仮想化技術を用いたリカバリ指向組み込みシステムの開発

石井　正将†　福島　拓†　巻島　一雄†　杵渕　雄樹†　中島　達夫†
早稲田大学理工学部コンピュータ・ネットワーク工学科†

1　序論

組込み機器を含め，ITシステムには多くの障害が発生する．障害の原因や影響の大きさなどは様々であるが，システムがクラッシュする原因の多くは，デバイスドライバで発生した障害であると言われている．デバイスドライバはOSと深く関わっているので，障害が発生した場合，復旧することが難しい．

本研究では，システムの中心となる汎用OSとデバイスドライバを実行するためのOSを，仮想化技術を用いて並列実行させる．これにより，デバイスドライバとOS本体の結合を疎にし，障害が発生した場合でも復旧可能なシステムを構築する．

2　関連研究

Microreboot

Microrebootとは，分散システムでの高可用性を実現するための仕組みである[1]．システムに障害が発生したとき，多くの場合はシステムの再起動により，復旧することができる．Microrebootでは，システムをいくつかのコンポーネントに分割し，障害が発生したコンポーネントのみを再起動することで，システムの復旧を行う．コンポーネントごとに再起動することで，再起動にかかる時間を短縮することができる．また，一つのコンポーネントを再起動している間でも，他のコンポーネントは実行可能であるため，システム全体が使用不能になることを防ぐことができる．システム全体を再起動する場合に比べ，Microrebootでは，失敗するリクエストの数を10分の1に抑えることに成功している．

今回の研究では，並列実行するOSのそれぞれを分散システムのノードと見なすことができる．単一マシン上で実行される複数のOSからなるシステムを，仮想的な分散システムと見ることで，Microrebootの手法を本研究に取り入れることができる．

ArcOS

ArcOSとは，リカバリによるデバイスドライバの自己修復を可能にしたOSである[2]．L4 microkernel上に，デバイスドライバの自己修復フレームワークを実装している．デバイスドライバに障害が発生した場合，デバイスドライバ単体での再起動を行い，障害を復旧する．デバイスドライバ単体を再起動させるので，復旧に必要な時間が短くなる．実験では，キャラクタデバイスドライバとVESAビデオドライバの二種類のデバイスドライバの実装を行い，再起動にかかる時間を測定している．共に，0.2msで再起動が可能となっている．

ArcOSでは，デバイスドライバのリカバリをOSの機能として実装している．今回提案する手法では，デバイスドライバのリカバリを，他のOS上に実装することで，既存のシステムへの導入を容易にできると考えられる．

3　技術背景

SPUMONE

本研究では，軽量CPUコア仮想化技術SPUMONEを使用している．SPUMONEは，図1のように，1つの物理CPUの上に，複数の仮想CPU(VCPU)を提供する．それぞれのVCPUにOSを割り当て，VCPUを切り替えながら実行することで，複数のOSを並列に実行することが可能になる．

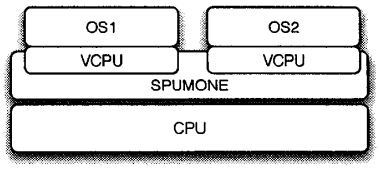

図1: SPUMONE概要

OS間通信

OS間でのデータのやり取りのために，OS間通信(IOC,Inter-OS communication)の仕組みが必要となる．

The development of recovery oriented embedded system using virtualization
†Masayuki ISHII, Taku HUKUSHIMA, Kazuo MAKIZIMA, Yuki KINEBUCHI, Tatuo NAKAZIMA, Dept. of Infomation and Computer Science, Faculty of Science and Engineering, Waseda University

SPUMONEは，VCPU間のプロセッサ間割り込み(IPI,Inter-processor interrupt)と共有メモリによって，OS間通信を実現している．

4 提案方式

システムの主な処理を行う汎用 OS とは別に，デバイスドライバを実行するための OS を用意する．これらの複数の OS を，仮想化技術を用いて並列実行させる．

OS 間通信を用いて，汎用 OS からデバイスドライバ用 OS にデータを送信する．デバイスドライバ用 OS は，受信したデータを元にデバイスの操作を行う．

実装

今回，デバイスドライバを実行するための OS(SOS,Simple-OS) の実装と，SOS 上で動作する SCIF(Serial Communication Interface with FIFO) シリアルドライバの実装を行った (図 2).

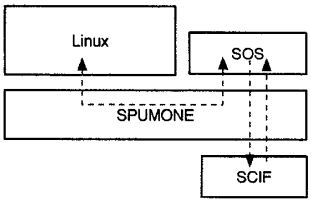

図 2: システムアーキテクチャ

SOS はラウンドロビン方式のスレッドが動作する OS である．CPU のタイマ割込みにより，定期的に SOS のコンテキストスイッチが実行される．SOS 上で行うべき処理がない場合，直ぐに Linux へ処理が切り替わる．Linux からデータを受信した場合，デバイスの操作を行い，結果を返信する．システム全体の動作は図 3 のようになる．

SOS，または SOS 上のデバイスドライバに障害が発生した場合，Linux 側から信号を送ることで，SOS のリブートを行うことができる．SPUMONE の IOC(OS 間通信) には複数のチャンネルが存在する．今回は，データ送受信用と SOS 制御用の計二つのチャンネルを用いた．SOS 制御用チャンネルにデータを書き込むと，SOS の再起動により，システムの復旧を図る．

5 評価と考察

評価環境

本研究では，SH4A プロセッサ評価用ボート SH-2007(CPU SH7780(400MHz)，RAM 128MB，Linux-2.6.20) での評価を行った．

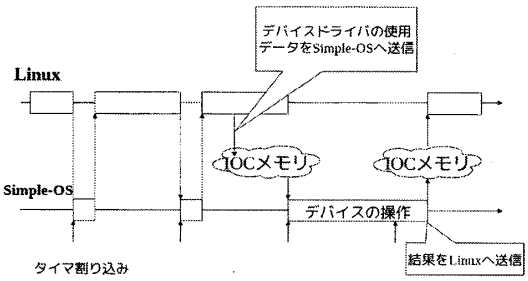

図 3: デバイスドライバ使用時の動作

再起動に要する時間

デバイスドライバに障害が発生したと仮定して，デバイスドライバの復旧に要する時間の測定を行った．結果として，平均で 56ms の時間で SOS を再起動することができた．SOS の機能を最小限に留めることで，復旧にかかる時間を抑えることができたと考えられる．

6 まとめ

本研究では，高可用性を実現するためのシステムアーキテクチャを提案し，その実装を行った．実験では，56ms での復旧が可能であった．

本方式によって，OS とデバイスドライバの結合が疎なシステムが実現できる．これにより，デバイスドライバに障害が発生したとしても，システムの復旧が可能になる．

今後は，複数のデバイスドライバの実装を行うとともに，性能評価を行っていく．

参考文献

[1] George Candea, Shinichi Kawamoto, Yuichi Fujiki, Greg Friedman, Armando Fox: Microreboot - A Technique for Cheap Recovery, 2004

[2] Hiroo Ishikawa, Alexandre Courbot and Tatsuo Nakajima: A Framework for Self-healing Device Drivers, 2008