組込み機器におけるメモリ監視機構の実現と評価

杉本晴秀† 福田亮平† 楯岡孝道† 鈴木貢† 中山泰一‡
†電気通信大学 情報工学科

1 はじめに

近年, 組込み機器がネットワーク上に配置され, そこでは特殊なハードウェアや専用の OS ではなく, ネットワークなどの豊富な機能を持った Linux のような汎用 OS を改良した OS を利用する場合が増えている. 一方で, ネットワークに接続されることで様々なリスクが増加した.

例えば, 攻撃の踏み台にされる, ウイルスなどの悪意あるコードを実行されるといったことが挙げられる. このような攻撃に対する防衛技術は汎用計算機向けには多数提案されているが, 基本的に組込み機器のような少ない資源上での利用を想定してはいない.

基本的に, 組込み機器は低いコストで構成されることが求められる. 汎用プロセッサには MMU（メモリ管理ユニット）が搭載されているが,MMU を用いてアドレス変換を行うと, そうでない場合に比べてコンテキストスイッチのオーバヘッドが大きくなる [1]. よって, より低いコストが求められる組込み機器では,MMU が搭載されないプロセッサも使われ続ける可能性がある.

そこで, 本研究では MMU がないプロセッサも視野にいれた, 組込み機器のセキュリティ向上を目的としたメモリ監視機構の設計と実装を行う.

2 関連研究・技術

2.1 メモリ保護ユニット（MPU）

ハードウェア的にメモリ空間を保護する手法としては, 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が挙げられる. 各方式の特徴を以下に考察する.

（1）ページング方式:メモリ空間をページ単位で管理・保護する方式. 一般的な MMU ではメモリ空間をページ単位で管理・保護する.

（2）アドレスマスク方式:サイズごとに可変で大きな領域を保護可能な方式. PowerPC のブロックアドレス変換などがこれにあたる. この方式ではアドレスの上位数ビットを比較してどの領域に属しているか判別する.

（3）リミットアドレス方式:古典的なリミットアドレス方式では上位, 下位アドレスを指定して任意サイズの領域を保護するのに対して, その発展系としてタグ付きリミットアドレス方式 [2] が提案されている.

その他のメモリ保護技術として Mondrian Memory Protection[3] がある. これはハードウェアのメモリ保護をワード単位で行う. ページ単位でのメモリ保護に比べ, ワード単位で保護する場合, メモリの未使用領域の割合を減らすことができ, メモリの有効利用が可能である. しかし,MMU でのページ単位保護に比べ, コストが非常に大きく, 組込み機器のような小規模環境下で利用するには, コストや機能の点で過剰性能である.

3 実装環境

FPGA 評価ボード SUZAKU-S 上に実装を行う. その仕様は, 表 1 の通りである.

表 1: SUZAKU-S の仕様

プロセッサ	Microblaze
周波数	51.6096MHz
Flash メモリ	4Mbyte
SDRAM	16Mbyte

4 設計

4.1 前提条件

設計のための前提条件として以下を設定する.

1. アクセス制御により悪意のあるユーザプログラムの実行は拒否可能
2. 異常発生時には再起動が許される.

第 1 の前提条件を設定可能である理由は, 組込み機器におけるアクセス制御機構によって, 強制アクセス制御が可能であるからである. 第 2 の前提条件を設定可能である理由は, 組込み機器が想定外のメモリアクセスを受けて暴走した場合, 再起動してメモリを初期化を行うほ

Memory Observation for Embedded System
Haruhide Sugimoto†, Ryohei Hukuda†, Takamichi Tateoka†, Mitsugu Suzuki†andYasuichi Nakayama‡
†Department of Computer Science, The University of Electro-Communications

うが, 再起動を行わない場合に比べて安全であるからである.

4.2 メモリ監視機構

SUZAKU-S など MMU が存在しない環境下では実メモリ上のいかなる場所でも自由に書き換えることが可能である. そこで, 本研究ではユーザプログラムの誤作動でメモリ上のコード領域が破壊されることを防ぐ事を目的とし, 以下のようにメモリ監視機構を設計する.

4.2.1 ハードウェアの設計

(1) FPGA 上に VHDL で記述することで実装
(2) CPU には OPB スレイブ (図 1) として接続
(3) 保護はページ (4kbyte) 単位
(4) ユーザロジック (Memory_Check_core)

Suzaku-S の場合は 16Mbyte の記憶領域を持つため,4096 ページ分の保護情報が必要. 保護ビットを 1 ビットとして, 全ページの保護情報として内蔵可能な BRAM を用いてユーザロジックの内部に作成する. 今回の場合,1/ (4096*8) =1/32768=0.03 パーセント程度の記憶オーバヘッドで済む. プロセッサから OPB バスを通じて出力されるアドレスを解析し, BRAM 上の保護ビットとの比較を行い, 割り込み信号を出力する. 処理の流れを図 2 に示す.

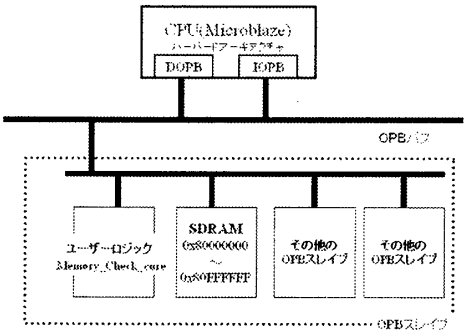

図 1: ハードウェアのブロック図

4.2.2 ソフトウェアの設計

(1) ユーザロジック内蔵 BRAM に対する保護ビット変更処理
(2) ユーザロジックからの割り込みを処理

プログラムがメモリに展開された際のコード領域が存在しているページに保護ビットを立て, プログラム終了時には保護ビットを下げるように設計する. ユーザロジックからの割り込み信号を受けた場合には, 再起動を行うことで, プログラムの予期しない誤動作によるシステム暴走を防ぐ.

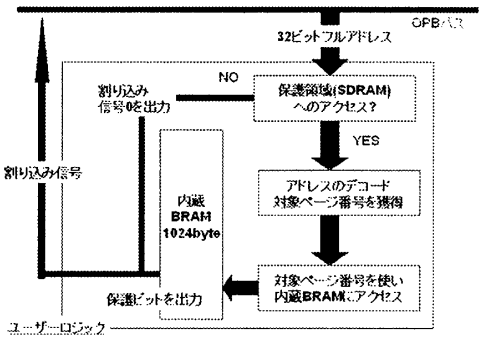

図 2: ユーザロジック処理の流れ図

5 まとめと今後の課題

本稿では MMU が搭載されていないプロセッサを対象とした, 組込み機器のセキュリティ向上を目的としたメモリ監視機構の設計について述べた.

今後の課題として, ユーザモードとカーネルモードの切り替えへの対応があげられる. 通常 MMU は CPU の一部として実装されるため,CPU 内の状態レジスタを参照することにより現在のプロセスがどのモードで動いているかを判断することが可能である. しかし, 本機構の実装形式では CPU から出ている OPB バスのスレイブとしてユーザロジックが実装されているため, 現在のプロセスがどちらのモードで実行されているかわからない. このためカーネル領域の保護ができないという問題がある. 対処方法としては以下の 2 つを考えている. (1) 全ての記述が公表されている CPU を利用し, その一部として検出機構を実装する. (2) カーネル内部でモードが切り替わる場所を特定し, そのタイミングでユーザロジック内のビットを切り替える.

参考文献

[1] http://opensrc.sec.samsung.com/document/ctx-perf-linux-2.6.11.6.pdf

[2] 西部満, 本田晋也, 冨山宏之, 高田広章, “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” 情報処理学会 研究報告 SLDM-119, pp115-120 (2005).

[3] E. Witchel, J. Cates, and K. Asanovic,“Mondrian Memory Protection” In proc. of ASPLOS (2002).