2003－OS－94　(1)
2003／8／4

複数実計算機における OS 切替え方式の実装

田渕正樹† 桝本圭† 伊藤健一†
乃村能成†† 谷口秀夫‡

一つの CPU を持つ計算機の上で,複数の OS が同時に走行する複数実計算機を試作している.複数実計算機の特徴は,一つの OS の停止が計算機全体の停止となる問題を解決し,他方の OS の動作に何ら影響を与えない点である.ここでは,各 OS の起動時に一つの計算機上で二つの OS が走行できる環境を構成する起動方式,および割込みを契機に CPU の使用権を切替える OS 切替え方式について,実現方式を述べる.

Implementation of Switching Multiple Real Machine Environments on a Single System

Masaki TABUCHI† Kei MASUMOTO† Ken-ichi ITOH†
Yoshinari NOMURA†† and Hideo TANIGUCHI‡

We have been developing and implementing a prototype for running multiple operating systems on a single computer. This method improves the virtual machine method's problem that has dependence of child operating systems on their mother VM. This paper describes how to boot up multiple operating systems and how to switch operating systems from one to another. Under this method, physical memory is exclusively divided and reserved for each operating system from beginning and the trigger of switching is hardware interrupts which have no dependence on any operating system.

1.はじめに

一つのパーソナルコンピュータ（以降,PC）には必ずオペレーティングシステム（以降,OS）が存在する.近年,様々な種類の OS が利用できるがそれぞれの OS に特徴があり,提供する機能も異なる.これら複数の OS を同時に走行させ,各 OS が持つ機能を合わせて利用できる環境を構築する技術が注目を集めている.このような,複数の OS が共存する環境においては,一つの OS 環境における障害が他の OS 環境に影響を及ぼすことを避けるために,各 OS の独立性を高めることが重要である.さらに,複数 OS 共存環境で周辺装置を利用する際,単独 OS 環境時と変わらない機能や性能を享受したい.

そこで,我々はハードウェア資源を分割し,各 OS に占有制御させることにより,複数の OS を独立に,かつハードウェア性能を十分に活かして走行させる複数実計算機（MRM:Multiple Real Machines）[1]を提案した.

MRM は以下の方針に基づいて構成する.

(1)OS 相互の影響を排除するため,OS 間での資源の共有は行わない.
(2)市販 OS を利用可能にするため,共存させる OS のソースコードの改造を行わない.
(3)各 OS の性能の低下を避けるため,エミュレートによる資源の共有を行わない.
(4)外部割込みの応答性を高めるため,可能な限り速く,該当する割込み処理ルーチンを実行可能とする.

†株式会社 NTT データ
†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
‡岡山大学工学部
‡Faculty of Engineering, Okayama University

上述の方針を満たすため,MRM では,各ハードウェア資源をそれぞれ一つの OS に占有させることで各 OS の独立性を確保する.ただし,メモリや CPU およびタイマなどの共有せざるを得ないハードウェア資源については,一つの資源を分割利用せざるを得ない.このため,メモリは,各 OS 毎に領域分割して利用する.また,CPU は時分割で各 OS に割り当て,タイマはタイマ割込みを各 OS に適切に供給する.一方,周辺装置は各 OS に分割占有させる.これを時分割分割に対し,空間分割と呼ぶことにする.

資源を時分割および空間分割するためには,計算機の起動方式,OS 切替え方式,および終了方式を確立する必要がある.ここでは,前者の二つの方式について,その実現方式および実装内容を述べる.

2.実現方式

2.1 起動方式

2.1.1 メモリ分割

MRM では,各 OS が互いに影響を及ぼし合わないようハードウェア資源を分割占有する.OS は起動時の処理においてメモリや周辺機器の初期化を行うため,起動時にハードウェアを分割占有する環境を構築する.メモリに関して,多くの OS ではカーネルを実メモリの下位アドレス部分に読込み動作させる.そこで,共存する OS を改造しないために,共存する OS 用に下位アドレス部分を使用させる.つまり,制御する OS は実メモリ領域の上位アドレス部分を使用する.

一つの計算機上に二つの OS が共存している様子を図１に示す.

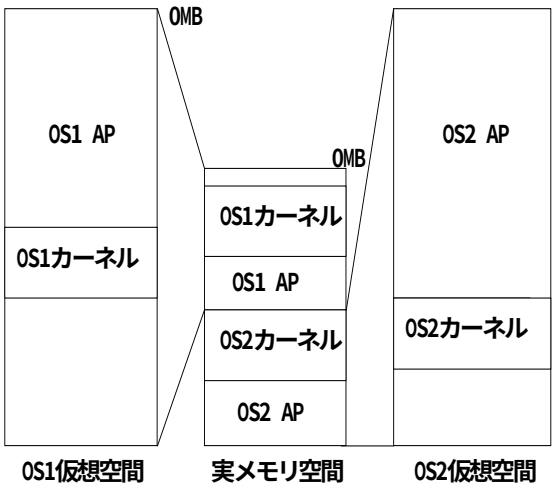

図１　メモリ分割の様子

各 OS は実メモリ領域を分割占有し,仮想空間は各 OS が占有している実メモリ領域に対応する空間を利用することで互いの OS は影響を及ぼさない.OS2 は,実メモリ領域の上位アドレス部分を使用し,最初に起動する OS である.OS1 は,下位アドレス部分を使用し,後に起動される OS である.

2.1.2 共存までの処理と課題

二つの OS が共存する環境を構築するまでの処理は,電源投入,BIOS 処理,OS2 起動,OS1 読込み,OS1 起動,OS1 と OS2 の共存（OS 切替え開始）の流れになる.この時,以下の課題がある.

(1)OS2 の使用可能実メモリ領域の制限

実メモリには二つの OS が読み込まれることになるため,互いのカーネルが上書きされないようにする必要がある.そのため,OS2 起動時に,OS1 が次に起動することを想定し,OS2 が使用できる実メモリ領域を制限し,OS1 カーネルと重ならないようにすることが必要である.

(2)OS1 の読込み

同時に二つの OS カーネルを実メモリに読み込めないので,OS2 が起動した後に OS1 を実メモリに読み込む必要がある.この際の,OS2 が OS1 読込みを開始する契機と OS1 読込み方法を確立する必要がある.

(3)OS2 環境保護のための監視処理

OS1 は,OS2 が動作していることを認識せずに初期化処理を行う.このため,OS1 初期化処理を OS2 が監視することで,OS2 の環境を破壊しないようにする必要がある.

2.2 OS 切替え方式

2.2.1 割込みによる OS 切替え

MRM では,割込みを契機に,各 OS が使用するメモリ領域およびレジスタ情報を切替えることにより OS 切替えを実現する.割込み発生時には,OS の上で走行する AP の走行を停止した状態で実行されるので,AP の走行状態に依存せずに切替えることが可能となる.

2.2.2 OS 切替えの流れと課題

MRM では,OS 切替えの前後でプログラムカウンタの値が連続するように割込み処理ルーチンを配置することにより,OS 間で共有部分を必要とせず,CPU 上のレジスタ値,ページテーブル,割込み管理テーブルおよびセグメントテーブルを切替えるだけで OS 切替えを実現している[2].

一般に割込みが発生すると,以下の処理が実行

される.

(A)CPU は自動的にフラグレジスタ,コードセグメントレジスタ(cs),プログラムカウンタ(eip)の値をスタックに積む.
(B)発生した割込み番号,および走行環境（他のレジスタ値）をスタックに積み該当する割込み処理を実行する.
(C)割込み処理が終了すると,スタックに積んでいた値を復元し,割込み発生時に実行していた処理を再開する.

上述の流れにおいて,OS 切替え処理は(A)の直後で行う.これは,CPU が自動的にスタックに積んだ直後に行うことで,OS の割込み処理内容に依存しないと共に,プログラムカウンタを合わせる際に必要な仮想アドレスが割込み管理テーブルより容易に取得できるためである.

次に,OS 切替え処理の流れを説明する.ここで,発生するハードウェア割込みは以下のように区別できる.

(1)I/O 機器などのハードウェア割込み
(2)タイマ割込み

以下,各割込みが発生した際の OS 切替えの流れを説明する.

(1)I/O 機器などのハードウェア割込み

OS 切替えは各 OS の走行環境,つまり CPU 上のレジスタ値を切替えることで実現する.ここで,共存する OS には改造を加えないとしたので,双方の OS 走行環境は OS2 側で保存,復元する.

(A)OS2 の AP 走行中に,OS1 の占有する周辺装置の割込み A が発生すると,OS2 の専有外割込みとして認識される.
(B)OS2 走行環境を保存し,前回保存されていた OS1 走行環境を復元する.
(C)OS2 が OS 切替え処理を実行,OS1 に切替える.
(D)OS1 では,当該周辺装置の割込み処理を実行した後,割込みから復帰しユーザモードに戻る.
(E)OS1 の AP 実行中に,OS2 の管理する周辺装置の割込み B が発生し,OS1 の専有外割込み処理として認識される.
(F)OS1 が,OS 切替え処理を実行し OS2 に切替える.
(G)OS1 走行環境を保存し,前回保存されていた OS2 走行環境に復帰する.
(H)OS2 では当該周辺装置の割込み処理を実行した後,割込み復帰してユーザモードに戻る.

図 2 に OS2 走行中に,OS1 が管理する周辺装置の割込み A が発生した場合の動作[前記(A)～(D)]および,OS1 走行中に,OS2 が管理する周辺装置の割込み B が発生した場合の OS 切替えの流れ[同(E)～(F)]を示す.ここで占有外割込みとは,自 OS 走行時に,自 OS で占有しないハードウェアへの割込みを表す.

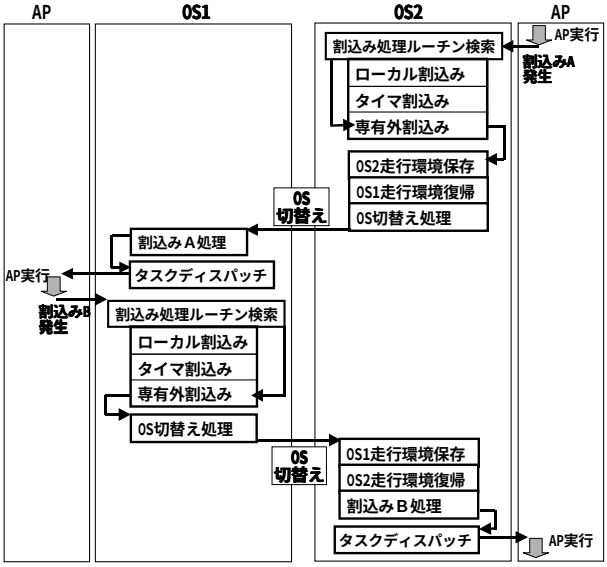

図２　OS 切替え処理の流れ

(2)タイマ割込み

OS 切替え自体の流れは,図 2 の流れと変わらない.しかし,タイマ自体はシステムで一つなので,タイマクロックが両方の OS に供給される必要がある.図 2 の動作をした場合にタイマ割込み処理はどちらか一方の OS でしか実行されない.従って,タイマ割込みに関しては各 OS がタイマ割込み処理を行うようにする必要がある.各 OS にタイマ割込みを行わせる方式としては以下が考えられる.

(方式 1)タイマ割込みが発生すると,走行している OS 側のタイマ割込み処理,OS 切替え,他方の OS のタイマ割込み処理を実行する.つまり,タイマ割込み毎に各 OS のタイマ割込み処理を実行する.
(方式 2)タイマ割込みの発生周期を半分にし,タイマ割込みが発生すると,OS 切替え,他方の OS のタイマ割込み処理を実行する.
(方式 3)全てのタイマ割込みを OS2 の処理として呼び出し,必要に応じて OS1 のタイマ割込み処理を実行する.

(方式 1)は OS1 側のタイマ割込み処理終了後に OS 切替え処理を実行する必要があるが,タイマ割込み処理の終了箇所を特定し,さらに OS1 には改造を加えず切替え処理を実行させることは容易

でない.

(方式 2)は,タイマ処理の終了箇所の特定を行う必要がない.しかし,タイマ割込み以外のハードウェア割込みで OS 切替えが発生した場合,一方の OS でしかタイマ割込み処理が実行できない可能性があり,タイマクロックを両方の OS に供給できない.

(方式 3)は,OS2 が制御を持っているため,OS1 と OS2 が異なるタイマ周期などの場合に有効である.しかし,OS1 のタイマ割込み処理を実行するかの判断処理が必要なため,性能低下が考えられる.

今回の実装では,上述の(方式 3)が複雑な判断処理を実装しない限り,性能低下は問題にならないと考えられる.従って,MRM ではタイマ割込みは OS2 が占有制御し,OS1 のタイマ割込み処理を行うか OS2 が判断する方式を実装する.(方式 3)を採用したタイマ割込み発生時の OS 切替え流れを図 3 に示す.OS2 側で判定処理を行い,OS1 側に切替えるか判断する.

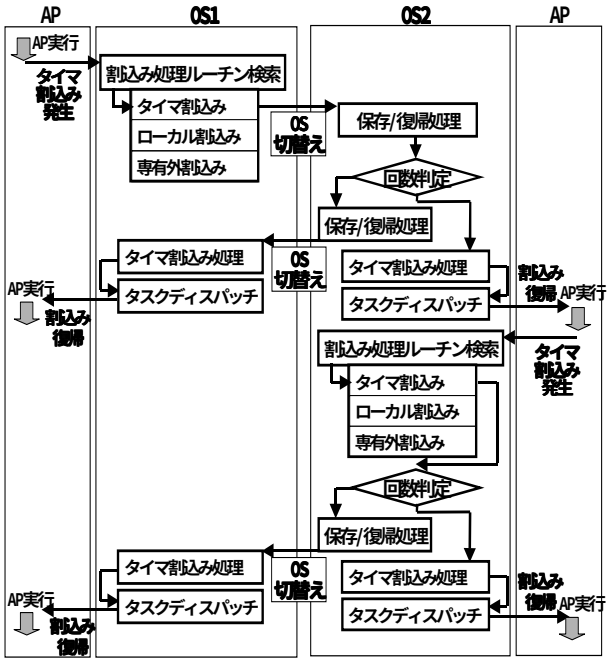

図３　タイマ割込み発生時の OS 切替え処理

上述した OS 切替えの流れにおいて、以下の課題がある.

(1)OS の走行環境保存/復元処理の実行

OS 切替えにおいて,切替わる側の OS 走行環境を保存し,切替える側の OS 走行環境を復元させる.ここで,OS の走行環境として必要な情報を保存/復元させる必要がある.

(2)OS1 から OS2 への OS 切替え処理

共存する OS1 には改造を加えないという前提のもと,OS1 で占有外割込みが発生した場合にプログラムカウンタを合わせることで OS2 へ制御を切替える方法を確立する必要がある.

(3)OS2 から OS1 への OS 切替え処理

(2)と同様に共存する OS1 には改造を加えず,プログラムカウンタを合わせることで,OS2 で占有外割込みが発生した場合に OS1 へ制御を切替える方法を確立する必要がある.

3.実装内容

ここでは,起動方式と OS 切替え方式の実装内容について述べる.なお,PC は Pentium の CPU を搭載した PC/AT 互換機を想定し,OS は Red Hat Linux7.2(kernel 2.4.7)を想定する.

3.1 起動方式

2.1.2 項で述べた,起動方式で課題となる三つの課題について,実現内容を説明する.

3.1.1　OS2 の使用可能実メモリ領域の制限

最初に起動した OS2 が OS1 利用領域を考慮してメモリマップを計画し,各 OS で使用する物理メモリ領域を設定する.具体的には,図 4 に示すように,OS2 は実メモリの上位アドレス部分を使用可能領域と認識し,下位アドレス部分は使用不可能領域と認識する.そして,OS2 は OS1 に実メモリの下位アドレス部分を使用可能領域と認識し,上位アドレス部分を使用不可能領域と認識するようなメモリマップを提供する.

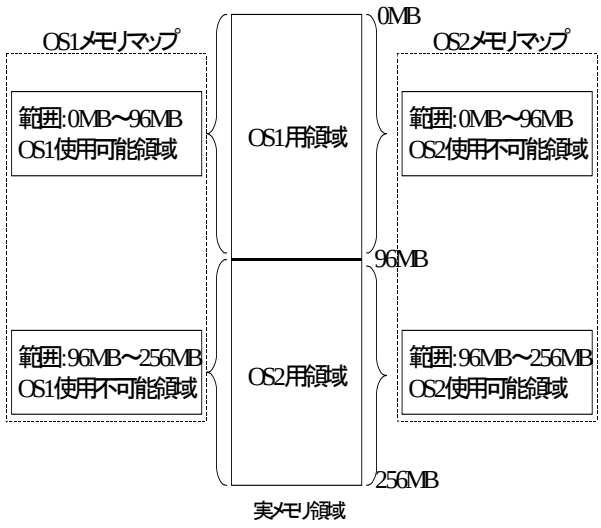

図 4　実メモリ領域の割り当て

3.1.2 OS1 の読込み

OS2 は,OS1 の読込み後に処理が再開できるように，現在の走行環境情報（レジスタ値など）を保存領域に格納する.その後,あたかも一つの OS が起動するかのように,つまりリセットボタンを押下したのと同じ状態で再起動を実行し,OS1 イメージを実メモリ上に読み込ませる.

3.1.3 OS2 環境保護のための監視処理

監視処理は,CPU のトレースモードを利用する.具体的には,１命令毎に発生する例外処理において,プログラムカウンタ値がスタックに積まれることを利用して OS1 の初期化処理を監視する.OS1 が OS2 の動作を妨げる不都合な命令を実行しようとした場合,OS2 は該当命令を実行させず代替処理を実行する.監視処理は,以下の処理から構成される.

(1)監視状態への移行
(2)命令シミュレート処理
(3)割込み管理
(4)監視状態の終了

各処理について実装内容を以下に述べる.

(1)監視状態への移行

監視状態,つまりトレースモードでの実行は,フラグレジスタのトレースビットをセットすることで実現する.トレースビットがセットされると,CPU は一命令実行毎に例外を発生させ,例外処理を実行した後,次の命令実行に移行する.OS2 はトレースモードへ移行した後,OS1 初期化処理の先頭へジャンプし OS1 初期化処理を開始する.

監視状態の開始時に以下の処理を行う.

(A)OS1 用メモリマップの用意

OS1 が使用制限された実メモリを認識するように,OS1 が参照するメモリマップの変更を監視状態開始時に行う.

(B)OS 切戻しを意識した監視状態の開始処理

OS2 監視状態での OS1 初期化処理が終了した後,各 OS は割込みを契機として OS 切替えを行い,OS 切替え後は割込みからの復帰処理を実行する.監視状態終了時には OS2 から OS1 への切替えを行うため,監視状態の開始をあたかも OS2 側で割込みが発生したかのような状態にする必要がある.これは,あらかじめ OS1 初期化処理の先頭にジャンプする処理を行う割込み処理を作成しておき,OS2 による監視状態に移行した直後に当該割込みを発生させることで割込みが発生した状況を実現する.

(2)命令シミュレート処理

OS2 は,OS1 が発行した命令が後の OS2 の走行を妨げないように,その実行を制御する.つまり,OS2 の走行を妨げない命令はそのまま実行させ,妨げる命令は実行させずに,代替処理を行う.以下に,代替処理を行う命令について,代替処理する理由および代替処理内容を述べる.

(A)フラグレジスタ設定命令

OS2 の監視状態はトレースモードでの走行により実現している.しかし,OS1 の初期化処理において,フラグレジスタ設定命令が実行された場合,フラグレジスタ上のトレースビットがクリアされ,監視状態が解除される可能性が生じる.従って,対処としてフラグレジスタ設定命令と認識した場合,設定値のトレースビットがセットされた状態で命令を実行するように設定値を変更する.

(B)割込み管理テーブル設定命令

本命令を実行すると例外および割込み発生時に参照する割込み管理テーブルが OS1 側のものになるため,次命令の内容をチェックする例外処理が実行できなくなる.従って,トレースモードでの監視状態を解除しないように,割込み管理テーブル設定命令は実行させず命令飛越しを行い,OS1 割込み管理テーブル情報として保存しておく.

(C)セグメントテーブル設定命令

本命令を実行するとページング機構において使用するセグメントテーブルが OS１側のものになるため,OS2 の動作が妨げられる.従って,セグメントテーブル設定命令は実行させず命令飛越しを行い,セグメントテーブル情報として保存しておく.

(D)ページテーブル設定命令

本命令を実行すると OS1 初期化処理が OS1 側仮想空間上で実行されることになり,監視状態が解除される.監視状態を保持するためには,OS2 側のページテーブルを使用し,OS2 の仮想空間上で OS1 の初期化処理を実行する必要がある.従って,OS1 初期化時に実行されるページテーブル設定命令は実行させず,ページテーブル設定情報として保存しておく.

(E)制御レジスタ設定命令

制御レジスタ(cr0,cr4)はプロセッサの動作モードと状態を制御するシステム制御フラグ等が格納されているレジスタである.従って,OS2 の状態を変更させないために制御レジ

スタへの処理は実行させず,制御レジスタの設定情報として保存しておく.

(3)割込み管理

上述したように,割込み管理テーブルの設定命令は実行させない.しかし,初期化処理において OS1 側の割込み処理を実行する必要がある場合が存在するため,例外処理内で割込み管理テーブルに対して,大きく以下の二つの処理を行う.

(A)システムコールのエントリ変更

システムコールは割込みで実行される.監視状態では OS1 側割込み管理テーブルの設定命令を実行させないため,OS1 初期化処理内のシステムコールは OS2 側のシステムコールが実行される.従って,OS1 初期化処理内のシステムコールは OS1 側のシステムコールを行い,例外処理中のシステムコールは OS2 側のシステムコールを実行する様に割込み管理テーブルのシステムコールのエントリを変更する.この際,割込みは 1 命令と認識されるため,システムコール割込み処理内では例外は発生しない.従って,OS1 初期化処理内でのシステムコールと,例外処理内でのシステムコールを上手く区別することができる.OS2 による監視状態の終了時には,OS1 側の割込み管理に切替えるため,OS1 側システムコールエントリを設定しない.

(B)共有資源への割込みのエントリ変更

監視状態下での共有資源への割込みは,OS2 側が処理する.しかし,OS1 側の割込み処理を実行する必要がある場合がある.例えば,初期化処理において OS1 はタイマ割込みが正しく発生しているか確認処理を行う.従って,監視状態において,タイマ割込みに関しては OS1 側の割込み処理を行う必要がある.しかし,監視状態下において OS1 がタイマ割込みに対する処理を割込み管理テーブルに登録する前にもタイマ割込みは発生する.そのため,OS1 側割込み管理テーブルにタイマ割込みエントリが登録された時点で OS2 側割込み管理テーブルのタイマ割込みエントリを OS1 側のエントリに変更する.OS2 による監視状態終了時には OS2 側割込み管理テーブルに,保存していた OS2 側タイマ割込み処理のエントリを設定する.

(4)監視状態の終了

監視状態の終了は,OS1 の初期化処理において監視状態を終了する契機となる命令の OS2 仮想空間および OS1 仮想空間上の仮想アドレスを一致させ,OS2 仮想空間上での命令を適切な命令に変更することによって実現する.OS2 による OS1 初期化処理の監視状態終了の流れを図 5 に示す.

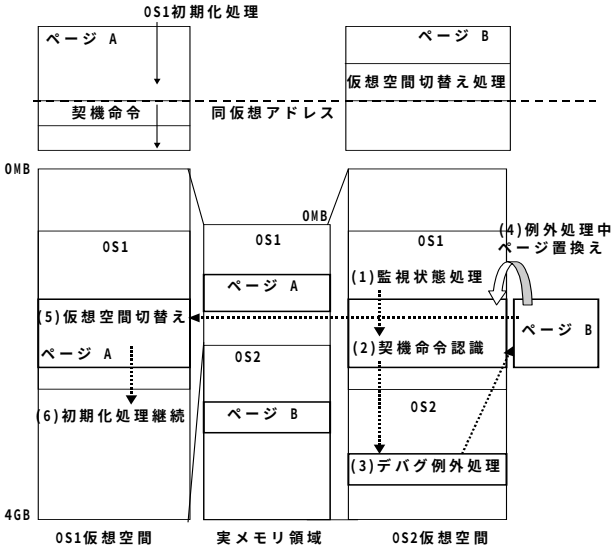

図 5　監視状態の終了流れ

監視状態での OS1 の初期化処理は OS2 仮想空間上で実行される.そして,監視状態を終了する契機となる命令を認識した場合,以下の流れにより監視状態を終了し,OS1 仮想空間上での実行に移行する.

(A)OS1 初期化処理の 1 命令毎に発生する例外処理において,監視状態終了の契機となる命令を認識する.
(B)例外処理内において,監視状態終了処理を実行する.
(C)OS1 の初期化処理が実行されている仮想空間が OS2 のものから OS1 のものに切替える.
(D)OS1 仮想空間上で OS1 初期化処理が継続され,OS1 が起動する.

上述の流れは,実メモリ領域上では異なるページＡとページＢにおいて,OS2 仮想空間上で,ページＡが対応付けられている仮想アドレスにページＢが対応付けられているように,ページテーブルのエントリを変更することで実現する.ここで,監視状態終了の契機となる命令がページＡに書かれており,OS2 仮想空間から OS1 仮想空間への切替え処理がページＢに書かれているとする.これにより,OS2 仮想空間上で実行していた OS1 初期化処理を契機となる命令からは OS1 仮想空間上で実行するように切替えることができる.

例外処理内の監視状態終了処理では,割込み管理テーブルエントリおよびセグメントテーブルエントリの復元,OS 切替え環境の構築,および監視状態終了処理を行う.以降で各処理について説

明する.

(A)割込み管理テーブルおよびセグメントテーブルエントリの復元

OS2 による監視状態においては OS2 側の割込み管理テーブルのシステムコールエントリおよびセグメントテーブルのタスクディスクリプタエントリなどが,OS1 の環境に変更されている.従って,OS2 の監視状態終了時には OS2 側の設定に復元する必要がある.これは,エントリ内容をあらかじめ保存しておき終了時の例外処理内で復元させることで実現する.

(B)OS 切替え環境の構築

共存させる OS の改造を行わないため,OS2 による監視状態の終了処理時には,各 OS が割込みを契機として切替わる環境が構築されている必要がある.これは,監視状態終了時の例外処理内で,割込みルーチンの仮想アドレスを一致させておくことで実現する.詳細は次章で述べる.

(C)監視状態終了処理

監視状態を終了するため,以下の処理を行う.

(a)OS2 仮想空間から OS1 仮想空間への切替え処理が書かれているページＢを用意する.この時,切替え処理は,契機命令の仮想アドレスから切替え処理の大きさを減算した仮想アドレスに書かれている.

(b)OS2 仮想空間上でのページＡの位置にページＢが対応付けられるようにページテーブルのエントリ内容を変更する.

(c)OS1 側割込み管理テーブル設定命令及び OS1 側セグメントテーブル設定命令を実行する.

(d)監視状態,つまりトレースモードでの実行を終了させるため,例外発生時に積まれているフラグレジスタの値から,トレースビットをクリアし,再度格納する.

(e)例外終了後,次に実行されるアドレスは例外発生時にスタックに積まれている.このアドレスは,監視状態を終了する契機命令のアドレスになっている.この値をページテーブル変更処理の大きさを減算した仮想アドレスに変更する.以上の作業により,例外処理終了後に実行される命令は,ページＡの契機となった命令ではなく,ページＢの OS2 仮想空間から OS1 仮想空間への切替え処理になる.

3.2 OS 切替え方式

OS 切替え方式の実現のために必要な OS 環境保存／復元処理,および OS 切替え処理について述べる.

3.2.1 OS 環境保存／復元処理

OS が交互に切替わる環境を構築するには,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際の環境を復元する処理が必要である.ここで,OS の環境とは以下の情報を指す.

(1)OS の仮想空間情報（セグメントテーブル,ページテーブル）
(2)割込み管理テーブル
(3)他のレジスタ情報

(1)OS の仮想空間情報

MRM では,各 OS は独立に動作するため,各 OS が独立な仮想空間のセットを持つ.そこで,OS の仮想空間の情報を保持するセグメントテーブルとページテーブルを OS 切替え時に切替える.従って,各 OS が使用するテーブルのアドレスを保持および復元する必要がある.セグメントテーブルについては,初期化処理において設定したテーブルを使用するため,OS1,OS2 ともに初期化処理においてセグメントテーブル情報を取得し,保存する.また,ページテーブルについては,実行されるプロセスによって異なるページテーブルを使用するため,OS 切替え時に保存／復元する必要がある.従って,OS 切替えを実行する毎にページテーブル情報を保存／復元する.

(2)割込み管理テーブル

割込みが発生した場合,CPU は割込み管理テーブルとして参照し,該当する割込み処理を実行する.OS 切替えを実行するには各 OS で各 OS 用の割込み管理テーブルを用いて割込みを行う必要があるため,OS 切替え時に,割込み管理テーブル設定命令を実行し,各 OS が参照すべき割込み管理テーブルを復元する.

(3)他のレジスタ情報

OS 切替えは OS の走行中に動的に発生するため,発生時の切替わり元 OS の走行環境を保持しておく必要がある.プロセスは CPU 上のレジスタを用いて動作を行うため,これらのレジスタ値を復元することで OS 切替え発生時の環境に復帰できる.具体的に以下のレジスタ値を OS 切替え時に保存／復元する.

(A)汎用レジスタ
(eax,ebx,ecx,edx,edi,esi,ebp,esp)
(B)セグメントレジスタ(cs,ds,fs,gs,ss)
(C)フラグレジスタ
(D)制御レジスタ(cr0,cr4)

3.2.2　OS1 から OS2 への OS 切替え

割込みが発生すると CPU は OS1 側割込み管理テーブルを参照する.テーブルのエントリが指し示すアドレス,つまり OS1 側の OS 切替え処理として OS1 から OS2 への仮想空間切替え処理が行われる.

これは,CPU がページング機構において参照するページテーブルを OS2 側のページテーブルに変更することで実現する.これにより,次命令以降,CPU は新しく設定されたページテーブルの対応に従って処理を継続する.変更後のページテーブルは OS1 側の割込み管理テーブルが指し示す割込み処理への仮想アドレスと,OS2 側で継続する OS 切替え処理の仮想アドレスが連続するように設定されている.

次に,OS2 側の OS 切替えでは以下の処理を行う.

(1)OS2 側セグメントテーブルおよび割込み管理テーブルへの切替え
(2)OS1 環境保存
(3)OS2 環境復元
(4)該当する割込み処理へのジャンプ

3.2.1 節の(1),(2)及び,(3)に示す保存／復元処理を行い OS2 側の走行環境に復帰した後,発生した割込みに該当する割込み処理を OS2 制御下で実行し,OS2 から OS1 への OS 切替えが発生した時点に復帰する.

3.2.3　OS2 から OS１への OS 切替え

割込みが発生すると CPU は OS2 側の割込み管理テーブルを参照する.ここで,割込み管理テーブルのエントリをあらかじめ変更し,以下の処理を行うルーチンへジャンプする.この際,OS2 側で使用するページテーブルは OS 切替え用のテーブルを使用する.

(1)OS2 走行環境保存
(2)OS1 環境復元
(3)OS1 側セグメントテーブルおよび割込み管理テーブルへの切替え
(4)OS1 側タスクレジスタの設定

上述の処理を実行した後,OS 切替え処理へジャンプする.OS 切替え処理では,OS1 側のページテーブルのアドレスを cr3 レジスタへ設定する.これにより,OS2 から OS1 への OS 切替えを行い,OS1 側割込み処理を実行する.OS1 側では,該当する割込み処理を OS1 制御下で実行し,OS1 から OS2 への OS 切替えが発生した時点に復帰する.OS 切替え環境を図 6 に示す.

OS1 から OS2,OS2 から OS1 への OS 切替え共に,プログラムカウンタが連続するようにページテーブルを設定し,切替わる側のページテーブルのアドレスを cr3 レジスタに設定する処理により OS 切替えを行う.

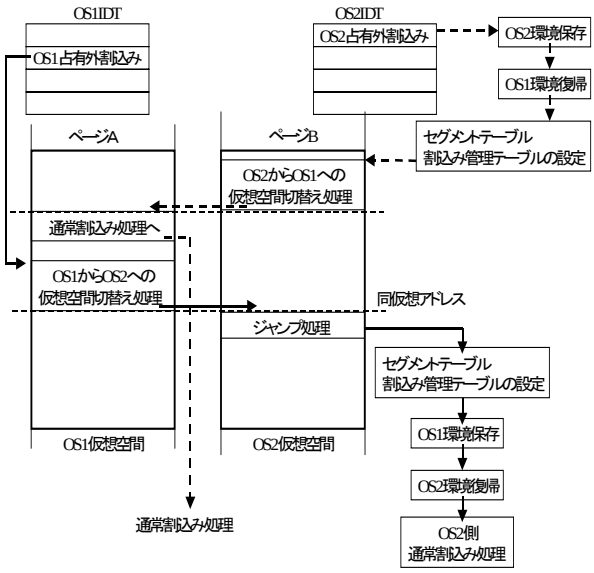

図 6　OS 切替え環境

4.おわりに

複数実計算機（MRM）実現時の課題である各 OS の起動方式,切替え方式について実装内容も含め述べた.具体的には,一方の OS が他方の OS の初期化処理内容を監視できる状態をトレースモードの利用により実現し,各 OS の使用するメモリ領域の排他的な確保および各 OS の走行環境を破壊しないように初期化処理において不許可とする命令を示した.また,実メモリ領域において共有領域を持たない各 OS が制御を移行するための OS 切替え方式として,プログラムカウンタの値が連続するように割込み処理ルーチンを配置することで割込みを契機として OS が切替わる際の処理内容を示した.今後は MRM の起動および切替えに関しての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]桝本圭,中島雄作,伊藤健一,乃村能成,谷口秀夫,“複数 OS 環境における OS 切替え方式,”情報処理学会研究報告,Vol.2003,No.19,pp.23-30,2003