システムソフトウェアと
オペレーティング・システム 92－4
（２００３．２．２７）

複数 OS 環境における OS 切替え方式

桝本　圭†　　中島　雄作†　　伊藤　健一†
乃村　能成‡　　谷口　秀夫‡

計算機を用いたサービスを実現する上で，一つのプロセッサの上に二つの OS を同時に走行させる技術の有用性が注目されている．しかし，既存の方式には，主となる一つの OS の停止が，計算機全体のシステムダウンとなる問題が残る．そこで我々は，これらの問題を解決する構成法を提案し，検討を進めている．ここでは，この構成法の課題の一つである OS 切替え方式について述べる．提案する方式は，割込みを契機に CPU の使用権を切替える方式であり，切替える直前の OS の状態に依存しないことを特徴とする．また，一方の OS が管理するハードウェア資源を他方の OS と共有する方式を述べる．

A Method for Switching Multiple Real Machine Environments on a Single System

Kei MASUMOTO†, Yusaku NAKAJIMA†, Ken-ichi ITOH†,
Yoshinari NOMURA‡ and Hideo TANIGUCHI‡

Due to the dramatic growth of average PCs, virtual machine software for running multiple operating systems on a single system has become popular, today. Existing virtual machine softwares are built on the similar concept of the conventional VM, which has dependence of child operating systems on their mother VM. This fact causes some problems such as performance limitation and collective security. In other words, accidental stop of the mother operating system leads to the corruption of the whole family. We have been research on the new framework for solving these problems. As a key technology of the framework, this paper describes a method for switching operating system from one to another safely. Under the method, the trigger of switching is a hardware interrupt which has no dependence on any operating system. Also we propose a method for sharing hardware resource among the coexistent operating systems.

１．はじめに

パーソナルコンピュータ(以降，PC と略す）に代表される計算機の普及は目覚しく，様々なサービスで利用されている．そして，そのとき利用するオペレーティングシステム(以降，OS と略す）には，いくつかの選択肢がある．例えば文書処理や電子メールの送受信などの個人的な利用目的の場合には，GUI に優れた Windows が多く用いられ，Web サーバのようにトランザクション処理を行う場合には，Unix 系 OS が用いられることが考えられる．このように利用用途と各 OS が持つ特徴に合わせて，利用者は OS を選択する．このとき，通常は一つの計算機に対して一つの OS が選択されるが，一つの計算機上で，同時に複数の OS を利用可能であると，より多くの利点を持つことができる．

そこで，一つの計算機に複数のプロセッサ(以降，CPU と略す）を搭載して，複数の OS を走行させる技術がある．また，より汎用性を高めるために，一つの CPU 上で複数の OS を同時に走行させる技術がある．ここでは，一つの CPU 上で同時に走行可能である複数の OS を複数 OS と呼び，これを実現するための構成法を複数 OS 構成法と呼ぶ．

従来の複数 OS 構成法は，各 OS に主従関係が存在するため，主となる OS がシステムダウンした際に，他の OS もその影響を受ける問題がある．

そこで，我々は一つの CPU 上に存在する各 OS の独立性を高めたハードウェア非共有である複数 OS の構成法を提案した[1]．その構成法には，以下の三つの課題がある．

(1) OS 立ち上げ方式

†株式会社 NTT データ技術開発本部
Research and Development Headquarters, NTT DATA Co.
‡九州大学大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University

(2) OS 切替え方式
(3) 非共有としたハードウェア資源の共有方式
ここでは，項目(2)の OS 切替え方式，項目(3)の非共有ハードウェア資源の共有方式について述べる．

2．複数 OS の利点と従来の構成法

2.1 利点

一つの計算機上に複数の OS を走行させる際に得られる主な利点として，次の二つがある．

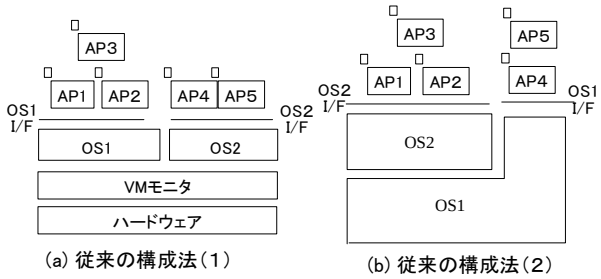
(1) 異なる OS の組み合わせによる機能補完
リアルタイム OS と，GUI に優れた OS の組み合わせを例に挙げる．リアルタイム OS は割込みの処理速度を追求した OS であり，GUI 機能に関しては貧弱である．一方，GUI に優れた OS は，割込み処理速度は遅いが，豊富な API により GUI 機能が充実している．両 OS を同時に走行させ，制御部をリアルタイム OS に，ユーザインタフェース部を GUI に優れた OS に分担させると，より高機能なシステムが低コストで実現することができる．このように異なる特徴を持つ OS を適切に組み合わせることで，OS 相互の機能が補完できる．

(2) 資源の分割による有効活用
計算機を複数の目的で使用する場合に，複数の OS に目的を振り分けることで，利点が得られる場合がある．例えば，二つの OS が一つの計算機で走行するとする．このとき一方の OS は，計算機の所有者が使用し，もう一方ではグリッド処理をするために貸し出す．近年の計算機の性能向上は目覚しく，一般の個人ユースとしての使用用途では，計算能力は余ることが多い．そこで計算能力を貸し出すが，一つの OS ではグリッド処理中に計算機所有者の本来の処理を妨害する恐れがある．そこで複数 OS を用いることにより，所有者本来の処理が妨害されることがなくなる．

2.2 従来の構成法

従来の構成法では，図１で示すように，複数 OS 間の調停を実現する方式が多い．文献[2]のように VM モニタ(主 OS)上に他の OS(従 OS)が走行する形や，文献[3]のように一方の OS の上に他の OS(従 OS)が走行する形をとる．これらの実現法には次の特徴がある．

(利点 1) 従 OS は，主 OS が全て制御している．そのため，主 OS 上で従 OS のデバグを制御できる．
(利点 2) 従 OS は，バイナリ互換で走行できる．そのため，様々な OS が適用しやすい．なお，バイナリ互換ではなく，何らかの改変を必要とする場合もある．
(欠点 1) 主 OS の停止は，計算機全体のシステムダウンとなる．
(欠点 2) 従 OS の機能は，主 OS が提供する機能に制約される．例えば主 OS で使用できないハードウェアは従 OS でも使用できない．
(欠点 3) 従 OS の性能が低い．主 OS を通じて，ハードウェアと通信を行うため，ハードウェアを占有した場合と同じ性能を得ることは難しい．

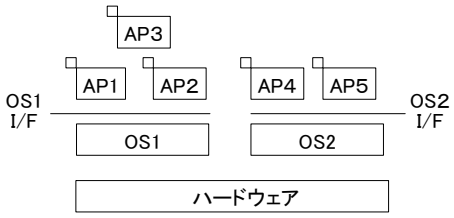


(a) 従来の構成法（1）　(b) 従来の構成法（2）

図1　従来の複数OS構成法

3．本構成法の特徴

２章で述べた従来の構成法の欠点の大きな原因の一つは，各 OS 間に存在する主従関係だといえる．そこで我々は，図 2 に示す各 OS の独立性を高めた，ハードウェア非共有とする複数 OS の構成法を提案した[1]．この構成法の特徴を以下に述べる．



AP：アプリケーションプログラム

図2.　提案方式による複数OS構成法

(特徴 1) 各 OS を独立に動作させる．これにより他の OS のシステムダウンによって影響を受けることがなく，主 OS を通じて従 OS が動くことがないため，従 OS の性能が大きく低下する問題を軽減できる．
(特徴 2) ハードウェアは基本的に非共有とする．主 OS が提供可能な機能しか従 OS が持てないとい

う問題を解決する．

(特徴3) 一方のOSには汎用OSを適用することを可能にするため，一方のOSに対する変更は極力少ないものとする．

そして，この構成法を実現するための技術的課題として，以下の三つがある．

(1) 起動方式

主従関係が存在する従来の構成法では，従OSは主OSが起動した後に，主OSのアプリケーションとして立ち上げることが考えられる．だが本構成法では，VMモニタのような主OSが存在せず，また独立性を高めるため，各OSが使用するメモリも独立となる．そこで各OSの起動方式を検討することが必要である．

(2) OS切替え方式

CPU資源は一つのため，ある時間においては一方のOSのみがCPUを使用する時分割方式とならざるを得ない．OS切替えのタイミング(リソース配分)の観点と，実際の切替え方式の検討が必要である．

(3) 非共有としたハードウェア資源の共有方法

本構成法では，各OSの独立性を高め，各OSの性能を維持するためにハードウェアを非共有としている．一方，他のOSが管理する資源をどうしても使いたいという要望がある．例えばディスプレイへの入出力機能となるビデオカードは各OS間でも共有させたいように，ハードウェアをOS間で共有したい場合も考えられる．このような場合でも，各OSは，OS間の共有部分を持たないので，制御方法の検討が必要である．

上記三つの課題のうち，ここでは関連性のある(2)(3)について述べる．

また，これらの方式を述べるにあたり，二つのOSをOS1，OS2と呼ぶ．また，想定環境として，CPUにはIntel社のPentiumを用いることとし，各OSは仮想記憶を利用するものとする．

4．OS切替え方式

本章では，本構成法において，二つのOSを切替える方式について述べる．

4.1 基本方式

複数OSを実現する上で，OSを切替える際に考慮すべき事項は，処理内容と，その処理が開始する契機である．以降，処理内容について以下(1)，(2)で述べ，処理が開始する契機について(3)で述べる．

(1) 走行していたOS環境の保存と復元

OSを切替える処理には，走行していたOSの環境を保存する処理と，これから走行するOSの，OS切替えにより停止したときの環境を復元する処理が必要である．また，切替える際に，各OSや各OSの上で走行するアプリケーションに影響がないようにしなければならない．

これと類似する目的を持つ処理として，OS上で走行するプロセスを切替えることがあるが，OSの場合も，同様な処理で切替えることができる．OSを切替える際に保存／復元が必要である事項は，次の四点となる．

(a) セグメントディスクリプタテーブル
(b) 割込みディスクリプタテーブル
(c) ページング機構
(d) プロセス切替えに必要な事項

本構成法において，各OSは独立に動作するため，独立な仮想空間のセットを持つ．そこで，(a)，(c)の切替えが必要である．また，ハードウェアも非共有としたため，割込みの種類が各OSで異なる部分も生じる．しかし，ハードウェアからの割込みは一つのCPUに対して行われるため，OSを切替える際には，各OSが各OS用の割込みディスクリプタテーブル(IDT)を用いて割込みを処理する必要があるため，(b)の切替えも必要となる．さらに，(a)～(c)のように通常初期化時に設定するのみという情報だけでなく，走行中に動的に切替わる走行状態もあり，それらを保存する必要があるため，(d)も必要である．

(a)～(d)に該当する具体的内容は，以下(A)～(G)となる．(a)，(b)には，(F)のGDTR，IDTRが該当し，(c)には(E)のうち，CR3レジスタが該当する．また(d)に該当する事項は，(E)のCR3レジスタとその他の(A)～(G)全てである．

(A) 汎用レジスタ
(eax,ebx,ecx,edx,edi,esi,ebp,ess,esp)
(B) セグメントレジスタ(cs,ds,es,fs,gs,ss)
(C) EFLAGSレジスタ
(D) EIPレジスタ
(E) コントロールレジスタ（CR0～CR4）
(F) メモリ管理レジスタ（GDTR,IDTR,LDTR,タスクレジスタ）
(G) デバッグレジスタ（DR0～DR7）

(2) 仮想アドレスの一致

(1)に示す値の保存と復元を行い，OS を切替えたとき，CPU が次に実行しようとする命令の仮想アドレスの値は，切替え前後で同じアドレスである．しかし，各 OS は任意の動作をするため，切替え後走行する OS が，切替え前の OS が使用していた仮想アドレスを用いたとき，正常に走行可能とならない場合が生じる．

そこで，切替え前に走行する OS 切替え処理と，切替え後走行する OS が行う処理の仮想アドレスについて，連続性が要求される．このため，本方式では，仮想アドレスの配置を調節し，連続性を持たせる．

(3) OS 切替え処理が開始する契機

走行中の OS では，OS 自体の処理や，アプリケーションの処理が行われるなど，様々な処理が行われている．これらの処理に対して，OS 切替えを行うことによる影響が発生しないように考慮する必要がある．

本方式では，割込みを契機として OS 環境の保存，復元処理を行い，OS を切替える．割込み発生時には，OS やアプリケーションの状態を保存するため，割込みを利用することで，走行していた OS や，OS の上で走行するアプリケーションの状態に依存せず，切替えることが可能となる．

4.2 処理の流れ

4.1 節の(1)～(3)を考慮した OS 切替え方式の流れとして，図 3 に OS1 または OS2 走行時に，OS2 が処理する割込みが発生した場合，図 4 に OS1 または OS2 走行時に OS1 が処理する割込みが発生した場合を示す．図 4 に関する説明は図 3 と同様のため，ここでは省略する．なお，切替えの際に必要な OS やアプリケーションの環境保存処理は，OS2 が行う．これにより，OS1 の変更を不要とすることができる．

図 3 の流れについて述べる．

(a) OS1 走行時に OS2 が処理する i 番目の割込み が発生した場合

(a-1) OS1 が所持する IDT の i 番目のエントリを参照し，割込み処理ルーチンのアドレスを得る．

(a-2) OS1 上の i 番目の割込み処理ルーチンを実行する．この処理ルーチンは，OS 切替え処理のみ行う．

(a-3) OS 切替え処理により，OS2 に切替える．OS2 は OS1 の環境保存及び OS2 の環境復元を行う．

(a-4) OS2 の i 番目の割込み処理ルーチンにジャンプする．

(a-5) OS2 が i 番目の割込み処理ルーチンを実行し，処理終了後は OS2 のユーザモードに戻る．

(b) OS2 走行時に OS2 が処理する i 番目の割込み が発生した場合

(b-1) OS2 が所持する i 番目の IDT のエントリを参照し，割込み処理ルーチンのアドレスを得る．

(b-2) OS2 が i 番目の割込み処理ルーチンを実行し，処理終了後は OS2 のユーザモードに戻る．

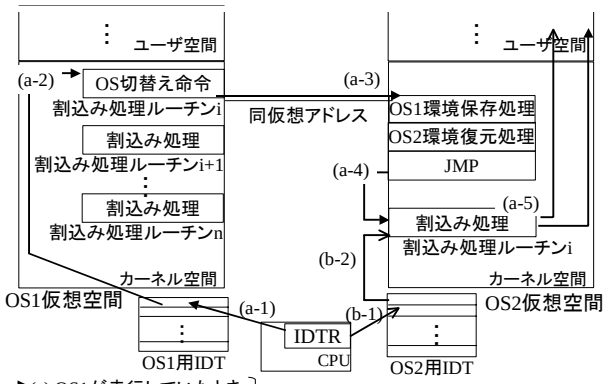


図3 OS切替えの流れ（OS2への割込み発生時）

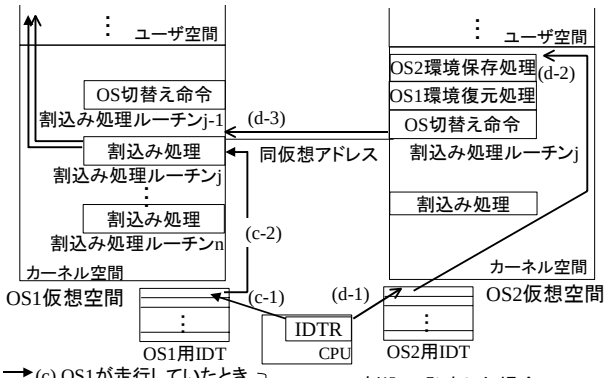


図4 OS切替え処理の流れ(OS1への割込み発生時)

5．非共有ハードウェアの資源共有方式

ここでは，一方の OS が管理するハードウェア資源を，他方の OS が利用する方式を述べる．具体的には，NIC を例として資源共有方式を述べる．

5.1 想定環境

ここで述べる資源共有方式は，OS から見た場合，Bus Master DMA 方式を用いた PCI タイプの NIC インタフェースを提供する．この条件を満たす NIC は現

在主流なものであり，このDMA方式を採用しているハードウェアは多い．従って，本方式は様々なハードウェアに対して応用可能である．

想定するNICは，一般に図5に示す基本動作を行う．図5の(1)～(4)について述べる．

(1) 送受信バッファ

NICドライバ上でパケットを格納する領域．

(2) 送受信FIFO

NIC上で送受信するパケットを格納する領域．

(3) レジスタ，RAM，EEPROM

NICの設定情報を保存する領域で，NICドライバによって読み込み／書き込みされる．またNICドライバがNICに対して何らかの処理を行わせる場合にも，この領域に書き込みを行う．

(4) 送受信エンジン

送受信バッファと送受信FIFO間でパケットのコピーをする部分．送受信FIFOとネットワーク間でパケットデータの通信も行う．

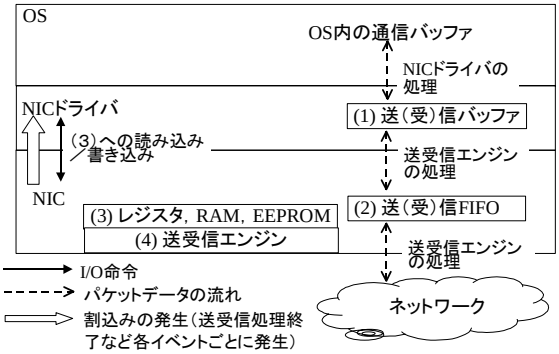


図5 Bus Master DMA方式を用いたNICの基本動作

5.2 NICの基本動作

図5に示すBus Master DMA方式を用いたNICの基本動作とは，NICドライバが，NIC上のレジスタに対してI/O命令を発行し，NICがレジスタ，RAM，EEPROMを用いて様々な動作を行う形式である．OSは，これらのレジスタをOSが使用するI/O空間にマッピングすることにより，これらのレジスタへのI/Oを可能とする．I/O命令を受けることでNICの基本動作が始まるので，基本動作をI/O命令の種類という観点から把握することが重要である．各I/O命令は，I/Oの対象となるレジスタによって分類が可能であり，分類は，レジスタの機能とレジスタがマッピングされるI/O空間を考えればよい．表1にI/O命令を分類する．

表1では，まずレジスタがマッピングされる領域について，I/O命令を(処理A)，(処理B)に分類した．レジスタがマッピングされるOS上のI/O空間には二種類あり，PCIデバイスが共通で保持するレジスタをマッピングするPCI設定空間と呼ばれる領域と，NIC固有のレジスタをマッピングするI/O空間と呼ばれる領域がある．これら二つは，PCI設定空間に属するレジスタを用いて，I/O空間に属するレジスタ群をマッピングする場所のベースレジスタを決定する関係にある．

分類は，(処理A)がPCI設定空間に属するレジスタ，(処理B)がI/O空間に属するレジスタとなっている．(処理B)については，レジスタの機能という観点から細分化が可能であるため，(処理B-1)～(処理B-5)までの五つに分類している．

表1 レジスタの種類と，I/O命令の種類の関係

I/Oの種類	I/Oを行うレジスタの種類	I/Oが行われるレジスタの例
(処理A) PCIハードウェア制御に関する処理	PCI設定空間	I/O空間に属するレジスタ群のベースアドレス設定，設定を有効にする命令，PCIバス上でのエラー表示，割り込みラインの設定，ハードウェア識別情報の表示
(処理B) NIC固有の処理	I/O空間	(B-1)～(B-5)で分類する
(処理B-1) 送受信処理	I/O空間	送受信開始命令 送受信バッファアドレス通知
(処理B-2) 割込み処理	I/O空間	割込み種類特定 NICが発生させる割込みの限定
(処理B-3) 統計，NICの状態診断，エラー処理	I/O空間	計測データ種類設定 計測データ読み出し
(処理B-4) EEPROMに対してI/Oを実行する処理	I/O空間	MACアドレスの設定
(処理B-5) その他	I/O空間	タイマ機能設定，電源管理設定，送信モード(全(半)二重)設定，設定情報読み出し

表1の中でNICの主要な動作として，(処理B-1)の送受信処理がある．図6に送信処理の流れ，図7に受信処理の流れを示す．

以下，図6に示す送信処理の流れを述べる．

(a) NICドライバの処理

(a-1) OS内の通信バッファからパケットをコピーする．

(a-2) NICのレジスタに送信バッファのアドレスを格納する．

(a-3) 送信開始命令を発行する．

(b) NIC内の送受信エンジンの処理

(b-1) 送信バッファから送信FIFOにパケットをコピーする．

(b-2)送信FIFOから，パケットをネットワークに送信する．

以下，図7に示す受信処理の流れを述べる．

(a) NICドライバの処理

NICドライバ初期化時に，NICのレジスタにに受

信バッファのアドレスを格納する．受信時は，受信バッファに空きがあると，NIC が任意のタイミングで受信 FIFO から受信バッファにパケットをコピーする．

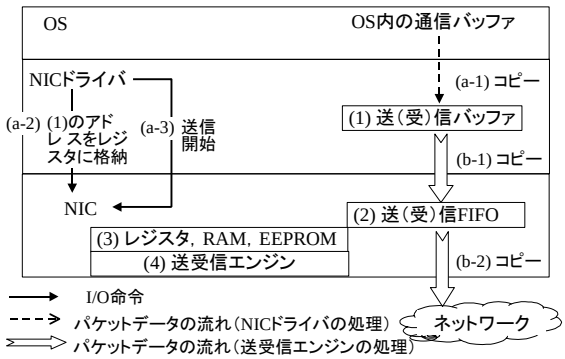


図6 Bus Master DMA方式を用いた NICの送信処理

(b) 送受信エンジンの処理
(b-1)ネットワークからパケットを受信する．
(b-2)受信バッファにパケットをコピーする．
(b-3)受信完了の割込みを発生させる．
(c) NIC ドライバの処理
(c-1) (b-3)による割込みが入ると，割込みに対する応答を返す．
(c-2) OS 内通信バッファにパケットをコピーして，その後，受信バッファを空にする．

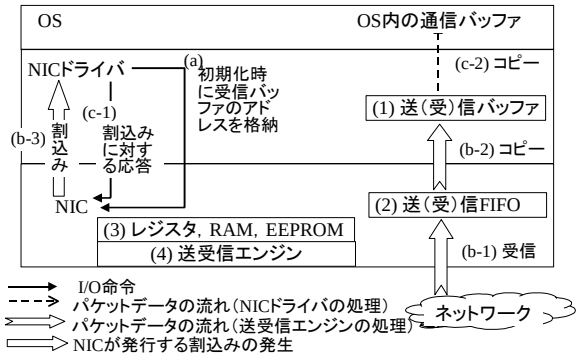


図7 Bus Master DMA方式を用いた NICの受信処理

5.3 基本方式

本方式は，NIC を管理する OS(OS2)が，他の OS(OS1)に NIC インタフェースを擬似提供することで NIC の資源を提供する．NIC の資源とは，表 1 で示した動作を行うための機能である．ここでは，このインタフェースを擬似NIC と呼ぶ．

実際の通信において，OS1 は OS2 が提供する擬似 NIC に OS 切替えにより処理を依頼し，擬似 NIC が OS1 に OS 切替えにより実行結果を返却する機能を有する．図 8 に基本方式を示す．

OS2 は，図 8 に示すように，OS1 に対して，NIC の資源として擬似 NIC を提供する．このとき，NIC は OS2 が所有するため，OS1 も OS2 の NIC を使用するポリシー（通信ポリシー）に従う必要がある．このため，通信パケットや I/O 命令は OS2 を経由する形式として，OS1 への割込みも擬似 NIC から発生させる形式としている．OS1 から NIC に対して発行される I/O 命令とパケットの流れは以下の通りになる．なお，OS2 が送受信するパケットの流れは図 6，図 7 と同様のため省略する．

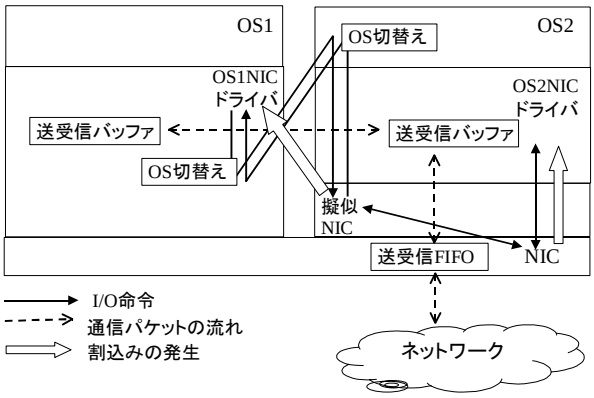


図8 基本方式

(1) OS1NIC ドライバから NIC への I/O 命令の流れ
OS1 ⇔ OS1NIC ドライバ ⇔ (OS 切替え) ⇔OS2 ⇔ OS2NIC ドライバ⇔ 擬似 NIC ⇔ NIC
(2) OS1 が送受信するパケットの流れ
OS1 送受信バッファ ⇔ OS2 送受信バッファ ⇔ 送受信 FIFO ⇔ ネットワーク
(3) OS2NIC ドライバから NIC への命令の流れ
OS2NIC ドライバ ⇔ 擬似 NIC ⇔ NIC

(3)について，OS2NIC ドライバから発行される命令が擬似 NIC を経由するのみとなるため，直接 NIC に命令を発行することと同様である．

図 8 に示すように，本方式を実現するために必要な処理は，二つの OS 間の通信を実現する OS 切替え処理と，資源共有を実現する擬似 NIC である．このうち，OS 切替え処理は 4 章で述べた．そこで，ここでは擬似 NIC のみ考えることとし，簡単のために OS を一つとして方式を述べる．OS が一つであるとき，OS 切替え処理，OS2，OS2NIC ドライバは考えなくともよい．すなわち，考える基本方式は図 9 となる．図 9 に関する説明は，図 8 の説明と同様のため省略する．

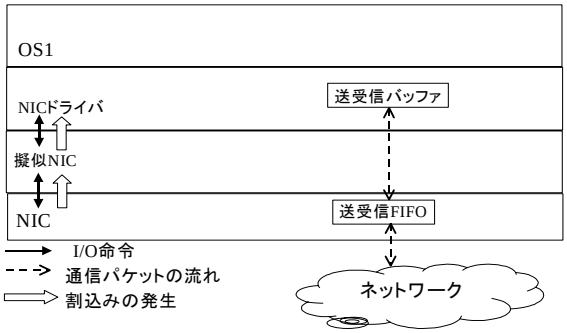


図9 基本方式 (OSが一つの場合)

5.4 擬似 NIC の設計

擬似 NIC は，NIC ドライバから I/O 命令を受け，何らかの処理を行う部分である．そして本方式を実現するためには，擬似 NIC の処理内容を規定することが必要である．そこで本節では，設計として，擬似 NIC が取り得る動作の規定を行う．その際に，以下二点を基本方針としている．

一点目に，実際の NIC は OS2 の資源であり，OS1 の擬似 NIC に対して行う操作の全てを実際の NIC に伝えてしまうと，OS2 の通信処理を妨害してしまうことが想定される．例えば NIC は，受信 FIFO から受信バッファに対して，受信バッファに空きがある場合，パケットを任意のタイミングでコピーする．このとき，NIC 上の受信バッファのアドレスを格納するレジスタの値を変更してしまうと，OS2 がパケットを受信できなくなるという問題が発生する．擬似 NIC を検討する際に，OS1 からの処理により OS2 の通信処理に影響が出ることを避けたい．そこで，OS1 による NIC 操作要求により OS2 の通信処理に影響が出ると考えられる場合には，直接 NIC のレジスタや，RAM，EEPROM などを操作せず，擬似 NIC 用にこれらのイメージ（以降，仮想レジスタと呼ぶ）を OS2 側に別途用意し，命令内容を格納する．そして，処理を行うタイミングを OS2 の通信処理の中で計りながら実行するという形式としている．

二点目に，処理を依頼する際には，通常一つの NIC ドライバが NIC を使用する場合より，明らかにパケット処理速度が低下する．そこで，命令の内容が擬似 NIC と OS1 の間で処理可能であれば，OS1 と擬似 NIC 間で処理する形式としている．

上記二点と図 9 を考慮して，OS1 に提供される擬似 NIC の動作を次の三種類に分類する．

(分類 1) 命令を実行するもの．OS2 の管理する NIC の状態を変更しないため，NIC に処理を依頼し，結果を返す．このとき，以降 OS1 から(分類 2-a)に該当する命令が発行されることや，ある命令の結果を期待して発行される命令もある．そのため，擬似 NIC 上の仮想レジスタに対しても命令を実行する．

(分類 2) 制限付きでその命令を実行するもの．次の三種類がある．

(分類 2-a) NIC に処理を依頼せず，擬似 NIC が，擬似 NIC 上の仮想レジスタに対して命令を実行するもの．OS2 の管理する NIC の状態を変更してしまうため，NIC に処理を依頼することはできないが，擬似 NIC 上のみで処理可能であるものが該当する．また，処理の高速化を図る目的で，擬似 NIC 上で処理させたいものが該当する．

(分類 2-b) NIC に処理を依頼するが，命令の内容や，NIC からの戻り値を加工した上で命令を依頼するもの．このとき(分類 1)と同様に，加工した命令を擬似 NIC の仮想レジスタに対して実行する，または加工した戻り値を擬似 NIC 内で保存する．OS1 が擬似 NIC に対して行う設定と，OS2 が NIC に対して行う設定が異なる場合に，何らかの影響が出る命令が該当する．

(分類 2-c) NIC に処理を依頼せず，擬似 NIC 上の仮想レジスタに対して命令を実行するもの．ただし，命令を加工した後実行する，または命令の戻り値を加工して返す．(分類 2-a)に該当する命令で，仮想レジスタに対して命令を実行する際に，特別な処理を必要とするものが該当する．

(分類 3) 命令の実行を依頼せず，擬似 NIC 上でも処理しないもの．このとき，書き込みを行う命令であれば，NIC，擬似 NIC 両方に書き込むことを行わず，読み込みを行う命令であれば，擬似 NIC が常にエラーとなる値を返すもの．NIC の状態を変更してしまい，かつ擬似 NIC 上でも処理不可能である命令が該当する．すなわち，OS1 には提供できない NIC の機能を実現するために必要な命令である．電源管理など，ハードウェアの機能を OS2 が常に占有している必要があり，擬似 NIC 上でもその機能を実現できない場合が一例として挙げられる．

5.5 擬似 NIC の動作例

擬似 NIC は，OS1NIC ドライバが発行した I/O 命令の種類に基づき，適切な処理を行う必要がある．そのためには，先述した(処理 A)～(処理 B-5)で示す I/O 命令の種類に，(分類 1)～(分類 3)で示す擬似 NIC が行う処理の分類を適用すればよい.表 2 に

適用結果を示す．また，表 1 で示した各処理の例をもとに理由を述べる．

(処理 A)について，ハードウェアの識別情報の表示に関しては，OS2 の通信処理を妨害しないため，(分類 1)とする．また，レジスタを I/O 空間へマッピングする命令に関しては，OS2 が行った設定を再設定することになるため，OS2 の通信処理を妨害する．そのため，NIC に処理依頼はできないが，擬似 NIC が設定内容を保存しておけば，以降 OS1 が NIC に対して通信をするための I/O ポートを把握することが可能になり，擬似 NIC 上で同機能を実現できる．従って，(分類 2-a)とする．その他の例については，OS2 が行った設定を再設定することになるため，NIC に処理依頼はできない．さらに，例えば割込みラインの設定による効果は擬似 NIC では実現困難である．従って，(分類 3)とする．

(処理 B-1)について，OS1 の送受信パケットは，擬似 NIC が一度受け取り，OS2 の通信処理の妨害をしないようにネットワークと送受信する形式としている．擬似 NIC とネットワークに関しては，OS2 の送受信処理を妨害しないタイミングで行われるため，ここでは考えなくともよく，OS1 と擬似 NIC 間の通信処理のみ考えることで十分である．OS1 と擬似 NIC 間の処理であるため，(分類 2-a)とする．

(処理 B-2)について，図 8，9 に示すように，OS1 に関する割込み処理は，擬似 NIC から発生させる形式としている．そのため，擬似 NIC と OS1 との処理になる．従って，(分類 2-a)とする．

(処理 B-3)について，計測データ種類設定は，OS2NIC ドライバが行う設定を変更し，OS2 の通信を妨害する．しかし，設定情報を擬似 NIC 上で保存することで対処可能であるため，(分類 2-a)とする．計測データ読み出しについては， OS2 の通信を妨害しないため，(分類 1)としする．しかし，例えば OS1 で設定した事項が OS2 では設定していない場合がある．その場合には，擬似 NIC 上の設定情報と比較し，エラー値を返す．そのため，(分類 2-b)も該当する．なお，エラー処理に関する計測データ種類設定，計測データ読み出しについて，OS2 が NIC を管理するため，エラー処理は OS2 側で処理されるものであるとしてよい．従って，ここでは考慮しないものとする．

(処理 B-4)について，MAC アドレス設定は，擬似 NIC 上で設定情報を保存しておき，通信時に擬似 NIC 上でパケットを書き換えることで処理可能である．従って，(分類 2-a)とする．

(処理 B-5)について，タイマに関する機能は，OS2 のタイマ機能で代用できる．従って，(分類 2-a)とする．しかし，タイマ以外の例は，OS2 が常に占有していなければならない命令であり，擬似 NIC 上では実現困難である．従って，(分類 3)として，この命令よる効果は OS1 に提供しないものとする．

表2 擬似NICの動作例

I/O命令の種類の分類	擬似NICの動作分類
(処理A) PCIハードウェア制御に関する命令	(分類1) NICに処理依頼 (分類2-a) 擬似NIC内で処理 (分類3) NICに処理依頼せず，擬似NICでも処理しない．
(処理B-1) 送受信処理	(分類2-a) 擬似NIC内で処理
(処理B-2) 割込み処理	(分類2-a) 擬似NIC内で処理
(処理B-3) 統計，NICの状態診断，エラー処理	(分類1) NICに処理依頼 (分類2-a)擬似NIC内で処理 (分類2-b) 擬似NICはエラー値を返す
(処理B-4) EEPROMに対してI/Oを実行する処理	(分類2-a) 擬似NIC内で処理
(処理B-5) その他	(分類2-a) 擬似NIC内で処理 (分類3) NICに処理依頼せず，擬似NICでも処理しない．

6．おわりに

複数 OS 構成法 [1] の課題である OS 切替え方式について述べ，また非共有ハードウェアの資源共有方式を NIC を例として述べた．OS 切替え方式は，割込みを契機として，切替えの際に考慮するべきアプリケーションなどの状態に依存せず，OS を切替えることを特徴とする．また，非共有ハードウェアの資源共有方式について，NIC を直接制御する OS が，NIC の擬似インタフェースを提供し，他の OS は OS 切替え方式を利用して，擬似インタフェースを使用する方式を示した．

今後は，各方式の詳細検討を進め，実装及び評価を行う予定である．

参考文献

[1] 谷口秀夫,乃村能成,田中一男,大塚作一,井上友二,“ハードウェアを非共有する複数オペレーティングシステムの構成法,”情報処理学会研究報告,Vol.2002,No.79,pp.47-54,2002.

[2] 新井利明，関口知紀，佐藤雅英，井上太郎，中村智明，岩尾秀樹，“ナノカーネル方式による異種 OS 共存技術「DARMA」の提案,”情報処理学会，第 59 回全国大会，講演論文集(1)，pp.139-140，1999.

[3] VMWare 社ホームページ，http://www.vmware.com/，最終アクセス日 2003 年 1 月 21 日．