A-11

複数メモリ領域保護機能を有するリアルタイム OS の設計

Design of Real-Time Operation System with Multiple Memory Protection Mechanism

山田 晋平†　　中本 幸一†
Shimpei Yamada　Yukikazu Nakamoto

1．はじめに

近年，組込みシステムの大規模化に伴いプログラムが複雑になり，プログラムの想定外の動作が他のプログラム領域に影響を与え，信頼性が低下する問題が起きている．このような，想定外の動作によるアクセスを防ぐための手段として，メモリ保護機構の利用が挙げられる．

PC やサーバといった汎用システムは，プロセッサのメモリ管理ユニット(MMU : Memory Management Unit)によるメモリ保護の機構を備えている[1]．Linux における典型的なメモリ保護では，特権レベルと非特権レベルの 2 レベルを用いることで，カーネル領域を保護する．各プロセスの領域は複数の非特権レベルの多重仮想空間によって保護される．

組込みシステムでは，保護すべき資源が OS からミドルウェアやアプリケーションにまで分散する傾向にあるため，それぞれの領域に対する保護が必要である．

本研究では，組込みシステムにおいて，一つのモジュールの想定外の動作が，他のモジュールやシステム全体に与える影響を抑えることを目的とする．これを実現するため，複数の領域間において許可されたアクセスのみを可能とするメモリ保護機構を提案し，このメモリ保護機構を用いたリアルタイム OS の設計について述べる．

2．関連研究

PC・サーバにおいて，粒度の細かいメモリ保護機構を提供する研究が行われている．文献[2, 3]では，プロセス内部に，拡張コンポーネントのための粒度の細かいメモリ保護機構を提案している．これはプロセス内の拡張コンポーネントに異なる保護特性（読込み，書込み，実行）を設定できるものである．Nooks では，Linux のデバイスドライバのバグにより，他の領域へ書き込むことによる誤動作を防ぐため，デバイスドライバをカーネルアドレス空間内で，異なる保護機能を有する低い特権レベルで動作できるようにしている[4]．また，動的にメモリ保護情報を変更するものとして，文献[5]がある．ここでは，非特権レベル空間に書き込まれた悪意のあるプログラムを Linux カーネルから実行できないようにするために，システムコール実行時に仮想マシンモニタでユーザプロセスの保護情報を変更している．

組込みシステムにおいては，メモリ保護機能を提供する ITRON4.0/PX 仕様[6]と，その実装である IIMP カーネル[7]がある．これは複数のタスクをまとめて保護ドメインに置き，保護ドメイン間のタスクアクセスをできないようにしている．保護ドメインは非特権レベル空間，特権レベル空間の両方に置くことができる．しかし，これらの関連研究では，必要最低限のメモリのみをアクセス可能とする機能，さらにこれを特権レベルでも実現したものはない．

† 兵庫県立大学大学院 応用情報科学研究科
Graduate School of Applied Informatics, University of Hyogo

3．組込みシステムで求められるメモリ保護

組込みシステムでメモリ保護機構を用いる場合に考慮すべき点を以下に述べる．

3.1　複数領域の保護

1 章で述べたとおり，組込みシステムでは，OS のみならず，アプリケーションまでもが，保護対象である資源を直接管理することが多い．そのため，それぞれの領域を保護できることが求められる．

3.2　特権レベル領域の保護

特権レベルと非特権レベルの 2 レベルを用いた保護では，非特権レベルのプログラムから特権レベルの OS を呼び出すために，時間のかかるシステムコールトラップを用いる必要がある．これは時間制約の厳しい組込みシステムでは大きなオーバヘッドとなる．そこで図 1 のように，特権レベル内にもう一つのレベルを設け，従来の特権レベルを L1，新たに設けたレベルを L2，非特権レベルを L3 とする．L1 と L2 は同じ特権レベルに存在するため，システムコールトラップと比べて高速な関数呼出しで互いを呼び出すことができる．

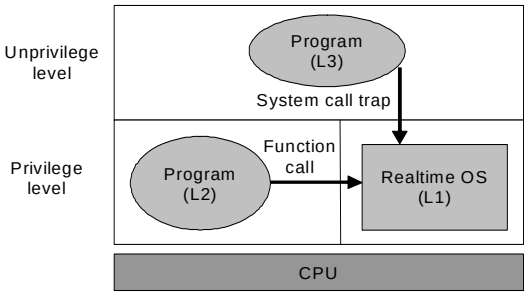

図 1　保護のレベル

リアルタイム性の求められるプログラムは L2 に配置し，ダウンロードされるプログラムなど OS に影響を与える可能性があるものは L3 に配置するものとする．

L2 のプログラムは L1 に配置される OS に影響を与えないことを前提とするが，想定外の動作により影響を与えてしまうことが考えられる．そのため，L1 と L2 の間でもメモリ保護が必要となる．

4．組込みシステム向けメモリ保護機構

本メモリ保護機構の目標は，以下のとおりである．

- プログラム実行時に必要最小限のメモリ領域をアクセス可能とすること
- 組込みシステム特有の，分散した資源管理に対応したアクセス制御を行うこと
- 上記の制御を特権レベルで実現することにより，非特権レベル実行時に発生するシステムコール呼出しのオーバヘッドをなくすこと

4.1 リージョン

本メモリ保護機構では，メモリ保護の単位をリージョンと呼ぶ．リージョンとは，メモリに対する同じ保護情報を有する複数のプログラムをまとめ，それらのプログラムのメモリ上における配置に必要な情報を合わせたものである．リージョンには非特権リージョンと特権リージョンが存在する．非特権リージョンはプロセッサの特権モード・非特権モードに関わらずアクセスでき，特権リージョンは特権モードでのみアクセスできるリージョンである．リージョンは複数定義でき，システム内のすべてのプログラムは，定義されたリージョンのいずれか一つに所属する．

現在実行中のプログラムが存在するリージョンを，現在のリージョンと呼ぶ．また，あるリージョンから別のリージョンの関数を呼び出し，現在のリージョンが変更することを，リージョンの遷移と定義する．

4.2 リージョン間の関係

本メモリ保護機構では，リージョン間でどのようなアクセスが可能であるかを示す accept という関係を定義する．accept 関係は，図 2 のように，あるリージョンから別のリージョンへの，読込み(r)，書込み(w)，実行(x)のそれぞれについてのアクセスの関係を表す．読込み，書込み，実行のすべてのアクセスが禁止される場合，accept 関係は存在しない．

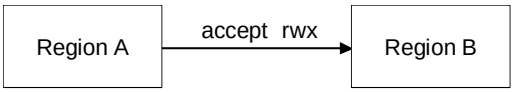

図 2　accept 関係

4.3 メモリ保護機構の動作

本メモリ保護機構ではリージョン遷移時に保護情報を変更する changeProt という API を提供する．本メモリ保護機構の accept 関係は changeProt を利用して実現される．changeProt は図 3 に示すように，保護状態を遷移先リージョンに応じたものに変更してから目的の関数を呼び出す．呼出しが終了すると保護状態を遷移前に戻し，呼出し元に処理を戻す．保護状態の変更には，プロセッサの MMU のメモリ保護機能を用いる．

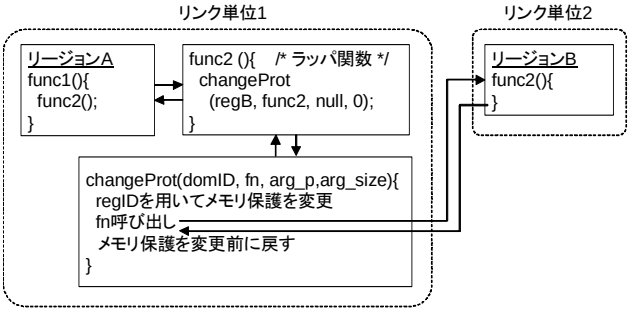

図 3　changeProt を用いたリージョン遷移

5. メモリ保護機構のリアルタイム OS への適用

本メモリ保護機構をリアルタイム OS に適用し，組み込みシステムで求められる保護機能を備えた OS を作成する．本研究ではメモリ保護機構の適用対象として，T-Engine[8]用の OS である T-Kernel を用いる．

メモリ保護機構を適用する OS において，タスク毎に独立してリージョンの遷移を行うため，タスクの TCB に現在のリージョンの情報を追加し，タスクディスパッチ時に TCB のリージョン情報に基づいて保護状態を変更する．

T-Kernel 本体は L1 のリージョンに配置する．T-kernel におけるシステムコールは，tk_cre_tsk()のような関数呼出しのインタフェースで提供されており，これらの関数からシステムコールトラップを用いてシステムコールエントリルーチンを呼出し，システムコール関数コードを用いて目的のシステムコール本体を呼出す．本メモリ保護機構では非特権リージョンから特権リージョンへ遷移する場合，ラッパ関数で遷移先リージョンなどの情報をセットした上で，システムコールトラップを用いて changeProt を呼び出し，保護状態を変更した後，目的の関数を呼び出す．この手順は前述のシステムコール呼出しと同じであり，システムコールのインタフェースである関数にラッパ関数の処理を，図 4 のようにシステムコールエントリルーチンに changeProt の処理を組込むことで，L3 から L1 への呼出しに対応できる．

L2 と L1 は同じ特権レベルであるため，システムコールトラップを用いないインタフェースの関数を設けることで，より高速にシステムコールを呼び出すことができる．

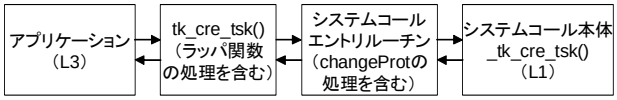

図 4　システムコール呼出しとリージョン遷移

6. おわりに

本稿では，組込みシステムの特徴を考慮したメモリ保護機構と，これを有するリアルタイム OS の設計について述べた．今後は設計に基づき実装・評価を行う．

参考文献

[1] A. Silbeschatz, P. Galvin and G. Gagne, Operating System Concepts, 6th edition, John Wiley & Sons, Inc., 2002.

[2] 品川高廣，河野健二，高橋雅彦，益田 隆司，"拡張コンポーネントのためのカーネルによる細粒度軽量保護ドメインの実現，" 情報処理学会論文誌，vol.40，no.6, pp.2596-2606，June 1999.

[3] M. Takahashi, K. Kono and T. Masuda, "Efficient Kernel Support of Fine-Grained Protection Domains for Mobile Code," Proc. 19th IEEE International Conference on Distributed Computing Systems, pp.64-73, May 1999.

[4] M.M. Swift, B.N. Bershad and H.M. Levy, "Improving the reliability of commodity operating systems," Proc. the 19th ACM Symposium on Operating Systems Principles, pp.207-222, Oct. 2003.

[5] A. Seshadri, M. Luk, N. Qu and A. Perrig, "SecVisor: a tiny hypervisor to provide lifetime kernel code integrity for commodity OSes," Proc. 21st ACM SIGOPS symposium on Operating systems principles, pp.335-350, Oct. 2007.

[6] TRON 協会，" μ ITRON4.0 仕様 保護機能拡張 Ver.1.00.00，" 2002.

[7] TRON 協会，"IIMP プロジェクト". http://www.assoc.tron.org/iimp/

[8] T-Engine フォーラム，http://www.t-engine.org/