2N-5

オペレーティングシステム ＯＭＮＩの概要

飯間　豊　　宮脇　誠　　岸田　一

（沖電気工業(株)　コンピュータシステム開発本部）

１．はじめに

ＯＭＮＩは、我々が試作中の密結合マルチプロセッサ用オペレーティングシステムである。仮想マシンを提供するカーネルとＯＳインタフェースを提供するサービスの２階層より構成される。複数スーパバイザの搭載により異なるＯＳインタフェースを同時に提供できる。オブジェクト指向概念により設計され、記述言語にはＣ＋＋[1]を用いた。現在、密結合マルチプロセッサシステムＯＲＩＯＮ[2]上にＵＮＩＸ*インタフェースを実現中である。

本稿ではＯＭＮＩの概要について述べる。

２．ソフトウェア構成

ＯＭＮＩの構成を図１に示す。ＯＭＮＩはカーネルおよびサービスの２層より構成される。カーネルはマシンアーキテクチャを隠蔽し、プロセッサ台数、実装メモリ量といったハードウェア的制限を取り除いた仮想マシンをサービスに提供することを目的とし、プロセッサ管理、メモリ管理および入出力管理より構成される。サービスは仮想マシン操作により、アプリケーションプログラム（ＡＰ）にＯＳインタフェースを提供することを目的とし、スーパバイザとファイルシステムより構成される。複数のスーパバイザを搭載可能である。

カーネル／サービス間のインタフェースは、カーネルが提供するプリミティブ関数およびサービスへの仮想割込みによって規定される。カーネルとサービスとは異なるモードで走行する独立してたソフトウェアである。カーネルはサービスから保護されており、サービスはソフトウェア例外によるカーネルプリミティブ呼出しのみによりカーネルを操作できる。

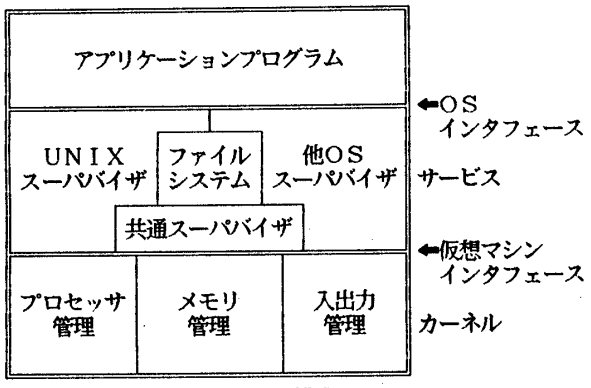

図1　ソフトウェア構成

An Overview of the Operating System OMNI
Yutaka IIMA, Makoto MIYAWAKI, Hajime KISHIDA
OKI Electric Industry Co.,Ltd.

２．１　カーネル

カーネルはプロセッサ管理[3]、メモリ管理[4]および入出力管理[5]より構成され、それぞれが実制御部と仮想制御部とに分割される。実制御部はハードウェアの制御を行い、仮想管理部は仮想マシンの実現を担当する。

カーネルが提供する主な仮想マシンは以下のとおりである。

- 仮想プロセッサ
  プログラムの実行主体であるプロセッサを仮想化したものである。仮想プロセッサ個数は実プロセッサ台数に依存しない。
- 仮想メモリ
  仮想プロセッサ単位の多重論理空間、セグメント等を提供する。仮想記憶管理もカーネルにおいて実現する。
- 仮想デバイス
  実デバイスの一部（例えばディスクの１パーティション）や複数デバイスの結合（例えばマルチボリュームディスク）を１つのデバイスであるかのように見せかけたデバイス。　ハードディスク、フロッピディスク、ストリーマテープ、端末およびプリンタをサポートする。

２．２　サービス

サービスは仮想マシン上にＯＳインタフェースを実現するものであり、共通スーパバイザ、特定ＯＳ用スーパバイザおよびファイルシステムより構成される。複数の特定ＯＳ用スーパバイザを搭載することにより、複数のＯＳインタフェースを同時に提供可能である。複数ＯＳ実現のため、ＯＭＮＩではサービス内共通機能を共通スーパバイザにより定義した。すべての特定ＯＳ用スーパバイザは、共通スーパバイザを利用して構成される。また、ファイルシステムは共通スーパバイザとのみ通信を行う。これにより、１つのファイルシステムを共有して、複数のスーパバイザを搭載することが可能となる。サービスは仮想プロセッサ上で動作し、仮想マシンのみを操作する。

現在、特定ＯＳ用スーパバイザとしては、ＵＮＩＸスーパバイザ[6]を試作中である。Ｃ＋＋によりすべて新規にコーディングし、マルチプロセッサ対応のため、サービスオブジェクトにセマフォによる排他制御機能を付加するといった機能追加を行った。

３．メモリ構成

ＯＭＮＩを搭載するＯＲＩＯＮのマシンアーキテクチャを図２に示す。

ＯＭＮＩは、実装プロセッサ台数の増加を目標としている。プロセッサ台数を増加した時、共有バスの競合が

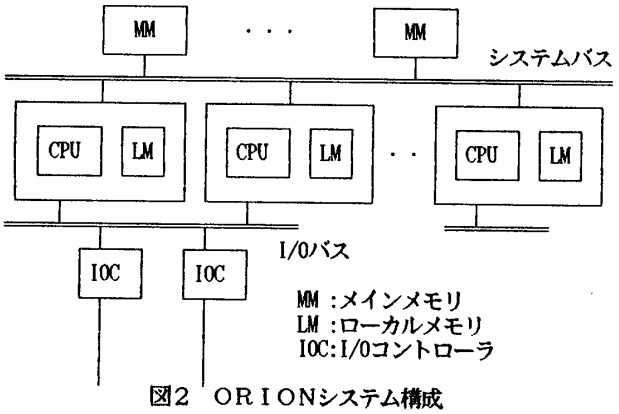

図2　ＯＲＩＯＮシステム構成

システム性能上のネックとなる。ＯＭＮＩでは、仮想プロセッサ私有資源をＬＭに配置することでＭＭアクセスを削減し、１２台程度までのプロセッサが実装可能である。具体的資源配置は以下のとおりである。

- ＭＭ
  - ・ＯＳオブジェクト
  - ・共有メモリ
- ＬＭ
  - ・ＯＳのテキスト、スタック
  - ・ＡＰのテキスト、データ、スタック

サービスおよびＡＰはカーネルが提供する仮想プロセッサ上で動作する。各仮想プロセッサは固有な論理空間を持つ。論理空間のメモリマップを図３に示す。

論理空間中の下位２Ｇバイト空間は各論理空間で共有されるシステム共有空間であり、上位２Ｇバイトは仮想プロセッサごとに異なる私有空間である。ＯＳはシステム共有空間に、ＡＰは仮想プロセッサ固有空間にマッピングされる。ただし、ＯＳのスタック及び一部のデータは仮想プロセッサ固有空間にマッピングされる。

ＡＰ等がＬＭに配置されるため、仮想プロセッサは特定の実プロセッサ上でのみ実行可能である。ロードバランシングのため、仮想プロセッサを他実プロセッサへ移動する必要がある。ＯＭＮＩでは仮想プロセッサ私有空間のスワッピングにより仮想プロセッサの移動を実現している。

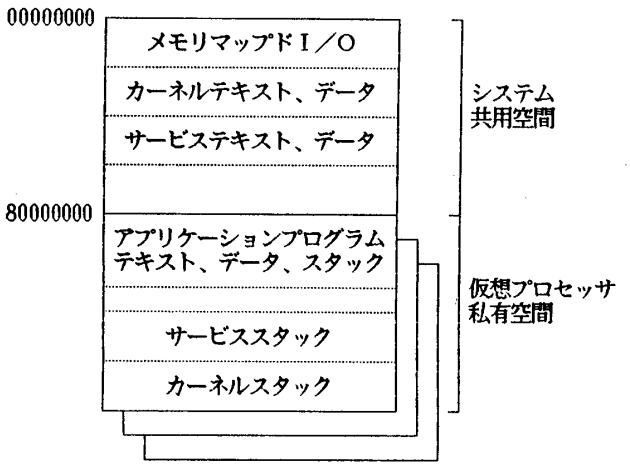

図3　論理空間マップ

４．オブジェクト指向

ＯＭＮＩはカーネル、サービスともにオブジェクト指向概念により設計された。オブジェクト指向概念導入の目的は、理解しやすく、保守性の優れたシステムを構築することにある。オブジェクトの集合としてシステムを記述することは、各モジュールの独立性を高め、モジュール間インタフェースの明確化に役立った。また、システム中のマシン依存部をいくつかのクラスに封じ込めることにより、移植性の向上にも効果的である。

カーネルにおいては、ハードウェアを直接制御する実管理部と、仮想マシンを提供する仮想管理部とを別クラスとして設計し、仮想管理クラスが実管理クラスを継承するというクラス構成をとっている。カーネルの実管理クラスの書き直しにより、ＯＭＮＩを他マシンに移植可能である。また、サービスにおいてはサービス内共通機能を提供する共通スーパバイザをミックスインクラスの集合として定義し、特定ＯＳ用スーパバイザはこれらミックスインクラスを継承するという方式をとった。

記述言語にはＣ＋＋を採用した。Ｃ＋＋は効率的にオブジェクト指向プログラミングを可能とする言語であり、オブジェクト指向的設計の直接的実現に有用である。また、厳密な型チェックにより、コンパイル時に多くのエラーを発見できることもＣ＋＋の長所である。Ｃ＋＋の言語仕様上の問題点として以下の２点が挙げられる。一点は、多重継承が許されないことである。システム設計の初期段階においては、特定の言語に依存しない設計を行ったため、Ｃ＋＋でのコーディング時にクラス構成の変更が必要となった。もう一点はクラスオブジェクトの欠如である。ＯＭＮＩにおいては、同一クラスから生成されるオブジェクトを管理する別のオブジェクトが必要となった。これを解決するため、我々はインスタンスオブジェクト管理専用の別個のクラスを定義した。

５．おわりに

本論文ではオペレーティングシステムＯＭＮＩの概要について述べた。ＯＭＮＩのＯＲＩＯＮ上での試作はほぼ終了し、今後ハードウェアを含めたシステム性能の評価を行う予定である。

参考文献

[1] Stroustrup, "The C++ Programming Language", Addison Wesley, 1986
[2] 阿部,他,"ＯＲＩＯＮアーキテクチャの概要", 情報処理学会第３７回全国大会　２Ｎ－５
[3] 長谷部,他,"オペレーティングシステムＯＭＮＩのプロセッサ管理",情報処理学会第３８回全国大会
[4] 矢野,他,"オペレーティングシステムＯＭＮＩのメモリ管理",情報処理学会第３８回全国大会
[5] 竹元,他,"オペレーティングシステムＯＭＮＩの入出力管理",情報処理学会第３８回全国大会
[6] 竹内,他,"オペレーティングシステムＯＭＮＩのＵＮＩＸスーパバイザ",情報処理学会第３８回全国大会

＊ ＵＮＩＸはＡＴ＆Ｔが開発したＯＳです。