1J-5

組み込み用途向けSMP環境におけるリアルタイム性能の向上

斎藤奨悟† 追川 修一‡
†筑波大学情報学群情報科学類 ‡筑波大学コンピュータサイエンス専攻

1 はじめに

近年、組み込み機器に高度な機能を持ったオペレーティングシステムを採用する事が一般的になりつつある．それに伴い，組み込み機器向けのプロセッサも汎用アーキテクチャ用プロセッサと同様に，オペレーティングシステムを支援する機能を持つ事が増えてきている．本研究では，広く普及している組み込み向けアーキテクチャである ARM アーキテクチャ上に小型のオペレーティングシステムを移植し，これらの機能について考察する．

2 研究概要

本研究では，小型のオペレーティングシステムを ARM9 アーキテクチャに移植し，性能の測定を行った．拡張する小型のオペレーティングシステムには，xv6 を利用する．xv6 は UNIX version6 をベースに開発された，x86 マルチプロセッサアーキテクチャ上で動作する教育用の汎用オペレーティングシステムである．xv6 は非常にシンプルな構造を持つため、独自の拡張に適している．動作の検証には，QEMU によるフルシステムエミュレーションを利用した．QEMU はプロセッサおよび，周辺のシステムをエミュレーションするソフトウェアである．

3 ARM アーキテクチャの特徴

ARM9 アーキテクチャは，汎用機において広く普及している x86 アーキテクチャと同様に，MMU(Memory Management Unit) による仮想メモリ機能を提供している．ARM9 アーキテクチャ特有の MMU の拡張機能として，FCSE(Fast Context Switch Extension) が挙げられる．通常，仮想メモリ空間の切り替えは，ページテーブルの書き換えおよび，ページテーブルのキャッシュである TLB のクリアを必要とする．これらの処理には数サイクルの命令が必要となり，また切り替え後の TLB ミスに伴うオーバーヘッドが発生する．ARM アーキテクチャでは，pid と呼ばれる値を元に，FCSE が仮想アドレスの修飾を行うことができる．具体的には，仮想アドレスの上位 7bit が pid の値に変換されてから MMU に渡され，ページテーブルの参照および物理アドレス

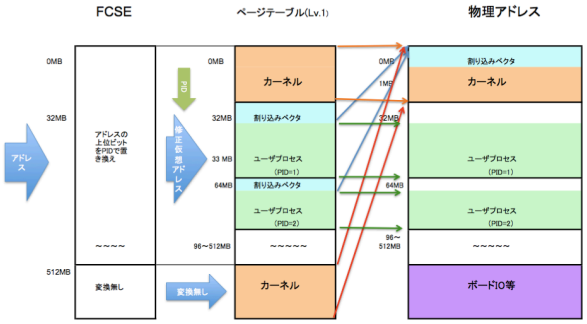

図 1: メモリマップ

への変換が行われる．この際，FCSE はキャッシュに含まれる pid の値を考慮し仮想アドレスと実アドレスの対応付けを行うため，従来の MMU によるコンテキストスイッチに必要な TLB のクリア，メモリ空間の書き換えが不必要となり，高速なコンテキストスイッチが可能となる．FCSE の欠点として，pid の数に上限があるため，一定数以上のコンテキストを扱う場合にページテーブルの書き換えが必要となる点，アドレス空間が 32MB で区切られるため，広いメモリ空間を扱う必要があるシステムに適さない事が挙げられる．本研究では，コンテキストスイッチにおいて FCSE を利用し，コンテキストスイッチのオーバーヘッドの削減について検証を行った．

4 xv6 の ARM9 への移植

本研究では FCSE を利用するため，xv6 のメモリマップを下の図 1 のように変更した．ARM アーキテクチャに搭載されている MMU は複数レベルのページテーブルにより，緻密に仮想メモリ空間と物理メモリ空間を対応づける事が出来る．本研究では FCSE を利用するため，仮想メモリと物理メモリを対応づけるページテーブルは，先頭の割り込みハンドラを持つ 0 番地以外は全て同じ値を対応づける，ストレートマッピングとして実装を行った．

x86 用オペレーティングシステムである xv6 を ARM アーキテクチャに移植するにあたり，プログラムのロードの手順を変更した．xv6 ではユーザプログラムはロード時に，ハードディスクから物理メモリ空間にロードされ，実行が行われる．しかしながら，ARM アーキテクチャが搭載されている機器やボードでは，ハードディ

Realtime MultiProcessing for embedded systems
†Shogo Saito ‡Shuichi Oikawa
†school of informatics ‡department of computer science University of Tsukuba

スクなどの大規模ストレージが搭載されていない場合も多い．このため本システムでは，起動時にカーネルとともにユーザプログラムをメモリ空間上にロードし，実行時に再びユーザ空間にユーザプログラムをコピーするよう実装を行った．この方法では，常にメモリ空間上の一部をユーザプログラムが存在するうえ，実行時にロードするためにユーザプログラムが二重にメモリ空間を占有することとなりメモリ効率が悪い．本システムでは比較的メモリ空間に余裕のあるシステムを想定し，ストレージへのアクセスによる遅延の影響を防ぐため，このような手法を採用した．

5 システムコールの追加

本研究では，FCSE によるコンテキストスイッチの性能向上の検証にあたり，xv6 に計測用のシステムコールを新たに追加した．xv6 では，プロセス間の通信に用いるシステムコールとして，UNIX の pipe にあたる sys_pipe システムコールを持つ．しかしながら，この pipe の実装ではシステムコール時にコンテキストスイッチが瞬時に発生せず，プロセスに割り当てられた時間が終了するまで，次のプロセスへのコンテキストスイッチが行われない．このため，プロセス間通信を行う独自のシステムコール sys_chg をシステムに追加した．このシステムコールは相手プロセスの ID および，相手プロセスに渡す数値を引数としてとる．sys_chg はコールされた時点で通信内容をバッファに保存し，呼び出し元のコンテキストスイッチ迄のタイマを 0 に設定する．これにより，システムコールの実行終了と同時にコンテキストスイッチのためのタイマ割り込みが発生し，即座にスケジューラに処理が移る．スケジューラは優先度を考慮した上で，先ほどのシステムコールの引数の相手プロセス ID を元にコンテキストスイッチを行う．このシステムコールにより，コンテキストスイッチにかかる時間の計測を行った．

6 実験環境

xv6 の ARM アーキテクチャへの移植にあたり，先述の qemu によりエミュレートした次の環境を実験環境として利用した．

- ボード　ARM RealView BaseBoard
- CPU　ARM926EJ-S
- メモリ　128MB

7 測定

上述の sys_chg システムコールを使い，100 回のコンテキストスイッチにかかる時間を測定した．FCSE を使用した場合の比較対象として，FCSE を使用せず，通常のページテーブル更新によりコンテキストスイッチを行うシステムを実装し，比較を行った．上の表 1 が計測結果である．表 1 の結果からは，FCSE を使用したことによる，明らかな速度向上は見られなかった．この原因として，計測環境にエミュレータである qemu を使用しているため，ページテーブルのアクセスにおける TLB キャッシュの効果が出ない点が考えられる．

表 1: 測定結果

	FCSE 未使用	FCSE 使用
計測時間 (s)	7.5s	7.43s

8 考察

ARM アーキテクチャのような組み込み向け機器を中心に採用されているアーキテクチャは，汎用機を対象とする x86 と異なり，先述の FCSE によるコンテキストスイッチの高速化などの，ハードウェアによるさまざまな支援機能が存在する．これらの組み込み機器向けアーキテクチャの拡張機能は，汎用のオペレーティングシステムにおいても活用することができる．移植を行う際にはそれぞれのアーキテクチャ特有の支援機構を利用する事が性能向上に貢献すると言える．

9 まとめ

本論文では，オペレーティングシステムを ARM アーキテクチャに移植する際の性能向上について，メモリ管理およびコンテキストスイッチの高速化を中心に述べた．FCSE 今後は移植したオペレーティングシステムのリアルタイム性能の向上および，SMP アーキテクチャへの対応を目指す．

参考文献

[1] Andrew N. Sloss, Dominic Symes, Chris Wright: Arm System Developer's Guide Designed and Optimizing System Software, Morgan Kaufmann Pub ,2004.

[2] Febrice Bellard, QEMU, a Fast and Portable Dynamic Translator, USENIX 2005.

[3] Andrew N. Sloss, Dominic Symes, Chris Wright. Arm System Developer's Guide Designed and Optimizing System Software. Morgan Kaufmann Pub, 2004.

[4] Carl van Schaik , Heiser, Gernot . High-performance microkernels and virtualization on ARM and segmented architectures . 1st International Workshop on Microkernels for Embedded Systems. Sydney, Australia: NICTA. pp. 11-21, 2007.