VMMによる耐故障性環境の構築*

青柳 信吾 † 追川 修一 ‡
筑波大学 システム情報工学研究科 ††

1 はじめに

近年，自動車や航空機，基幹設備など生活基盤を支える多くシステムに組込み機器が利用され，それぞれの設計において耐故障性が重要となっている．耐故障性とはシステム内で発生した故障の回復，分離により，システム障害を回避する特性のことであり，もし生活基盤を支えるシステムにおいて耐故障性を含めた設計が行われていない場合，社会へ重大な影響を及ぼす危険性がある．

耐故障性の解決手法としては，一般にシステム構成要素の冗長化がある．各構成要素を複数個実装することにより，単一要素の故障を隠蔽し，システム障害を回避する手法である．しかし構成要素がソフトウェアの場合，要素の内部状態がシステム全体に強く影響するため，内部状態を含め冗長化する必要がある．

この解決手法として，プライマリバックアップモデルがある [1]．プライマリバックアップモデルは，主資源となるプライマリと，冗長資源となるバックアップから構成され，障害が発生していない場合には主資源のプライマリを，プライマリに障害が発生した場合にはバックアップを利用することで，単一障害の隠蔽を実現する．またプライマリバックアップモデルでは，プライマリの内部状態をバックアップへ複製することにより，状態を含めたの冗長化を実現する．

本研究では，耐故障性を含む組込み機器実現のため，プライマリバックアップモデル機能を提供する仮想マシンモニタ (Virtual Machine Monitor : VMM) の実装を行う．VMM は実マシン上に仮想マシン（Virtual Machine : VM）を提供するソフトウェアであり，VMM が耐故障性機構を提供することで，実マシン上で動くプログラムに変更を加えることなく，それぞれへ耐故障性を提供することができる．以上の実装を行い，組込み機器の耐故障性向上のために最低限必要な構成要素，および必要となる実装量の解析を行う．

2 設計と実装

本研究の VMM 構成を図 1 に示す．VMM は仮想 CPU，仮想デバイス，デバイスドライバ，通信機構から構成され，これらにより VM 上で実行される OS (ゲスト OS) Linux 2.6.23 が管理される．VM は仮想 CPU，仮想デバイスから構成され，ゲスト OS の状態の管理機能やゲスト OS 実行機能を提供する．デバイスドライバは実マシンのデバイス制御を行い，VMM に対してデバイス操作機能を提供する．通信機構はデバイスドライバと VM を制御し，ゲスト OS をプライマリからバックアップへ移送する．

2.1 仮想マシンの設計

本研究で利用する IA-32 アーキテクチャは VMM の実装が困難なアーキテクチャであるため [2]，ハードウェア仮想化機構 Intel VT-x を利用する．VT-x は VMM 実装をサポートするハードウェア機構であり [3]，ゲスト OS の発行する特定の命令を検出機構を持つため，容易な VMM の設計，実装が可能となる．本システムでは，VT-x によりゲスト OS の発行する命令を検出し，一部の命令を VMM が管理することで仮想 CPU を実現する．また，移送の際にプライマリからバックアップへ転送する仮想 CPU の状態は，VT-x の管理する状態のうちゲスト状態（Guest-state）と VM 実行状態（VM-execution control）のみである．

プライマリバックアップモデルの実現には，ゲスト OS が管理しているデバイスの内部状態も転送する必要がある．本システムではデバイス状態の取得を容易にするため，VMM はゲスト OS に対して仮想デバイスを提供する．これらの仮想デバイスはゲスト OS からの I/O 命令により操作され，VMM がこれらの操作を実デバイスへ通知することで実際の I/O が行われる．デバイス状態取得のため，VMM は仮想デバイスの初期化操作を監視し，内部状態を取得する．これにより，VMM はプライマリからバックアップへのデバイス内部状態を通知できる．

2.2 通信機構の設計

通信機構の処理を図 2 に示す．まず，プライマリは仮想 CPU を操作しゲスト OS を停止する．次に，バックアップへ移送のリクエストを発行する．バックアップ

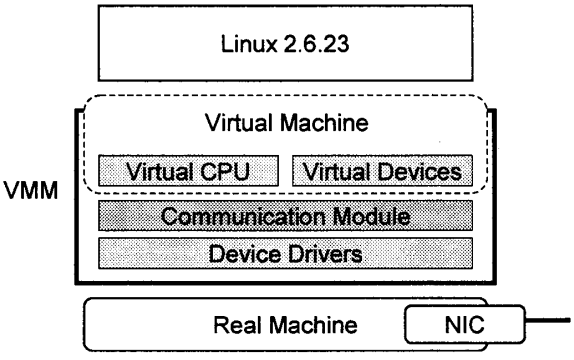

図 1: VMM の構成

*Implementation of a VMM for Fault-Tolerant Systems
†Shingo Aoyagi
‡Shuichi Oikawa
††Department of Computer Science, University of Tsukuba

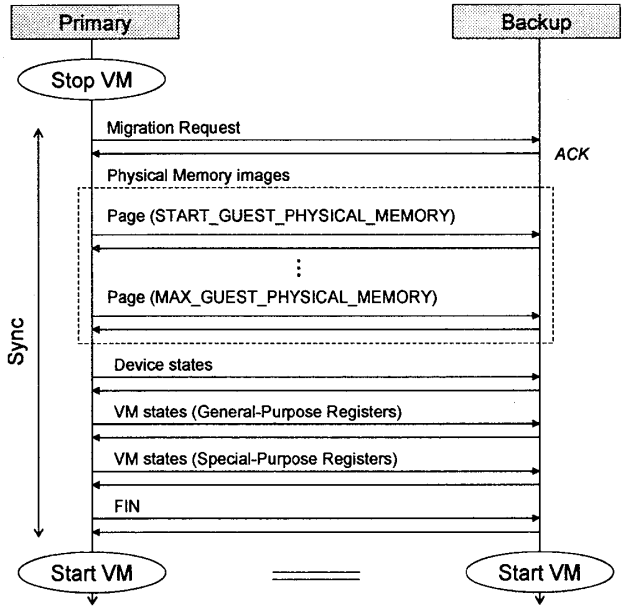

図 2: 仮想マシンの移送

がこのリクエストに対して応答を返すと，プライマリはバックアップへ物理メモリ状態，仮想 CPU 状態，仮想デバイス状態を転送する．最後にプライマリが移送完了を通知し，ゲスト OS の移送が完了する．

本システムの状態転送はプライマリからの状態通知とバックアップからの ACK によって構成される．各状態通知は通知内容を示すヘッダと内容により構成される．プライマリは状態通知に対応した ACK が一定時間以上到着しなかった場合，状態通知を再送する．またヘッダに状態の識別子を含めることにより，バックアップへ同じ状態通知が 2 回到着した場合も解決することができる．これにより，本システムはネットワーク上で発生するエラーも回避可能である．

2.3 実装

ゲスト OS の移送機能を持つ VMM の実装は C 言語 7000 行，アセンブラ 2000 行程度となった．また，ゲスト OS として Linux 2.6.23 を利用する．本環境は VT-x により，ゲスト OS 内の仮想化に必要な命令を実行中に検出するため，ゲスト OS に対しての変更は不要である．

3 結果

以上を実装しプライマリバックアップモデルの検証実験を行った．実験環境として Dell Precision 490 (Xeon 5130 2.0GHz, Memory 2GB, NIC Intel 82546GB Copper-Cable 1Gbps) を 2 台用意し，それぞれをプライマリ，バックアップとした．これらをミラーリングハブ Corega SSW08GTR によりイーサネット接続する．

以上の環境を構築し，プライマリバックアップモデルが実現できたことを確認した．プライマリは VMM によってゲスト OS を VM 上で実行し，特定のタイミングでバックアップへゲスト OS を移送する．移送完了後，プライマリとバックアップが同じ結果を返すことを確認した．

また，プライマリからバックアップへの Linux 移送時間は 31.6 秒となった．この移送時間の大半は物理メモリ状態の転送時間であり，31.5 秒であった．ACK を含めたプライマリバックアップ間の転送量は 151MB であることから，移送中の平均回線占有率は 3.8% (38Mbps/1Gbps) となった．回線占有率が低くなる主な原因としては，VM 状態を通知する処理のレイテンシが高いためであり，通信機構の改善することで移送時間の大幅な削減が期待できる．

4 まとめと今後の課題

本研究では，耐故障性向上に向けて，VMM を利用したプライマリバックアップモデルの設計と実装を行った．これにより，ゲスト OS Linux2.6.23 をプライマリ上で実行し，バックアップへ移送，それぞれの上で同じ結果が得られることを確認した．また，プライマリバックアップモデルの実現には仮想 CPU と仮想デバイス，デバイスドライバ，再送付き通信により実現可能であり，これを実装するためのコード量は 9000 行程度であった．また，冗長化機能を追加するソフトウェアに対しての変更は不要であるため，既存のソフトウェアシステムの冗長化は容易であり，耐故障性向上への利用が期待できる．

現在は，プライマリからバックアップへゲスト OS の状態を一括で転送するため，ゲスト OS 移送中にはプライマリゲスト OS が停止している．また，ゲスト OS の状態通知は毎回図 2 の流れを繰り返すため，冗長な通信が発生してしまう．そのため，今後は物理メモリやデバイス状態の差分転送によりゲスト OS の停止時間を削減する機構を設計する必要がある．また，耐故障性を向上させるためには，プライマリ-バックアップを外部環境に対して故障発生時に透過的に切り替える必要があり，今後はプライマリの故障検出や透過的な切り替え機構の設計を行う予定である．

参考文献

[1] Thomas C. Bressoud, Fred B. Schneider: “Hypervisor-based fault tolerance” , SOSP, 1995.

[2] John Scott Robin, Cynthia E. Irvine: “Analysis of the Intel Pentium ’ s Ability to Support a Secure Virtual Machine Monitor” , 9th USENIX Security Symposium, 2000

[3] Yaozu Dong, Shaofan Li, Asit Mallick, Jun Nakajima, Kun Tian, Xuefei Xu, Fred Yang, Wilfred Yu: “Extending Xen with Intel Virtualization Technology” , Intel Technology Journal, Volume 10, Issue 3, 2006