マルチメディア通信と分散処理 31 － 3
（１９８６. ９. ２９）

マルチプロセサ指向高機能リアルタイムＯＳの一構成法

横畑　静生*　　青木　久延**　　杉田　由美子*

*（株）日立製作所　システム開発研究所
** 日立マイクロコンピュータエンジニアリング（株）

プロセス制御から会話処理用まで広範囲の分野に適用できるマルチプロセサ指向のモジュラー型オペレーティングシステム（ＯＳ）の基本部を開発した。本ＯＳはリアルタイム・カーネルを中核として、会話処理、バッチ処理機能等の多くの機能を機能モジュールとして構成することにより、ユーザ・システムの要求に従って再構成できる。タスク間は基本的にはメッセージ通信機能により結合されており、その機能をプロセサ間に拡張することにより密結合のプロセサ間にまたがるタスク間通信も実現している。また将来のサーバーによる機能拡張に備えて、デバイスおよびソフトウエア割込みに対応するサーバ定義機能を実現している。本論文では、本ＯＳの構造、基本機能、タスクの実行環境について述べる。

A Multi-processor Oriented High Level Function Realtime OS

Shizuo YOKOHATA*　　Hisanobu AOKI**　　Yumiko SUGITA*

* Systems Development Laboratory, Hitachi, Ltd.
1099 OHZENJI ASAO-KU KAWASAKI-SHI, 215 JAPAN
** Hitachi Microcomputer Engineering Co.,Ltd.
1479 JYOUSUIHONMACHI KODAIRA-SHI, 187 JAPAN

A multi-processor oriented module-structured operating system(OS) has been developed. It has been designed to be used in wide variety of microcomputer applications such as process control systems, interactive systems. It has a compact real-time kernel. Many functions such as interactive processing and batch processing are implemented as functional modules under the kernel, so that it can be re-configured according to the user's system requirement. Inter task communication is performed using message transfer method. Communications between tasks on tightly-coupled multi-processor is also implemented using enhanced message transfer method. Prepared for a future extention, it has server-definition-function for software interrupts and devices. This paper describes structure, organization, basic function of this OS and executing environment for a task.

１．はじめに

制御分野におけるリアルタイムＯＳに要求される機能は、従来の制御機能だけのものから最近ではさらにデータの蓄積、加工といった情報処理機能や、プログラム開発機能まで広範囲にわたり多様化してきた。一方マイクロコンも１６ビット、３２ビットと急速な発展をし処理能力も飛躍的に向上した。しかしさらに速い応答性や処理能力を要求するシステムに対して、マルチプロセサ構成により対応できるＯＳが重要になりつつある。今回、機器組込から制御とプログラム開発が同時に行える汎用システムまで、モジュール選択により広範囲な用途に適用できるモジュラー型マイクロコン用マルチプロセサ指向高機能リアルタイムＯＳの基本部を開発したので報告する。

２．開発の狙い

ＯＳ開発の狙いは以下の通りである。

(１)制御分野、情報処理分野の両分野に適応できる統一思想を持つＯＳとする。

制御分野は、単一の機器を制御するコンピュータ群を別のコンピュータと接続してシステム全体を制御するＦＡ(Factory Automation)化の時代に入ってきた。この分野で今後重要になるのはネットワークを通じての機器の制御、情報の収集、蓄積、加工機能、最終的に出力された情報を見て判断する人間に分かり易い情報を提供する入出力機能であると考えられる。

また情報処理の分野は、スタンドアロンのワークステーションをＬＡＮ(Local Area Network)で結んだ水平分散、メインフレームと接続した垂直分散へと発展してきた。またこの分野は、ユーザとの直接的なインターフェイスが基本と成るため、ユーザにとってさらに使い勝手の良いマンマシン・システムおよびネットワークシステムが今後とも発展してくると思われる。

これから分かるように制御分野、情報処理分野共に今後増々重要になってくるのは、アプリケーションによる機能も含めた高度情報処理機能である。情報処理と制御の両分野に適用できる高度情報処理機能を持つリアルタイムＯＳとする。

(２)リアルタイム制御機能と高度情報処理機能を併せ持ち、アプリケーションに応じて最適なＯＳを構成できるモジュラー型ＯＳとする。

前述したように制御分野、情報処理分野共に高度情報処理機能が必要である。しかしユーザシステムが必要とする機能は多様であり、常にＯＳの全機能を常備したシステムではシステムに無駄が多くなり、性能にも影響する。このことからシステム全体をモジュラー構造として、各ユーザ・システムに必要な機能だけを選択してユーザシステムに最適なＯＳを構成できるＯＳとする。またそのＯＳのカーネルは、リアルタイム処理機能および性能を持つものとする。

(３)マイクロコン用他ＯＳのアプリケーションの実行環境を提供できるＯＳとする。

マイクロコンの分野においては流通ＯＳが広く使われており、それらＯＳ用に多数のベンダーからアプリケーション・プログラムが供給されている。それらのアプリケーションが実行できることは、ユーザにとって多大なメリットがあることから、本ＯＳでは広く利用されている他ＯＳのインターフェイスを提供する。

(４)マルチプロセサ構成に対応できるＯＳとする。

制御分野、情報処理分野共に今後は扱うデータ量が飛躍的に増大して来ると思われる。それに伴いファイルシステムやデータベースといった機能を別プロセサ(密結合)で実行することにより性能を向上させたいという要求が高まってくるものと思われる。またマンマシン・システムにおいては、会話処理のレスポンスを一定時間内に保ために特定の処理を別プロセサで実行したり、制御システムにおいては特定タスクのレスポンス確保のために別プロセサで実行できる様なマルチプロセサ対応のＯＳとする。

３．ＯＳの構成と特徴

本ＯＳの特徴は次に示す通りである。

(１)機能の取外しが容易なモジュラー構造

リアルタイムＯＳを利用するシステムの形態も機器組込形のシステムから会話処理によるプログ

ラミング機能を必要とするシステムまで多岐にわたる、そしてそれらのＯＳに対する性能，機能および容量などへの要求も多様である。そこで本ＯＳはＯＳ自身をモジュール構造として構成し、必要なモジュールの選択により各ユーザシステムに最適なＯＳの構築が可能なビルディング・ブロック方式となっている。

(２)高速リアルタイム性

リアルタイム環境下のアプリケーションの実行に十分に対応できる応答性を持つリアルタイム・カーネルを核としたＯＳ構造になっている。

(３)マルチプロセサ対応

アプリケーション負荷の増大および，データーベース，ファイルシステム等のシステム・プログラムの負荷の増大に容易に対応できるように、本ＯＳではバス結合方式のマルチプロセサ構成に対応できる機能をもっている。

(４)高度情報処理機能

前述したように制御分野及び情報処理分野の両方において情報処理機能が今後増々重要になることから、本ＯＳでは，データベース、グラフィックといった高度情報処理機能を装備している。

(５)マイクロコン用他ＯＳインターフェイス

マイクロコン用ＯＳとして事実上の標準ＯＳであるいくつかの流通ＯＳに対して、それらのＯＳ用アプリケーションを実行できる他ＯＳインターフェイス機能を持っている。これにより流通ＯＳの多様なアプリケーションを有効活用できる。

(６)マルチユーザＴＳＳ環境

制御を行いながらデータを編集、加工するアプリケーションを実行したり、プログラムを開発したりする機能へのニーズが高いことから、本ＯＳでは制御を行いながらマルチユーザＴＳＳ環境を同時にサポートする。

(７)ユーザ・フレンドリ・コマンドシステム

ユーザの会話処理を支援するためにユーザ・フレンドリなコマンド・システムを提供する。

図１に本ＯＳの全体構成を示す。ＯＳは次に示すように次の５つのブロックに大別される。図における論理Ｉ/Ｆ部、物理Ｉ/Ｆ部、ゲストＯＳコネクタ部、コマンド・ユーティリティ部に含まれる機能はタスクとして構成され、カーネルとの間および相互間はすべてカーネルのメッセージ機能により結合されている。

(１)カーネル部

カーネル部は本ＯＳの中心となるブロックで、リアルタイム制御機能を提供する。この部分はハードウエアとの関連が一番強い部分であるため、移植性向上を目的としてハードウエアに関連するモジュールを機種依存部として別モジュールに分離している。

カーネル部は、マルチタスキング用基本機能の他に、デバイス管理機能をもつ。デバイス管理機能は、デバイスの排他制御を行ったりデバイスの状態(オンライン／オフライン)変化の処理等を行う。

(２)物理Ｉ/Ｆ部

物理Ｉ/Ｆ(インターフェイス)部は、デバイス・ドライバ群からなる。デバイス・ドライバは、各入出力デバイスに依存したデバイスとの間のフィジカル入出力を行

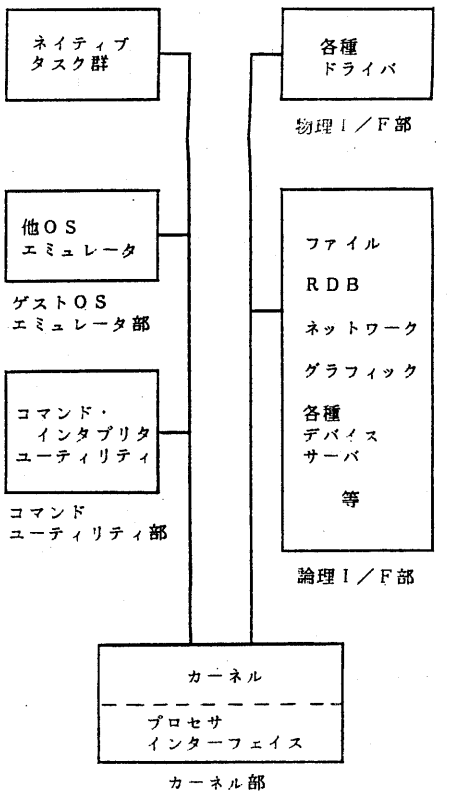

図１　全体構成

う。これらのドライバは手続き型及びタスク型の両形式で作成することが可能である。

(3)論理I/F部

論理I/F(インターフェイス)部は、ファイルシステム，データベース，ネットワーク等の機能モジュールから成る。この部分はハードウェアから独立させるために、入出力はすべて物理I/F部を経由して行う。これらの論理モジュールはタスクにより構成され、論理モジュール間およびカーネルとの情報交換はすべてメッセージ通信により行われる。またこれらサーバを必要に応じて容易に追加できる機能を持つ。

(4)コマンド・ユーティリティ部

ユーザの入力したコマンドを解析実行するコマンド・インタプリタと、ユーザのプログラム開発を支援する各種ユーティリティ群から成る。コマンド・インタプリタには、コマンドの実行を制御するコマンド制御言語機能、実行したコマンド履歴を記憶しておき再利用できる機能、入力コマンドおよび再利用コマンドの編集機能といったユーザにとって使い易い機能を有している。

4．利用形態

本OSの適用分野としては、リアルタイム処理の分野とワークステーション等の情報処理の分野がある。基本となる利用形態は、モジュール選択に応じて構成できる次の3つである。図2参照。

(1)リアルタイム核モード

リアルタイム核だけから成る基本システム。計測機器やマシン制御の分野で組み込み型として使われる。

(2)リアルタイム核＋ファイルモード

リアルタイム核とファイルシステムから成り、データの蓄積・加工といった情報処理を含むリアルタイム処理が適用分野で、プロセス制御、簡易情報処理システムに使われる。また場合によっては、さらにデータベースを含めた構成とし、より高度な処理を行うことも可能である。

(3)フルシステムモード

フル機能から構成されるシステム。情報処理の分野では、リアルタイム処理カーネルを中心として、ネットワーク機能、データベース機能、マルチウインドウ機能等のマンマシン機能を使ってOA用ワークステーション、エンジニアリング用ワークステーションに適用できる。制御の分野では、

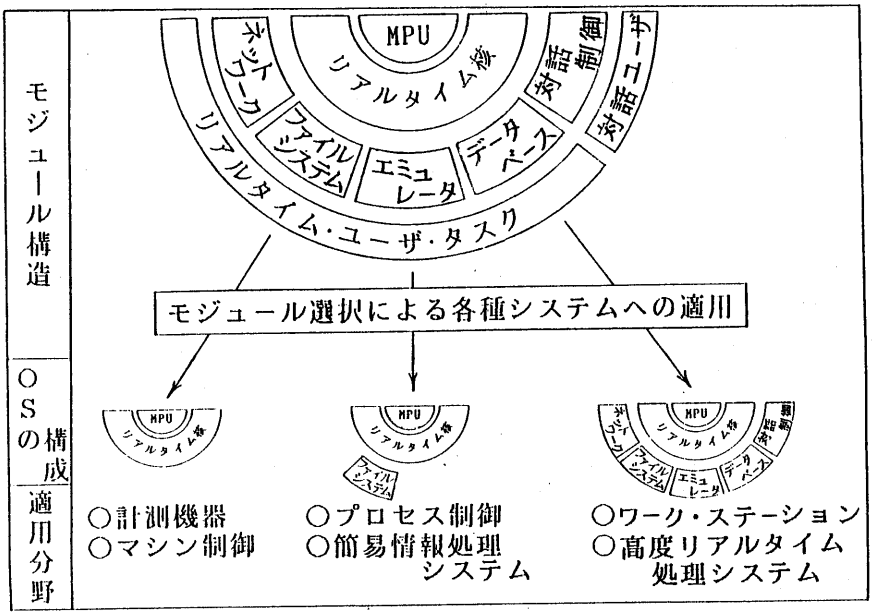

図2　システム構成と
各種システムへの適用

制御を行いながら収集した情報を分析、加工したり、プログラム開発を行う高度リアルタイム処理システムに適用できる。

また上記基本利用形態の他に、ユーザが図１に示すモジュールを自由に選択することにより、自システムに最適なＯＳを構築することができる。必要によってはマルチプロセサ構成を取ることで、さらに高性能のシステムを構築することも可能である。

５．カーネル機能

５．１　タスク実行環境

図３にタスクの実行環境を示す。実行環境は、システム、リアルタイム処理、会話処理、バッチ処理の４つの環境(エンバイロメント)に分類されている。内部的には、エンバイロメントはタスクの集合(トリー構造)で構成されており、各エンバイロメントはユニークな識別子(エンバイロメントＩＤ)により区別される。

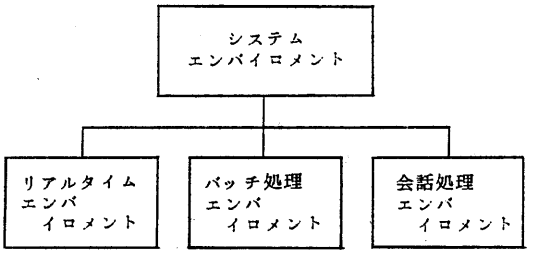

図３　実行環境とエンバイロメント

(１)システム・エンバイロメント

本エンバイロメント内には、図１の全体構成に示した論理Ｉ/Ｆ部、物理Ｉ/Ｆ部のタスクが置かれ実行される。すべてのシステム・タスクは、このエンバイロンメント内で実行される。ユーザシステムに必要な機能のシステム・タスクだけをエディション・テーブルと呼ばれるテーブルに登録しておくことで、それらのシステム・タスクだけがこのエンバイロメント内で実行される。

(２)リアルタイム処理エンバイロメント

本エンバイロメント内には、ユーザが作成したリアルタイム処理タスク群が置かれ実行される。このタスク群は、システム・エンバイロメントと同様にエディション・テーブルに登録しておくことにより自動的に生成される。

(３)会話処理エンバイロメント

入出力用の端末が割り当てられたエンバイロメント(セッション)で、本エンバイロメントはユーザによる端末からのログイン操作により生成される。複数ユーザがログイン操作を行なうと、独立したエンバイロメントとして複数の会話処理エンバイロメントが生成される。

(４)バッチ処理エンバイロメント

ＳＵＢＭＩＴコマンド・ファイルから入力を行ない、出力をリスティング・ファイルに行なうエンバイロメントで、ユーザがＳＵＢＭＩＴコマンドを端末から入力することにより、本エンバイロメント内で実行されるタスク(バッチ・ジョブ)が生成される。

システム起動時には、初期エンバイロメントとしてシステム・エンバイロメントが生成され、その中の初期タスクとしてシステムタスク・マネジャが生成される。システムタスク・マネジャは、エディション・テーブルに登録してある全システムタスクを生成し、その後リアルタイム処理エンバイロメントとバッチ処理エンバイロメントを生成し、各エンバイロメントの初期タスクを生成する。リアルタイム処理エンバイロメント内には、初期タスクとしてリアルタイム・マネジャが生成され、リアルタイム・マネジャはユーザが登録している全リアルタイムタスクを生成する。バッチ処理エンバイロメント内には初期タスクとしてバッチ・マネジャが生成され、バッチ・マネジャはＳＵＢＭＩＴコマンドによりバッチジョブが入力されるとバッチジョブを起動する。

５．２　タスクのプロテクトとタスク空間

本ＯＳではタスク空間を考えるにあたり次の３つのシステム構成を考えている。

(１)メモリ保護機構なしシステム

(２)メモリ保護機構ありシステム

(３)仮想記憶サポート・システム

現システムでは仮想記憶はまだサポートしていない。以下メモリ保護機構ありシステムにおけるタスク空間の設定とプロテクトについて述べる。

ここで述べるメモリ保護機構(ＭＭＵ)とは、①メモリの保護機能、②論理空間と物理空間の分離機能、の二つの機能を持つものを指す。そしてメモリ保護機構ありシステムとは、その機能のうちの一つであるメモリ保護機能だけを使ってタスクのプロテクトのみを行なうシステムである。

基本的にはタスクはＭＭＵを使って個別にプロテクトされる。タスクがディスパッチされる時、タスクのコード、データ、スタックおよびタスクが使用しているプール領域がＭＭＵによりプロテクトされる。

会話処理、バッチ処理エンバイロメント内タスクは個別にプロテクトされる。リアルタイム処理エンバイロメント内常駐タスク(常にメモリ上に存在するタスク)は、タスク生成時にタスクの存在する空間を指定することにより空間毎にプロテクトされる。この空間をタスク毎に別々に指定することによりタスク毎プロテクトが可能であり、また複数タスクを１つの空間内に定義することにより空間毎のタスクのグループ化が可能である。システム・エンバイロメント内タスクは、カーネル以外の全エリア(システム、リアルタイム処理、バッチ処理、会話処理の全エンバイロメント)を含めた領域としてプロテクトされる。

１つの空間は複数のセグメントにより構成され、各セグメントは用途に応じてＲｅａｄ、Ｗｒｉｔｅ、Ｒｅａｄ／Ｗｒｉｔｅの属性をつけることができる。これらのセグメントの定義および空間とセグメントの対応づけは、ユーザがエディション・テーブルに登録することにより行なわれる。空間当たりのセグメント数はハードウェアに依存する。

５．３　システム・コール

図４にタスクの状態遷移を示す。タスクの状態としては、Non-Existent(タスクが存在しない状態)、Dormant(実行禁止状態)、Idle(起動待ち状態)、Suspend(実行中断状態)、Wait(イベント待ち

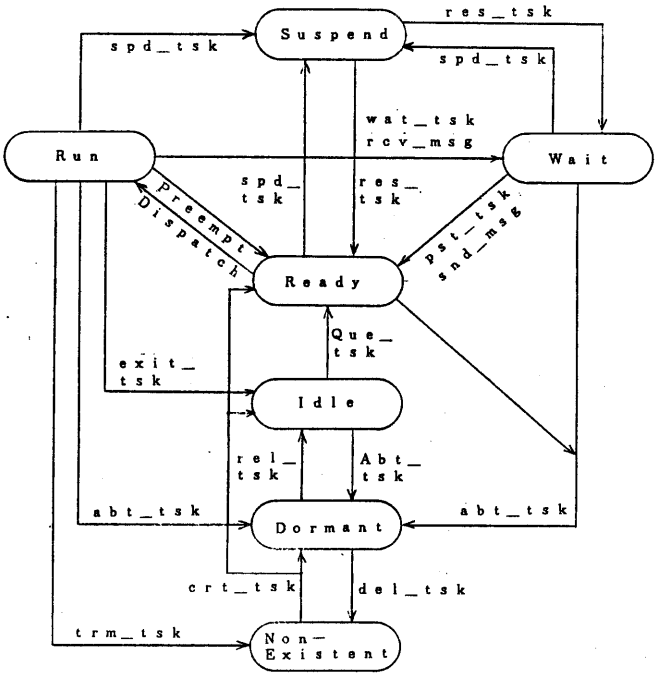

図４　タスク状態遷移図

状態)、Run(実行中状態)の７つがある。

システムおよびリアルタイム処理エンバイロメント内タスクは上記７つの状態を使うことができるが、会話処理エンバイロメントおよびバッチ処理エンバイロメント内のユーザタスクにとっては、Dormant、Idle状態は必要ないことから、Ready、Run、Wait、Runの状態だけが使われる。すなわちタスク起動はタスク生成用システムコール(crt_tsk)によりNon-Existent状態からReady状態にされ、終了は終了用システムコール(trm_tsk)によりRun状態からNon-Existent状態にされる。

表１にカーネル・システムコール一覧を示す。カーネル機能としてはエンバイロメント管理、タスク管理、同期管理、資源管理、メモリ管理、時計管理などの機能がある。

リアルタイム処理タスクと会話処理タスクを同時実行するＯＳでは、システムの信頼性を高めることが重要である。そのために一つの方法としてプリビレッジによるタスク権限のチェックを行なう。基本的にはタスクは自タスクがどのエンバイロメントに属しているかによって発行できるシステムコールが制限される。

５．４　マルチプロセサ機能

カーネルはバス結合マルチプロセサ対応の機能を持っている。メモリ方式は共有メモリだけを使うグローバルメモリ方式と、ローカルメモリだけまたはローカルメモリとグローバルメモリ両方を持つローカルメモリ方式の両方をサポートしている（図５）。ローカルメモリ方式におけるアドレス空間と実メモリのマッピングは図６のようになっている。すなわちアクセスした番地に応じて、そこにマップされているローカルメモリの内容がアクセスされる。

本ＯＳにおけるマルチプロセサ対応機能は次の２つである。

(１)ＯＳ機能分散

これはファイルシステム、データベースといったシステム機能の一部を別プロセサで実行することにより負荷分散を行なうものである。先に述べたように、これらのシステム機能はカーネルのメッセージ通信機能により結合されており、このメッセージ通信機能をバス結合のＣＰＵ間に拡張することによりおこなっている。メッセージ内容はシングル・プロセサと同一である。（図７）

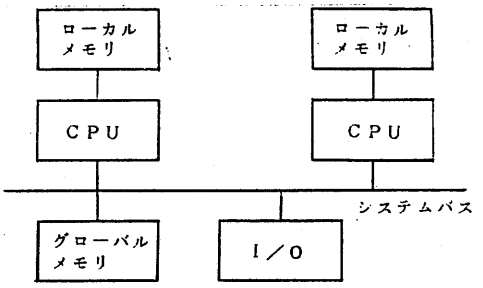

図５　ローカルメモリ方式

(２)アプリケーションの分散

これは特定アプリケーションを別プロセサで実行することにより、一定の応答性を確保したり負荷分散を行なったりするものである。本ＯＳではこの実現のために、タスクと実行プロセサを対応づけるプロセサ選択マスクによる方式を用いた。プロセサ選択マスクはタスク毎に存在し、そのタスクを実行できるプロセサのマスクを表している。各プロセサのスケジュラは、次に実行するタスクをサーチする時にこのプロセサ選択マスクを参照して、自プロセサで実行できるタスクだけを取り出して実行する。

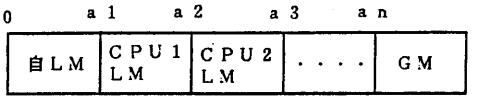

図６　ローカルメモリ方式のアドレスマップ

６．ファイル機能

６．１　特徴

本ファイルシステムの特徴を次に示す。

(１)デバイス毎にサーバを定義できるデバイス・サーバ機能

(２)固定長レコード、可変長レコードによるレコード編成、バイトストリームによるストリーム編成が

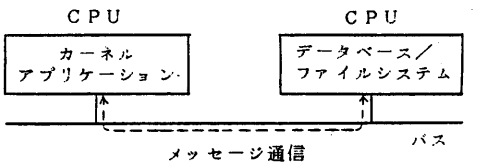

図７　バス結合によるマルチプロセサ化

可能
(3)高速アクセス用の連続ディスクブロック・アロケーション機能
(4)主記憶を仮想デバイスとする高速ファイル処理(RAMファイル)が可能
(5)階層ディレクトリ構造
(6)ファイル、レコード単位での排他制御が可能

本システムのファイル機能としては、上記ファイルシステム以外にリレーショナル・データベースがある。その特徴を次に示す。
(1)データベースはファイルシステムの上に実現されており、ファイルシステムと親和性が良い。
(2)データベースが扱うメディアとして、フロッピーディスク、ハードディスクが可能。
(3)関連演算として、射影、選択、結合をサポートしている。

6.2 ファイル編成法

本OSのファイル編成法としては、ディスクファイル用として連続ファイル、順編成ファイル、バイト・ストリーム・ファイルの3種と、CRT、プリンタ等のデバイスをファイルとして扱うデバイス・ファイルの計4種がある。表2にファイルシステムのシステムコール一覧を示す。

(1)連続ファイル

ロードモジュール用のファイルで、ファイルはディスク上の連続した領域に確保される。このサイズはファイル作成時に決められ拡張できない。アクセスは論理ブロック単位に行われ、シーケンシャルと論理ブロック番号によるランダムアクセスが可能。

(2)順編成ファイル

ファイルはレコードの集合として構成され、ファイルは自動拡張されるが物理的には連続しない。アクセスはレコード単位または論理ブロック単位でのシーケンシャルまたはランダムアクセスが可能。

(3)バイト・ストリーム・ファイル

レコード形式を持たないバイト列から成るファイルで、ファイルは自動拡張されるが物理的には連続しない。シーケンシャル・アクセスとバイト・アドレスによるランダム・アクセスが可能。

(4)デバイス・ファイル

コンソールやプリンタ等のデバイスをディスク・ファイルと同様に扱うもので、一般には読み込みと書き込みが可能であるが、実際にはデバイスの属性により異なる。アクセス単位およびアクセス順序(シーケンシャル、ランダム)はデバイスにより異なる。

表1 カーネル・システムコール一覧

マクロ名	機能
crt_env	エンバイロメントを生成
del_env	エンバイロメントを削除
crt_tsk	タスクを生成
del_tsk	タスクを削除
que_tsk	タスクをReady状態に
ext_tsk	タスクを終了(Idle)
trm_tsk	タスクを終了(Non-Existent)
abt_tsk	タスクを強制終了(Dormant)
rel_tsk	タスクをdormantからidle状態
spd_tsk	タスクを一時中断
res_tsk	タスクの一時中断を解除
get_tsk	タスク情報の取得
set_tsk	タスク情報の設定
frz_tsk	タスクを主メモリにロック
unf_tsk	タスクをアンロック
set_nam	タスクに名前を付ける。
get_tid	タスク名のタスク番号を取得
wat_sev	イベント発生待ち (シングル)
wat_mev	イベント発生待ち (マルチ)
pst_sev	イベントをポスト (シングル)
pst_mev	イベントをポスト (マルチ)
red_evt	イベント・フラグを読む。
clr_evt	イベント・フラグをクリア
alc_res	リソースを確保
fre_res	リソースを解放
con_dev	デバイスをシステムに接続
dis_dev	デバイスをシステムから切る
snd_msg	メッセージを送信
rcv_msg	メッセージを受信
chg_msg	メッセージ受け付け状態を変更
alc_mem	プールからメモリを割り付け
fre_mem	プールへメモリを返す。
set_vct	割込みベクタを設定
ocr_int	ソフト割り込みを発生
dcl_srv	ソフト割り込みのサーバ宣言
ack_srv	サーバ処理終了を通知
get_tim	時刻の読み出し
set_tim	時刻の設定
dly_tsk	タスクを遅延起動、周期起動
can_tsk	遅延、同期起動のキャンセル

表2 ファイル・システムコール一覧

マクロ名	機能
alc_fil	ファイルを割り付ける。
asg_fil	ファイルをオープンする。
cls_fil	ファイルをクローズする。
del_fil	ファイルを削除する。
chk_fil	バッファ内容をディスクへ出力
get_fil	ファイル情報を取得する。
set_fil	ファイル情報を設定する。
rew_fil	ファイルのリワインド
spc_fil	デバイス毎に定義された機能
mak_dir	ディレクトリの作成
rem_dir	ディレクトリの削除
chg_dir	カレント・ディレクトリ、デバイスの変更
red_rec	レコードのリード
wrt_rec	レコードのライト
upd_rec	レコードの修正
pst_rec	レコードのポジショニング
lck_rec	レコードのロック
ulk_rec	レコードのアンロック

６．３　サーバ定義機能

本ＯＳではユーザが各種サーバを容易に追加できる機能をサポートしている。サーバとしては、コンソール、プリンタ等のデバイスを扱うサーバをデバイス毎に定義できるデバイス・サーバ定義機能と、ソフトウェア割込み番号に対応してサーバを定義できるソフトウェア・サーバ定義機能の２つがある。

(１)デバイス・サーバ定義機能

デバイスもディスク・ファイルのオープンと同様に asg_fil SVC においてデバイス名に：を付けて指定することにより、以降のデータの読み書きはディスク・ファイルと同様に行なえる。ユーザはこのデバイス毎にそれに対するサービスを提供するサーバ・タスクを定義することができる。この機能によりデバイスの追加によるサーバの追加、サーバの変更が容易に行なえる。

(２)ソフトウェア・サーバ定義機能

本機能はソフトウェア割込み番号に対応してサーバを定義するものである。一般にはソフトウェア割込み命令はハードウェアに依存するが、本ＯＳでは仕様を非機種依存とする観点からソフトウェア割込み命令および割込み番号は仮想的なものとした。具体的には、本サーバ定義はソフトウェア割込み番号をパラメタとするシステムコールにより行なわれる。サーバへのサービス要求は、ソフトウェア割込み番号とサーバへの引数をパラメタとするシステムコールにより行なわれる。

７．論理インターフェイス部

論理インターフェイス部として現在考えているのは、図１に示したようにファイルシステム、リレーショナル・データベース、ネットワーク、グラフィックの４つである。

ファイルシステム、リレーショナル・データベースに関しては、６章に示したような特徴を持つものを既に開発した。現在ネットワーク機能とグラフィック機能の開発を行っている。

ネットワーク機能はＩＳＯ規格であるＯＳＩに準拠したものを、そしてグラフィック機能はＡＮＳＩ規格のＶＤＩに準拠したものを考えている。

８．おわりに

本稿ではマルチプロセサ指向のマイクロコン用高機能リアルタイムＯＳの一構成法について述べた。本ＯＳはリアルタイム制御用の機器組込用から、制御とプログラム開発が同時に行なえる汎用システムまでモジュールの選択により適用でき、さらに速い応答性や処理性能を要求するシステムに対してはマルチプロセサで対応できる。本ＯＳはリアルタイム制御、パソコン、ワークステーション用としての利用が可能である。

参考文献

［１］横畑他：マルチプロセサ指向高機能リアルタイムＯＳの一構成法、
　　情報処理学会第３３回全国大会論文集（１９８６）

［２］日立製作所：６８０００ＲＭＳ５ユーザーズマニュアル

［３］ＩＳＯ／ＴＣ９７／ＳＣ２１／ＷＧ３　Ｎ１１８：
　　Database Language Extended ＳＱＬ