B-043

組込み用 OS のネットワーク機能の設計と形式手法 SPIN による検証

Design of the network function for an operating system for embedded systems and its verification by Spin Model Checker

笠原 良太† 望月 祐希‡ 野口 健一郎‡
Ryota Kasahara Yuuki Mochizuki Kenichiro Noguchi

1. はじめに

安全な組込みシステムを実現するためのオペレーティングシステムとして神奈川大学 野口研究室で試作中の OS にネットワーク通信機能を追加する試作研究を行った。

本研究の目的は次のものである。

(1) 組込みシステム用 OS に求められる高信頼・高セキュリティ・高性能のネットワーク機能の実現
(2) (1)の正当性証明の為の形式手法 SPIN を用いた検証

2. 背景

2.1 OS-K

OS-K は神奈川大学 野口研究室で試作中の組込み用 OS([1]～[3])である。IA-32 アーキテクチャのプロセッサ上で動作する。特徴として、セパレーションカーネル方式を採用し、デバイスドライバなど、通常の OS ではカーネル空間で動作させるコンポーネントをユーザ空間で動作させる点などがある。

2.2 SPIN

SPIN はモデル検査ツールである。並行プロセスや、個々のプロセス内での非決定的な振る舞いを記述したものに対し、可能な動作を網羅的に探索して、指定した性質が成立するかどうか自動的にチェックする。振る舞いは仕様記述言語 Promela で記述する。

3. ネットワーク対応 OS-K の試作

3.1 概要

本研究で試作したネットワーク対応 OS-K は以下の特徴を持つ。

(1) パーティション別役割

ネットワーク機能を実装するために、パーティション OS への機能追加と、二つのシステム用パーティションの追加を行った。

(a) パーティション OS(P-OS) : AP へソケット通信を行うためのインターフェースの提供。ネットワークサーバとのインターフェースを持つ。
(b) ネットワークサーバ(NS) : 主に通信パケットの制御を行う。P-OS とイーサネットドライバとのインターフェースを持つ。
(c) イーサネットドライバ(ED) : イーサネットコントローラ(EC)の制御を行う。ネットワークサーバとのインターフェースを持つ。

(2) ネットワーク機能の非特権モード動作

OS-K のアーキテクチャに準じ、ネットワーク機能はユーザ空間(非特権モード)で動作させるものとした。このことによって、以下の利点などが得られる。

(a) 設計が簡素化できる。
(b) ネットワーク機能にバグが存在してもシステム全体が止まらない。
(c) アクセスできるメモリ空間を限定することでセキュリティの向上に繋がる。

(3) 非特権モードでの割り込みの扱い

上位層から下位層への要求をダウンコールといい、下位層から上位層への通知をアップコールというが、従来の OS-K のメッセージ通信機能ではダウンコールのためのメッセージ通信待ちとアップコールのための割り込み待ちを同時に行えなかった。これでは性能の確保が難しい。そこで、この問題を解決するために(a), (b)の変更を加えた。

(a) 割り込み待ちをメッセージ通信として扱えるようにした。
(b) メッセージ通信に通知(notify)機能を追加した。

3.2 構成

全体は図 1 のように構成した。

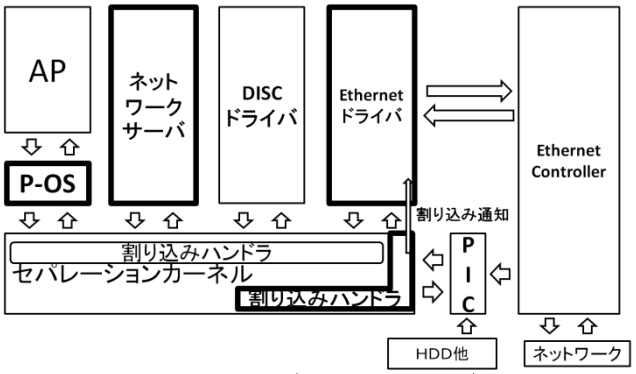

図 1: 全体構成(太枠部分を追加)

3.3 前提とするハードウェア環境

以下にネットワーク対応 OS-K を動作させるために本研究で使用したハードウェア環境を示す。

(1) システムボード:ALIXsystemboards(Model:alix1d)
(2) イーサネットコントローラ:VIA 社 VT6105M

3.4 セパレーションカーネルへの仕様変更・追加

従来の OS-K のセパレーションカーネルの機能の中で、以下の機能について、仕様変更・追加を行った。

(1) メッセージ通信機能

receive に、notify による通知メッセージと、割り込みの通知を受け取る機能を追加した。また、SK コールとして notify を追加した。

(2) ハードウェア割り込みハンドラ

イーサネットコントローラからの外部割り込みを受け付ける際の割り込みハンドラを追加した。また、従来の OS-K は外部割り込みの処理方法として dwait (指定した割り込み番号に対応するハードウェア割り込みの発生を待つ) 方式が取られていたが、これをやめ、割り込み通知を receive で受け取れるように変更した。

3.5 開発結果

開発には C 言語とアセンブリ言語を用いた。各構成要素の開発量を次に示す。

† 神奈川大学大学院理学研究科情報科学専攻 Kanagawa University（現在 日立情報通信エンジニアリング株式会社 Hitachi Information & Communication Engineering, Ltd.）
‡ 神奈川大学大学院理学研究科情報科学専攻 Kanagawa University

P-OS	430 ステップ
ネットワークサーバ	596 ステップ
イーサネットドライバ	668 ステップ

4. 実機テスト

4.1 テスト環境

ネットワーク対応 OS-K のテスト環境として図 2 のように配置した。(Wireshark：ネットワークフレームキャプチャソフト)

アプリケーションドライバはコンソールを利用してコマンドを打ち込むことで任意の文字列を送信できるようにした。また、受信の場合も、コンソールを利用してコマンドを打ち込むことで受信している(または受信する)データを受け取り、画面に出力できるようにした。

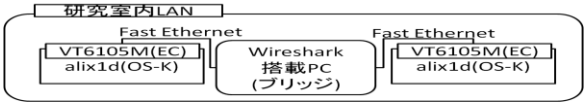

図 2: テスト環境

4.2 テスト結果

テストの結果、送受信ともに問題なく動作した。

5. SPIN による検証

文献[1] にて発表した OS-K の SPIN モデルに本研究のネットワーク通信機能のモデルを追加する事で検証を行った。

5.1 仕様の記述・モデル化

設計を図 3 のようにモデル化した。さらに、イーサネットコントローラとネットワーク(NW)についてスタブとしてモデル化した。

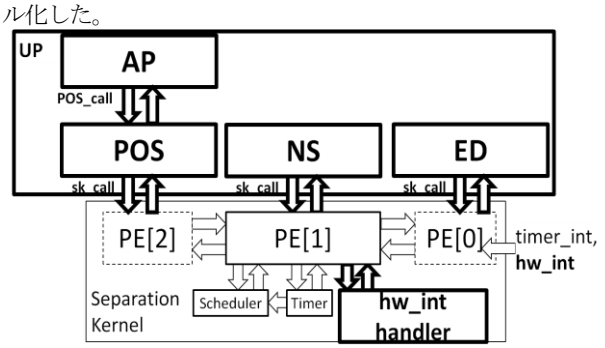

図 3: OS-K モデル全体構成(太枠部分を追加)

5.2 前提・事前条件

送信ケース・受信ケースを別に行い、AP スタブはどのタイミングかで必ず 1 回送信/受信要求を P-OS へ送る。NW スタブは受信ケースの場合のみ、受信データを EC スタブへ送る。

5.3 検査内容の決定と検証結果

SPIN による網羅検証を行う際の検証項目として定めたものは以下である。

(1) NW システムがデッドロックに陥らない
(2) NW システムがライブロックに陥らない
(3) 各プロセスが妥当な状態で終了する

以上の項目を検査できるようにコーディング内に記述を加え、送信と受信のそれぞれのケースの網羅探索を行った。結果、エラーケースは発見されなかった。

6. 評価・考察

得られた成果は以下の通りである。

(1) 試作結果について

(a) 一通りのプロトコルスタック(ソケット機能、UDP、IPv6、Ethernet)を、一部は簡易実装ではあるが、実現できた。
(b) セパレーションカーネル上に、ユーザモードプログラムとして、ネットワーク機能を実現できた。
(c) OS-K が提供する仮想的分散環境を利用し、複数パーティションに分割して実現した。(P-OS、NS、ED)
(d) ネットワーク機能の実現で課題となるアップコール問題については、OS-K のメッセージ通信機能に機能追加を行うことにより、性能的に問題のない機能を実現できた。

以上より、一定の信頼性、セキュリティ、及び性能を確保したネットワーク機能を実現出来た。

(2) 形式手法を用いた検証について

(a) 本研究の OS-K は一種の分散環境であり、やりとりの同期などでバグが出やすい。それに対して網羅探索をすることで、高信頼性を保証できた。
(b) 後付けの検証になってしまった。しかし、実機テストでは網羅テストは不可能であるが、SPIN では、制限の元ではあるが、それができた。
(c) ただし、仕様のバグは見つからなかった。

(3) セパレーションカーネル方式の OS-K の評価

仮想的分散環境の提供はネットワーク機能をモジュール化して実現するのに有効だった。

(4) 形式手法適用の評価

SPIN モデルの記述には仕様記述言語 Promela を用いた。各構成要素の開発量を表 1 に示す。

表 1: 開発量一覧

	SK(従来[1])	SK(本研究追加部)	NWコンポーネント部	スタブ部	計
SPINモデル	851	224	301	147	1523
試作OS-K	6670	128	1781		8579

このことから、SPIN を使ったモデル検証は実際の実装と比べると、かなり大きな比率のコーディングが必要であるということがわかった。

7. 今後の課題

今後の課題を以下に示す。

(1) 試作について

(a) ARP のサポート
(b) TCP のサポート

(2) テストについて

(a) 複数ホストからの受信
(b) OS-K 以外の OS との通信

(3) SPIN モデル検証について

複数 AP の検証。

参考文献

[1] Kei Kawamorita, Ryouta Kasahara, Yuuki Mochizuki, and Kenichiro Noguchi: Application of Formal Methods for Designing a Separation Kernel for Embedded Systems, World Academy of Science, Engineering and Technology Issue 68, July 2010, pp. 506-514.
[2] 川守田 慶, 野口 健一郎:形式手法を用いた組込み用 OS の試作-B メソッドによる仕様検証実験-, FIT2008.
[3] 野口 健一郎, 川守田 慶:拡張状態遷移技法を用いた仕様検証の実験　組込み用 OS 試作への適用, FIT2008.