Vol.2017-ARC-225 No.31
Vol.2017-SLDM-179 No.31
Vol.2017-EMB-44 No.31
2017/3/10

エンジン制御プログラムのコア間排他制御機構

奥原 誠[†1]

概要：燃費向上や排ガス規制，機能安全対応などによるシステムの複雑化に伴い，エンジン制御の性能向上ニーズが高まっており，これに対応するためにエンジン制御 ECU にもマルチコアマイコンの導入が始まっている．現在は 2 コアマイコンが主流だが，今後は 3 コア以上のマルチコアマイコンが主流となる．そのためコア数増加に伴い，新たに排他制御の選定が必要になる．本研究ではエンジン制御で求められるマルチコアでの排他制御の要件を明確化し，定性的な評価を行った結果 MCS ロックを選定した．また選定した MCS ロックを定量的に評価しエンジン制御に適しているアルゴリズムであることを確認した．

キーワード：エンジン制御，マルチコア，スピンロック

Exclusive Control Mechanism for Engine Control Program

MAKOTO OKUHARA[†1]

Abstract: The performance requirements of automotive engine control are increasing, for instance to comply the improvement of fuel efficiency and emission constraint and functional safety. These requirements for engine control ECU with high performance lead to multicore approaches to reach the desired featured. Now dual core is mainstream,but over the 3 core will become mainstream in future. Because the number of the core increases ,the choice of exclusive control mechanisam is necessary. In this paper , I clarified requirements of exclusive control demanded by engine control. As a result of having carried out a qualitative evaluation, I chose MCS locks. As a result of having evaluated MCS lock quantitatively, I confirmed that it was the algorithm that was suitable for engine control.

Keywords: Engine Control，MultiCore，Spinlock

1. はじめに

燃費向上や排ガス規制，機能安全対応などによるシステムの複雑化に伴い，エンジン制御へ要求される制御性能向上ニーズは年々増加し続けている．増加するニーズに対応するためパワトレ系マイコンでは CPU 周波数を上げることにより性能向上を実現してきたが，消費電力や冷却装置のコストアップなどの課題があり，CPU 周波数を上げることによる性能向上は見込めなくなった．そのため安価で省電力，高性能なマルチコアマイコンのエンジン制御への導入が始まっており，「2 コア」，もしくは周辺回路の制御に特化したペリフェラルコアを搭載した「2 コア＋ペリフェラルコア」構成のマルチコアマイコンが主流となってきている．今後も制御性能向上ニーズは増加し続けると予想され，マイコンメーカーは 3~6 コアのマルチコアマイコンのリリースを計画している．マルチコアマイコンを使用する場合，コア間で共有するデータへのアクセスにはコア間排他制御機構が必須である．クリティカルセクションが短いコア間排他制御機構ではスピンロックと呼ばれる方式が一般的に用いられており，現在エンジン制御では主にこの方式が使用されている．エンジン制御はデータが複雑に絡みあい複数のプライオリティの処理間でデータのやりとりを行うことで干渉のおそれがあるため，スピンロックを使用する頻度が多い．その結果スピンロックがエンジン制御に大きな影響を与えている．そこで本論文ではエンジン制御で求められるコア間排他制御機構の要件を明確にし，エンジン制御で最適なスピンロックアルゴリズムの評価を行う．本論文の構成は次の通りである．2 章では，本論文の対象となるエンジン制御のソフトウェア構成について述べると供にエンジン制御で排他を行う際に必要となる要件を述べる．3 章では，これまでに提案されている 4 つのスピンロックアルゴリズムについて説明を行う．4 章では，このスピンロックアルゴリズムを 2 章で示した要件に基づき定性的に評価すると供に，評価の結果選定した MCS アルゴリズム[1]の有効性を実機で定量的に確認する．

2. エンジン制御

本論文が対象とするエンジン制御ソフトウェアの構成を述べ，マルチコア化におけるコア間排他制御機構の課題を挙げる．その上でエンジン制御にコア間排他制御機構を採用する際に必須となる排他要件を明確化する．

†1 富士通テン(株)
Fujitsu Ten Ltd.

Vol.2017-ARC-225 No.31
Vol.2017-SLDM-179 No.31
Vol.2017-EMB-44 No.31
2017/3/10

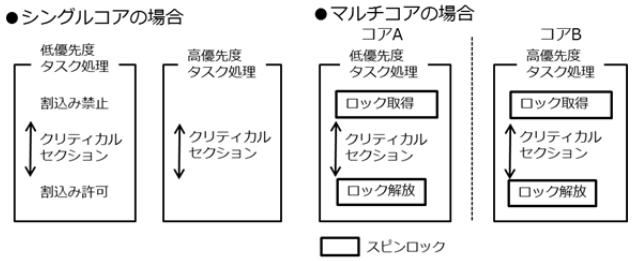

図 1　シングル・マルチコアの排他制御機構の違い

2.1 エンジン制御ソフトウェア構成

エンジン制御のソフトウェアは数十のセンサからの入力割込みと時間周期で動作する時間同期タスクから成り立っている．入力割込みの中でもクランク角センサからの回転信号による回転信号入力割込み処理は，エンジン回転数に応じて処理回数が変動し高回転になるほど処理回数が多くなる．この回転信号入力割込みと数ミリ～数秒オーダーで起床される時間周期タスク処理がエンジン制御ソフトウェアで構成されている．

この回転信号入力割込み処理と各時間同期タスクは異なるプライオリティの処理間でデータのやりとりを行っており，共有するデータ数は数千に及ぶ．そのためデータの干渉を起こさないように排他制御機構が重要となる．

2.2 マルチコア化におけるコア間排他制御機構の課題

エンジン制御ソフトウェアをマルチコアマイコンに搭載する上でのコア間排他制御機構の課題を以下に挙げた．

(a)　排他処理時間増加による処理性能悪化
(b)　ロック取得待ち時間によるエンジンシステムへの影響
(c)　汎用的なスピンロックの実現

(a) 排他処理時間増加による処理性能悪化

図 1 に示すようにシングルコアとマルチコアで排他制御機構が異なる．シングルコアではプライオリティの異なる処理間でデータをやりとりすることにより、干渉が発生するため，優先度の低いタスク側でクリティカルセクション箇所前後に割込み禁止/許可を実施することにより干渉を防げたが，マルチコアの場合はコア間に優先度の概念がないため干渉箇所すべてにスピンロックを実施する必要がある．さらにスピンロックは割込み禁止状態で実施する必要があるため割込み禁止処理よりも処理量が多く，排他の処理時間が増加する．特にエンジン制御は共有するデータ数が数千あり，排他処理の実行回数は 1 秒間で数万回にのぼるため，処理性能悪化に繋がる．

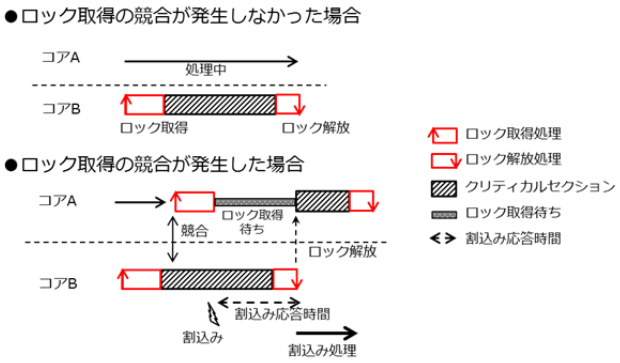

図 2　スピンロックの対象処理

(b) ロック取得待ち時間によるシステムへの影響

図 2 のようにロック取得の競合が発生しなかった場合，クリティカルセクション箇所を実行する前後でロック取得処理とロック開放処理が実行される．ロック取得の競合が発生した場合は後から実行するコアはロック取得待ちが発生する．このロック取得待ちはクリティカルセクション実行中の処理時間に依存する．またロック取得，開放処理は割込み禁止状態で実行する必要があるため，割込み応答性が悪化する．エンジン制御では排他処理の実行回数が多いため，ロックを実施する単位(ロック単位)は細粒度ロックのように共有データ個々で排他を実施し，競合によるロック取得待ち時間を抑えることが望ましい．しかし細粒度ロックの場合，複数のロックをネストして取得した場合にデッドロックが発生する恐れがある．このデッドロックを回避する方法として Totally FIFO[2]などが提案されているが，実現するための処理量が多いため，エンジン制御における排他処理の実行頻度を考えると処理性能がかえって悪化することになる．そのためロックの単位はジャイアントロックが現実的な手段と考える．ジャイアントロックを使用した場合，共有データへのロック取得競合が発生する頻度が増え，ロックを取得するまでの時間が競合タイミングに依存する恐れがある．エンジン制御のようなリアルタイムシステムにおいて，ロック取得要求から取得までのロック取得待ち最悪時間が規定できないことは制御破綻を引き起こす恐れがある．たとえば噴射量演算など，回転周期に依存する処理の場合，次の噴射タイミングに演算が間に合わなければ噴射量が正しく算出されず，最適な噴射量による噴射が実施されなくなる．割込み応答性についても同様のことが言える．

(c) 汎用的なスピンロックの実現

スピンロックを実現するためにはアトミック命令が必要となるが，各マイコンメーカーが用意しているアトミック命令はそれぞれ仕様が異なる．そのため，各社に搭載されているアトミック命令の仕様でスピンロックを実現する必要

Vol.2017-ARC-225 No.31
Vol.2017-SLDM-179 No.31
Vol.2017-EMB-44 No.31
2017/3/10

がある．マイコンによってはスピンロックのアルゴリズムを実現できない恐れもある．
エンジン ECU メーカーは各エンジン制御の制御処理量，ROM/RAM スペック，コストからその都度マイコンを選定するため，どのマイコンでも実現できる汎用的なスピンロックが求められる．

2.3 排他要件

2.2 のマルチコア化における課題より本論文の主題であるエンジン制御に必要なコア間排他制御機構の要件を 4 つ挙げる．

(1) ロック取得・解放処理時間が短いこと
(2) ロック取得待ちの最悪時間が規定できること
(3) 割込み応答時間が短いこと
(4) マイコンアーキへのポータビリティがあること

(1)に関しては課題(a)より処理性能に影響しないように処理時間の短い単純なアルゴリズムが求められる．(2)(3)は課題(b)よりマルチコア化におけるシステムへの影響を考慮して，この要件が求められる．(4)に関しては課題(c)より汎用的なスピンロックが求められるため要件として記載した．

3. スピンロックアルゴリズム

スピンロックアルゴリズムとして TAS ロック，MCS ロック，中断可能なキューイングスピンロック[3]，チケットロック[1]が提案されている．それぞれのアルゴリズムについて特徴を説明する．

（a）TAS ロック

最も単純なスピンロックで，ロックが取得できるまでチェックを繰り返す．アトミック命令 Test and Set を使用して同時にロックを取得する干渉を防ぐ．2 コアでは 1 つのコアがロックを開放した場合，必ずもう片方のコアがロックを取得できるが，3 コア以上の排他で使用した場合，ロックの取得順序はランダムに決まる．そのためロックの取得要求タイミング次第ではロックを取得できないリソーススタベーションが発生する恐れがある．またロック取得待ちの間に割込み処理を受け付けることができないため，割込み応答性は悪い．

（b）MCS ロック

ロックの取得順をキューで管理するキューイングスピンロック手法の一つである．ロック取得要求順にキューに並ぶ．キューが空である場合はすぐにクリティカルセクションへアクセスでき，空でない場合はキューの最後尾に並び順番を待つ．クリティカルセクションへのアクセスが終わり，ロックを開放するコアは次に並んでいるコアへロック取得を通知する．この時，次のコアにロック取得を通知するタイミングとロック取得要求でキューの最後尾に並ぶ処理が同時に発生した場合，通知先に通知することができずにデットロックが発生する恐れがある．これを防ぐためにアトミック命令 Compare and Swap を使用し干渉を防ぐ．ロックの取得はキューの順で実行されるため，最悪実行時間はコア数のリニアオーダーで規定することができる．一方で MCS ロックはロック取得待ちの間に割込み処理を受け付けることができないため，割込み応答性は悪い．

（c）中断可能なキューイングスピンロック

(b)の MCS ロックの課題である割込み応答性を改善するために提案されたロック取得待ち中に割込み処理を受け付けることができる２種類のアルゴリズムである．1 つはキューに並んでいるときは割込みを許可し，割込み処理から復帰後にキューの最後尾に並ぶ方法，もう１つは割込み処理から復帰後もキューの位置を変えない方法である．どちらの方法もスピンロックのアルゴリズム内で割込みを検出する必要があり OS での対応が必要となる．そのため MCS ロックと比較すると処理が複雑になる．また割込み処理の発生頻度が多いエンジン制御では，キューの待ちの間に割込み処理を実施することにより最悪実行時間が規定できなくなる恐れがある．

（d）チケットロック

番号札(チケット)を用いて取得順番を管理する方法である．ロック取得要求時に番号札(チケット)を取り，チケット番号を１つすすめる．ロック取得可能なチケットがロック単位毎に掲示されており，ロック取得できるまでチケットとの照合を続ける．掲示番号が自分のチケットと一致した場合にロックを取得できる．ロック解放時は掲示番号を１つすすめる．ロック取得要求を同時に行った場合，同じチケットをとる恐れがあるため，チケットの取得とチケット番号を１つすすめる処理をアトミック命令 Fetch and Add で実現する必要がある．取得順番を管理しているため最悪実行時間はコア数のリニアオーダーで規定することができる．一方で MCS ロックと同じようにロック取得待ちの間に割込み処理を受け付けることができないため，割込み応答性は悪い．

Vol.2017-ARC-225 No.31
Vol.2017-SLDM-179 No.31
Vol.2017-EMB-44 No.31
2017/3/10

表 1　排他要件評価結果

要件＼アルゴリズム	(a)TASロック	(b)MCSロック	(c)中断可能なキューイングロック	(d)チケットロック
(1)ロック取得・解放時間	○	○	×	○
(2)ロック取得待ちの最悪時間	×	○	×	○
(3)割込み応答時間	×	×	○	×
(4)ポータビリティ	○	○	×	×
評価（○の数）	2	3	1	2

表 2　パワトレマイコン主要コアのアトミック命令

Core	G3MH	TriCore	POWER	ARM
主要アトミック命令	CAXI、LDL、STC	CMPSWAP	LAS1	LDREX、STREX
TASロック	○	○	○	○
MCSロック	○	○	○	○
チケットロック	○	○	×	○

4. 評価

3 章で説明した 4 つのアルゴリズムに対して 2 章で明確化した 4 つの要件に対して定性評価を行った。その結果、選定した MCS アルゴリズムで実機を使用して定量評価を実施した.

4.1 定性評価

定性評価結果を表 1 にまとめた．要件（1）は処理が複雑な中断可能なキューイングスピンロックが満たさず，要件（2）はロック取得がランダムな TAS ロックと割込みを許可してロック取得待ちを行っている中断可能なキューイングスピンロックが満たさない．要件（3）は割込み禁止状態でロック取得待ちを行っている TAS ロック，MCS ロック，チケットロックが要件を満たさない．要件（4）については，表 2 のパワトレマイコンで使用している主要なコアの命令コードを調査し判定を行った．G3MH はルネサス，TriCore はインフィニオン，POWER は NXP と ST マイクロが，ARM は Cypress が主に使用している．各コアで用意されているアトミック命令でそれぞれのアルゴリズムが実現可能かを確認した．なお中断可能なキューイングスピンロックは OS での対応も必要なため，表 2 からは除外した．この結果，チケットロックのみ POWER 系のコアで実現することが困難であった．以上，4 つの要件を最も満たしている MCS ロックがエンジン制御での排他要件を満たしていると判断した. 割込み応答性が悪いという問題があるが，ロック取得待ちの最悪実行時間が規定できるため，割込み応答性の最悪時間も規定できる．そのため設計段階でクリティカルセクションの最大時間に規定を設けておけば，割込み応答性がエンジン制御システムに影響を及ぼさない対策をとることが可能である.

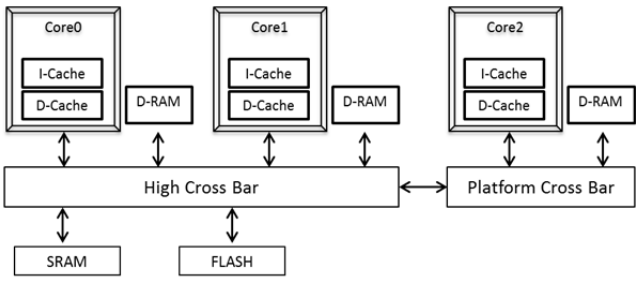

図 3　POWER 系マイコンアーキテクチャ

4.2 定量評価

4.1 で選定した MCS ロックと 2 コアマイコンで一般的に使用されている TAS ロックを 3 コアマイコンの実機環境上で性能比較を実施した．これは排他アルゴリズムを TAS ロックから MCS ロックに変更したことにより性能悪化が発生していないことの確認（要件（1））と MCS ロックが排他要件（2），(3）を満たしていることを確認するためである.

4.2.1 評価項目

2.3 排他要件の定量化できる要件（1）～（3）を評価項目として確認した.

・評価 1．ロック取得・解放処理時間
・評価 2．ロック取得待ちの最悪時間
・評価 3．割込み応答時間

4.2.2 評価環境

NXP 製 MPC5746M マイコンを使用して評価を行った．対象マイコンのアーキテクチャの特徴を図 3 に示す.
このマイコンは 3 つの POWER 系コアを搭載している．全て同じコア，動作周波数も全て最大 200MHz の設定が可能である．メモリは全コアからアクセスできる共有 RAM である SRAM と命令コードを書き込む ROM 領域の FLASH がある．FLASH も全コアからアクセスできる．各コアごとに用意されている専用 RAM 領域として D-RAM を搭載している．SRAM よりもアクセス応答時間が早い．バスへの接続は HighCrossBar と PlatformCrossBar がある．マイコン内部とのバスにつながる HighCrossBar は最大 200MHz の設定, 周辺ペリフェラルとのバスと接続する PlatformCrossBar は最大 100MHz の設定が可能である.
MPC5746M は 3 コアとも同じコアを使用しているが，コア 2 のみ PlatformCrossBar と接続しているため, ROM/RAM へのアクセスレイテンシが遅い．また定量評価ではコア間で排他の競合をさせる必要があり，競合させる確立を上げるため全てのコア，バスを遅い周波数 50MHz に統一した．I-Cache は使用し，D-Cache はコヒーレンシーの問題があるため使用しない．D-RAM は各コアのロック情報格納先と

Vol.2017-ARC-225 No.31
Vol.2017-SLDM-179 No.31
Vol.2017-EMB-44 No.31
2017/3/10

して使用した.

4.2.3 評価方法

各評価項目についての評価方法を示す.

4.2.3.1. 評価 1. ロック取得・解放処理時間

コア間で競合が発生していない場合のロック取得，開放処理時間を測定する．各コアでロック取得要求を行い SRAM に配置した共有データ 10 個の読み出しと書き込みを行う．実施後，ロックを解放する．これをクリティカルセクション処理と呼ぶことにする．これを各コア競合しないように 100μs ずつ間隔をあけて実施し，各コア 1000 回測定した結果の平均値を算出した．各コアで測定するのはコア 2 のバスの違いによる影響有無を確認するためである．なお, このクリティカルセクション処理は 1μs の処理時間がある.

4.2.3.2. 評価 2. ロック取得待ちの最悪実行時間

2 コアで競合が発生した場合，3 コアで競合が発生した場合それぞれのロック取得までの最悪実行時間を測定する.

・2 コア競合

4.2.3.1 と同様，クリティカルセクション処理を各コアで実施する．競合を発生させるために各コアで時間の同期をとり，2 コアで同じタイミングでクリティカルセクション処理を実施させた. コア競合は全 3 パターン(コア 0－コア 1，コア 1－コア 2，コア 2－コア 0)，それぞれ 1000 回ずつ実施し, 各コアのロック取得待ちの最悪実行時間を測定する.

・3 コア競合

2 コア競合と同様，クリティカルセクション処理を各コアで実行する．各コアロック解放後，即時にロック取得要求をすることにより 3 コア競合を発生させる.

各コアのロックの取得，解放を 1000 回実施し，各コアのロック取得待ちの最悪実行時間を測定する.

4.2.3.3. 評価 3. 割込み応答時間

3 コアで競合が発生した場合の割込み応答時間を確認する.

4.2.3.2 ロック取得待ちの最悪実行時間の 3 コア競合と同様にクリティカルセクション処理を各コアで実行し，各コアロック解放後に即時にロック取得要求を実施した．また各コア 100μs 毎に 1μs の処理時間がある割込み処理を発行する．各コアのロックの取得，解放を 1000 回実施し，割込み発生から実際に割込み処理が始まるまでの最悪時間を測定した.

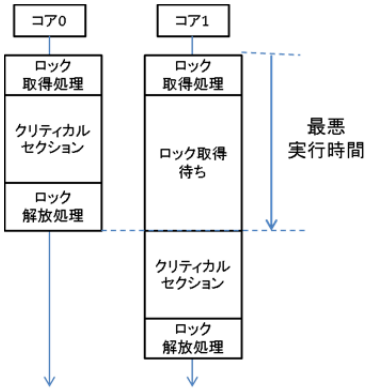

図 4 2 コア競合時の最悪実行時間

4.2.4 評価結果

4.2.4.1. ロック取得・解放処理時間

表 3 よりロック取得処理時間についてはどちらも 650ns 前後とほぼ同等の性能であった．一方でロック開放処理は MCS ロックの場合, 次にロック取得を渡すという後処理が必要である分，処理時間が長くなっている.

表 3 ロック取得・開放処理時間測定結果

	MCSロック			TASロック		
	コア0	コア1	コア2	コア0	コア1	コア2
ロック取得処理時間[ns]	667	667	688	646	670	646
ロック解放処理時間[ns]	695	694	688	618	604	583

4.2.4.2. ロック取得待ちの最悪実行時間

・2 コア競合

表 4 よりロック取得時間は MCS ロックが TAS ロックよりも約 200ns 短い結果となった．これは TAS ロックが競合発生時にロック変数のチェックを繰り返し行うことによりバスが混雑することが原因と考えられる.

2 コア競合時の MCS ロックの最悪実行時間は図 4 より

ロック取得時間 + ロック解放時間 + クリティカルセクション時間であり，4.2.4.1 の結果よりロック取得時間 + ロック解放時間は 1376ns（コア 2)，クリティカルセクション時間は 1000ns なので 2376ns が机上計算での最悪値となり，測定結果は同等の値が得られている.

表 4 2 コア最悪実行時間測定結果

	MCSロック			TASロック		
	コア0	コア1	コア2	コア0	コア1	コア2
2コア競合最悪実行時間[ns]	2542	2625	2438	2792	2813	2563

Vol.2017-ARC-225 No.31
Vol.2017-SLDM-179 No.31
Vol.2017-EMB-44 No.31
2017/3/10

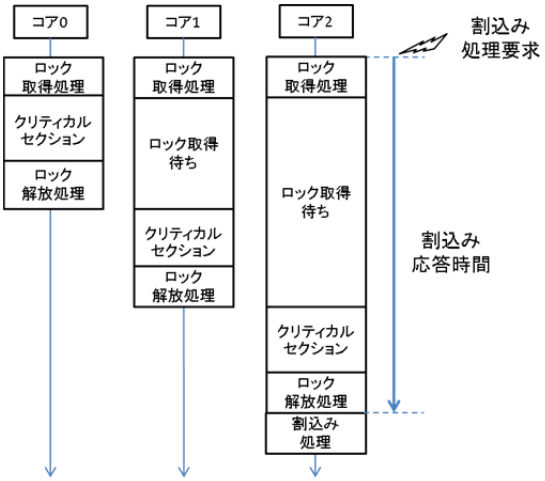

図 5　割込み応答時間の最悪値

・3 コア競合

表 5 よりロック取得時間が TAS ロックのコア 2 で 2 秒以上かかり，リソーススタベーションが発生したといえる．コア 2 のみ発生していることより今回使用したマイコンアーキの影響の恐れもあるが，他の評価項目で影響がでていなかったため, TAS ロックの問題だと考える. これは要件(2)ロック取得待ちの最悪実行時間が規定できることを満たしていない．一方で MCS ロックは机上計算での最悪値 2 コア競合時の最悪実行時間 × 2 コア分 − ロック取得時間の約 4100ns と近い値が得られた．

表 5　3 コア最悪実行時間測定結果

	MCSロック			TASロック		
	コア0	コア1	コア2	コア0	コア1	コア2
3コア競合最悪実行時間[ns]	4667	4167	4083	2104	2042	2242479

4.2.4.3.　評価 3　割込み応答時間

表 6 より MCS ロック，TAS ロックとも同等の割込み応答時間となった．図 5 より割込み応答の机上最悪値は 3 コアの最悪実行時間 ＋ 自コアのクリティカルセクション時間＋ロック解放処理分で MCS ロックではクリティカルセクション 1 μs, ロック解放処理 688ns(コア 2)より約 5800ns となり、測定結果は同等の値が得られている．

表 6　割込み応答時間測定結果

	MCSロック	TASロック
割込み応答時間[ns]	5666	6166

4.2.5 考察

2 コアで主流に使用されている TAS ロックと MCS ロックでは，ロック取得・解放処理時間は MCS ロックの処理時間増加割合が TAS ロックの約 10%以内に収まっており ほぼ同等の測定結果になったと考える．また競合発生時は TAS ロックではリソーススタベーションが発生したが MCS ロックでは発生しないというメリットが実機により確認できた．割込み応答性は机上最悪値と同等の測定結果が得られ，クリティカルセクション区間の上限値を規定できれば割込み応答性の上限値も決まることが確認できた．以上よりマルチコアでのスピンロックアルゴリズムは MCS ロックが有効であると判断できる．

5. おわりに

本研究ではエンジン制御で求められる排他制御の要件を明確化した上で最適なアルゴリズムは MCS ロックであると選定した．また 2 コアから 3 コアへのソフト移植時に排他アルゴリズムによる処理性能悪化にはつながらないことが確認できた．

謝辞　本研究を進めるにあたりご協力いただいた名古屋大学大学院情報科学研究科　本田晋也氏に深く御礼申し上げます．

参考文献

[1] Mellor-Crummey, J.M. and Scott, M.L.: Algorithms for scalable synchronization on shared-memory multiprocessors, ACM Trans. Comput. Syst., Vol.9, No.1, pp.21–65(1991).
[2] Takada, H. and Sakamura, K.: Real-Time Scalability of Nested Spin Locks, International Workshop on Real-Time Computing Systems and Applications, pp.160–167(1995).
[3] 高田広章，坂村健：中断可能なキューイングスピンロックアルゴリズム，電子情報通信学会論文誌 D-I，Vol.J78-D-I, No.8, pp.661‐669 (1995).