SPUMONE:軽量な CPU 仮想化手法

SPUMONE:Light weight CPU virtualization layer

湯村 悠 (Yu Yumura) 早稲田大学
神田 渉 (Wataru Kanda) 早稲田大学
香取 知浩 (Tomohiro Katori) 早稲田大学
杵渕 雄樹 (Yuki Kinebuchi) 早稲田大学
中島 達夫 (Tatsuo Nakajima) 早稲田大学

1 はじめに

近年，組み込み機器の高性能化，多機能化が進み，それに伴いソフトウェア開発のコストが問題になっている．このような状況を受け，組み込みシステムのプラットフォーム化が進められている．この一手法として Linux などの既存の汎用 OS を組み込み機器において利用する事例が増加している．しかし，汎用 OS は組み込み機器での使用を前提としていないため，組み込み機器で要求されるリアルタイム応答性を考慮されていない．そのため，既存の汎用 OS で組み込み機器に求められる高いリアルタイム応答性を達成することは困難である．この問題を解決するため汎用 OS をリアルタイム処理に対応するよう拡張する取り組みがなされているが，結果としてシステムの複雑化，不安定化を招いている．

2 提案手法

仮想化技術を利用したマルチ OS システムは，開発コストの問題に対する有効な解決策であると考えられる．ハードウェアを仮想化しリアルタイム OS と汎用 OS を同時に動作させることで，既存のソフトウェア資産を利用しつつリアルタイム応答性と高機能性が両立が可能となると思われる．

しかし，既存の仮想化技術である仮想マシンモニタなどはリアルタイム OS を動作させることを考慮して設計されていないため，ゲスト OS のリアルタイム応答性を保証することはできない．そこで本研究では軽量な CPU 仮想化手法 SPUMONE を提案する。

SPUMONE(Software Processing Unit, Multiplexing ONE into two)は CPU を仮想化し複数の OS を 1 つの CPU 上で動作させることを可能とする．SPUMONE を使ったシステムの例を図 1 に示す．

SPUMONE はできるだけ軽量にするというコンセプトに基づき，CPU のみを仮想化するという方法をとっている．これにより，仮想化のオーバーヘッドを最小限にし，ゲスト OS がリアルタイム応答性を損なわず動作することを可能とする．また特権命令のエミュレーションによるオーバーヘッドを避けるためゲスト OS はそれぞれ特権状態で動作することを許している．これはゲスト OS の修正コストを小さくするという点でも有効である．[1]

SPUMONE は CPU のみを仮想化し，基本的にデバイスの仮想化はしない．そのため，ゲスト OS 間でデバイスの共有を行わないようにする必要がある．しかし，仮想デバイスを実装することにより，OS 間でデバイスを共有することも可能である．

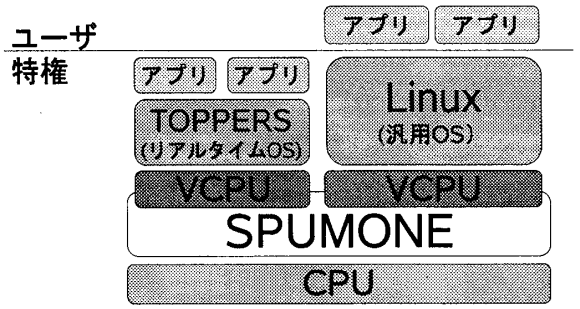

図 1:SPUMONE システム概要

3 実装

SH-4A アーキテクチャの CPU 上で SPUMONE を実装した．ゲスト OS としてリアルタイム OS である TOPPES/JSP と、汎用 OS である Linux を利用したマルチ OS システムを構築した．

3.1 SPUMONE

SPUMONE の機能としては，ゲスト OS の起動，ゲスト OS のスケジューリング，割り込みの振り分け，がある．

SPUMONE はゲスト OS のイメージをメモリ上に配置し，それぞれの仮想 CPU が OS のエントリポイントから実行するように設定する．

ゲスト OS のスケジューリングは固定優先度で行われ，リアルタイム OS が高優先度，汎用 OS が低優先度で動作するようになっている．

SPUMONE は割り込みを受けとり，それぞれのゲスト OS に振り分けを行う．SPUMONE のゲスト OS のスケジューリングはこの割り込みの振り分けの際に行われる．SPUMONE は割り込みが発生するとあらかじめ登録された情報をもとに割り込みを振り分ける．その際ゲスト OS の切り替えが必要となる場合，SPUMONE は仮想 CPU の切り替えを行う．CPU の切り替え時は，レジスタ群など必要な値を仮想 CPU 構造体に保存し，次に起動するゲスト OS 用の仮想 CPU 構造体の値をロードすることによって行われる．

割り込みの振り分けの設定はビルド時に行うか，動作時にAPIを呼び出すことによって行うことができる．

また汎用OSへの割り込みによってリアルタイムOSから汎用OSへ制御が移らないように，それぞれのゲストOSの割り込み優先度を変更している．

3.2 ゲストOSの変更

ゲストOSの変更点としては物理アドレス上の配置位置，割り込み優先度の設定および，割り込みベクターの登録，一部の特権命令の置き換えがある．

それぞれのゲストOSは使用する物理アドレスの領域が衝突しないように配置位置および使用領域を変更する必要がある．

割り込みはSPUMONEが受け取る必要があるので，割り込みベクターの登録をSPUMONEへの通知APIに置き換える．これにより通知された値を元にSPUMONEは割り込みを振り分ける．

ゲストOSのスケジューリングを実現するために，それぞれのゲストOSのつかうデバイスの割り込み優先度を変更する必要がある．

リアルタイムOSのSLEEP命令をSPUMONEのAPIへ置き換えている．このAPIが呼ばれると，SPUMONEはリアルタイムOSがアイドル状態になったと判断し，汎用OSへ制御を移す．これによりリアルタイムOSがアイドル状態の時のみ汎用OSが動作するというスケジューリングを実現している．

4 評価と考察

SPUMONEの有効性を示すため仮想化によるオーバーヘッドの大きさと開発にかかるコストの評価を行った．評価で使用した環境を次の表1に示す．

表 1:評価環境のスペック

デバイス	SH-2007
CPU	SH7780
CPUクロック数	400MHz
メモリ	128MB
ゲストOS	TOPPERS/JSP 1.3 Linux 2.6.20.1

SPUMONEによる仮想化のオーバーヘッドを評価するためにTOPPERS上でシリアル割り込みの遅延の計測を行った．割り込みが発生してから，実際に割り込みハンドラが呼ばれるまでにかかったクロック数の計測結果を次の表2に示す．

表 2:割り込み遅延の計測結果

環境		クロック数	時間(us)
TOPPERS	平均値	8144	0.2
	最悪値	260	0.65
TOPPERS and Linux on SPUMONE	平均値	348.46	0.87
	最悪値	37135	92.84
TOPPERS and Linux with stress on SPUMONE	平均値	351.78	0.88
	最悪値	24819	62.05

この計測によりSPUMONEのオーバーヘッドは平均的には小さいことがわかった．これはリアルタイムOSの性能を損なわないのに十分だと考えられる．しかし最悪値のオーバーヘッドは非常に大きくなっており，現在のままではこのシステムがリアルタイム応答性を満たしているとは言い難い．これは汎用OSがリアルタイムOSの動作に影響を与えているためと考えられ，ゲストOSの振る舞いを解析しリアルタイム応答性を高める必要がある．

次に開発コストを評価するため，ゲストOSの変更量を調べた．それぞれのゲストOSの変更量を表3に示す．

表 3:ゲストOSの変更量

ゲストOS	変更量(行)
Linux 2.6.20.1	7
TOPPERS/JSP 1.3	14

開発コストに関してはゲストOSの変更量が少なく，SPUMONEは有効な手法であると考えられる．

5 まとめ

本研究はリアルタイム応答性を保証できるような，軽量なCPU仮想化手法，SPUMONEを提案、実装、評価した．

評価によりSPUMONEは組み込みシステムにおけるリアルタイム応答性と高機能性という相反する要求を満たすシステムを、過去の資産を活用することにより，低コストで実現するのに有効な手法であることが示せた．

しかし，現時点ではゲストOSに対してリアルタイム応答性を保証できていないので，実用化するためにはさらなるゲストOSの解析が必要であると考えられる．

参照

[1]Yu Murata, Wataru Kanda, Kensuke Hanaoka, Hiroo Ishikawa, Tatsuo Nakajima: A Study on Asymmetric Operating Systems on Symmetric Multiprocessors. EUC 2007: 182-195