6L−3

マルチプロセッサワークステーション "Stonehigh" — 機能分散型OSの設計と実現 —

鈴木茂夫　宮本剛　伊達厚　岩本信一　柴山茂樹
キヤノン（株）情報システム研究所

1. はじめに

キヤノン情報システム研究所で試作されたワークステーションのハードウェア（Stonehigh）は，最大4個のメインCPUからなる密結合マルチプロセッサシステムであり，機能分散の一環としてIOC（I／O Channel）を持つ構成になっている．我々は，この Stonehigh上に，カーネギーメロン大学で開発されたMach2.5をベースとしたStonehigh-OSの構築を行なった．Stonehigh-OSは，メインCPU上で動作するMachカーネル部と，IOC上で動作し各種I／O装置の制御を行なうIOCS（IOC Control System）からなる．

本報告では，Machカーネル部とIOCSのインタフェース，IOCSの内部構成およびIOCSの機能拡張について述べる．

2. Stonehigh-OSの構成

Stonehighは，メインCPUとしてモトローラ社のMC68040，IOC，そして64MBytesのメインメモリ（メインCPU，IOC両方からアクセス可能）を持つ構成になっている．IOCは，各種I／O装置（ハードディスク装置，LANなど）のコントローラ，そしてそれらを制御するプロセッサであるモトローラ社のMC68030，高速な4MBytesのローカルメモリからなる．

Stonehigh-OSは，上記のような構成のStonehigh上で動作するOSであり，メインCPU上で動作するMachカーネル部とIOC上で動作するIOCSから構成される．このような構成をとることで，従来OS内部のデバイスドライバが担当していたI／O処理の多くの部分を，IOCSに代行させた機能分散が可能となり，メインCPUの負荷を軽減することができる．

また，IOCのプロセッサであるMC68030の性能，そして高速なローカルメモリを活用することで，メインCPUとは独立に，IOC内部でI／O処理の性能向上を図ることができる．例えば，IOCS内部にディスクキャッシュ機構などを実現することで，ハードディスク装置（HD）に対するI／O処理の性能向上が可能である．

3. Machカーネル部とIOCSのインタフェース

(1) I／O要求の発行方法

MachカーネルからIOCSへのI／O要求の発行，そしてIOCSからMachカーネル部への処理結果の返答は，次に述べるIOCプロトコルにしたがって行なわれる．IOCプロトコルは，CPU間割込み機能とメインメモリを利用した非同期方式による通信である（図1参照）．これは，Machカーネル部内部のデバイスドライバの構造に合わせた結果の仕様である．

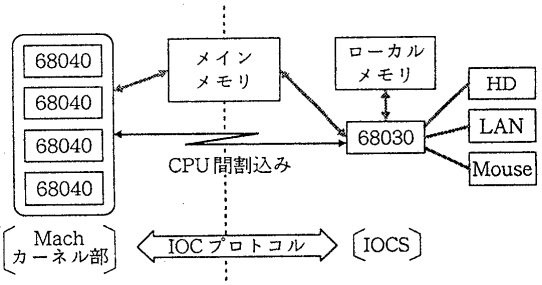

図1 Machカーネル部とIOCSのインタフェース

(2) コマンド領域とステータス領域

I／O要求発行時にMachカーネル部がI／O要求に必要な情報を伝えるための領域をコマンド領域と呼び，I／O処理結果の返答時にI／O処理結果の情報を読み出すための領域をステータス領域と呼ぶ．これらの領域は，メインメモリ上の固定領域に置かれ，それぞれ以下のような情報が格納される．

[コマンド領域]
- I／O装置のID
- 命令コード（PROBE, READ, WRITE, IOCTL）
- 2次記憶上の領域（ブロックアドレス，サイズ）
- データ転送領域（メインメモリ上の領域）

[ステータス領域]
- I／O装置のID

Multiprocessor Workstation "Stonehigh" — Design and Implementation of OS —
S. SUZUKI, T. MIYAMOTO, A. DATE, S. IWAMOTO and S. SHIBAYAMA
Information Systems Research Center, Canon Inc.

・命令コード（PROBE, READ, WRITE, IOCTL）
・状態コード（正常終了コード, エラー終了コード）

(3) データ転送領域の指定方法

Stonehigh-OSは仮想記憶OSである．したがって，I／O処理時のデータ転送領域も仮想空間で管理されることになる．このとき，データ領域は仮想空間では連続した領域であっても，物理空間では複数の分割された領域になっている可能性がある．IOCSに対する1回のI／O要求で，これら分割された領域へのデータ転送を可能にするため，それぞれの領域を指定する管理ブロックを任意個チェーンした構造でデータ転送領域の指定を行なうことにした．

4. IOCSの内部構成

IOCSは次の2種類のソフトウェアモジュールから構成される．

(1) IOCカーネル

マルチタスク機能を司るモジュールで，タスクのスケジューリング，I／O要求のキューイング処理などを行なう．

(2) ドライバタスク

IOCカーネルの制御下でタスクとして動作し，各種I／O装置の制御を行なう．特別なタスクとして，IOCプロトコルにしたがってMachカーネル部からのI／O要求を解釈するタスク（コマンド解釈タスク），Machカーネル部に各I／O処理結果の返答を行なうタスク（処理結果返答タスク）なども動作している．

ドライバタスクは，無限ループ構造をとり，自タスクが担当するイベント（I／O処理要求，I／O装置からの割込み）が発生するまで，同期プリミティブであるabandon_cpu（IOCカーネルが提供）を呼び出しSUSPEND状態となる．対応するイベントが発生すると，タスクはCPUが割り当てられ，自タスクが担当するI／O処理を行なう．それが終了すると処理結果返答タスクにI／O処理結果の返答を要求し，その後再びabandon_cpuを呼び出し，次にイベントが発生するまでSUSPEND状態となる．

5. IOCSの機能拡張

現在，IOCSは5種（キーボード，マウス，HD，LAN，シリアルポート）のI／O装置の制御を行なっている．これらのうち，HDに関しては，ローカルメモリ（約3.4MBytes）を利用してIOCS内部にディスクキャッシュ機構を実現し，I／O処理の性能向上を試みた．

ディスクキャッシュ機構の設計の際，Machカーネル部が有するキャッシング機構との役割分担を考慮して，キャッシング方式の検討を行なった．Machカーネル部には，約3.2MBytes（主記憶全体の容量である64MBytesの5％）のバッファキャッシュと，実行プログラムをキャッシングするオブジェクトキャッシュ機構がある．したがって，Machカーネル部のバッファキャッシュとほぼ同容量のメモリしか持たないIOCSで，リード処理時にキャッシングを行なっても大きな効果は期待できない．それよりも，ライト処理時のコピーバック方式によるキャッシング，そして先読み処理を実現した方がより大きな効果が期待できる．そこで，次のような方式でディスクキャッシュ機構を実現した．

・ライト処理時のキャッシング：コピーバック方式
・キャッシュ全体サイズ：約3.4MBytes
・キャッシュブロックサイズ：8KBytes
・先読みキャッシュサイズ：1キャッシュブロック

特性の異なる数種類のアプリケーションプログラム（コンパイル処理，ファイルのリモートコピー処理，ファイルのコピー処理など）を動作させて性能測定を行なった結果，アプリケーションの特性の違いによってキャッシュの効果が異なるものの，リード／ライト処理ともに最大で約6倍処理速度が向上することがわかった．例えば，ディスクのリード／ライト処理が全体処理時間の約20％を占めているコンパイル処理を例にすると，ディスクキャッシュ機構を導入したことにより，全体の処理時間で約10％短縮することができた．

6. おわりに

Machカーネル部とIOCSの2つのモジュールから構成され機能分散を指向したStonehigh-OSを設計，実現した．現在，研究業務のプラットフォームとして使用している．また，Machカーネル部内部のデバイスドライバとのマッチングを考慮に入れてMachカーネル部とIOCS間のインタフェースであるIOCプロトコルを設計，実現した．IOCSの機能拡張としてディスクキャッシュ機構を実現したことにより，メインCPUに負荷を与えることなくHDに対するI／O処理の性能を向上できた．

参考文献

[1] 伊達 他，「マルチプロセッサワークステーション“Stonehigh”―コンセプトとハードウェア概要―」，第45回情報処理学会全国大会，6L-02，1992