Vol.2013-SLDM-160 No.27
Vol.2013-EMB-28 No.27
2013/3/14

リアルタイムシステムにおけるTLBミスの影響調査と改善手法

加藤 寿和[1,a)]　石川 拓也[1]　本田 晋也[1]　高田 広章[1]

概要：近年，大規模化するリアルタイムシステムにおいて，メモリ保護機能が必要となっており，その実現のためにMMUが使用される．MMUでは，TLBというキャッシュ機構を使用するが，TLBミスの発生を予測することは難しく，そのために，最悪実行時間の予測が困難となる．本研究では，まず，リアルタイムシステムにおいて，TLBミスが最悪実行時間へ及ぼす影響を調査する．次に，その影響を低減させるための，TLBロック機能を用いた改善手法を提案する．そして，評価実験により，提案手法の有効性を示す．

キーワード：リアルタイムシステム，TLBミス，最悪実行時間，TLBロック機能

Investigation and Improvement Techniques on the Impact of TLB misses in Real-Time Systems

Kato Toshikazu[1,a)]　Ishikawa Takuya[1]　Honda Shinya[1]　Takada Hiroaki[1]

Abstract: In recent years, memory protection for real-time systems is needed to ensure safety of the systems. The MMU used for memory protection uses the TLB which is a caching mechanism. When using the TLB in real-time systems, the worst-case execution time (WCET) is estimated pessimistically, because it is difficult to predict the occurrence of the TLB misses. In this paper, first, we evaluate the impact of TLB misses on the WCET. Secondly, we propose methods to control the occurrence of TLB misses by using the TLB locking mechanism. The result of evaluation shows the effectiveness of the proposed methods.

Keywords: Real-time system, TLB miss, Worst-case execution time, TLB locking

1. はじめに

近年，リアルタイムシステムにおける大規模化，複雑化が進んでおり，メモリ保護機能を持ったリアルタイムOSの必要性が高まってきている．一般的に，メモリ保護機能の実現には，メモリ管理ユニット（MMU）が使用される．MMUは，アドレス変換機能を支援するためのハードウェアであり，仮想アドレスと物理アドレスの対応表であるページテーブルに従って，アドレス変換を実施する．ページテーブルの各エントリには，アクセス権を指定することができる．MMUは，メモリアクセス時に，アドレス変換を実施するとともに，アクセス権を確認し，不正なアクセスを検出することで，メモリ保護機能を実現する．MMUでは，アドレス変換とアクセス権の確認を高速に行うために，Translation Lookaside Buffer（TLB）という，ページのエントリ情報を保持しておくキャッシュ機構が用いられる．TLBが，要求されたページのエントリ情報を保持していない場合にはTLBミス例外となり，主記憶に置かれたページテーブルを参照し，該当するページのエントリ情報の取得（ページテーブルウォーク）と既存のTLBエントリとの置換を行う．ページテーブルウォークでは，主記憶に複数回アクセスするため，大幅な処理時間が必要となる．また，TLBには，TLBミス例外処理をハードウェアで実施する方式と，ソフトウェアで実施する方式があるが，後者の方が，より大きい処理時間を必要とする．

[1] 名古屋大学大学院情報科学研究科
Graduate School of Information Science, Nagoya University
[a)] t_kato@ertl.jp

一方，リアルタイムシステムにおける多くのスケジューラビリティ解析手法では，システム内における各処理の最悪実行時間が用いられるが，その正確な値を見積もることは，一般的に困難であり，悲観的な見積もり結果となることが多い．リアルタイムシステムで TLB を使用する場合には，TLB ミスの発生タイミングを予測することが困難であることから，最悪実行時間が悲観的に見積もられ，プロセッサの性能向上などの必要が生じ，開発コストが増加すると考えられる．しかし，リアルタイムシステムを対象として，TLB ミスが最悪実行時間に与える影響を評価したり，この問題に対する改善手法を提案している論文は発見できなかった．一方，リアルタイムシステムにおける最悪実行時間の予測可能性に関しては，キャッシュミスやページフォールトを対象にした研究がある [1], [2].

そこで，本研究では，まず，リアルタイムシステムを想定した評価アプリケーションを用いて，TLB ミスの発生が最悪実行時間へ及ぼす影響について評価する．本研究では，CPU として，SH-4 プロセッサ [3]，リアルタイム OS として TOPPERS/HRP2 カーネル [4]（HRP2 カーネル）を使用する．次に，TLB ミスによる影響を低減させるための方法として，TLB ロック機能に着目した TLB ミスの抑制手法を提案する．本研究では，TLB ロック機能を，デッドライン時間が設定された処理（リアルタイム処理）に対して適用することで，その最悪実行時間の正確な見積りを可能とする．また，システムの実行効率を考慮して，TLB ロック処理を静的に行う手法と動的に行う手法を提案する．最後に，提案手法を評価アプリケーションに適用し，評価実験を行うことで，提案手法の有効性を確認するとともに，システムの実行効率を各手法で比較評価する．

本論文の構成は，次のとおりである．まず 2 章で，評価環境について述べ，3 章で，TLB ミスが最悪実行時間へ及ぼす影響について評価する．そして，4 章で，TLB ロック機能を用いた改善手法について述べ，5 章で，提案手法の有効性を評価する．最後に，6 章で本論文をまとめる．

2. 評価環境

本研究では，SH-4 プロセッサを評価環境として使用する．SH-4 プロセッサは，64 エントリの命令，データ共用の TLB を持ち，TLB ミス例外処理はソフトウェアで行われる．なお，用いた SH-4 プロセッサの動作周波数は，235MHz であり，キャッシュは無効とした．また，リアルタイム OS として，HRP2 カーネルを使用する．HRP2 カーネルは，保護機能を実施する単位として，カーネルオブジェクトの集合である，保護ドメインを持つ．

本研究で使用した評価アプリケーションは，リアルタイム処理（車載系処理）と，比較的広範囲なメモリ領域へアクセスを行うメディア系処理が HRP2 カーネル上で混在する構成となっている．車載系処理やメディア系処理には，

表 1 使用ベンチマーク
Table 1 Benchmarks used for the evaluation.

タスク	ベンチマーク	メモリサイズ (byte)
車載系タスク 1	ttsprk01	58,164
車載系タスク 2	a2time01	6,564
車載系タスク 3	rspeed01	4,088
車載系タスク 4	puwmod01	13,392
車載系タスク 5	canrdr01	9,360
メディア系タスク	djpegv2data4	535,332

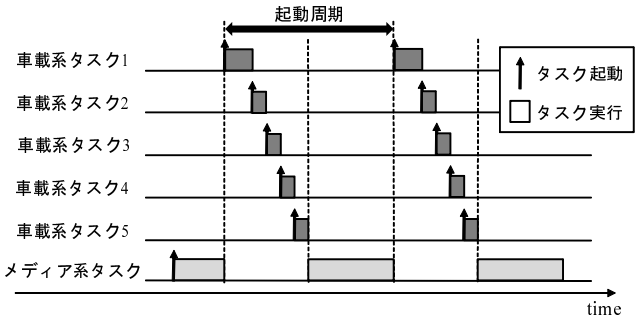

図 1 評価アプリケーションの実行イメージ
Fig. 1 Execution of the application for the evaluation.

EEMBC ベンチマーク [5] を用いて，異なる保護ドメインに所属する各タスクが，それぞれのベンチマークを実行する．**表 1** に，使用するベンチマークと，そのメモリサイズを示す．**図 1** に，評価アプリケーションの実行イメージを示す．まず，メディア系タスクが起動し，その後，周期的に車載系タスク 1 を起動させる．車載系タスク 1 は，ベンチマークを実行後，車載系タスク 2 を起動させ，自タスクを終了する．以降は同様に，各車載系タスクは，ベンチマークを実行後に，次の車載系タスクを起動させ，自タスクを終了する．車載系タスクは，すべて同一優先度に設定されており，メディア系タスクよりも高い優先度となっている．そのため，車載系タスク実行中には，メディア系タスクは，プリエンプトされて処理が実行されない．メディア系タスクは，車載系タスクが，設定された測定回数の実行を終えるまで終了しない．今回は，測定回数を 10,000 回と設定した．また，車載系タスクすべての実行時間は，平均で約 $3m$ 秒，最大で約 $5m$ 秒であり，車載系タスク 1 の起動周期は，$50m$ 秒に設定した．

3. TLB ミスによる影響調査

本研究では，TLB ミスによる最悪実行時間への影響を，TLB ミス影響度によって評価した．TLB ミス影響度は，測定対象とする処理において，TLB ミスが 1 回も起きない場合に想定される最良実行時間に対する，実測した TLB ミスの処理時間の割合として算出する．測定対象としたリアルタイム処理は，車載系タスク 1 であり，その実行時間と TLB ミス回数を計測することで，各測定における影響

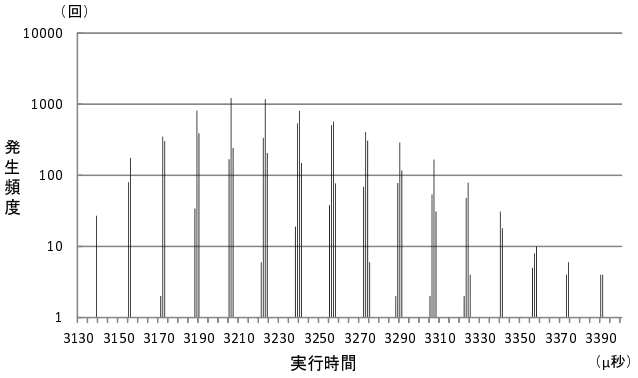

図 2 車載系タスク 1 の実行時間分布
Fig. 2 Execution time distribution of the automotive task1.

度を算出し，その平均値と最大値を求めた．今回，車載系タスク 1 に関して，ベンチマーク（ttsprk01）で使用する入力値セットを，その実行時間が最大になる特定の入力値セットに固定した．これは，実行パスの違いによる，実行時間の変動の影響を取り除くためである．そして，実行パスと TLB ミス以外には，車載系タスク 1 の実行時間を変動させる要因は存在しない．また，TLB ミス 1 回あたりの処理時間を測定した結果，平均 16.9μ 秒であった．

車載系タスク 1 の実行時間分布を**図 2** に示す．横軸は実行時間であり，縦軸は発生頻度を対数目盛により表している．最小実行時間は 3,139μ 秒であり，その時の TLB ミス回数は 0 回であった．また，最大実行時間は 3,391μ 秒であり，その時の TLB ミス回数は 15 回であった．そして，本測定における，平均影響度は 2.91%であり，最大影響度は 8.13%であった．図 2 において，実行時間の分布が集中している 2 箇所の時間の差分で，最も差分が短いものが，TLB ミス 1 回あたりの処理時間であると考えられる．この差は，平均 17μ 秒であり，事前に測定した TLB ミス 1 回あたりの処理時間とほぼ一致する．本計測において，最大影響度は 8.13%となったが，最悪実行時間を見積もる際には，この値以上の影響が及ぶことになると考えられる．したがって，最悪実行時間の見積もりにおいて，TLB ミスの影響は，無視できないものになると考える．

4. TLB ロック機能による改善手法

本研究では，TLB ミスを抑制し，最悪実行時間の予測可能性を向上させる手法として，TLB ロック機能に着目する．TLB ロック機能とは，特定ページのエントリ情報を TLB に登録する際に，そのエントリを次回からの TLB ミス時における置換対象から除外することで，そのページに関する TLB ミスが発生することを抑制する機能である．この TLB ロック機能を，リアルタイム処理でアクセスされるページに対して適用することで，リアルタイム処理における最悪実行時間の予測可能性の向上を狙う．

一般的に，TLB ロック機能では，通常の TLB エントリの一部をロックエントリとして割り当てるため，TLB ロックを実施しない場合には，通常の TLB エントリとして利用できる．TLB ロック機能をハードウェアで実装している組込みプロセッサには，ARM プロセッサ [6] がある．一方，ソフトウェアで TLB を管理する場合には，ソフトウェアの実装により，TLB ロック機能を実現することが可能となる．本研究では，SH-4 プロセッサを研究対象とするため，TLB ロック機能はソフトウェアにおける実装となる．

4.1 TLB ロック適用における課題

リアルタイム処理に TLB ロックを適用する場合に考えられる課題として，以下のことが挙げられる．

まず，TLB ロック処理を実施する際に発生する処理時間である．TLB ロック対象とするページのエントリ情報を，ページテーブルウォークにより取得する場合，TLB ミスが発生した場合と，同等の処理時間を要してしまう．そこで，本研究では，組込みリアルタイムシステムのメモリ配置が静的であることに着目した．TLB ロック対象とするページのエントリ情報を，静的に配列で用意しておき，TLB ロック処理時に，その配列への参照により，ページのエントリ情報を取得することで，TLB ロックの処理時間を低減した．

次に，リアルタイム処理に関連するページを TLB ロックすることで，非リアルタイム処理で使用できる TLB エントリ数が減少し，システムの実行効率が低下してしまうことがある．本研究では，TLB ロックの実施方法として，静的ロック手法と動的ロック手法を提案，評価することで，この課題の解決手法を検討した．これらの手法の詳細については，それぞれ，4.2 節，4.3 節で述べる．

最後に，TLB ロック対象とするページの選択問題がある．リアルタイム処理の規模が大きく，アクセスするページ数が，TLB ロック可能なエントリ数を超えてしまう場合，すべてのページのエントリ情報を TLB ロックできなくなるため，TLB ロック処理を実施するページを選択する必要がある．ただし，本研究では，この課題を今後の課題とし，リアルタイム処理でアクセスするページ数が，TLB ロック可能なエントリ数の範囲内であるという前提を置く．

4.2 静的ロック手法

静的ロック手法は，システムの起動時に TLB ロック処理を実行し，システムの実行中は，TLB 内のロック状態が維持される方式である．**図 3** に，静的ロック手法のイメージを示す．システムの起動時に，リアルタイム処理である高優先度タスクに関連するページのエントリ情報を TLB ロックし，以降は，TLB 内のロック状態が維持される．静的ロック手法の利点としては，まず，リアルタイム処理における最悪実行時間の予測可能性が高まり，その解析が容易になることが考えれられる．リアルタイム処理に関連す

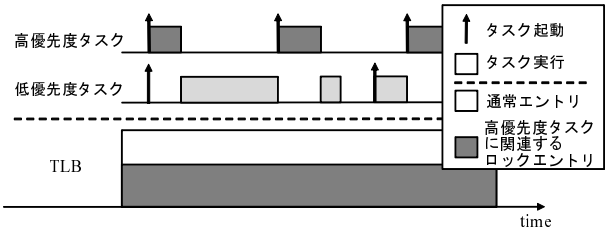

図 3 静的ロック手法
Fig. 3 Static locking method.

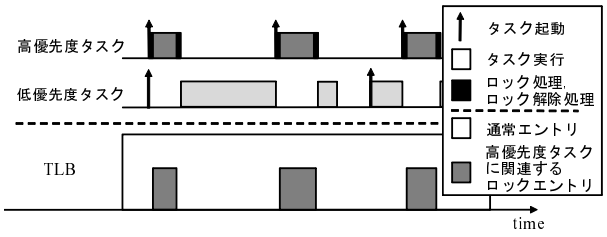

図 4 動的ロック手法
Fig. 4 Dynamic locking method.

るすべてのページを TLB ロックする場合には，TLB ミスは 1 回も発生しないため，実行時間の変動の要因が TLB ミスのみである場合に，実行時間は一意に決定することができる．また，他の利点として，TLB ロックに必要となる処理時間が，システムの起動時のみで済むことがある．一方，静的ロック手法の欠点としては，システム実行中に，非リアルタイム処理（図 3 では，低優先度タスク）が使用できる TLB エントリ数が減少するため，非リアルタイム処理における TLB ミスが増加し，システム全体の実行効率が低下する可能性があると考えれられる．

4.3 動的ロック手法

動的ロック手法は，TLB ロックの適用対象であるリアルタイム処理が実行を開始する際に TLB ロック処理を実行し，リアルタイム処理が終了する際に，TLB ロック解除処理を実施する方式である．TLB ロック解除処理とは，TLB ロックしているエントリを解放し，通常の TLB エントリとして使用できる状態にする処理である．なお，本研究では，問題を簡単にするために，リアルタイム処理は，最高優先度を持ったタスクであり，プリエンプトしない状況のみを考える．**図 4** に，動的ロック手法のイメージを示す．リアルタイム処理である高優先度タスクが実行状態になる際に TLB ロック処理を実行し，リアルタイム処理が完了して，タスクが終了する際に TLB ロック解除処理を実行する．動的ロック手法の利点としては，まず，静的ロック手法と同様に，リアルタイム処理における最悪実行時間の予測可能性が向上できることが考えれられる．次に，リアルタイム処理が実行状態から休止状態に移行する際に，TLB ロックされたエントリが解放されるため，非リアルタイム処理における TLB ミスが，静的ロック手法を用いた場合よりも低減され，システムの実行効率の低下を抑制できることが考えれられる．一方，動的ロック手法の欠点としては，リアルタイム処理が実行されるたびに，TLB ロック処理と TLB ロック解除処理を実行するため，実行時間のオーバヘッドが大きくなることが考えられる．

TLB ロック処理では，TLB ロック対象とするページのエントリ情報を，TLB から事前にフラッシュしておく必要がある（同じページに対するエントリ情報が TLB に存在すると，例外が発生するため）．静的ロック手法では，システムの起動時のみフラッシュ処理を実施するため，TLB 全体をフラッシュする手法が適切であると考えられる．これは，システム起動時に TLB 全体をフラッシュしたとしても，その後実行される処理に与える影響は少なく，さらに，TLB 全体フラッシュは，1 命令で実行できるためである．しかし，動的ロック手法では，処理の切替えが発生するたびに，このフラッシュ処理が必要となる．そのため，フラッシュ処理によって生じる実行時間のオーバヘッドや，TLB ロック処理を実施しない場合には削除されないエントリ情報が TLB から削除される可能性があることにより，システムの実行効率が低下してしまう問題が生じると考えられた．本研究では，システムの実行効率の低下を防止するために，動的ロック処理で用いるフラッシュ方法として，アドレス指定フラッシュを用いる，動的ロック手法 1 と, TLB 全体のフラッシュと TLB 内容の保存，復帰を組み合わせる，動的ロック手法 2 の，2 つの手法を提案する．

動的ロック手法 1 では，仮想アドレスを指定して，TLB ロック対象とするページのエントリ情報のみを TLB からフラッシュする．この方法では，無関係なエントリのフラッシュを防止できるが，TLB ロック対象のページごとに，フラッシュ処理を実行する必要があるため，フラッシュ処理の実行時間が増加してしまうことが懸念される．

動的ロック手法 2 では，TLB 全体をフラッシュするが，その実施前に，TLB ロック処理を実行する時点での TLB 内容を保存しておく．そして，リアルタイム処理が終了する際の TLB ロック解除処理を実行するタイミングで，保存しておいた TLB 内容を，TLB に復帰させる処理を実施する．TLB 内容の保存，復帰処理により，非リアルタイム処理に関連するページのエントリ情報が，リアルタイム処理の実行を通して，1 つもフラッシュされないことになるため，システムの実行効率の低下を抑制できると期待される．しかしながら，TLB 内容の保存と復帰に，多くの処理時間を要してしまうことが懸念される．

5. 提案手法の評価

提案する TLB ロック手法を，2 章で述べた評価アプリケーションに適用し，その有効性を評価した．評価内容としては，リアルタイム処理の実行時間分布と TLB ミス影

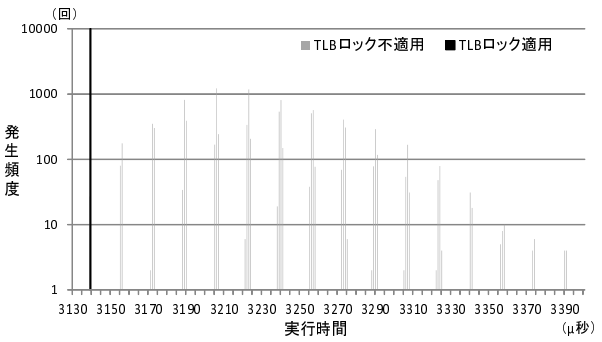

図 5 TLB ロック適用時の車載系タスク 1 の実行時間分布
Fig. 5 Execution time distribution of the automotive task1 with TLB locking.

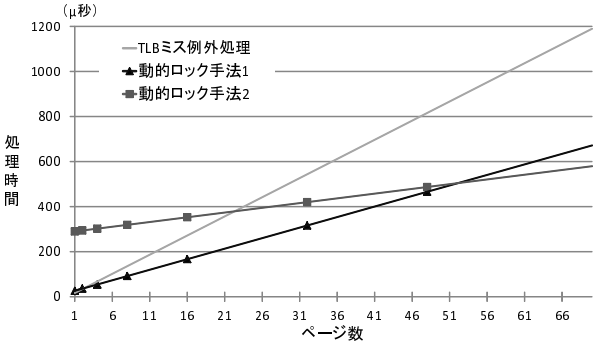

図 6 各 TLB ロック手法における処理時間
Fig. 6 Processing time for TLB locking methods.

響度，各 TLB ロック手法の処理時間，また，システムの実行効率である．

評価アプリケーションへの適用に際して，静的ロック手法では，メディア系タスクを起動する前に TLB ロック処理を実施する．動的ロック手法では，車載系タスク 1 が，周期的に起動される際に TLB ロック処理を実施し，TLB ロック対象とする最後のタスクが終了する際に TLB ロック解除処理を実行する．TLB ロック対象としたリアルタイム処理は，車載系タスク 1 のみの場合と，すべての車載系タスクの場合であり，TLB ロックを実施したページ数は，それぞれ，16 ページ，49 ページであった．ここで，車載系タスク 1 のみを TLB ロック対象とする場合には，3 章と同様の入力値セットを与える．

各 TLB ロック手法の処理時間測定では，TLB ロック対象とするページ数を変化させ，それぞれの，TLB ロック処理，および，TLB ロック解除処理を実施した際の処理時間を測定した．また，システムの実行効率を評価するために，非リアルタイム処理であるメディア系タスクの応答時間，実行時間，TLB ミス回数を計測した．応答時間は，メディア系タスクが起動されてからすべての処理を完了するまでの時間であり，実行時間はメディア系タスクが実行状態となり処理を行った時間である．また，メディア系タスクの TLB ミス回数の測定では，メディア系タスクが実行状態のときに発生した TLB ミス回数を計測した．

5.1 評価結果と考察

まず，車載系タスク 1 のみに TLB ロックを実施した場合の，車載系タスク 1 の実行時間分布を，図 2 の実行時間分布と重ねて，**図 5** に示す．横軸は実行時間であり，縦軸は発生頻度を対数目盛により表している．本測定において，TLB ミスは発生せず，TLB ミス影響度は 0% となった．なお，すべての TLB ロック手法について，車載系タスク 1 の実行時間分布は，図 5 のようになった．

図 5 より，車載系タスク 1 の実行時間分布が 1 箇所に集中していることが分かる．これは，TLB ロックの実施により，リアルタイム処理における TLB ミスが発生しなくなり，実行時間が一意になったためだと考えられる．したがって，TLB ロック機能は，TLB ミスが最悪実行時間に及ぼす影響を低減させる手法として，有効であるといえる．

次に，各 TLB ロック手法における処理時間を，TLB ミス例外処理の時間と比較した結果を，**図 6** に示す．横軸は TLB ロックを実施するページ数，または，TLB ミスを起こしたページ数であり，縦軸は処理時間を表している．

図 6 より，動的ロック手法 1 では，2 ページ以上を TLB ロックした場合に，TLB ミス例外処理よりも，処理時間が小さくなることが確認できる．TLB ミス例外処理の時間が，1 ページあたり 17μ 秒であるのに対して，動的ロック手法 1 の処理時間は，TLB ロック対象とするページが 1 ページ増えるごとに，約 9.6μ 秒の増加となる．これは，TLB ロック処理で使用するページのエントリ情報を，静的に用意したことで，ページテーブルウォークを行うことなく，TLB ロック処理ができたためであると考えられる．

一方，動的ロック手法 2 では，最初の 1 ページの TLB ロックにおいて，処理時間が 290μ 秒と，大きい時間を要している．これは，TLB の保存，復帰処理に必要となる処理が影響している．しかし，TLB ロック対象とするページが 1 ページ増加したときの，TLB ロック処理に要する時間の増加量は，約 4.2μ 秒であり，動的ロック手法 1 よりも小さいため，52 ページ以上では，すべての TLB ロック手法の中で最小の処理時間となる．

最後に，各 TLB ロック手法を適用した際の，メディア系タスクに関する評価結果を，**図 7** と**図 8** に示す．図 7 は，車載系タスク 1 のみを，図 8 は，すべての車載系タスクを TLB ロック対象とした際の評価結果である．グラフの横軸は，適用した TLB ロック手法を表し，左の縦軸は応答時間，および，実行時間を，右の縦軸は TLB ミス回数を，それぞれ表している．ここで，応答時間，実行時間，および，TLB ミス回数は，TLB ロック手法を適用しなかった場合の計測結果を 1 とした，相対比で表している．

まず，静的ロック手法に着目する．TLB ロック対象とするページ数が少ない場合（図 7），TLB ロックを適用しな

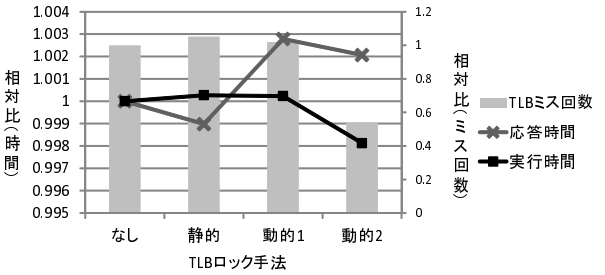

図 7 メディア系タスクの測定結果（車載系タスク 1）
Fig. 7 Results for the media task(the automotive task1).

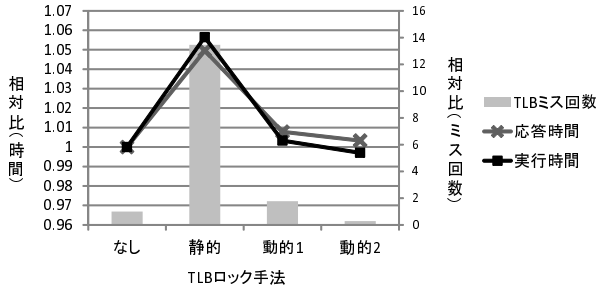

図 8 メディア系タスクの測定結果（全車載系タスク）
Fig. 8 Results for the media task(all automotive tasks).

かった場合に比べ，各測定結果の変動は，概ね 0.1%以内であることが分かる．応答時間が減少したのは，車載系タスク 1 の最大実行時間が，TLB ロック機能により，最小化されたためだと考えれられる．一方，TLB ロック対象とするページ数が多い場合（図 8），TLB ミス回数が約 13 倍になり，応答時間も約 5%増加した．これは，メディア系タスクが実行中に使用できる TLB エントリ数が減少し，TLB ミスが頻発したためと考えられる．

次に，動的ロック手法に着目する．図 7，図 8 ともに，動的ロック手法 1 よりも，動的ロック手法 2 の方が，メディア系タスクに与える影響が小さかった．この結果より，リアルタイム処理が実行される際に，メディア系タスクに関する TLB 内容を保存，復帰する処理の効果が大きいということが分かった．

以上の評価結果を踏まえた，各 TLB ロック手法の総合的な考察を述べる．まず，TLB ミスによる最悪実行時間への影響を低減させる上で，すべての TLB ロック手法は有効であるといえる．一方，各 TLB ロック手法で，TLB ロック処理と TLB ロック解除処理に必要な処理時間，システムの実行効率へ与える影響は，大きく異なることが分かった．総合的に判断すると，TLB ロック対象とするページ数が少ない場合には，静的ロック手法が有効であり，TLB ロック対象とするページ数が多い場合には，動的ロック手法 2 が有効であるといえる．ただし，この結論は，本評価環境におけるものであり，キャッシュメモリを有効にする場合や，異なるプロセッサを用いる場合には，必ずしも同じなるとは限らない．異なる評価環境において，提案手法を評価することは今後の課題である．

6. おわりに

本論文では，リアルタイムシステムにおいて，TLB ミスが最悪実行時間に及ぼす影響の調査と，TLB ロック機能を用いた改善手法について述べた．TLB ミスの影響調査では，リアルタイムシステムを想定した評価アプリケーションを用いて，TLB ミスの処理時間が最悪実行時間へ及ぼす影響の評価を実施した．評価の結果，リアルタイム処理の実行時間に対して，TLB ミスの処理時間が最大 8.1%影響することを確認した．次に，TLB ミスによる影響を低減させる方法として，TLB ロック機能を用いた TLB ミスの抑制手法を提案した．本研究では，TLB ロック機能をリアルタイム処理に対して適用し，その実施方法として，静的ロック手法と動的ロック手法を提案した．一方，TLB ロック処理を行う際には，該当ページのエントリ情報を TLB からフラッシュしておく必要があり，本研究では，動的ロック手法のフラッシュ処理方法として，2 つの手法を提案した．評価アプリケーションに，以上の TLB ロック手法を適用して評価実験を行った結果，すべての TLB ロック手法において，最悪実行時間の予測可能性が向上できたことを確認できた．また，システムの実行効率を考慮して，各 TLB ロック手法を使い分ける必要があることが分かった．

今後の課題としては，まず，本研究で実施した TLB ミスの影響調査と TLB ロック手法の評価実験を，異なる環境で実施することが挙げられる．次に，本研究では，考慮の対象外とした，TLB ロック対象とするページの選択方法の検討が挙げられる．また，TLB ロック機能を支援するハードウェアの提案も今後の課題として考えられる．

参考文献

[1] Liu, T., Li, M. and Xue, C. J.: Minimizing WCET for Real-Time Embedded Systems via Static Instruction Cache Locking, *Proceedings of the 2009 15th IEEE Symposium on Real-Time and Embedded Technology and Applications*, RTAS '09, Washington, DC, USA, IEEE Computer Society, pp. 35–44 (2009).
[2] Puaut, I. and Hardy, D.: Predictable Paging in Real-Time Systems: A Compiler Approach, *Proceedings of the 19th Euromicro Conference on Real-Time Systems*, ECRTS '07, Washington, DC, USA, IEEE Computer Society, pp. 169–178 (2007).
[3] ルネサスエレクトロニクス：SH-4 ソフトウェアマニュアル, Rev.6.00.
[4] 石川拓也，本田晋也，高田広章：静的なメモリ配置を行うメモリ保護機能を持ったリアルタイム OS，コンピュータソフトウェア，Vol. 29, No. 4, pp. 161–181 (2012).
[5] EEMBC: http://www.eembc.org/.
[6] ARM: ARM Architecture Reference Manual ARMv7-A and ARMv7-R edition.