2K-6

Ships1 におけるノード間接続装置の研究

松原 裕人† 和田 卓† 大谷 真†
湘南工科大学情報工学科†

1. はじめに

Ships1(SHonan Institute of technology Parallel System 1)では、安価な CPU を複数用いたコストパフォーマンスの高い並列コンピュータを研究・開発している。Ships1 ではデータ転送によるボトルネックを解消し、性能の向上を図る為に新たにノード間結合装置を開発しノード間を繋ぐ。この装置を介し、raw I/O でノード間のメモリの直接転送を行う。本論文では考案したノード間結合装置のソフトウェア/ハードウェアのインターフェース及び、開発したデバイスドライバの処理方式を述べ、プロトタイプによる評価結果を報告する。

2. Ships1 におけるノード間接続方式の概要

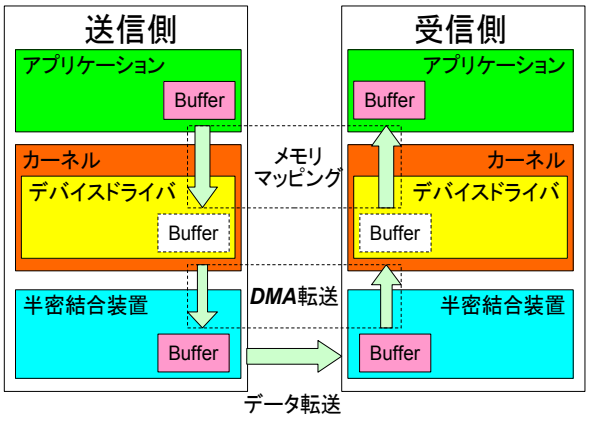

図 2-1 ノード間接続方式の概要

Ships1 では通信ノード間のアプリケーションバッファでの Zero-Copy 転送を目指すため、上の図の様な転送方式を取る。アプリケーション-デバイスドライバ間ではユーザ空間のメモリをカーネル空間にマッピングすることによってメモリをカーネル空間にコピーしたように見せる。次にデバイスドライバ-半密結合装置間は DMA(Direct Memory Access)転送を行う。DMA 転送を行うことにより CPU を介さずにメモリ-半密結合装置間のデータ転送が行え高速化が計れる。半密結合装置間は装置上のバッファにデータを滞留させること無く転送を行う。これによりデータをコピーすること無く高速のデータ転送を行うことを狙いとしている。

3. Ships1 におけるノード間接接続方式

半密結合装置の開発において新たに考案した処理方式案を提案した。以下でその処理方式案の解説を行う。

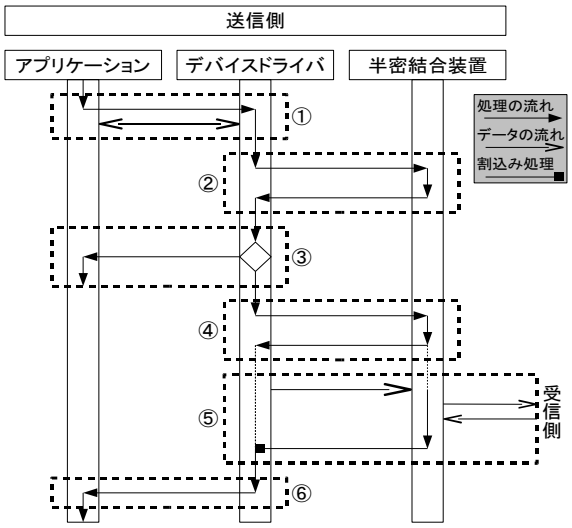

図 1. 送信ノードの処理の流れ

まず、データ送信時の具体的な処理の流れを説明する。下記の番号(①, ②, …, ⑪)は図 1 及び図 2 の図中の番号に対応している。

① アプリケーションが半密結合装置に対して write システムコールを実行する。デバイスドライバは write システムコールの呼出し時に渡されたバッファの仮想アドレスが指す物理アドレスにカーネル空間の仮想アドレスを割り当て、マッピングする。

② 半密結合装置から状態と転送待ちキューの長さを取得する。

③ 転送待ちキューが全て埋まっている場合はカーネルのバッファにコピーを行う。カーネルバッファにコピーできたサイズをアプリケーションに返す。転送待ちキューに空きがある場合は④に進む。

④ アプリケーションのバッファのバスアドレスとデータ長を転送待ちキューに追加する。

Inter-Node Connection in Ships1

Matsubara Hiroto†, Suguru Wada† and Makoto Oya†
† Department of Information Technology,
Shonan Institute of Technology

⑤ 半密結合装置は転送待ちキューの先頭のアドレスに対して DMA 転送を実行し、データを送信する。DMA 転送が終わったら転送待ちキューの先頭を削除し、IRQ を出す。IRQ で呼び出された割込みハンドラは転送待ちキューに入っていたバッファアドレスをアンマップする。
⑥ DMA 転送を行ったバイト数を返り値として返す。

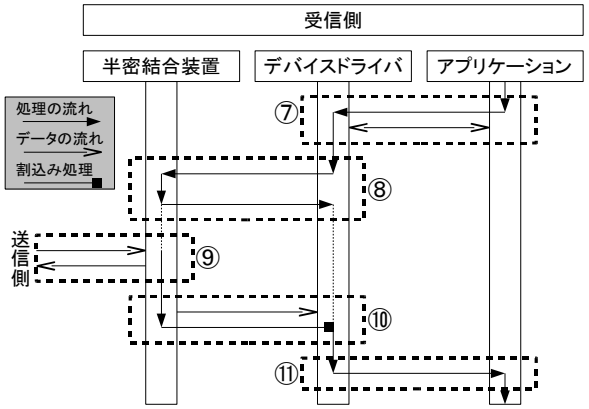

図 2: 受信ノードの処理の流れ

次にデータ受信時の具体的な処理の流れを説明する。図 2 はデータが届く以前に read システムコールが wait していた場合の図である。
⑦ アプリケーションが半密結合装置に対して read システムコールを実行する。デバイスドライバは read システムコール時に渡されたアプリケーション
ョンバッファの物理アドレスをカーネル空間にマッピングする。
⑧ 転送待ちキューに空きがあった場合、アプリケーションのバッファのバスアドレスとデータ長を転送待ちキューに追加し、スリープする。
⑨ 半密結合装置がデータを受信する。
⑩ 半密結合装置は転送待ちキューの先頭のアドレスに対して DMA 転送を実行する。DMA 転送が終わったら転送待ちキューの先頭を削除し、IRQ を出す。IRQ で呼び出された割込みハンドラは転送待ちキューに入っていたバッファアドレスをアンマップし、スリープしていたドライバを起こす。
⑪ 半密結合装置が DMA 転送したバイト数を read の返り値として返す。

4. プロトタイプによる検証

前略の方式の実現には、OS・ソフトウェアとして主に以下の点の実装を行う必要がある。
(i) ユーザ空間アドレスのカーネル空間アドレスへのマッピング
(ii) (i)に連動した DMA 転送
(iii) IRQ による割込み制御
(iv) デバイスドライバのバッファプールの制御
(v) ソフトウェア/ハードウェアの排他制御
これらのうち①, ② 及び③の一部についてデバイスドライバのプロトタイプを作り検証した。

プロトタイプの実装においては、OS は Linux 2.6.11_1.1369FC4smp、半密結合装置のシミュレータとして東京エレクトロンデバイス社の製品である TD-BD-PCI66 及び TD-PCI-FKip を利用した。以上のプロトタイプを用いて実験を行い①、②について考案した 3.の方式の妥当性が検証できた。プロトタイプの主要な考案点は以下の通りである。
・ユーザ空間アドレスのページ構造体に対するカーネル空間アドレスの割当て
・ユーザ空間アドレスのページ構造体の DMA バッファへのマッピング

5. まとめ

本論文では新たに考案した方式を示し、プロトタイプによってその実装可能性を示した。

今後の課題としては write システムコールが read システムコールに先行して実行された場合の動作の更なる検討、実装については scatter/gather の DMA バッファを用いた DMA 転送の実現。また、半密結合装置を使った動作と性能の検証などである。

参 考 文 献

1) Daniel P. Bovet, Marco Cesati. Understanding the Linux Kernel, 3rd Edition. O'reilly & Associates Inc, November 2005.
2) Honathan Corbet, Alessandro Rubini, Greg Kroah-Heartman. Linux Device Drivers, 3rd Edition. O'reilly & Associates Inc, February 2005.
3) Intel Corporation, IA-32 インテル アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル(上中下巻). Intel Corporation, 2004.