社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2005－ＳＬＤＭ－119 (20)
2005／3／18

ハードリアルタイムシステムに適したメモリ保護機構の提案と評価

西部 満[†] 本田晋也[††]
冨山宏之[†] 高田広章[†]

近年，ハードリアルタイム性が要求される組み込みシステムで，ソフトウェアの複雑化に伴う信頼性や開発効率の低下が問題となっており，その解決のためにメモリ保護機構の導入の必要性が高まっている．メモリ保護を実現するためには，ハードウェアとソフトウェア (OS) の両方でのサポートが必要である．本論文では，メモリ保護効率に優れてハードウェアのコストも低い，ハードリアルタイムに適したメモリ保護ユニットを提案し，回路面積や遅延の評価を行った．また，そのメモリ保護ユニットをプロセッサに組み込み，メモリ保護機能付きリアルタイム OS である μITRON/PX 仕様に準拠した OS を動かした．その上で，サービスコールや割り込み応答性能等のリアルタイム性の評価を行ない，ハードリアルタイムシステムにおいてもメモリ保護が使えることを示した．

キーワード ハードリアルタイム，メモリ保護，μITRON/PX

Proposal and Evaluation of Memory Protection Mechanism For Hard Real-Time Systems

Mitsuru Nishibe ,[†] Shinya Honda ,[††] Hiroyuki Tomiyama [†] and Hiroaki Takada [†]

Recentry, there is a problem in hard real-time embedded system that the increase of the development cost and the decrease of the quality and reliability because of increasing software complexity , the need for memory protection will only increase over time. Memory protection is necessary both hardware and software (OS) support.

First, it proposed the protection method to improve a past memory protection method, made the circuit of a past method and the proposed method, and evaluated each area of the circuit and the delay. Proposal MPU was mounted on 32 bit processor , and ported the kernel based on μITRON/PX. From the results the proposed protection unit is better than other method in the protection efficiency , real-time performance and the hardware cost. It made clear that memory protection is valid in hard real-time system.

Key Words hard real-time , memory protection , μITRON/PX

1. はじめに

近年，組込みシステムの大規模化や複雑化に伴う，開発コストの増大や品質・信頼性の低下が問題となっている．単一のプロセッサ上で実現できる機能が多くなっているため複雑なソフトウェアが実装され，その開発や検証にかかる時間が増大している．

このような場合に必要になるのが，各モジュール (タスク) 間の独立性の保証である．障害が発生した場合，システムがモジュール間の不用意な干渉がないことを保証しないと，その障害がどのモジュールにあるかを特定するのが難しい．同様の問題が，ハードリアルタイム性が要求される組込みシステムにおいても発生しており，メモリ保護機構の導入の必要性が高まっている．

メモリ保護を実現するためには，ハードウェアとソフトウェア (OS) の両方におけるサポートが必要である．ハードウェアとしては，仮想アドレスを実現するためのメモリ管理ユニット (MMU) によってメモリ保護を行うのが一般的である．しかし，MMU は一般的

† 名古屋大学大学院情報科学研究科
Graduate School of Information Science, Nagoya University
†† 豊橋技術科学大学情報工学系
Department of Information and Computer Sciences, Toyohashi University of Technology

にアドレス変換バッファ(TLB) に仮想ページから物理ページへの対応の一部をキャッシュしておくが，このキャッシュがミスした場合 (TLB ミス) のペナルティが非常に大きい．TLB ミスは予測が困難なため，割り込みなどに対するリアルタイム性を保証する事が難しい．このため，例えば自動車の制御系のようなハードリアルタイム性が要求されるシステムでは使用できない．アドレス変換は組込みシステムには不要であり，それを持たずリアルタイム性が高いメモリ保護ユニット（以下，MPU）も存在するが，メモリ保護効率が悪い．組み込みシステムにおいてはメモリ効率も重要なため，現状では一般的にハードリアルタイムシステムにはメモリ保護は使われない．

一方ソフトウェアとしては，OS が MMU または MPU への適切な設定を行ったり，サービスコールや例外処理における非特権モードと特権モードの切り替えに対応する必要がある．それらの処理の仕組みが MPU の設計に依存するため，ハードウェアによって OS の持つオーバーヘッドが変化する．このため，ハードウェアを変更した場合，OS についても評価しなくてはならない．

そこで，本研究はメモリ保護効率が高くリアルタイム性にも優れ，ハードウェアコストも低い MPU の方式を示し，それらを組込み向けリアルタイム OS（以下，RTOS）に適用し，その有用性を評価することを目的とする．

従来の MPU と提案する MPU の回路を作成し，面積や遅延を求めハードウェアのコストと性能を求めた．また，提案した回路をプロセッサに組み込み，μITRON/PX 仕様[1] に準拠したのカーネルを移植し，その上でアプリケーションを含んだ評価を行った．

本論文の構成を説明する．第 2 章では従来の MPU の方式について利点や欠点等の特徴を説明する．第 3 章では従来の MPU を改良した方式を提案し，その特徴とソフトウェア上の制約を説明する．第 4 章は提案した MPU を作成し，プロセッサ組み込んだ事と，μITRON/PX 仕様の RTOS をその環境に移植した事を述べる．第 5 章では，作成した MPU の回路面積・遅延の評価や RTOS のサービスコールや割り込み応答性能の評価を行ってる．最後に第 6 章で本論文のまとめを行っている．

2. 従来のメモリ保護ユニットと問題点

2.1 対象システムの特徴と要求仕様

従来の MPU とその問題点について説明する前に，対象とする組込みシステムの特徴について述べる．まず，プログラム全体がひとつのモジュールとしてリンクされて ROM に配置されるために以下の事がいえる．

- メモリ配置は静的
- アドレス変換は不要

また対象システムは，例えば自動車のエンジン制御のように，過酷な環境下でハードリアルタイム性が要求されるものとする．このため以下のような制約がある．

- 50MHz 程度の動作周波数
- 多くても数 MB のメモリ
- キャッシュ等のように正確な最悪時実行時間の予測が難しい機構は搭載しない

このように，キャッシュミスでさえ許容されないシステムにおいては，TLB ミスも同様に許されないため，TLB を用いる MMU は使用できない．

2.2 メモリ保護機能付きリアルタイム OS

本論文で想定するメモリ保護機能付き RTOS の特徴を述べる．まず，タスクやセマフォ等のオブジェクトは何らかの保護ドメインに属す．オブジェクトは自分の属している保護ドメインのオブジェクトに対してはアクセスできるが，それ以外の保護ドメインに属しているオブジェクトにはアクセスできない．

実装について説明する．1 つの保護ドメインに属するプログラムコードやデータは，同じ保護属性 (読み，書き，実行属性) ごとに連続する 1 つのメモリ領域にマッピングされる．例えば，プログラムコード，読み書き可能なデータ，読み込み専用のデータに分類して連続した領域にマッピングする．また，それらのメモリ領域以外に共有データや共有コードのメモリ領域にアクセスしなくてはならない．この事から 1 つの保護ドメインは，コード，データ，スタック，各種共有データやライブラリを含めて，8 個程度の連続したメモリ領域を保護できれば十分である．

2.3 従来のメモリ保護ユニット

現在の MPU を，ページング方式，アドレスマスク方式，リミットアドレス方式に分けて，2.1 を踏まえて各々の特徴と問題点について述べる．

まず，ページング方式について説明する．これは汎用システムで一般的な MMU を用いたメモリ保護であるが，ページ単位の保護しか行えず全て保護領域のページを MMU 内の TLB に保持しておくことが難しい．このため，TLB ミスが生じ最悪時実行時間を悪化させる．特に RISC プロセッサにおいては，TLB ミスに伴う TLB の入れ替えはソフトウェア例外で処理するため，キャッシュミスと比べても非常にペナルティが大きく，ハードリアルタイムシステムへの適用は困難である．また，アドレス変換のための余分なレ

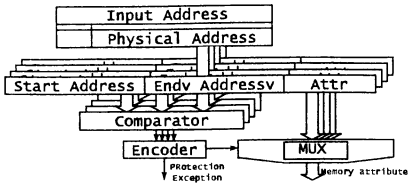

図 1　リミットアドレス方式の構成

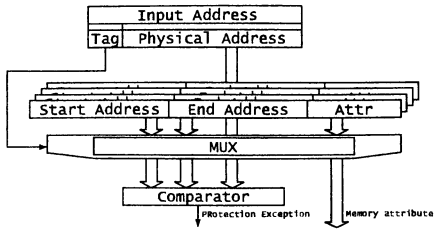

図 2　提案するタグ付きリミットアドレス方式の構成

ジスタやロジックがハードウェアのコストを高くする欠点がある．

次に，アドレスマスク方式について説明する．これはページごとにサイズを可変で大きな領域も保護できる方式である．1 メモリ領域に対して一つのエントリを割り当てればよく，エントリの数が十分であれば，TLB ミスに相当するペナルティは無くなる．ARM 740T/940T MPU[2)]，PowerPC のブロックアドレス変換がこれに相当する．これらの方式は回路を単純にしてコストを抑えるために，上位 n ビットのみの比較によって領域を判別しているのが特徴であり，欠点は保護効率が悪い事である．これは，2 のべき乗単位でしか保護領域を確保できず，未使用領域ができる事を意味する．ただし，ARM においては複数のエントリを利用して 2 のべき乗以外の領域を確保することもできる．また，上位 n ビットとは別にその下の 2 ビットを利用することにより，2 のべき乗とその 3/4 の大きさの領域を確保するような拡張 (2 ビット比較器付きアドレスマスク方式（以下，2 ビット比較方式）) も考えられているが，いずれも保護効率と回路面積とトレードオフになる．

リミットアドレス方式について説明する．これは上端と下端のアドレスを指定して任意サイズの領域を保護する方式である．回路構成を図 1 に示す．IBM System/360 等のメインフレームや Intel の 32 ビットプロセッサ [3)] のセグメントがこれに相当する．必要なエントリ数だけ載せれば，TLB ミスに相当するペナルティもメモリの無駄も無い利点があるが，1 つのエントリに比較器が 2 個必要であり，アドレスマスク方式と比べると制御レジスタの数も 2 つのアドレスを指定するために 2 倍になるため，ハードウェアのコストは説明した 3 種類の中で最も高くなる欠点がある．

3. 提案するメモリ保護ユニット (MPU)

3.1 特　　徴

本論文では，リミットアドレス方式のハードウェアコストを改善したタグ付きリミットアドレス方式を提案する．この方式の回路の構成を図 2 に示す．アドレスの上位ビットをエントリを選択するタグとして扱う事により，エントリ数に関らず比較回路の数を 2 つに固定する．その代わりにエントリを選択するためのマルチプレクサが必要になるが，全体的な回路面積の量は削減される．

問題点は，アドレスの上位ビットをタグとして使用するので扱えるアドレス空間が小さくなることである．しかし，1 つの保護ドメインに仮定の 2 倍の最大 16 個のセクションが必要であると仮定しても 4 ビットしかタグは必要ないため，例えば 32 ビットプロセッサならば，物理アドレス指定に使えるのは 28 ビットでアドレス空間は 256MB となる．これは本論文で想定するシステムに十分対応できる大きさであるため問題はない．

3.2 ソフトウェアが受ける制約

提案方式を利用することにより，2.2 で説明したようにメモリ領域ごとの上位 n ビットの値を変えるという制約を受ける．例えば，コード領域は 0，データ領域は 1，スタック領域は 2，共有ライブラリは 3，共有データは 4 等のようにである．この時に，異なるタスクで共有するメモリ領域が同一のタグを持つように調整するべきである．調整しない場合，ポインタを含んだデータ構造を扱うときにタスクごとにアドレスの解釈をする負担がかかるためである．

4. 実　　装

4.1 ハードウェア

提案したメモリ保護方式と従来の方式を verilog で記述し，ページング方式の MMU を持つ，ルネサス社の 32 ビットプロセッサ M32R-II に組み込んだ．これはプロセッサコアから各種ペリフェラルまで全て verilog で記述してあり，FPGA 上で動かすことができる．MMU はプロセッサコアからキャッシュや内部バスを隠蔽するように命令・データキャッシュ，内部バス間に配置されており，プロセッサコアから出るアドレスを変換したりそのアクセス権のチェックをしている．それを提案した MPU に置き換え，アドレス変換を行わずメモリ保護のみを行うようにした．アドレス変換

を行わないことにより，アクセス権の調査とメモリアクセスを並列に処理することができるため，MPU の遅延は MMU と比べてメモリアクセスの分だけ大きくても許される [4)5)]．また，5.3 で述べるメモリプローブに対応するための制御レジスタを追加した．

4.2 ソフトウェア

メモリ保護機能付き RTOS として μITRON/PX 仕様の IIMP カーネルを使用する．まず，標準の M32R-II に同 OS を移植した．これは M32R-II 標準の MMU によるページング方式によるメモリ保護を行う．この場合，保護ドメインを切り替える際に TLB の制御レジスタを変更する必要がないが，TLB に記憶されていないページを参照するたびに例外が発生し，ソフトウェアによる TLB の書き換えを行う．

次に，M32R-II の MMU を提案する MPU に載せ変えた M32R-II に移植した．この場合，保護ドメインを切り替える際に MPU の制御レジスタを全て書き換える．その代わりに保護ドメインの切り替え以外に MPU の制御レジスタの書き換えは発生しない．

5. 評　　価

5.1 ハードウェアの評価

表 1 に各方式を面積を優先にして合成した結果を示す．なお，合成した環境は次のとおり．

- コンパイラ：Design Compiler Version 2000.11-SP1
- ライブラリ：VDEC ROHM 0.35μm EXD

作成した回路は TLB に相当する部分のみである．よって MPU 全体では制御回路が加わるためこれより大きな面積となる．保護できる領域数は 2.1 で想定した仕様にそって 8 領域とした．対象とするシステムの場合，速度的にはよほど大きくない限り問題とならないので遅延制約に関しては省略する．

結果をみると，アドレスマスク方式は非常に低コストで高速だが，保護粒度を小さくするために 2 つのエントリを併用したり，2 ビット比較方式のように新しくハードウェアを追加したりすると面積または遅延の点での優位性が薄れる．リミットアドレス方式はアドレスマスク方式 (単一) と比べると回路は大きく遅延も大きい．タグ付きリミットアドレス方式は面積はアドレスマスク方式 (単一) に次ぐ小面積だが，遅延が最大という欠点がある．これはタグによって領域を選択するマルチプレクサの遅延による．

これよりコストを抑えるならアドレスマスク方式 (単一) かタグ付きリミットアドレス方式 が適している．ただし，タグ付きリミットアドレス方式は遅延が

方式	面積優先	
	面積 (mm^2)	遅延 (ns)
アドレスマスク方式 (単一)	0.10386	4.53
アドレスマスク方式 (併用)	0.20702	4.53
2 ビット比較方式	0.17038	4.57
リミットアドレス方式	0.16226	8.18
タグ付きリミットアドレス方式	0.11645	10.88

表 1　各方式の回路性能

保護	act_tsk	wai_sem	sig_sem	no_support
有り	2235	1332	1438	940
無し	500	480	560	—

表 2　メモリ保護有りと無しの各 API の所要クロックの比較

若干大きいが，ターゲットとするシステムのクロックは 50MHz 程度であることとを考えると許容範囲である．また，保護効率を考慮すると総合ではもっとも有効であるといえる．なお，プロセッサコアの回路面積は 1.687770mm^2 なため，コアに対しては 10%程度の占有率で済む．

5.2 サービスコールの評価

メモリ保護 OS が保護のない OS と比べてサービスコールがどの程度のオーバーヘッドを生じるかを評価した．以下の API について，μITRON/PX 準拠の IIMP カーネルと保護の無い μITRON 仕様の TOPPERS/JSP カーネル上での必要クロックを計測した．IIMP は JSP をベースにしており，機能的にはほぼ同じである．このため，両者を比較することにより保護の有無による違いを評価できる．なお，2.1 で述べたように断りのない限りキャッシュオフにおける評価である．

act_tsk　タスクの起床
wai_sem　セマフォの獲得待ち
sig_sem　セマフォの返却
no_support　呼び出し時のオーバーヘッド

メモリ保護のない JSP カーネルの各サービスの平均実行時間の対比を表 2 に示す．各 API に関しては常に成功する条件で呼び出しを行った．

no_support より，おおよそのオーバーヘッドが 1000 クロックと分かる．測定した API の JSP における実行時間は 500 クロック程度であり，メモリ保護を有効にすることによって API の実行時間が 3 倍にもなってしまう事になる．ただし，OS のサービスコールが CPU を占有する割合は，通常たかが 1%程度であるため，3 倍になっても全体に大きな影響は与えないともいえる．

リアルタイム性に関しては，例えば act_tsk ならばタスクのアクティベートが成功するか否かでクロック数が大きく変わるものの，同じ条件では実行時間の揺

アドレス	使用ソフト	ソフト	ハード
dom1_mem	オリジナル	460	253
0	オリジナル	540	253
dom1_mem	テストスイート	460	253
0	テストスイート	1014	253

表 3　ソフトまたはハードによるメモリプローブの所要クロック

らぎはみられなかった事から，提案手法を使ったメモリ保護 RTOS の場合は保護なし RTOS と同様にリアルタイム性を確保できる．

5.3　メモリプローブ

メモリ保護 OS では，サービスコール内でユーザタスクから渡されたアドレスにアクセスする時に，そのアドレスに対して現在の保護ドメインがアクセス権限を持っているかを調べなくてはならない．このアクセス権限があるか調べる事をメモリプローブと呼ぶ．メモリプローブはデータキューやメールボックス等のサービスを安全に使う上で必要である．

IIMP カーネルではメモリプローブをソフトウェアによって行っている．保護ドメインのメモリブロック情報を参照するのだが，保護ドメイン数に比例して処理にかかる最悪時間が増える問題があり，性能が悪化する要因となっている．

ところが，MPU を使い，現在の保護ドメインの保護情報が全て MPU に設定されている場合，それを利用してハードウェアでプローブすることが技術的に可能である．そこで，MPU にアドレスを制御レジスタに入力するとそのアドレスのアクセス権を返す仕組みを実装した．評価結果を表 3 に示す．

この結果は保護ドメイン 1 に属するタスクから，自ドメインとどこにも属さないアドレスにアクセスした場合のメモリプローブの所要クロック数について示してある．アドレスが 0 なのは，どのドメインにも所属しておらずアクセス権がない領域である事を示す．dom1_mem は保護ドメイン 1 上の int 型 (4 バイト) メモリを意味する．

表の上 2 行のデータは，ユーザドメイン数 2 のテスト用プログラム上で評価を行い，表の下の 2 行のデータは IIMP のテストスイート上で評価を行ったものである．参考までに静的に生成されたメモリブロックを管理する配列の大きさは，前者は 6 個，後者は 14 個である．テストスイートの配列が大きくなるのはメモリ保護プールや共有ドメインを幾つか持っているためである．

結果を見ると，ソフトウェアによるメモリプローブの場合は，アクセスするアドレスやメモリブロックの数で所要時間が大きく変化する事が分かる．対して今回提案したハードウェアでは一定の所要時間で済み，リアルタイム性の保証が容易である．

5.4　複数タスクが動作する場合の割り込み応答性能

ここでは，提案手法・リミットアドレス方式・アドレスマスク方式と汎用 OS が用いるページング方式における割り込み応答性能の変化を評価する．

ハードリアルタイムシステムにおいては，入力 (外部環境の変化) に対する出力 (アクチュエータ等の動作) までの時間が一定時間に抑えられることが要求される．一般に入力は割り込みによって通知される場合が多いため，この性能指標がリアルタイムシステムにおいて重要である．

この割り込み応答の評価では，何らかのタスクを常に実行状態にしている中で優先度の高い割り込みを発生させている．具体的には，最も低い優先度で JPEG デコーダを動かし，その次に高い優先度で周期的に mp3 デコーダを動かした上で，周期的に割り込みを発生させた．そして，その割り込みを受けたハンドラの中でタスクを起床する API を呼び，実際にタスクが起床するまでの時間を計測した．なお，これらの JPEG デコードタスク，mp3 デコードタスク，割り込みによって起床するタスク，全ては異なるドメインに配置した☆．各デコードタスクは対象プロセッサに浮動小数点演算ユニットが付いていないため，整数演算だけで処理できるライブラリを利用した．

図 3 に従来のページング方式の評価結果と提案した方式の評価結果を示す．従来のページング方式の場合，応答時間が 330 μ秒の場合が殆どであるが，僅かな確率で 450 μ秒前後の場合もある．これは起床したタスクが起動しようとした時，そのタスクの再開するアドレスのページが TLB に登録されていない場合に発生する命令 TLB ミスによるペナルティが原因である．今回の評価プログラムでは割り込みによって起床されるタスクは何もせずリターンするが，もしローカル変数やグローバル変数などを使用している場合はさらにスタックやグローバル変数のあるアドレスにおいてデータ TLB ミスが発生する可能性がある．また，プログラムのサイズが大きいまたはジャンプするなどでページをまたぐ場合に，さらに命令 TLB ミスを引き起こす可能性もある．

それらの可能性を全て考慮すると，1 回の TLB ミスで 100 μ秒程度のペナルティが発生するため，実際

☆ ここで，JPEG デコーダや mp3 デコーダを使用しているのは常にシステムに負荷がかかる状態における割り込み応答性能を計るのが目的なだけで，対象とする制御系のシステムにそれらのアプリケーションが関係するわけではない．

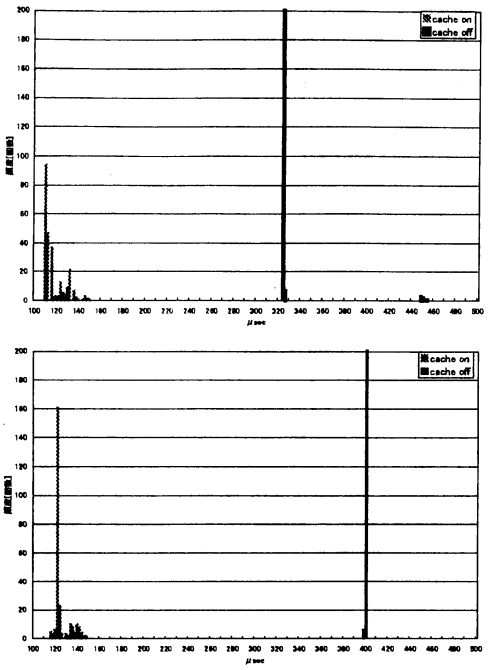

図 3 ページング方式を使用した時の応答性能 (上) と提案方式を使用した時の応答性能 (下)

のタスクの最悪応答時間は今回得たデータよりも遥かに大きくなる可能性を持っている．これはリアルタイム性を保証する上で障害となりうる．

対して提案した手法では，応答の時間が約 400 μ秒と一定で全く例外が無い．ただし，安定はしているが従来のページング方式の TLB なしの応答時間 330 μ 秒と比べると遅い．これはページング方式では起床するタスクの実行するページが TLB に登録されている場合，TLB ミスが発生せず MMU の制御レジスタの書き換えが必要ないためである．提案する手法ではドメインが変わったときに必ず MPU の制御レジスタを書き換えるため，平均的な応答時間は提案する手法のほうが遅くなる結果になる．つまり，平均的な速度をみるとページング方式のほうが有利であり，最大応答時間でみると提案する手法のほうが有利である．言い換えるならば，一般的なアプリケーションならばページング方式，リアルタイム性を要求されるアプリケーションでは提案手法の方が有利である．

参考にキャッシュを有効にした場合のデータも示す．この場合，提案方式の場合でもキャッシュミスによって実行時間が変化するため，実行時間の予測可能性がキャッシュが無効の時よりも劣る．

6. 結　　論

本論文では，組込み制御系で必要とされるハードリアルタイム性を持ったシステムにおける MPU を提案・評価した．まず，従来の MPU の特徴を調査し，柔軟でハードウェアコストのより小さい MPU を提案した．従来の MPU と提案した MPU を verilog で記述し，回路面積や遅延の評価を行い，そのハードウェアコストにおける優位性を確認した．

それから，M32R-II へメモリ保護機能拡張を行った RTOS である IIMP カーネルを移植し，提案した MPU を実装した M32R-II の上にも IIMP カーネルを移植した．その上で，メモリ保護機能の付いた RTOS が持つサービスコールやメモリプローブといったオーバーヘッドを測定した．また，mp3 デコーダや JPEG デコーダといったタスクをその上で動かし，ページング方式と提案方式でどのような性能差が出るかを評価した．

それらの評価の結果，提案した MPU と移植した RTOS 上の組み合わせで，従来の MMU を用いたシステムよりも最悪時実行性能が向上しリアルタイム性の確保が容易になったことが明らかになった．

本研究をまとめると次のようになる．

- 従来の MPU を改良し，評価を行い，保護粒度の改善・ハードウェアコストの削減を確認した
- MPU をプロセッサに組込み，RTOS を含めて評価し，そのオーバーヘッドを明らかにした
- MMU を用いた方式と比べ，提案手法を含め MPU のリアルタイム性における優位性を確認した
- ハードリアルタイムに耐えうるメモリ保護システムを構築できることを明らかにした

参 考 文 献

1) トロン協会バージョンアップ WG, 高田広章編: μITRON4.0 仕様保護機能拡張 (μITRON4.0/PX 仕様) Ver 1.00.00, トロン協会 (2002).
2) ARM Limited: *ARM940T Technical Reference Manual (Rev1)* (1999).
3) Intel Corporation: *IA-32 Intel Architecture Software Developer's Manual Volume 3: System Programming Guide* (2001).
4) Koldinger, E.J., L. H. C. J. and Eggers, S.: Architectural Support for Single Address Space Operating Systems, Technical report.
5) Koldinger, E.J., L. H. C.J. and Eggers, S.: The protection lookaside buffer: efficient protection for single address space computers, Technical report.