Mint オペレーティングシステムにおけるデバイス移譲方式

左海 裕庸
岡山大学 大学院自然科学研究科
電子情報システム工学専攻
平成25年2月14日

背景

＜ Mint＞
マルチコアプロセッサ上で複数のLinuxカーネルを独立に走行
計算機資源の動的割り当てが不可能

＜ 要求＞
計算機資源を動的に割り当てたい
⇨ Loadable Kernel Module(LKM)を用いてデバイスを移譲

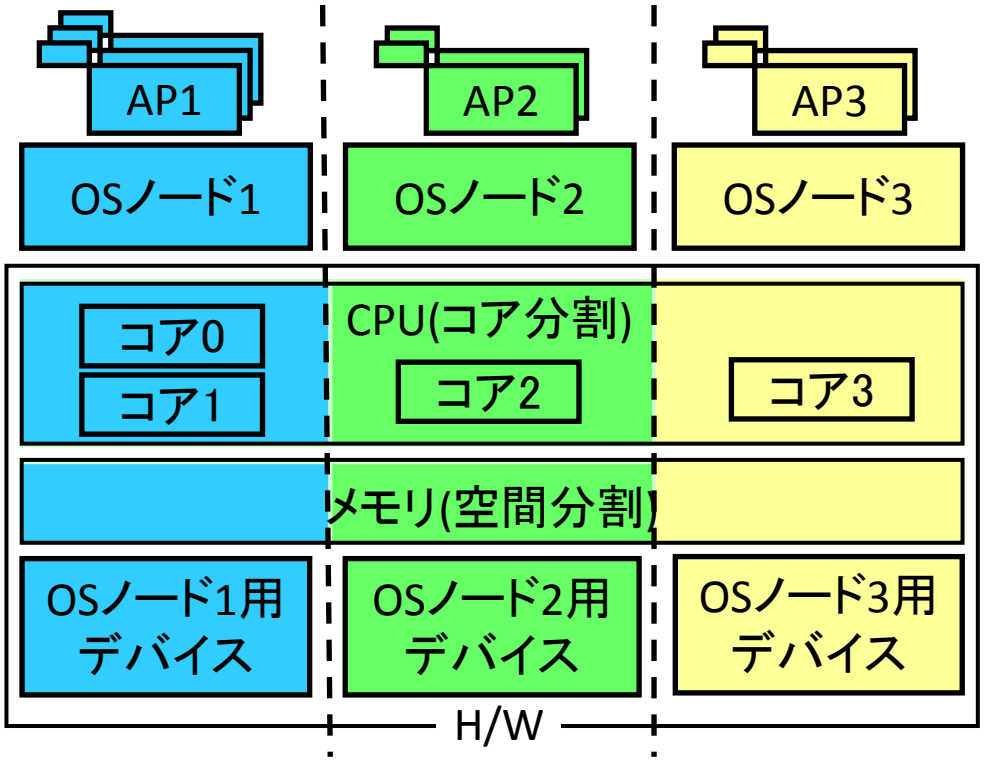
＜Mintにおけるデバイス移譲＞
複数のOS走行中にデバイスの占有状態を変更すること

＜デバイス移譲によって期待される効果＞
例：Mintの利便性の向上，負荷分散

＜ 目的＞
(1) LKMによるデバイス移譲方式の評価
(2) LKMによるデバイス移譲方式より短い時間での移譲可能な方式の実現と評価
⇨ 割り込みルーティング変更によるデバイス移譲方式の実現と評価

Mint オペレーティングシステム

(1) 1台の計算機上で複数のLinuxを独立に走行する
(2) 各OSノードは1つ以上のコアを占有する
(3) 全てのOSノードが相互に処理負荷の影響を与えない
(4)全てのOSノードが入出力性能を十分に利用できる



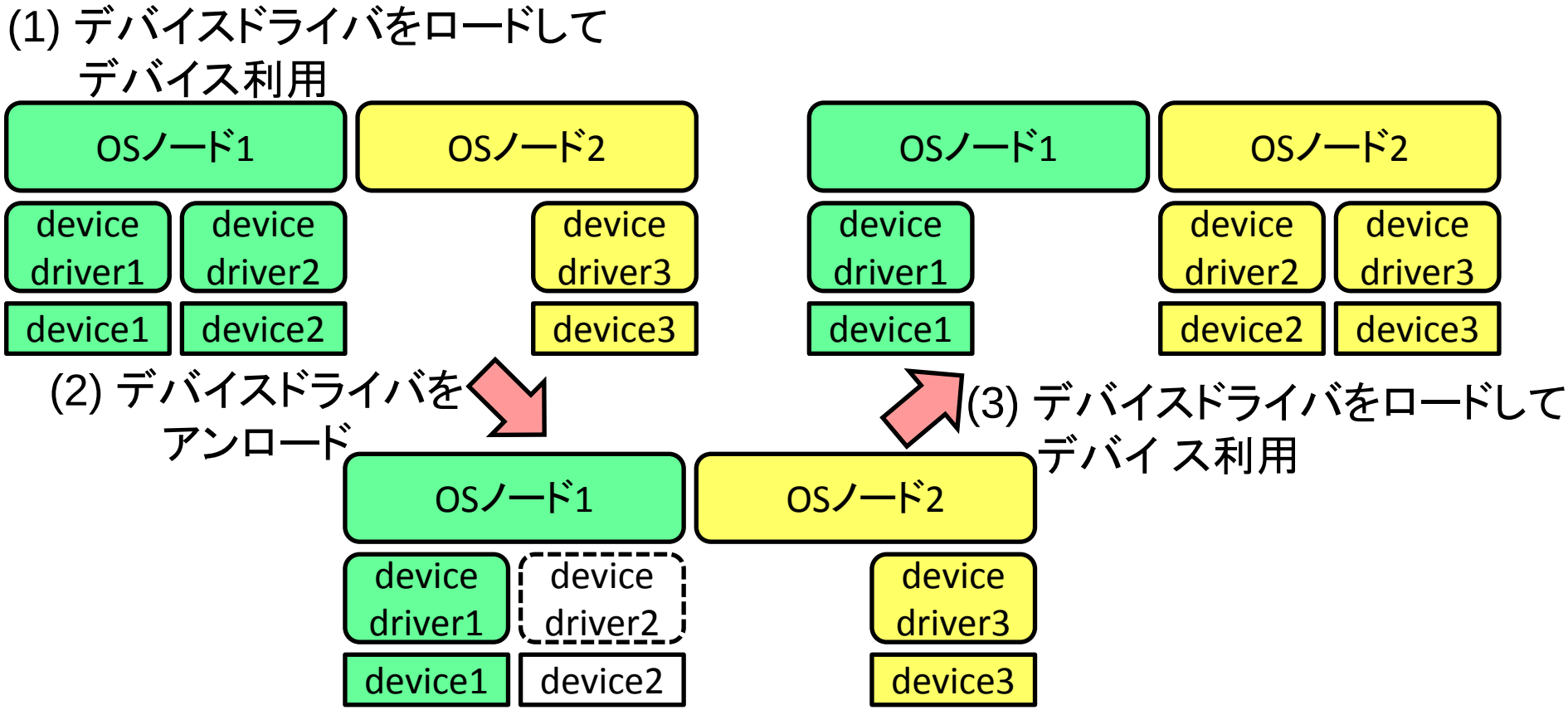
デバイス移譲の契機

デバイス移譲の目的によって要求される移譲速度は異なる

デバイス移譲の契機の例として以下の4つがある

契機	デバイス移譲の時間
(1) パケット受信ごと	パケット受信間隔よりも短い間隔
(2) タイムスライスごと	タイムスライス間隔(約10ms)よりも短い間隔
(3) 処理負荷が高まった際の負荷分散	比較的長くてもよい
(4) ユーザの任意	比較的長くてもよい

LKMによるデバイス移譲方式



LKMによるデバイス移譲の評価

処理		処理時間	合計
NICデバイスドライバロード処理	modprobe (前処理) init_moduleシステムコール modprobe (後処理)	0.60ms 77.8ms 0.0076ms	78.4ms
NICデバイスドライバアンロード処理	modprobe (前処理) delete_moduleシステムコール modprobe (後処理)	0.54ms 254.3ms 0.004ms	254.8ms

(1) LKMによるデバイス移譲時間はミリ秒単位のオーダー

(2) Linuxカーネルのタイムスライス間隔はミリ秒単位のオーダー

⇒ タイムスライス間隔でのデバイス移譲は不可能

(3) ユーザ任意でのデバイス移譲はタイムスライス間隔よりも長い周期

⇒ ミリ秒単位のオーダーで十分可能

割り込みルーティング変更による デバイス移譲方式

デバイスからの割り込みの通知先を変更することにより，
デバイスの占有状態を変更

(問題点) pin-base(割り込み線を使用する)割り込みにおいて
割り込み線を共有するデバイスが存在

⇒意図していないデバイスまで移譲される可能性

(対処) 割り込み線を使用しないMSIを利用

＜MSI(Message Signaled Interrupt)＞

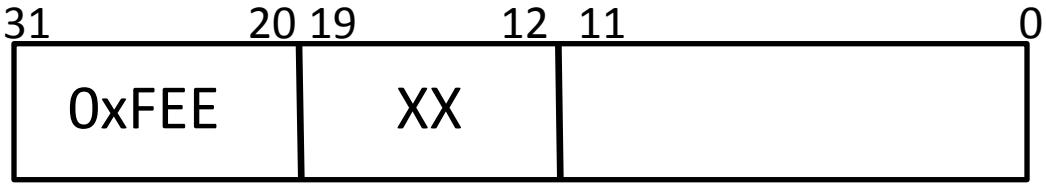
特定のデータを特定のアドレスへ書き込むことにより
発行される割り込み

MSI発行に関わるレジスタ

(1) Message Address Register

(A) Message Data Registerに格納するデータを書き込むアドレスを指定するレジスタ

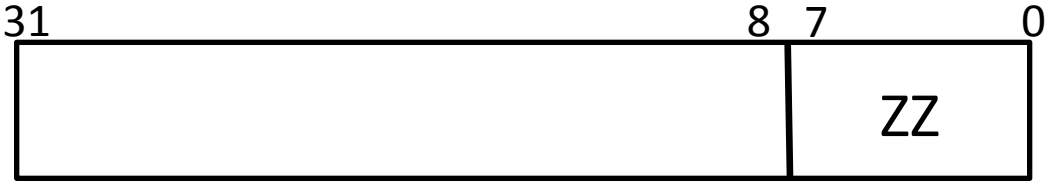
(B) 割り込み通知先情報(XX)を含む



(2) Message Data Register

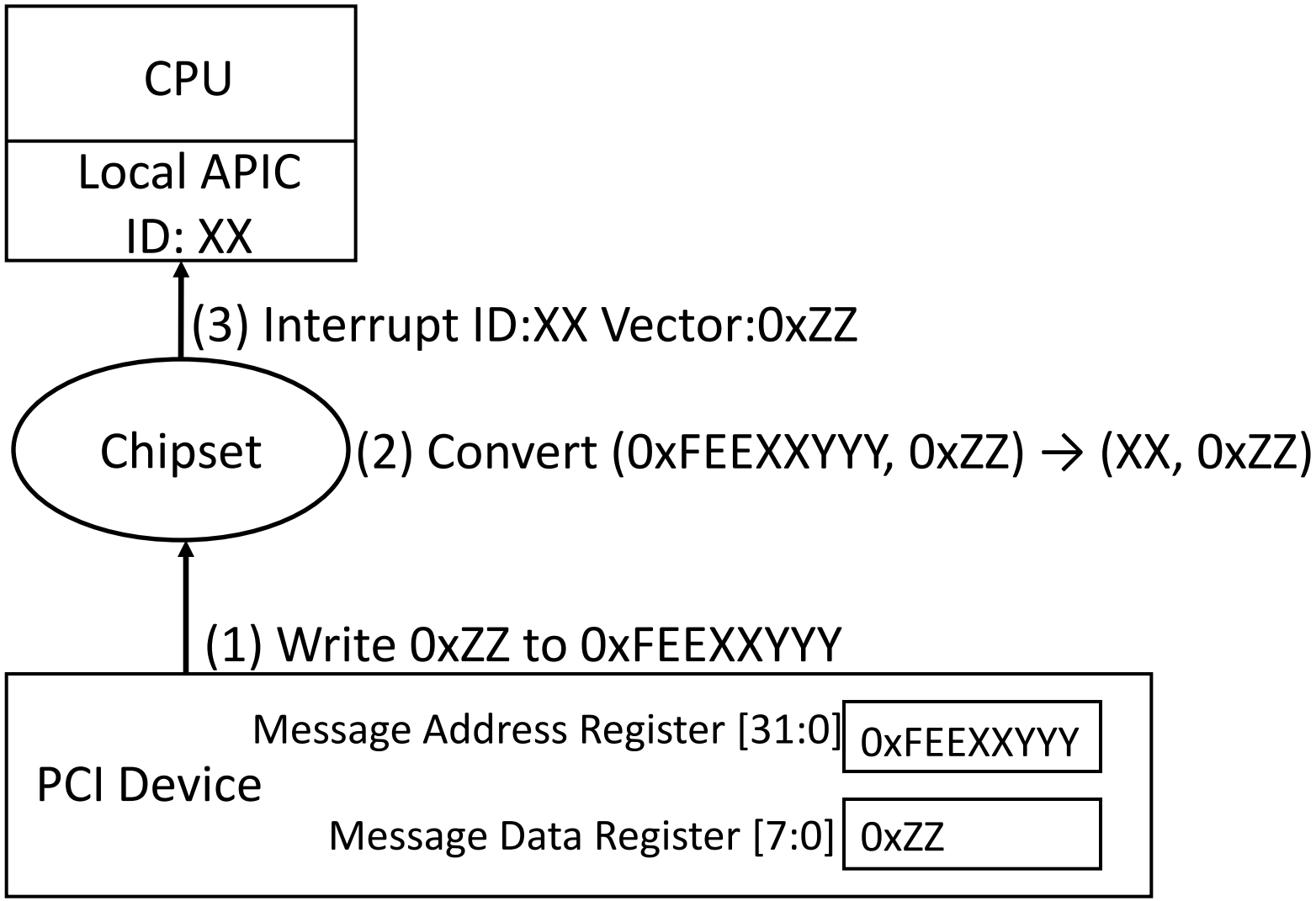
(A) Message Address Registerで指定するアドレスへ書き込むデータを格納するレジスタ

(B) 割り込みベクタ番号 (ZZ)を含む

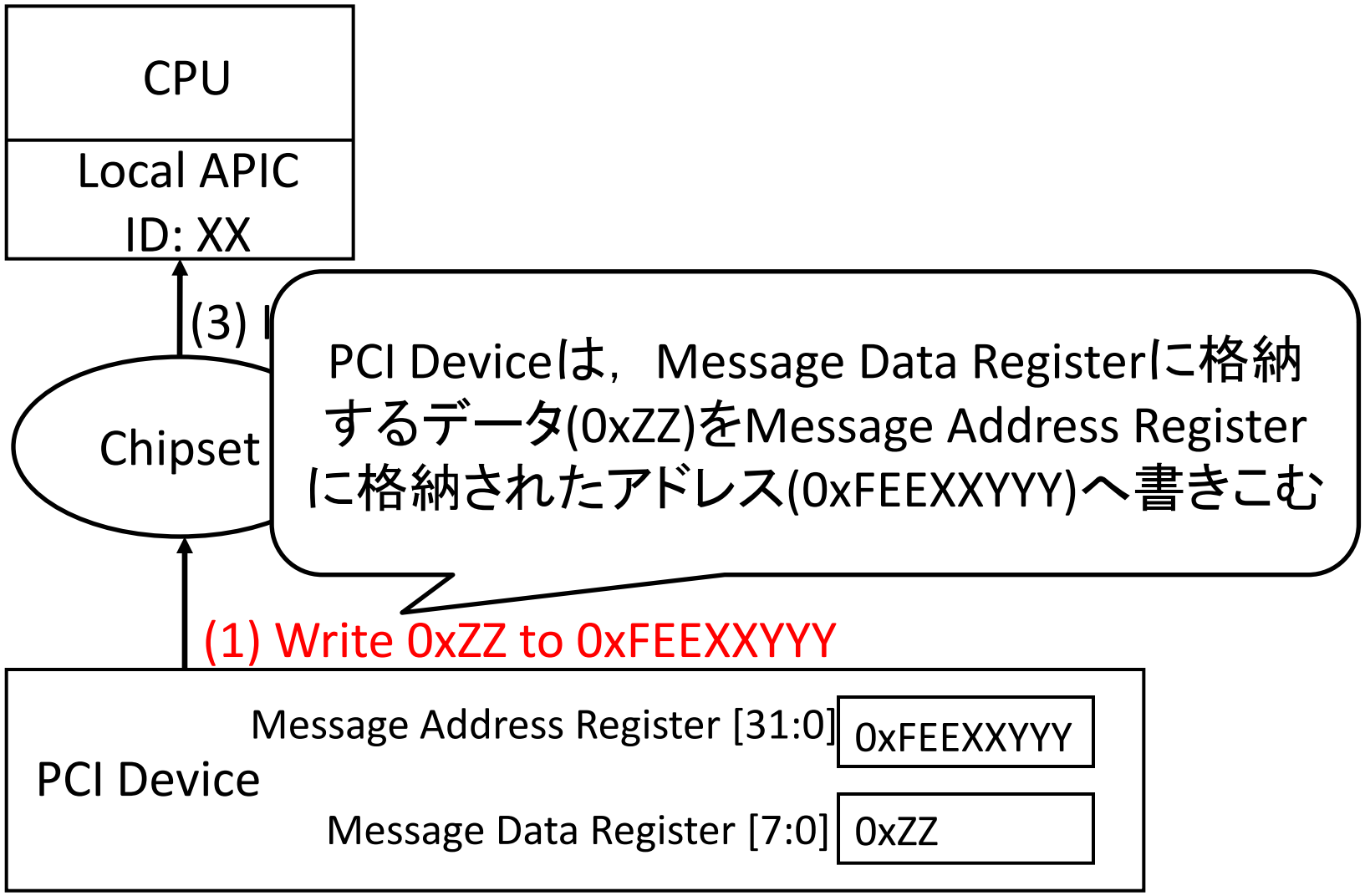


これら2つのレジスタはPCI Deviceごとに存在

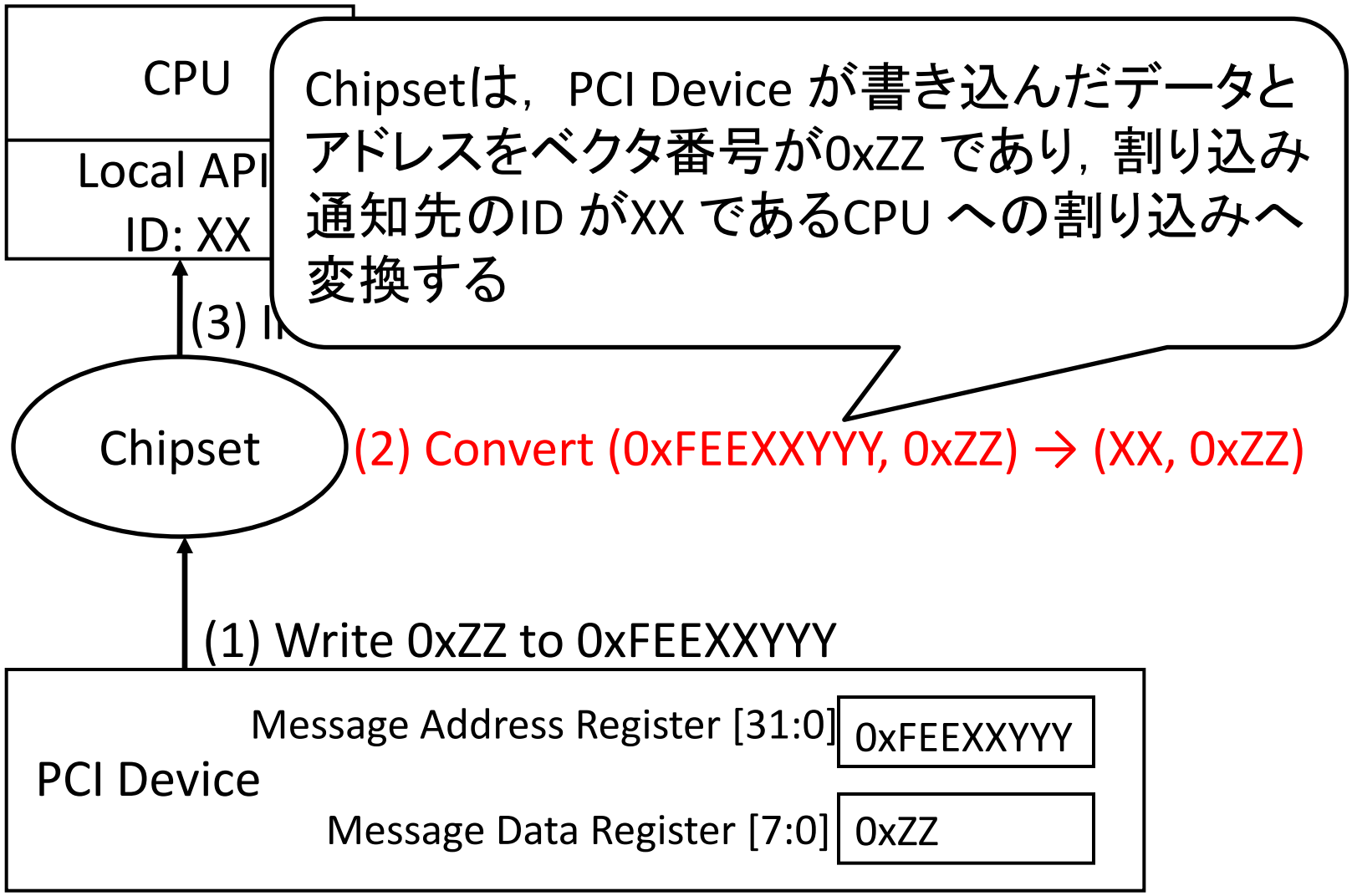
MSIによる割り込みの流れ(1/5)



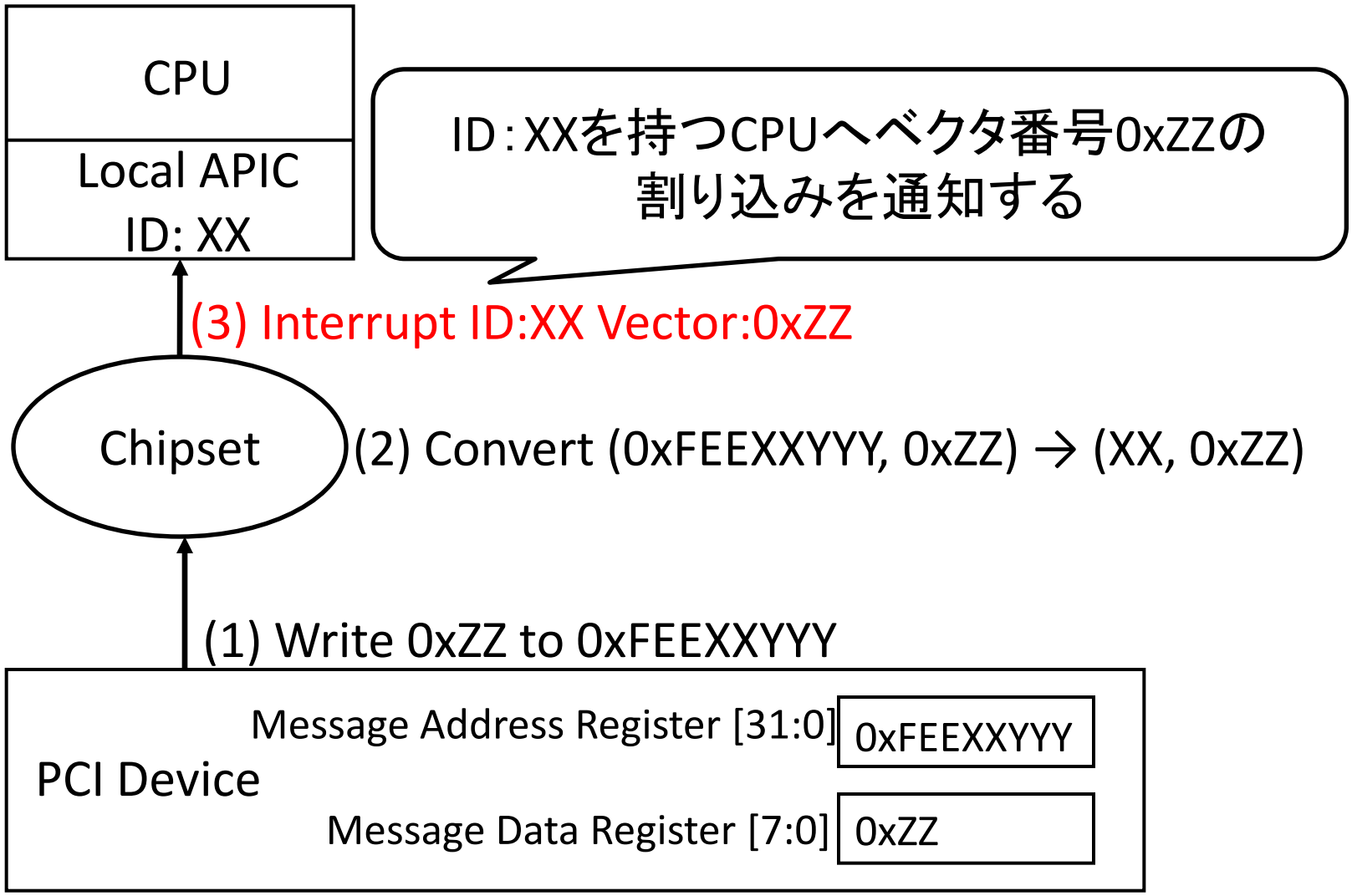
MSIによる割り込みの流れ(2/5)



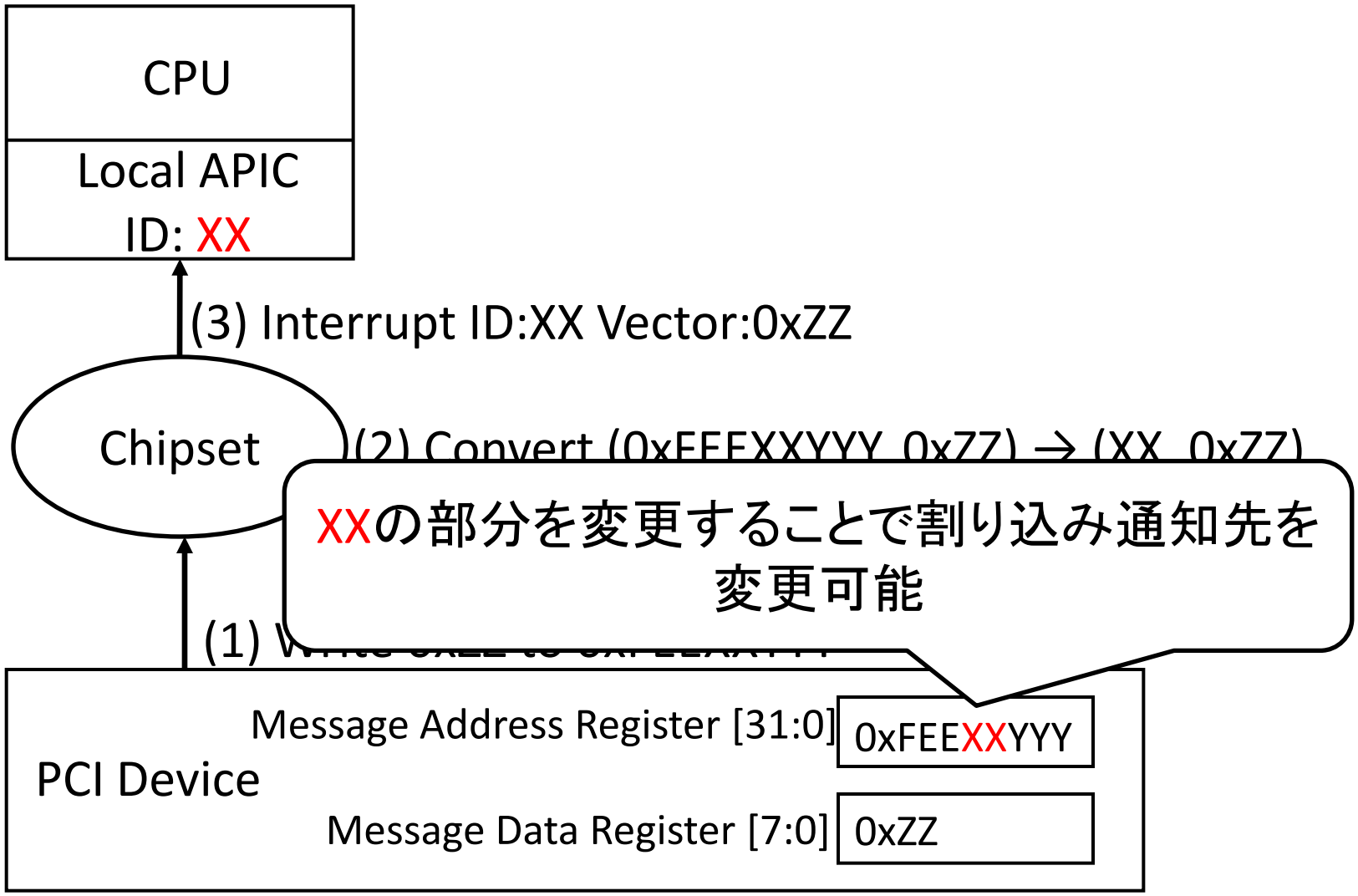
MSIによる割り込みの流れ(3/5)



MSIによる割り込みの流れ(4/5)



MSIによる割り込みの流れ(5/5)



MSIによるデバイス移譲の課題

(課題1) 移譲するデバイスの指定方法

(課題2) 割り込み通知先の指定と書き換え方法

割り込み通知先として移譲先のOSノードを指定する方法と割り込み通知先の書き換え方法が必要

(課題3) 割り込みベクタ番号の変更

各OSノードごとに割り込みベクタ番号を設定するため，デバイス移譲時に各OSごとの割り込みベクタ番号へ変更

(課題4) デバイス固有の処理

デバイスによって，割り込み通知先変更以外の処理が必要

対処

(対処1) デバイスIDによる移譲デバイスの指定

Linuxはデバイスごとに一意のデバイスIDを割り当て，管理

⇨デバイスを一意に特定できるため，これを利用

(対処2) 移譲時における割り込み通知先の指定と書き換え

(A) Mintは各OSノードが占有するコアに一意のLogical APIC IDを割り振り，割り込み通知先の指定に使用

⇨移譲時における割り込み通知先をLogical APIC IDで指定

(B) PCI Deviceのレジスタ読み書きを行う関数により書き換え

(対処3) 移譲時における割り込みベクタ番号の変更

デバイス移譲時に割り込みベクタ番号を移譲先OSノードの割り込みベクタ番号に変更

(対処4) デバイスドライバによる初期設定の解析と変更

各OSノードごとに設定する値が存在する場合，移譲先OSノードが設定する値へ変更

NICデバイス移譲時の固有処理

以下の2つの値を設定する必要がある

(1) Transmit Normal Priority Descriptors

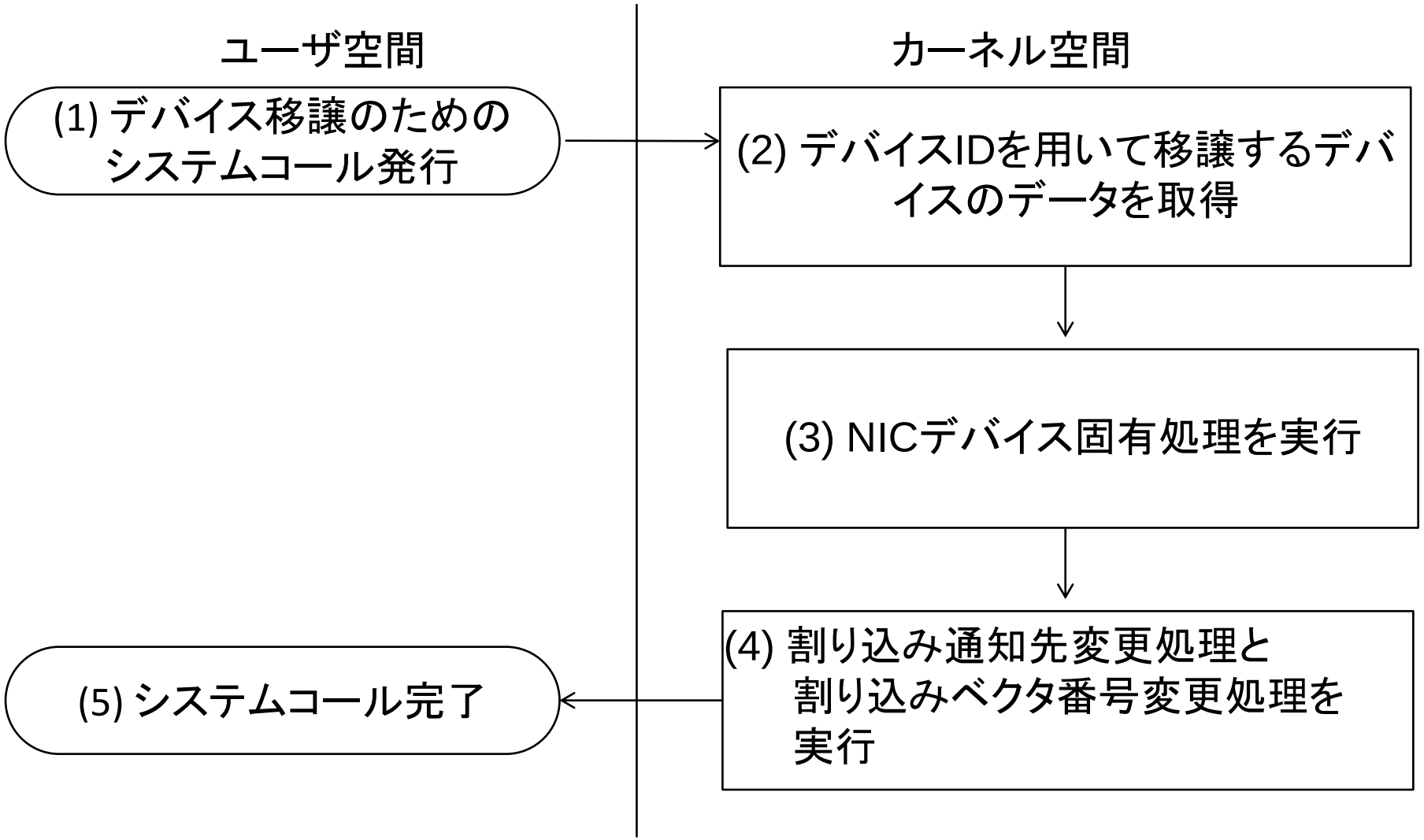
送信バッファの管理に使用するTx Descriptor Ringの
スタートアドレスを設定するレジスタ

(2) Receive Descriptor Start Address

受信バッファの管理に使用するRx Descriptor Ringの
スタートアドレスを設定するレジスタ

送受信バッファとこれを管理するDescriptor Ringは，各OSノードごとに確保するため，デバイス移譲時に移譲先OSノードのものに変更

MSIによるNICデバイス移譲の流れ



MSIによるデバイス移譲方式の評価

測定箇所	時間	合計
(2) デバイスIDを用いてNICデバイスのデータの取得	5.26µs	9.08µs
(3) NICデバイス固有処理	1.68µs	
(4) 割り込み通知先変更処理と割り込みベクタ番号変更処理	3.65µs	

(1) MSIによるデバイス移譲はマイクロ秒単位のオーダー

⇒ タイムスライス間隔でのデバイス移譲が可能

(2) ユーザ任意のデバイス移譲にも対応可能

(3) デバイスドライバの解析が必要なため，工数が必要

⇒ 移譲対象デバイスはMSIによるデバイス移譲が必要であるかの検討が必要

LKMによるデバイス移譲方式とMSIによるデバイス移譲方式の比較

	LKMによるデバイス移譲方式	MSIによるデバイス移譲方式
対応可能な契機	処理負荷が高まった際の負荷分散 ユーザの任意	パケット受信ごと タイムスライスごと 処理負荷が高まった際の負荷分散 ユーザの任意

(1) MSIによるデバイス移譲方式は，デバイスドライバの解析が必要

(2) LKMによるデバイス移譲方式は，Linuxカーネルへの改変が不要

LKMによる方式が対応可能な契機はLKMによる方式

LKMによる方式が対応不可能な契機はMSIによる方式

おわりに

(1) LKMによるデバイス移譲方式の評価

NICデバイスを例にロード/アンロードの時間を測定

(2) 割り込みルーティング変更によるデバイス移譲方式

デバイスからの割り込みの通知先を変更することにより，
デバイスの占有状態を変更する方式を実現

(3) 割り込みルーティング変更によるデバイス移譲方式の評価

NICデバイスの移譲時間を測定し，対応可能な契機について評価

(4) 各方式の比較評価

各方式がどのような移譲契機に適しているか評価