ハードウェア仮想化技術を用いた仮想計算機モニタの構築に向けて

尾 上 浩 一[†]　　大山恵弘[††]　　米 澤 明 憲[†]

1. はじめに

仮想計算機モニタは仮想計算環境を提供するためのひとつの手法である．仮想計算機モニタに関連する研究および開発が盛んに行われ，高性能で計算資源を多く持つサーバ用の計算機から個人用の計算機までの広い範囲において仮想計算機モニタが実用化されている．仮想計算機モニタは，仮想計算機が発行する特権命令などの仮想化および多重化に必要な操作を捕捉し適切な処理への変換を行う．ここで対象とするアーキテクチャは，現在の多くの計算機や仮想計算機モニタで利用されている x86 である．

x86 では，特権命令ではないが仮想計算環境を提供するためには捕捉すべき命令（機密命令）が存在する．たとえば，コントロールレジスタ CR3 の読み込み命令は機密命令である．CR3 の読み込み命令が捕捉されない場合，仮想計算環境で利用している CR3 の値ではなく，仮想計算機モニタが利用している CR3 の値が読み込まれてしまう．このため，機密命令へ対応することは x86 上で仮想環境を提供するひとつの必要条件となる．

仮想化を行うためのソフトウェアおよびハードウェア技術はこれまでいくつか提案されている．ソフトウェア技術のみを利用して仮想計算環境を提供する手法としては VMware [7] によるバイナリ変換や Xen [4] による準仮想化がある．しかしながら，ソフトウェア技術のみを利用して仮想計算機モニタを提供した場合（ソフトウェア仮想計算機モニタ），バイナリ変換に伴うオーバーヘッドや準仮想化による仮想計算環境上で動作する OS(ゲスト OS) の修正の必要性が生じる．他方，近年ではハードウェア (CPU, I/O) サポートによる仮想化技術が Intel や AMD から提案されている [3, 6]．ハードウェア仮想化技術を利用することにより仮想計算機モニタの実装が容易になる（ハードウェア仮想計算機モニタ）．仮想環境を提供するためにゲスト OS を修正する必要もなくなる．ソフトウェア仮想計算機モニタに対する，ハードウェア仮想計算機モニタの仮想化に伴う実行時オーバーヘッドの大小は，アプリケーションに依存する [1]．

ここでは我々が今回実装した仮想計算機モニタの設計および実装について述べる．我々は AMD のハードウェア仮想化技術 AMD-V を用いて仮想計算機モニタの設計，実装を行った．我々はシミュレータ SimNow [2] を利用して仮想計算機モニタを設計，実装し，簡単なプログラムの動作確認を行った．

2. AMD-V による仮想計算環境の提供

AMD-V では，仮想計算機モニタの実行（ホストモード）からその上で動作する仮想計算機の実行（ゲストモード）への切り換えを行うために VMRUN 命令を利用する．VMRUN 実行時の流れは次の通りである．最初に仮想計算機モニタの実行状態が保存される．保存領域は仮想計算機モニタの初期化時に用意する．仮想計算機モニタの実行状態の保存後，動作させたい仮想計算環境の実行状態を Virtual Machine Control Block (VMCB) に設定する．最後に設定された VMCB に基づいてゲストモードの実行を開始させる．

他方，ゲストモードからホストモードへの切り換えは VMEXIT 例外が発生したときに行われる．VMEXIT 例外が発生するのは，ゲストモード実行時に特権命令や機密命令の実行，例外および割り込みなどが発生したときである．仮想計算機モニタは VMEXIT の発生原因を VMCB から取得することができる．

仮想計算機モニタはホスト・ゲストモード間の切り換え時に 2 つの目的で VMCB を利用する．ひとつの目的は，VMRUN 実行時，VMEXIT 例外発生時にレジスタなどのゲストモードの実行状態を設定，通知するためである．もうひとつの目的は，仮想計算機モニタが捕捉すべきゲストモードの処理，すなわち VMEXIT を発生させるべき処理を設定するためである．たとえば，仮想計算機モニタがゲストモードにおけるコントロールレジスタ CR3 への書き込みを捕捉

† 東京大学
†† 電気通信大学

したい場合，VMCB 中の CR3 の書き込み用制御ビットをセットする．

AMD-V は仮想計算機モニタにおける MMU 操作を容易にするために，Nested Paging 機能を提供している．ただし，プロセッサが Nested Paging 機能を有していることが必要である．Nested Paging 機能は，ゲストモードにおける物理アドレスとホストモードにおける物理アドレスの間のアドレス変換をハードウェアによって実現している．Nested Paging を有効するためには，VMCB 中の Nested Paging 機能用の制御ビットをセットする．

3. 設計・実装

我々が設計，実装した仮想計算機モニタは，ハードウェアのすぐ上で動作する (Type-I VMM[5])．近年ではマルチコアを含めた複数の CPU を持つ計算機が普及してきているため，提案する仮想計算機モニタは SMP 計算機のサポートを考慮した設計にしている．複数の CPU を利用できることは仮想計算機モニタの高速化に役立つ．また，用途に応じて分類された仮想計算機を異なる CPU で実行することも可能になる．たとえば，ウィルス検査を行う仮想計算機をひとつの CPU で実行させ，他の仮想計算機を残りの CPU で実行させることにより，Performance Isolation を実現することが可能になる．

ゲストモードで特権命令の実行などにより VMEXIT 例外が発生したときには，仮想計算機モニタが適切な処理を行う必要がある．現在のプロトタイプシステムにおける VMEXIT 発生時の処理では，捕捉した実行命令をエミュレートし，インストラクションポイントを次の実行命令に設定する．その後，仮想計算機モニタが VMRUN を用いてゲストモードを再開させる．上記の VMEXIT 発生時の処理を実現するために次のような操作を行う．まず，エミュレートすべき命令の物理メモリ上の位置を検出する．その後実行命令の解釈を行い，次の実行命令のインストラクションポイントを決定する．最後に VMRUN を用いてゲストモードを再開させる．

さらに，現在のプロトタイプシステムでは，マルチコアを含む複数 CPU を起動し，各 CPU で APIC のタイマー割り込みを利用することが可能である．

4. まとめと今後の課題

ハードウェア仮想化技術 AMD-V を用いた仮想計算機モニタの設計，実装を行った．現在のプロトタイプシステムでは，マルチコアを含む複数の CPU を起動させ，APIC のタイマー割り込みが利用可能である．プロトタイプシステムは，仮想計算環境で動作するプログラムが特権命令を実行したときに，仮想計算機モニタがプログラムの実行の捕捉，捕捉時の実行命令のエミュレーション，実行の再開を行う．

今後は，仮想計算機モニタにスケジューリング機能を追加し，複数のプログラムを動作可能にしたい．ストレージ処理やネットワーク処理などに対応するために，入出力処理の設計および実装も必要である．そして，仮想計算環境で Linux などの既存の OS が動作するようにしていきたい．

参 考 文 献

1) K. Adams and O. Agesen. A comparison of software and hardware techniques for x86 virtualization. *SIGARCH Comput. Archit. News*, Vol. 34, No. 5, pp. 2–13, 2006.
2) AMD. AMD SimNow Simulator. `http://www.amd.com/simnow/`.
3) AMD. Introducing AMD Virtualization. `http://www.amd.com/virtualization/`.
4) P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, pp. 164–177, New York, December 2003.
5) Robert P. Goldberg. Survey of virtual machine research. *IEEE Computer*, Vol. 7, No. 6, pp. 34–45, June 1974.
6) Intel. Intel Virtualization Technology. `http://www.intel.com/technology/virtualization/`.
7) VMware. VMware. `http://www.vmware.com/`.