

Kexecを用いた TwinOS起動方式の検討

岡山大学 工学部 情報工学科
中原 大貴

研究背景

<TwinOS方式>

1台の計算機上で、2つのLinuxが同時走行する方式

<現在のTwinOSの問題点>

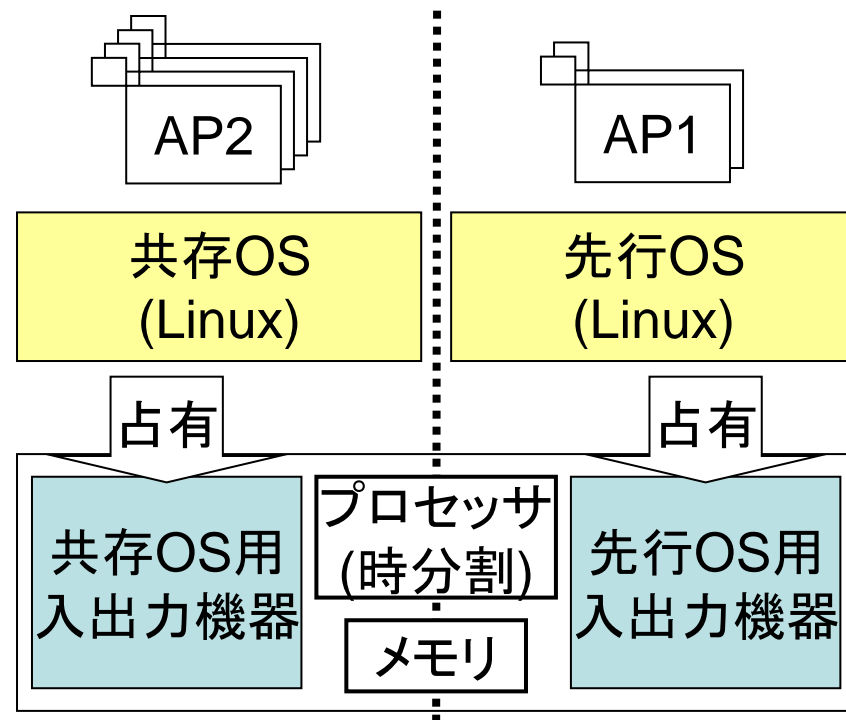
先行OSの環境を保存したまま共存OSを起動

➡ 共存OSの起動処理が複雑

<Kexec>

カーネルの高速な再起動を実現する機能

➡ TwinOSへの応用を検討



Kexec

高速な再起動を実現する機能

<特徴>

- (1) Linux Kernelに標準搭載
- (2) セットアップルーチン, BIOS, およびブートローダを通らないことで高速なカーネルの再起動を実現

<動作>

- (1) 再起動に用いるカーネルイメージの読み込み
- (2) 走行中のカーネルの終了処理
- (3) カーネルイメージを利用可能なメモリ領域に展開
- (4) カーネルを起動するための前処理
- (5) 展開したカーネルイメージの先頭にジャンプ

Kexecを用いた共存OS起動処理

KexecをTwinOS用に改変することで共存OS起動処理の簡略化

(要求1) 先行OSの走行環境の保存と復元

(要求2) 先行OSの走行環境の保護

(要求3) メモリマップの領域のタイプを無視してカーネルを展開

(要求4) カーネルを任意のメモリアドレスに展開

先行OSの走行環境の保存と復元

共存OS起動時，共存OSの初期化処理が先行OS走行環境を破壊

➡ 先行OS走行環境の保存と復元が必要

(対処1) Kexecを用いたカーネル走行環境の保存と復元

既存のKexecの機能を利用

＜Kexecのカーネル走行環境保存機能＞

再起動前のカーネル走行環境を保存する機能

この機能には共存OS起動処理において必要の無い処理が存在

➡ 必要な処理だけを呼び出すように改変

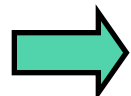
- カーネル走行環境保存機能をTwinOS用に改変
- カーネル走行環境復元処理を追加

先行OSの走行環境の保護

Kexecの再起動処理による終了処理の省略

	ファイル	関数	対象
(1)	drivers/base/core.c	device_shutdown	デバイス
(2)	drivers/base/sys.c	sysdev_shutdown	システムデバイス
(3)	arch/x86/include/asm/smp.h	smp_send_stop	SMP
(4)	arch/x86/kernel/apic.c	lapic_shutdown	APIC
(5)	arch/x86/kernel/hpet.c	hpet_disable	HPET

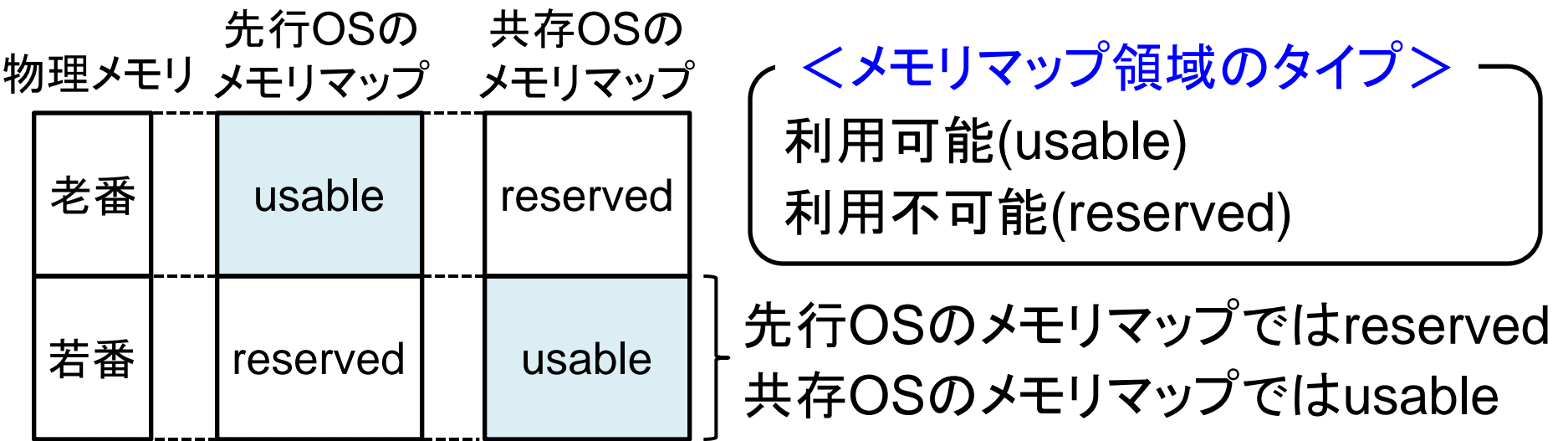
Kexecのカーネル起動機能を用いた再起動による終了処理が
先行OSの走行環境を破壊

 先行OSの走行環境を破壊されないように保護が必要

(対処2) 終了処理を省略

- 終了処理を行う(1)から(5)の関数の呼び出しを省略

メモリアップの領域のタイプを無視してカーネルを展開



TwinOSは、メモリアップの領域のタイプを意図的に変更

➡ メモリを若番と老番で2分割

先行OSのreserved領域に共存OSのカーネルを配置

Kexecはメモリ領域のタイプを参照し、配置可能かどうかを判定

➡ 共存OSのカーネルを配置しようとするエラーと判定

(対処3) メモリアップのタイプによるエラー判定を省略

- Kexec内のメモリアップのタイプによるエラー判定を省略

カーネルを任意のメモリアドレスに展開

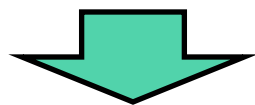
共存OSのカーネルをメモリの若番アドレスに展開しなければならない

➡ Kexecにカーネルの展開先を指定する機能が必要

(対処4) Kexecを用いたカーネル展開先の制限

(課題) カーネルのリアルモード部分の配置における制限

リアルモード部分は1MBより前のアドレスに配置する必要がある



Kexecの再起動処理ではリアルモード部分の走行を行わない

➡ リアルモード部分の配置における制限を解消可能

- ・リアルモード部分を1MB以上の領域に配置可能に変更

本発表のまとめ

<実績>

KexecをTwinOS用に改変

(1) Kexecを用いたカーネル走行環境の保存と保護

Kexecが持つカーネル走行環境保存機能をTwinOS用に改変

(2) 終了処理を省略

(3) メモリマップのタイプによるエラー判定を省略

(4) Kexecを用いたカーネル展開先の制限

リアルモード部分を1MB以上の領域に配置可能に変更

<今後の課題>

(1) 共存OS起動後に先行OSに処理を戻す機能の実装

(2) マルチコアCPU上でKexecを用いて各コアを占有し、カーネルを起動する機能の実装