マルチOS環境におけるアカウンティングシステムの実装

Implementation of Accounting System on Multi-OS Systems

巻島 一雄 Kazuo Makijima
杵渕 雄樹 Yuki Kinebuchi
菅谷 みどり Midori Sugaya
中島 達夫 Tatsuo Nakajima

早稲田大学 基幹理工学部 情報理工学科
Department of Computer Science Waseda University

概要

マルチOS環境にアカウンティングシステムを実装して，ゲストOSのCPU使用率を制限できるようにした．マルチOS環境の構築者はアカウンティングシステムを利用する事により，ゲストOSにCPU使用率の上限を設定する事ができる．アカウンティングシステムはゲストOSのCPU使用時間を記録し，CPU使用率の上限を超過しないように監視する．この機能によって1つのOSがCPUを占有してしまう事態を回避して，他のゲストOSがCPUを使える事を保障する．

1 序論

現在の情報家電が必要とする機能は多岐に渡っている．たとえば携帯電話ではリアルタイム性とGUI・ネットワーク・マルチメディア処理などが必要とされる．これらを実現するために組み込みシステムではマルチOS環境が広く使われている．組み込みシステム開発者は1つのシステム上にRTOSとWindowsやLinuxといった汎用OSを乗せることで，リアルタイム性を確保し，かつ汎用OSの豊富なソフトウェアが使用できるシステムを作る．しかしこうした様々なマルチOS環境の内，プロセッサが1つのシステムにおいては，RTOSがCPUを長い間占有して汎用OSに制御を渡さない事態が起こりうる．つまりリアルタイム性を保障するために，汎用OS側のCPU利用時間が保障されない事がある．

こうした状況を回避するためにはCPUの利用を制限する機能を作る必要がある．たとえばLinuxではプロセスごとにCPU使用率の上限を定めるアカウンティングシステム[1][2]が存在する．本研究ではこのアカウンティングシステムをSPUMONEという仮想マシンモニタ上に移植し，ゲストOSごとにCPU使用率を制限できるようにした．これによりリアルタイム性と汎用OSの実行時間の保障を両立するシステムを作った．

2 マルチOS環境

システムの構築に当たって，筆者が所属する中島研究室で開発中のソフトウェアSPUMONEを使った．SPUMONEは仮想マシンモニタであり，複数の仮想的なCPUをゲストOSに提供する．SPUMONEを用いて本研究ではRTOSであるTOPPERS/JSPと汎用OSであるLinuxを1つのCPUで動かす．

2.1 SPUMONEのスケジューリング方式

SPUMONEは固定優先度でゲストOSのスケジューリングを行う．すなわちTOPPERSの優先度が高く，Linuxが動作するのはTOPPERSがスリープ命令を発行した時に限られる．再びTOPPERSが動作を開始するのはTOPPERSへの割り込みが起こった時であり，TOPPERSへの割り込みはLinuxの動作中でも受付可能な状態になっている．TOPPERSが動作している間はLinuxが動かず，Linuxへの割り込みも起こらない．これによりTOPPERSのリアルタイム性が保障される．

2.2 問題点

この仕組みではTOPPERSがCPUを占有してしまう状況が起こる．例えばTOPPERSへの割り込みが短い間隔で起こり続けた時（図1）である．そこで筆者はアカウンティングシステムの機能をSPUMONEに追加して解決を図った．（図2）

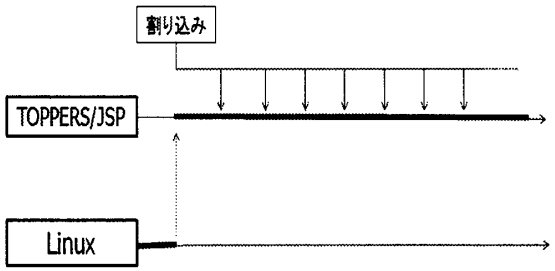

図1 Linuxが長時間動かないスケジューリング例

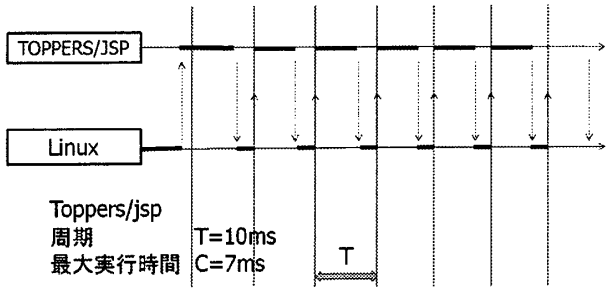

図2 アカウンティングシステムを導入した例

3 アカウンティングシステム

3.1 仕様

SPUMONEを使ってシステムを開発する者は，周期と最大実行時間の2つの値をゲストOS1つに対して定める．ゲストOSのCPU使用率の上限は最大実行時間/周期となる．

ゲストOSがひとつの周期内でCPUを使うことができる時間はこの最大実行時間までである．アカウンティングシステムはゲストOSがCPUを使用した時間を記録する．この値までCPUを使ったゲストOSは，アカウンティングシステムによってCPUの利用が不可能な状態に置かれる．

周期が終わる度にCPUの利用が不可能な状態は解除され，記録されていたCPU使用時間は0にリセットされる．したがって再び最大実行時間までCPUを使うことができるようになる．

3.2 実装

アカウンティングシステムは周期が終わるタイミングとゲストOSが最大実行時間を超過するタイミングで処理を行う必要がある．そのため実装にはタイマ割り込みが必要となる．

アカウンティングシステムはゲストOSの実行時間超過を阻止するために，ゲストOSが最大実行時間を超過した時にタイマ割り込みが起こるようにする．そしてこのタイマ割り込みハンドラ内で，実行中のゲストOSがプリエンプトされる．

また周期の長さをTとするとアカウンティングシステムは時間Tが経過する度にタイマ割り込みを入れて，CPU利用禁止状態の解除とCPU使用時間のリセットを行う．

しかしアカウンティングシステムだけで複数のタイマデバイスを占有するとゲストOSが充分な数のタイマデバイスを利用できなくなる．そのため筆者は1つのタイマデバイスを，複数のワンショットモードのタイマに見せかけて動作するように実装した．これを周期のタイマ，ゲストOSの実行時間超過阻止のタイマとして利用した．

また既存のゲストOSスイッチを行う箇所に新しく関数呼び出しを挿入した．以下この関数をフックと呼ぶ．フック内ではゲストOSのCPU利用時間を記録し，また切り替え先のゲストOSに対して実行時間の超過を阻止するためのタイマをセットする．このタイマの割り込みハンドラ内で，実行中のゲストOSがプリエンプトされる．

4 評価

4.1 動作

TOPPERSの周期を10ms，最大実行時間を7msに設定して，TOPPERSが暴走状態にあってもLinuxのシェルを快適に動かせる事が確認できた．

4.2 オーバーヘッド

ゲストOSスイッチの処理時間が増えると予想される．なぜならOSスイッチ時にフックが呼ばれるためである．またアカウンティングシステムによるゲストOSスイッチは，本研究によって新しく増えたオーバーヘッドである．

4.3 計測値

表1にオーバーヘッドの計測値を示す。

表1 アカウンティングシステムのオーバーヘッドの計測値

アカウンティングシステムによるスイッチの処理時間の平均値	3.77 μs
フックの処理時間の平均値	1.58μs

5 結論

アカウンティングシステムを実装してゲストOSごとにCPU使用率を制限できるようにした．またオーバーヘッドの値は小さく，リアルタイム性と汎用OSの実行時間の保障を両立するシステムを作ることができた．

参考文献

[1] Midori Sugaya, Shuichi Oikawa, Tatsuo Nakajima: "Accounting System: A fine-granined CPU resource protection mechanism for Embedded System", In Proceedings of ISORC 2006: pp.72-84

[2] Midori Sugaya, Takeharu KATO, Tatsuya Ohigashi, Yoichi Yuasa: "Linux へのQOS(Quality Of Service)機能の実装および標準化 概要設計書 ver1.10", http://sourceforge.jp/projects/cabi/document/specification/ja/1/specification.pdf