰＵ＿Ｌ１がプロセスを実行する（ステップＳ５）。ここでＣＰＵ使用時間取得部１５が収集するＣＰＵ使用時間は，課金情報記憶部１１の非競合ＣＰＵ使用時間に格納される。

【００５３】

さらに，ＣＰＵ情報取得部１７は，ハードウェアリソース管理部１２０から，例えばＣＰＩなどのＣＰＵ情報を収集する（ステップＳ６）。ここでＣＰＵ情報取得部１７が収集するＣＰＵ情報は，課金情報記憶部１１の課金モードの設定によって，競合ＣＰＵ情報または非競合ＣＰＵ情報として課金情報記憶部１１のＣＰＵ情報に格納される。

【００５４】
なお，現在プロセスを実行していない論理ＣＰＵを調べて，空きＣＰＵがなければ（ステップＳ２のＮＯ），ディスパッチャ１４０がそのプロセス（タスク）をタスクキュー１５０に追加するので，課金処理装置１は処理を終了する（ステップＳ７）。
【００５５】
その後，競合判定部１６は，論理ＣＰＵ＿Ｌ１で実行されていたプロセスが休止状態となったことを検出すると（ステップＳ１０），プロセススケジューラ１００を介して，他の論理ＣＰＵ＿Ｌ０でプロセスが実行中であるか否かを調べる（ステップＳ１１）。他の論理ＣＰＵ＿Ｌ０でプロセスが実行中であれば（ステップＳ１１のＹＥＳ），その他の論理ＣＰＵ＿Ｌ０で実行中のプロセスの課金情報の課金モードに非競合状態を設定する（ステップＳ１２）。
【００５６】
そして，ＣＰＵ使用時間取得部１５は，プロセス実行時間管理部１１０からＣＰＵ使用時間（クロックサイクル数）の収集を停止し，ＣＰＵ情報取得部１７は，ハードウェアリソース管理部１２０からＣＰＵ情報の収集を停止し，論理ＣＰＵ＿Ｌ１で実行されていたプロセスが停止される（ステップＳ１３）。
【００５７】
一方，他の論理ＣＰＵ＿Ｌ０でプロセスが実行中でなければ（ステップＳ１１のＮＯ），ＣＰＵ使用時間取得部１５は，プロセス実行時間管理部１１０からのＣＰＵ使用時間（クロックサイクル数）の収集を停止し，ＣＰＵ情報取得部１７は，ハードウェアリソース管理部１２０からのＣＰＵ情報の収集を停止する。そして，ＳＭＴプロセッサ１３０の論理ＣＰＵ＿Ｌ１で実行されていたプロセスが停止される（ステップＳ１３）。
【００５８】
その後，ユーザプログラムの実行が終了すると，以下の課金処理が行われる。
【００５９】
１つの課金処理として，ＣＰＵ情報を用いた課金処理を説明する。
【００６０】
ＣＰＵ使用時間換算部１８は，課金情報記憶部１１から当該プロセスの課金情報を抽出する。課金情報として，非競合ＣＰＵ使用時間，競合ＣＰＵ使用時間，ＣＰＵ情報（競合ＣＰＵ情報，非競合ＣＰＵ情報）を取得する。ここで，ＣＰＵ情報としてＣＰＩが格納されているとする。
【００６１】
抽出した課金情報は以下のとおりとする：
競合ＣＰＵ使用時間＝１０［秒］
非競合ＣＰＵ使用時間＝２０［秒］
競合ＣＰＵ情報（ＣＰＩ）＝２．６
非競合ＣＰＵ情報（ＣＰＩ）＝１．１
ＣＰＵ使用時間換算部１８は，以下のようにしてＣＰＵ換算使用時間を算出する場合に用いる重み付け値を決定する：
重み付け値＝非競合ＣＰＵ情報／競合ＣＰＵ情報＝１．１／２．６＝０．４２
さらに，決定した重み付け値を用いて，競合ＣＰＵ使用時間からＣＰＵ換算使用時間を算出する：
ＣＰＵ換算使用時間＝（競合ＣＰＵ使用時間）×（重み付け値）
＝１０×０．４２＝４．２［秒］
その後，ＣＰＵ使用時間換算部１８は，課金情報記憶部１１から読み出した課金情報，算出したＣＰＵ換算使用時間を課金算出情報へ設定し，課金算出情報記憶部１３へ格納する。
【００６２】
図４に，課金算出情報の例をテーブル形式で示す。課金算出情報は，プロセスごとに，競合ＣＰＵ使用時間(図４中で「競合あり」の「ＣＰＵ時間」と示す），競合ＣＰＵ情報

（同「競合あり」の「ＣＰＩ」と示す），非競合ＣＰＵ使用時間（同「競合なし」の「ＣＰＵ時間」と示す），非競合ＣＰＵ情報（同「競合なし」の「ＣＰＩ」と示す），ＣＰＵ換算使用時間（同「換算時間」と示す），競合ＣＰＵ使用時間と非競合ＣＰＵ使用時間の合計時間（同「合計」と示す），課金対象となる実効時間，算出された課金額などの項目で構成される。なお，合計，実効時間，課金額の項目は，課金算出部１９によって処理される。

【００６３】

課金算出部１９は，課金算出情報のＣＰＵ換算使用時間と非競合ＣＰＵ使用時間とを合算して実効時間を算出し，実効時間に所定の課金単位を乗算して課金額を算出する。ここでは課金単位を１０円／秒として，プロセスｐ１を例にすると，以下のとおりとなる：

実効時間＝（ＣＰＵ換算使用時間）＋（非競合ＣＰＵ使用時間）

＝４．２＋２０＝２４．２［秒］

課金額＝（実効時間）×（課金単位）＝２４．２×１０＝２４２［円］

課金算出部１９は，算出した実効時間，課金額などを課金算出情報記憶部１３の課金算出情報に格納する。

【００６４】

これにより，競合状態において実行されたプロセスに対しても，論理ＣＰＵの競合状態をより正確に反映させ，競合状態におけるＣＰＵの性能低下を調整した公平な課金処理を行うことができる。

【００６５】

なお，詳細な実効時間を算出するために，課金情報記憶部１１内にプロセス間での組み合わせ数分だけのＣＰＵ情報（ＣＰＩ）テーブルを備えておき，それぞれの比率を用いてユーザ換算使用時間を算出するようにしてもよい。

【００６６】

次に，別の課金処理として，所定の重み付け値を用いた課金処理を説明する。

【００６７】

ＣＰＵ使用時間換算部１８は，課金情報記憶部１１から当該プロセスの課金情報を抽出する。課金情報として，競合ＣＰＵ使用時間，非競合ＣＰＵ使用時間を取得する。

【００６８】

抽出した課金情報は以下のとおりとする：

競合ＣＰＵ使用時間＝１０［秒］

非競合ＣＰＵ使用時間＝２０［秒］

ＣＰＵ使用時間換算部１８は，ＣＰＵ換算使用時間を算出する場合の重み付け値として，予め設定された，競合状態においてプロセスを同時実行しうる論理プロセッサ数を用いる。ここで，ＳＭＴプロセッサ１３０に構築される論理ＣＰＵは２つであるので，

重み付け値＝１／（論理ＣＰＵ数）＝１／２＝０．５

とする。

【００６９】

そして，競合ＣＰＵ使用時間からＣＰＵ換算使用時間を算出する：

ＣＰＵ換算使用時間＝（競合ＣＰＵ使用時間）×（重み付け値）

＝１０×０．５＝５［秒］

その後，ＣＰＵ使用時間換算部１８は，課金情報記憶部１１から読み出した課金情報，算出したＣＰＵ換算使用時間を課金算出情報記憶部１３の課金算出情報へ格納する。

【００７０】

図５に，課金算出情報の例をテーブル形式で示す。課金算出情報は，プロセスごとに，競合ＣＰＵ使用時間(図４中で「競合あり」と示す），非競合ＣＰＵ使用時間（同「競合なし」と示す），ＣＰＵ換算使用時間（同「換算時間」と示す），競合ＣＰＵ使用時間と非競合ＣＰＵ使用時間の合計時間（同「合計」と示す），課金対象となる実効時間，算出された課金額などの項目で構成される。

【００７１】

課金算出部１９は，課金算出情報のＣＰＵ換算使用時間と非競合ＣＰＵ使用時間とを合算して実効時間を算出し，実効時間に所定の課金単位を乗算して課金額を算出する。プロセスｐ１を例にすると，以下のとおりとなる：

実効時間＝（ＣＰＵ換算使用時間）＋（非競合ＣＰＵ使用時間）

＝５＋２０＝２５［秒］

課金額＝（実効時間）×（課金単位）＝２５×１０＝２５０［円］

課金算出部１９は，算出した実効時間，課金額などを課金算出情報記憶部１３の課金算出情報に格納する。

【００７２】

この場合には，競合状態と非競合状態でのＣＰＵ使用時間を区別して収集するだけでよく，図１に示すＣＰＵ情報取得部１７および図３の処理フローにおけるステップＳ６の処理は不要である。これにより，負荷の少ない処理で，競合状態において実行されたプロセスに対しても公平な課金処理を行うことができる。

【００７３】

次に，第２の実施例として，共有リソースでの状態情報を使用して課金する処理を説明する。

【００７４】

図６は，本発明の第２の実施例における構成例を示す図である。本発明にかかる課金処理装置１は，第１の実施例の場合と同様に，コンピュータのＯＳのプロセススケジューラ１００内に組み込まれるプログラムとして実施される。

【００７５】

課金処理装置１は，課金情報記憶部２１，課金算出情報記憶部２３，ＣＰＵ使用時間取得部２５，リソース状態情報取得部２７，および課金算出部２９を備える。

【００７６】

課金情報記憶部２１は，プロセスごとのＣＰＵ使用時間，リソース状態情報などの課金に使用する情報などの課金情報を記憶する手段である。

【００７７】

課金情報のＣＰＵ使用時間は，ユーザプログラムのプロセスの実行において論理ＣＰＵを使用した時間を記録する項目である。リソース状態情報は，各プロセスの実行時のＳＭＴプロセッサ１３０の共有リソースの利用状態を示す情報であり，例えば，共有リソースの各演算／処理ユニットにおけるプロセスごとの命令待ち時間などである。

【００７８】

課金算出情報記憶部２３は，課金算出部２９において使用される課金処理に必要な情報および課金処理の結果である課金算出情報を記憶する手段である。

【００７９】

課金算出情報は，プロセスごとに，ＣＰＵ使用時間，競合状態での実行と判定されたプロセスのＣＰＵ使用時間，競合状態でのＣＰＵ使用時間の調整処理を行った換算時間，課金対象とする実効的なＣＰＵ使用時間である実効時間，実効時間をもとに算出した課金額などである。

【００８０】

ＣＰＵ使用時間取得部２５は，プロセス実行時間管理部１１０から，ユーザプログラムのプロセスの実行について論理ＣＰＵを使用した時間を収集し，課金情報記憶部２１に格納する処理手段である。

【００８１】

リソース状態情報取得部２７は，ハードウェアリソース管理部１２０から，ＳＭＴプロセッサ１３０の共有リソースの各ユニットにおけるプロセスごとの命令の待ち時間（クロックサイクル数）を収集し，課金情報記憶部２１の課金情報に設定する処理手段である。

【００８２】

課金算出部２９は，課金情報記憶部２１から課金情報のリソース状態情報を読み出し，ユーザプログラムの各プロセスが，他の論理ＣＰＵがプロセスを実行中である状態（競合

状態）か否か（非競合状態）を判定し，課金情報記憶部２１から読み出したＣＰＵ使用時間を競合状態におけるＣＰＵ使用時間と非競合状態におけるＣＰＵ使用時間とに区別し，それぞれのＣＰＵ使用時間をもとに課金額を算出する処理手段である。

【００８３】

図７に，リソース状態情報取得部２７の共有リソースのリソース状態情報の収集例を示す。リソース状態情報として，例えば命令待ち時間（クロックサイクル数）を使用する場合は，演算ユニット（ＡＬＵ），浮動小数点演算ユニット（ＦＰＵ），ロード／ストアユニット（ＬＤ／ＳＴ）の各ユニットの命令スケジューリングユニット（ＳＵ）での命令待ち時間を利用する。２つの論理ＣＰＵの命令デコーダから，それぞれの命令が必要なユニットに投入される場合に，いったんＳＵに蓄えられてから命令が発行される。２つの論理ＣＰＵの命令が同じユニットへ投入されていれば，実際に発行されるまでのＳＵでの命令待ち時間が競合状態の目安を示す情報となる。また，システムバスに対しても同様に，バスリクエストキューでの命令の待ち時間を収集する。

【００８４】

リソース状態情報取得部２７は，図７にリソース状態情報テーブルとして示すような情報を課金情報記憶部２１のリソース状態情報に格納する。

【００８５】

課金算出部２９は，課金情報記憶部２１の課金情報のＣＰＵ使用時間，およびリソース状態情報を抽出する。ここで，リソース状態情報は，図７のリソース状態情報テーブルの内容とする。

【００８６】

抽出した課金情報は以下のとおりとする：

ＣＰＵ使用時間＝２４．０［秒］

ＳＭＴプロセッサ１３０の共有リソースのＡＬＵやＦＰＵなどのユニットでは競合状態と非競合状態とを明確に区別することができない。そこで，課金算出部２９は，リソース状態情報の各プロセスでの命令待ち時間が所定時間を超える場合に，当該プロセスの命令が競合によって待機させられたとみなし，プロセスの実行時のＣＰＵ使用時間を，競合状態におけるＣＰＵ使用時間であると判定する。すなわち，課金情報のＣＰＵ使用時間から，競合状態におけるＣＰＵ使用時間と判定した時間以外の時間を非競合状態におけるＣＰＵ使用時間とする。

【００８７】

ここで，ＡＬＵでの命令待ち時間から，競合状態のＣＰＵ使用時間とされた時間を「０．５秒」とし，ＦＰＵでの命令待ち時間から，競合状態のＣＰＵ使用時間とされた時間を「３．５秒」とする。

【００８８】

課金算出部２９は，競合状態および非競合状態におけるＣＰＵ使用時間を以下のように特定する：

競合状態のＣＰＵ使用時間＝０．５＋３．５＝４［秒］

非競合状態のＣＰＵ使用時間＝２４．０－４＝２０．０［秒］

そして，競合状態のＣＰＵ使用時間について，所定の重み付け値を用いてＣＰＵ換算使用時間を算出する。ここで，重み付け値は，予め設定された，競合状態においてプロセスを同時実行しうる論理プロセッサ数を用いる：

重み付け値＝１／（論理ＣＰＵ数）＝１／２＝０．５

そして，競合状態のＣＰＵ使用時間からＣＰＵ換算使用時間を算出する：

ＣＰＵ換算使用時間＝（競合ＣＰＵ使用時間）×（重み付け値）

＝４×０．５＝２［秒］

さらに，ＣＰＵ換算使用時間と非競合のＣＰＵ使用時間とを合算して実効時間を算出し，実効時間に所定の課金単位を乗算して課金額を算出する：

実効時間＝（ＣＰＵ換算使用時間）＋（非競合ＣＰＵ使用時間）

＝２＋２０＝２２［秒］

課金額＝（実効時間）×（課金単位）＝２２×１０＝２２０［円］
課金算出部２９は，課金情報，算出したＣＰＵ換算使用時間，実効時間，課金額などを課金算出情報記憶部２３の課金算出情報へ格納する。

【００８９】
図８に，課金算出情報の例をテーブル形式で示す。

【００９０】
課金算出情報は，プロセスごとに，共有リソースの各演算ユニットのリソース状態情報である命令待ち時間を利用して算出した競合状態でのＣＰＵ使用時間（図８中，例えば，「ＡＬＵ競合あり」の「ＣＰＵ時間」と示す），課金情報のＣＰＵ使用情報から，競合状態でのＣＰＵ使用時間を差し引いた非競合状態におけるＣＰＵ使用時間（同「競合なし」の「ＣＰＵ時間」と示す），課金情報のＣＰＵ使用時間（同「合計」と示す），実効時間，課金額などからなる。

【００９１】
これにより，共有リソースの各ユニットの利用状態を反映して，より公平な課金を行うことができる。

【００９２】
次に，第３の実施例として，各ユーザプログラムの処理が，物理的ＣＰＵの性能をどの程度の割合で使用していたかという情報を用いて課金する処理を説明する。

【００９３】
第３の実施例は，図１に示す構成の課金処理装置１において実施できる。本例では，ＳＭＴプロセッサ１３０で２つの論理ＣＰＵでプロセスが同時に実行される場合を想定し，その実行内容が，図４に示す課金算出情報のプロセスｐ１およびｐ２のものとする。また，ＣＰＵ情報取得部１７は，ＣＰＵ情報としてＣＰＩを収集し，課金情報記憶部１１に格納するものとする。

【００９４】
ＣＰＵ使用時間換算部１８は，課金情報記憶部１１の課金情報を抽出し，プロセスｐ１およびｐ２のそれぞれのＣＰＵ使用時間が，これらのプロセスの実行中のＣＰＵ使用時間に対して，どの程度の割合であったかを算出する：

プロセスｐ１のＣＰＵ使用時間の比率＝ＣＰＵ使用時間全体の２０％
プロセスｐ２のＣＰＵ使用時間の比率＝ＣＰＵ使用時間全体の３０％
２つのプロセスが競合しているＣＰＵ使用時間＝ＣＰＵ使用時間全体の１５％

ＣＰＵ使用時間換算部１８は，ＣＰＵ使用時間の全体（クロックサイクル数）をＣとすると，プロセスｐ１およびｐ２の実行時間，２つのプロセスが競合状態である場合の実行時間は次のようになる：

プロセスｐ１のＣＰＵ使用時間＝Ｃ×０．２
プロセスｐ１の非競合状態でのＣＰＵ使用時間＝Ｃ×（０．２－０．１５）
プロセスｐ２のＣＰＵ使用時間＝Ｃ×０．３
プロセスｐ２の非競合状態でのＣＰＵ使用時間＝Ｃ×（０．３－０．１５）
プロセスｐ１とプロセスｐ２の競合状態でのＣＰＵ使用時間＝Ｃ×０．１５

さらに，プロセスｐ１およびｐ２の実行命令数を算出する：

プロセスｐ１の実行命令数＝（Ｃ×０．１５）×（１／競合ＣＰＩ）
＋（Ｃ×０．２－０．１５）×（１／非競合ＣＰＩ），
プロセスｐ２の実行命令数＝（Ｃ×０．１５）×（１／競合ＣＰＩ）
＋（Ｃ×０．３－０．１５）×（１／非競合ＣＰＩ）

ここで，全ＣＰＵ使用時間でプロセスｐ１とｐ２とが実行されていたと仮定すると，

プロセスｐ１のＣＰＵ使用時間＝Ｃ×（１／競合ＣＰＩ）＝０．３８×Ｃ
プロセスｐ２のＣＰＵ使用時間＝Ｃ×（１／競合ＣＰＩ）＝０．３２×Ｃ

となる。すなわち，プロセスが常に実行されていた場合には，プロセスｐ１は３．７２倍の命令数を，およびｐ２は，３．５８倍の命令数を実行できる計算となる。

【００９５】

そして，この値から，プロセスｐ１およびｐ２の実質的なＣＰＵの使用率を算出する：
プロセスｐ１の実質的ＣＰＵ使用率＝（０．１０／０．３８）×１００＝２６．３％
プロセスｐ２の実質的ＣＰＵ使用率＝（０．０９／０．３２）×１００＝２８．１％
課金算出部１９は，ＣＰＵ使用時間換算部１８が算出したプロセスｐ１およびｐ２の実質的ＣＰＵ使用率を用いて，予め定めておいたＣＰＵ使用率当たりの課金単価をもとに課金額を算出する。これにより，より精度の高いＣＰＵ使用率を用いて課金を行うことができる。

【００９６】

なお，ＣＰＵ使用時間換算部１８によって算出されるプロセスｐ１およびｐ２の実質的ＣＰＵ使用率は，ＳＭＴプロセッサの性能予測処理にも使用することができる。例えば，プロセッサの性能予測処理において，従来のような単純な使用率の測定法による場合は，プロセスｐ１のＣＰＵ使用時間の比率は２０％となることから，性能予測は５倍と算出される。しかし，実質的なＣＰＵ使用時間の比率は，２６．３％であり，性能予測は５倍ではなく，約３．８倍にしかならないことがわかる。精度の高い実質的ＣＰＵ使用率によるＣＰＵ性能予測処理は，例えば，計測した実質的ＣＰＵ使用率が一定の値以上であれば，ユーザにＣＰＵの負荷状態についてより正確に警告を行うシステムなどを構成することも可能となる。

【００９７】

さらに，上述した種々の実施例において説明した課金処理装置１は，算出した実効使用時間を用いてＳＭＴプロセッサ１３０で実行されるユーザプログラムの利用課金額を算出するプログラム利用課金算出部を備えることができる。

【００９８】

プログラム利用課金算出部は，ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して前記ユーザプログラムの利用課金額を算出する。これにより，ＳＭＴプロセッサにおけるアプリケーションプログラムなどの利用料金についても課金を行うことができる。

【００９９】

以上，本発明をその実施の形態により説明したが，本発明はその主旨の範囲において種々の変形が可能であることは当然である。

【０１００】

例えば，第１の実施例などにおいて，ＣＰＵ使用時間取得部１５は，プロセス実行時間管理部１１０から，各プロセスのＣＰＵごとのユーザ使用時間（ＣＰＵ使用時間）を収集し，課金モードの設定に従って，収集したＣＰＵ使用時間を，課金情報記憶部１１の競合ＣＰＵ使用時間または非競合ＣＰＵ使用時間のいずれかに格納するものとして説明した。しかし，課金情報記憶部１１にユーザプログラムのプロセスで使用した全ＣＰＵ使用時間を非競合ＣＰＵ使用時間の代りに設けて，プロセスごとの全ＣＰＵ使用時間および非競合ＣＰＵ使用時間とを収集し格納するようにしてもよい。図９（Ａ）に，この場合のＣＰＵ使用時間取得部１５の例を示す。また，図９（Ｂ）に課金情報記憶部１１の課金情報の例を示す。課金情報は，プロセスごとに，全ＣＰＵ使用時間としてＣＰＵごとのユーザ使用時間，非競合ＣＰＵ使用時間としてＣＰＵごとの競合していたユーザ使用時間，課金モードを格納する。

【０１０１】

本発明の形態および実施例の特徴は，以下のとおりである。

【０１０２】

（付記１）　ＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行う装置であって，

論理プロセッサでユーザプログラムのプロセスの実行が開始される時に，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記判定結果として競合状態または非競合状態のいずれかの状態を競合状態

記憶部に設定する競合判定部と，
前記ユーザプログラムのプロセスが論理プロセッサを使用した時間を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ使用時間と，前記非競合状態での実行における非競合ＣＰＵ使用時間とを区別してＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得部と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間および前記非競合ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して課金額を算出する課金算出部とを備える
ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１０３】
（付記２）　付記１記載のＳＭＴプロセッサ用課金処理装置において，
所定の重み付け値を用いて，前記競合ＣＰＵ使用時間をもとに，前記ユーザプログラムのプロセス実行時の実質的な使用時間であるＣＰＵ換算使用時間を算出する使用時間換算部を備えるとともに，
前記課金算出部は，前記ＣＰＵ換算使用時間および前記ＣＰＵ使用時間記憶部から読み出した前記非競合ＣＰＵ使用時間をもとにＣＰＵ実効使用時間を算出し，所定の課金単価によって前記ＣＰＵ実効使用時間から課金額を算出する
ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１０４】
（付記３）　付記２記載のＳＭＴプロセッサ用課金処理装置において，
前記所定の重み付け値として，競合状態においてプロセスを同時に実行した論理プロセッサ数を用いる
ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１０５】
（付記４）　付記２記載のＳＭＴプロセッサ用課金処理装置において，
前記ユーザプログラムのプロセスの実行の開始から，前記プロセスを実行する論理プロセッサの性能指標であるＣＰＵ情報を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ情報と前記非競合状態での実行における非競合ＣＰＵ情報とを区別してＣＰＵ情報記憶部に格納するＣＰＵ使用情報取得部を備えるとともに，
前記使用時間換算部は，前記ＣＰＵ情報記憶部から読み出した前記競合ＣＰＵ情報および前記非競合ＣＰＵ情報の比率を用いて，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間をもとにＣＰＵ換算使用時間を算出する
ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１０６】
（付記５）　付記４記載のＳＭＴプロセッサ用課金処理装置において，
前記ＣＰＵ情報取得部は，前記ＣＰＵ情報として，１命令当たりのクロックサイクル数またはクォンタム当たりの命令実行数のいずれかの情報を収集する
ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１０７】
（付記６）ＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行う装置であって，
論理プロセッサでユーザプログラムの各プロセスを実行中のＳＭＴプロセッサのハードウェアリソースの利用状態に関するリソース状態情報を収集し，リソース状態情報記憶部に格納するリソース状態情報取得部と，
前記ユーザプログラムのプロセスが前記論理プロセッサを使用したＣＰＵ使用時間を収集しＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得部と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記リソース状態情報取得部から読み出した前記各プロセスのリソース状態情報を用いて，前記ユーザプログラムの各プロセスが，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態

ではない非競合状態であるかを判定し，前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間を前記競合状態におけるＣＰＵ使用時間と前記非競合状態におけるＣＰＵ使用時間とに区別し，前記競合状態および前記非競合状態のＣＰＵ使用時間を区別して課金額を算出する課金算出部とを備える

ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１０８】

（付記７）　付記６記載のＳＭＴプロセッサ用課金処理装置において，

前記リソース状態情報取得部は，論理プロセッサでユーザプログラムのプロセスが実行される場合に，前記プロセスの命令が，前記ハードウェアリソースの少なくとも演算ユニットまたは処理ユニットのいずれかのユニットに対して投入されるまでの待ち時間を収集し，前記待ち時間をリソース状態情報として前記リソース状態情報記憶部に格納し，

前記課金算出部は，前記リソース状態情報記憶部から読み出した前記各プロセスの待ち時間が，所定時間以上である場合に前記プロセスの実行が競合状態であると判定し，所定の重み付け値を用いて前記競合状態のＣＰＵ使用時間をもとに，前記ユーザプログラムのプロセス実行時の実質的な使用時間であるＣＰＵ換算使用時間を算出する

ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１０９】

（付記８）　付記６または付記７のいずれか一項に記載のＳＭＴプロセッサ用課金処理装置において，

前記課金算出部は，所定の重み付け値を用いて，前記競合状態のＣＰＵ使用時間をもとにＣＰＵ換算使用時間を算出し，前記ＣＰＵ換算使用時間および前記非競合状態のＣＰＵ使用時間をもとに，ＣＰＵ実効使用時間を算出し，所定の課金単価によって前記ＣＰＵ実効使用時間から課金額を算出する

ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１１０】

（付記９）　付記２記載のＳＭＴプロセッサ用課金処理装置において，

前記所定の重み付け値として，競合状態においてプロセスを同時に実行した論理プロセッサ数を用いる

ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１１１】

（付記１０）　付記１または付記６のいずれか一項に記載のＳＭＴプロセッサ用課金処理装置において，

前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して前記ユーザプログラムの利用課金額を算出するプログラム利用課金算出部を備える

ことを特徴とするＳＭＴプロセッサ用課金処理装置。

【０１１２】

（付記１１）　コンピュータのＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行うための処理方法であって，

前記コンピュータが，

論理プロセッサでユーザプログラムのプロセスの実行が開始される時に，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記判定結果として競合状態または非競合状態のいずれかの状態を競合状態記憶部に設定する競合判定処理過程と，

前記ユーザプログラムのプロセスが論理プロセッサを使用した時間を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ使用時間と前記非競合状態での実行における非競合ＣＰＵ使用時間とを区別してＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得処理過程と，

前記ユーザプログラムのプロセスの実行が終了した後に，前記ＣＰＵ使用時間記憶部か

ら読み出した前記競合ＣＰＵ使用時間および前記非競合ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して課金額を算出する課金算出処理過程とを有する
ことを特徴とするＳＭＴプロセッサ用課金処理方法。

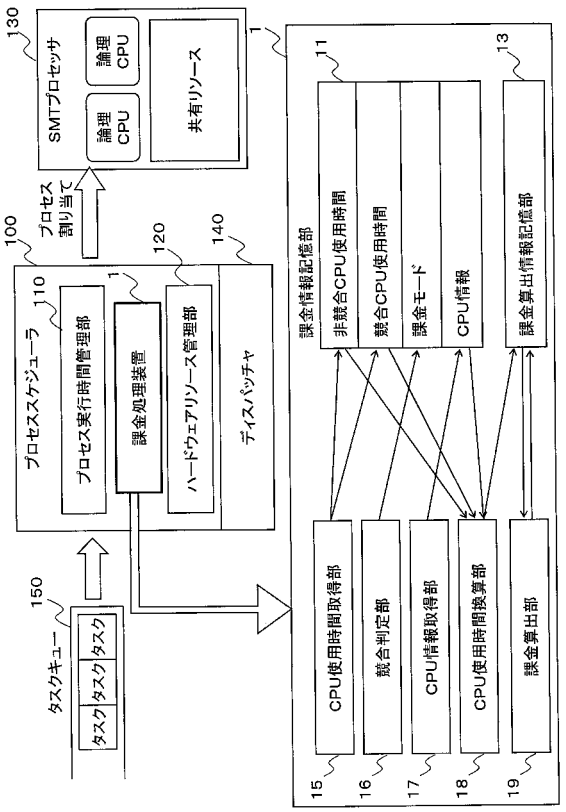
【０１１３】
（付記１２）　コンピュータのＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行うための処理方法であって，
前記コンピュータが，
論理プロセッサでユーザプログラムの各プロセスを実行中のＳＭＴプロセッサのハードウェアリソースの利用状態に関するリソース状態情報を収集し，リソース状態情報記憶部に格納するリソース状態情報取得処理過程と，
前記ユーザプログラムのプロセスが前記論理プロセッサを使用したＣＰＵ使用時間を収集しＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得処理過程と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記リソース状態情報取得部から読み出した前記各プロセスのリソース状態情報を用いて，前記ユーザプログラムの各プロセスが他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間を前記競合状態におけるＣＰＵ使用時間と前記非競合状態におけるＣＰＵ使用時間とに区別し，前記競合状態および前記非競合状態のＣＰＵ使用時間を区別して課金額を算出する課金算出処理過程とを有する
ことを特徴とするＳＭＴプロセッサ用課金処理方法。

【０１１４】
（付記１３）　コンピュータのＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行うための処理方法をコンピュータに実行させるためのプログラムであって，
前記コンピュータに，
論理プロセッサでユーザプログラムのプロセスの実行が開始される時に，他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記判定結果として競合状態または非競合状態のいずれかの状態を競合状態記憶部に設定する競合判定処理過程と，
前記ユーザプログラムのプロセスが論理プロセッサを使用した時間を収集し，前記競合状態記憶部を参照して，前記競合状態での実行における競合ＣＰＵ使用時間と前記非競合状態での実行における非競合ＣＰＵ使用時間とを区別してＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得処理過程と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記ＣＰＵ使用時間記憶部から読み出した前記競合ＣＰＵ使用時間および前記非競合ＣＰＵ使用時間にもとづいて，前記競合状態におけるユーザプログラムの実行と前記非競合状態におけるユーザプログラムの実行とを区別して課金額を算出する課金算出処理過程とを実行させるための
ＳＭＴプロセッサ用課金処理プログラム。

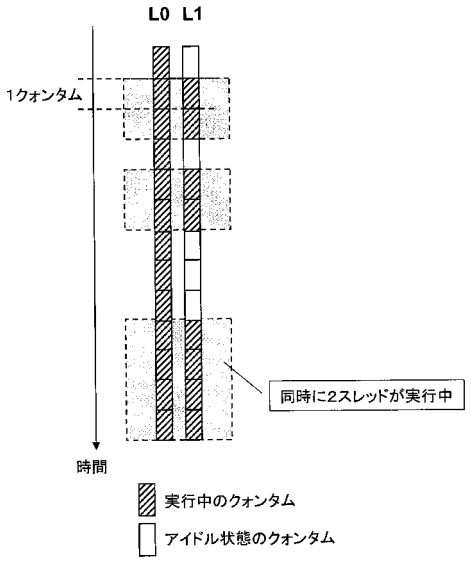
【０１１５】
（付記１４）　コンピュータのＳＭＴプロセッサで実行されるユーザプログラムに対する課金処理を行うための処理方法をコンピュータに実行させるためのプログラムであって，
前記コンピュータに，
論理プロセッサでユーザプログラムの各プロセスを実行中のＳＭＴプロセッサのハードウェアリソースの利用状態に関するリソース状態情報を収集し，リソース状態情報記憶部に格納するリソース状態情報取得処理過程と，
前記ユーザプログラムのプロセスが前記論理プロセッサを使用したＣＰＵ使用時間を収集しＣＰＵ使用時間記憶部に格納するＣＰＵ使用時間取得処理過程と，
前記ユーザプログラムのプロセスの実行が終了した後に，前記リソース状態情報取得部

から読み出した前記各プロセスのリソース状態情報を用いて，前記ユーザプログラムの各プロセスが他の論理プロセッサでプロセスが実行中である競合状態または前記競合状態ではない非競合状態であるかを判定し，前記ＣＰＵ使用時間記憶部から読み出した前記ＣＰＵ使用時間を前記競合状態におけるＣＰＵ使用時間と前記非競合状態におけるＣＰＵ使用時間とに区別し，前記競合状態および前記非競合状態のＣＰＵ使用時間を区別して課金額を算出する課金算出処理過程とを実行させるための
　ＳＭＴプロセッサ用課金処理プログラム。
【図面の簡単な説明】
【０１１６】
【図１】本発明の第１の実施例における構成例を示す図である。
【図２】論理ＣＰＵ＿Ｌ０，ＣＰＵ＿Ｌ１でのプロセスの実行と競合状態の例を示す図である。
【図３】第１の実施例における本発明の処理の流れを示す図である。
【図４】第１の実施例におけるＣＰＵ情報を含む課金算出情報の例をテーブル形式で示す図である。
【図５】第１の実施例における課金算出情報の例をテーブル形式で示す図である。
【図６】本発明の第２の実施例における構成例を示す図である。
【図７】リソース状態情報取得部の共有リソースのリソース状態情報の収集例を示す図である。
【図８】第２の実施例における課金算出情報の例をテーブル形式で示す図である。
【図９】本発明の他の実施例におけるＣＰＵ使用時間取得部および課金情報記憶部の課金情報の例を示す図である。
【図１０】ＳＭＴプロセッサおよび非ＳＭＴプロセッサのプロセス実行の相違を説明するための図である。
【符号の説明】
【０１１７】
１　課金処理装置
１１　課金情報記憶部
１３　課金算出情報記憶部
１５　ＣＰＵ使用時間取得部
１６　競合判定部
１７　ＣＰＵ情報取得部
１８　ＣＰＵ使用時間換算部
１９　課金算出部
２１　課金情報記憶部
２３　課金算出情報記憶部
２５　ＣＰＵ使用時間取得部
２７　リソース状態情報取得部
２９　課金算出部
１００　プロセススケジューラ
１１０　プロセス実行時間管理部
１２０　ハードウェアリソース管理部
１３０　ＳＭＴプロセッサ
１４０　ディスパッチャ
１５０　タスクキュー

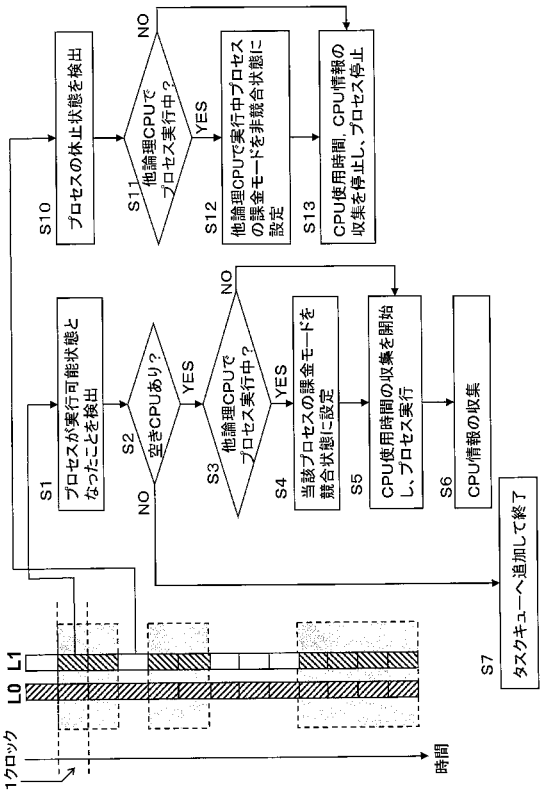
【図１】



【図２】



【図３】



【図４】

	競合あり		競合なし		合計 [sec]	換算時間 [sec]	実効時間 [sec]	課金額 [円]
	CPU時間 [sec]	CPI	CPU時間 [sec]	CPI				
プロセスp1	10	2.6	20	1.1	30	4.2	24.2	242
プロセスp2	320	3.1	40	1.2	360	123.9	163.9	1,639
プロセスp3	200	6.7	100	2.0	300	59.7	159.7	1,597

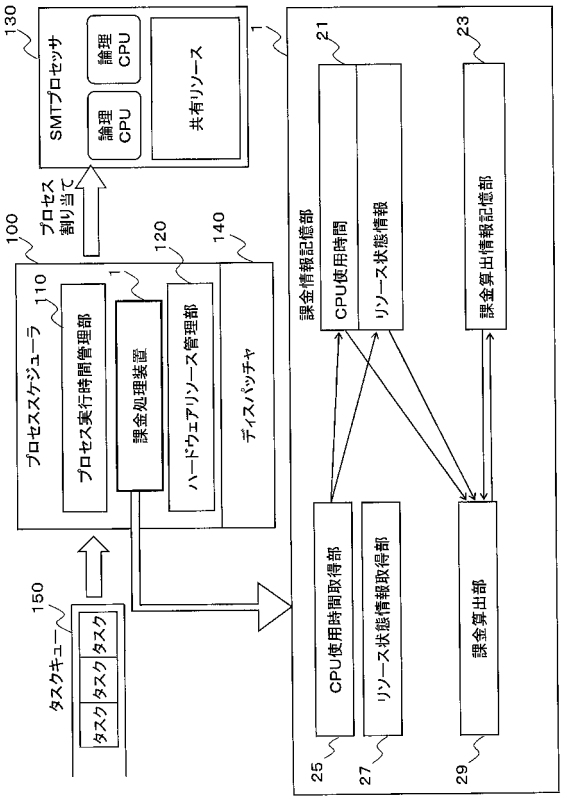
課金：10円／sec

【図５】

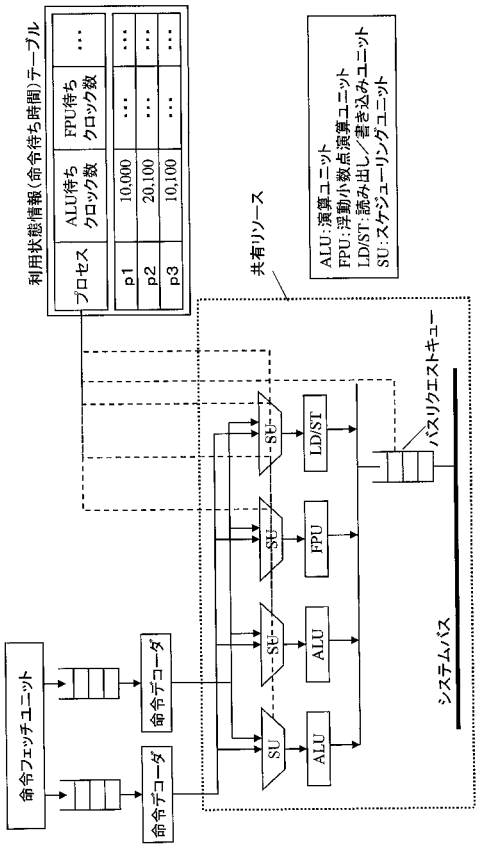
	競合あり [sec]	競合なし [sec]	合計 [sec]	換算時間 [sec]	実効時間 [sec]	課金額 [円]
プロセスp1	10	20	30	5	25	250
プロセスp2	320	40	360	160	200	2,000
プロセスp3	200	100	300	100	200	2,000

課金:10円／sec

【図６】



【図７】



【図８】

	ALU競合あり	FPU競合あり	…	競合なし	合計 [sec]	実効時間 [sec]	課金額 [円]
	CPU時間 [sec]	CPU時間 [sec]	…	CPU時間 [sec]			
プロセス1	0.5	3.5	…	20.0	24.0	22.0	220
プロセス2	200.5	1.0	…	40.0	241.5	140.8	1,408
プロセス3	120.5	0.0	…	100.0	220.5	160.3	1,603

課金:10円／sec

【図９】

(A)

```
void update_one_process(struct task_struct *p,
struct kernel_stat_tick_times *time, int cpu)
{
kernel_timeval_add_usec(&p->per_cpu_utime[cpu],
time->u_usec +time->n_usec);
   if (p->isConflict)
kernel_timeval_add_usec(&p->conf_per_cpu_utime[cpu],
time->u_usec + time->n_usec);
do_process_times(p, time);
}
```

(B)

task_struct 構造体

kernel_timeval per_cpu_utime[]

(CPUごとのユーザ使用時間)

kernel_timeval conf_per_cpu_utime[]

(CPUごとの競合していたユーザ使用時間)

int isConflict

(課金モード:現在論理CPU間で競合しているか)

【図１０】