7N−3　磁気ディスク制御装置用マルチプロセッサ対応実時間ＯＳ

横畑静生 *、　阿部行雄 **、　中村勝憲 ***
*(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)
***(株)日立製作所　小田原工場

１．はじめに

各種制御装置にマイクロプロセッサが使用されるようになってから久しい。そして近年、それら制御装置の高機能化、高性能化、高信頼化の要求に応えるために、マルチプロセッサ構成を採る装置が増えてきた。

また従来、この装置制御の分野では実時間ＯＳが広く使用され、この実時間ＯＳには割込み応答性に代表される対イベント高速応答性だけが特に要求されていた。しかし近年では、それに加えて高スループット、対故障性を要求する制御装置も徐々に増えてきた。

今回大型磁気ディスク制御装置を対象に、それらの要求に応えるマルチプロセッサ対応ＯＳを開発したので、本稿ではそのうち特にマルチプロセッサ対応機能とディスク制御装置特有機能について述べる。

２．ハードウェア構成

図１にディスク制御装置のハードウェア構成を示す。ＭＰは専用のマイクロプロセッサで、チャネル側の処理を行うチャネルプロセッサ群とデバイス側の処理を行うデバイスプロセッサ群の２つに分けて、機能分散により処理の多重度を上げる構成とした。各プロセッサはローカルメモリを持ち、共通メモリとしてキャッシュメモリとシステム全体の制御情報を記録するシステム制御メモリがある。

チャネルとデバイス間のデータ転送であるスルー処理はチャネルプロセッサとデバイスプロセッサの同時実行により処理し、デバイスからキャッシュへの先読み処理はデバイスプロセッサだけで処理する。全プロセッサのローカルメモリにはＯＳ及びタスクが搭載され、全プロセッサは対等関係にある。

３．マルチプロセッサ対応機能

３．１　実行プロセッサ選定機能

機能分散によりタスクを実行できるプロセッサ群は固定されているが、各プロセッサ群内では基本的には任意のプロセッサで実行可能である。しかしプロセッサとデバイスを結ぶ物理的なパス(ケーブル)が接続されていないデバイスプロセッサはそのデバイスの処理を行えない。またパスが存在しても障害により使用できなくなったり、メンテナンスのために電源がOFFになっているケースもある。

従って実際には任意のプロセッサで実行させると、プロセッサによってはタスクを実行できずタスク起動オーバヘッドだけが発生する。そこでＯＳのタスク生成機能には、デバイスやキャッシュへのパス障害、プロセッサ障害等のハードウェアの物理的や論理的な縮退復旧を管理するシステム制御メモリ内にある装置の構成制御情報を使って、ＯＳが実行可能プロセッサを自動選定する機能と、実行させるプロセッサをタスク自身が指定する実行プロセッサ指定機能の２つを設けた。

内部的にはタスク毎に実行可能プロセッサリストを持たせて、各プロセッサのＯＳは、このプロセッサリストを使って自律的に自プロセッサが実行できるタスクを選択して負荷分散を行う方式とした。マスタプロセッサを持たない本負荷分散制御方式は、プロセッサ障害に対する信頼性の向上及び性能上のあい路の回避を狙ったものである。(図２参照)。

３．２　タスク同時実行機能

スルー処理はチャネルプロセッサとデバイスプロセッサの両プロセッサ上でタスクを同時に実行する必要があるが、従来の実時間ＯＳにある一般的なタスク同期機構では複数タスクの複数プロセッサ上での同時実行は保証されず、片方のタスク起動後に別タスクの起

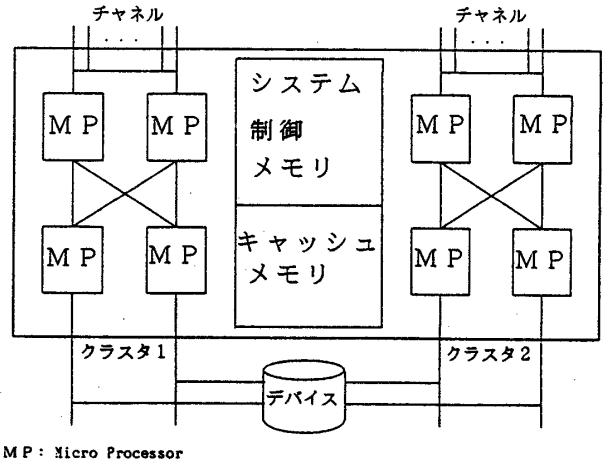

図１　ハードウェア構成

Multi-Processor Realtime OS for Disk Controller
Shizuo YOKOHATA*, Yukio ABE**, Katsunori NAKAMURA***
*Systems Development Lab., HITACHI,Ltd., **Hitachi Microcomputer Engineering CO.,Ltd
***Odawara Works, HITACHI,Ltd.

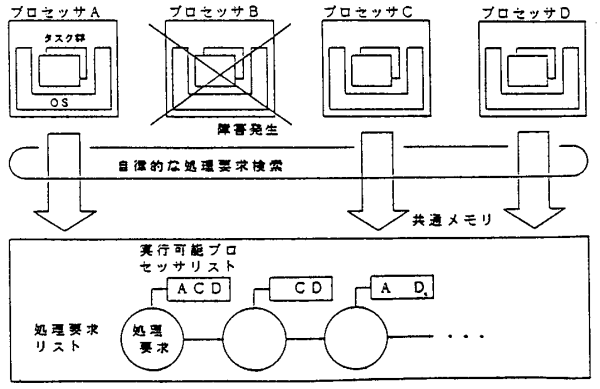

図２　自律的負荷分散制御方式

動を待つか実行をあきらめることになり、これが大きなオーバヘッドとなる。

これを防止するためにタスクの同時実行機能を設けた。タスクはWAIT状態になるときに、本タスクの再起動時に別タスクの同時実行が必要である旨を指定する。ＯＳは、WAIT状態が解除されたタスクをディスパッチする直前に、同時実行タスクの有無をチェックして、有ればその同時実行タスクを実行できる空きプロセッサを探して、プロセッサ間通信により他プロセッサにその同時実行タスクの起動を依頼した後に、タスクを起動する。空きプロセッサがない場合は、本タスクの起動を一旦あきらめ別の処理を行う。

この実現のためにWAIT要因として「同時実行待ち」を設けた。同時実行すべきタスクは、この要因待ちのWAIT状態にしておく。この機能により無駄なタスク起動によるオーバヘッドを防止できるようにした。

３．３　タスク間通信機能

マルチプロセッサ上のタスク間通信には、異プロセッサ間の同期(同時)通信と、非同期(非同時)通信の２つがある。一般的な通信方式には、プロセッサ間の共通メモリを用いたＯＳ提供のメールボックス等の通信路方式、タスクが共通にアクセスできるメモリを直接アクセスするシェアドメモリ方式、プロセッサ間の高速な通信レジスタ等を用いてタスクが直接受け渡すダイレクト方式等がある。

本ＯＳでは高速化の観点から、同期(同時)通信用にはダイレクト方式、非同期(非同時)通信用にはシェアドメモリ方式を採用した。

本制御装置においては通信が親子タスク間にのみ必要であることから、シェアドメモリ機能を親子タスク間に限定した。共通のメモリエリアもタスクに固定的に割り付けて、メモリ確保用のシステムコールを発行しなくてもいつでもアクセスできるようにして、低オーバヘッドで実現できる仕様とした。

４．ディスク制御装置特有機能

４．１　チャネル接続機能

タスク状態としては、一般的なＯＳと同一のRUN,WAIT,READY,INACT(存在しない状態)の他に、ディスク制御用に新たに追加したWAIT-RI(待ち状態)がある。WAIT-RIにおける各種待ち要因はWAITと同一であり、チャネル接続要求があることだけがWAITと異なる。

ディスク制御装置においては、チャネルは重要な共有資源の一つでタスクの実行に必要なケースが多く、タスク間で有効利用してチャネルの使用効率を向上させることが重要である。このためにタスクはWAIT状態になるときチャネルの使用権を放棄するためにチャネルを切り離す。従って再起動時にはチャネルの再接続が必要になる。しかし一般資源と同様にタスク自身がチャネルを確保(再接続)するのでは、チャネルはビジー率が高いため接続が行えずに、このタスク起動がオーバヘッドとなるケースが多発する。

WAIT-RIはこのオーバヘッドの低減を目的に追加した状態である。タスクの待っているイベントが発生したときにタスクがWAIT-RI状態であれば、ＯＳがチャネルの再接続処理を行って、接続できた場合だけタスクを起動することによりオーバヘッドを防止している。

４．２　ジョブ管理

ジョブは制御装置の外部や内部で発生する仕事である。チャネルから流入するチャネルコマンドのまとまりは、１つのジョブである。チャネルからの処理要求に対しては、ＯＳがジョブの生成を行う。装置内で発生する先読み処理も１つのジョブであり、このジョブはタスクが生成する。

資源の割付けはジョブに対して行っている。これは１つのジョブを複数のタスクが処理するケースでは、資源をタスクに割り付けるとタスク間での資源の引渡しが必要になり、オーバヘッドになるためである。そのために資源をジョブに対して割り付け、タスク間での共有を可能にしている。

また、ＯＳはこれらのジョブの生成消滅を管理し、デバッグ用にシステム稼働時にトレース情報を残す。

５．おわりに

本稿ではディスク制御装置を対象に、マルチプロセッサ対応実時間ＯＳの主にマルチプロセッサ対応機能とディスク制御装置特有機能について述べた。

６．参考文献

(1) 佐藤他：キャッシュ付き磁気ディスク制御装置のマルチプロセッサ制御方式、情報処理学会第４１回全国大会

(2) 猪股他：磁気ディスク制御装置におけるマルチプロセッサ制御方式のシミュレーション性能評価、情報処理学会第４１回全国大会