5N-5

可変構造型並列計算機の並列オペレーティング・システム - プロセス管理の概要 -

福田晃　村上和彰　末吉敏則　富田眞治
(九州大学)

1. はじめに

我々は、多様な並列処理に柔軟に対処できるマルチプロセッサ『可変構造型並列計算機』の開発を進めている(1)。本システムは128台のプロセッシング・エレメント(PE)からなる。本稿では、その並列オペレーティング・システム(OS)におけるプロセス管理の概要について述べる。

2. プロセスの分割

計算機システムは物理的、仮想的資源から構成され、プロセスはこれらを用いて活動する。我々は、プロセスを資源割当ての単位とその中で動く活動体とに分割する。前者をタスクとよび、後者をスレッドとよぶ。これらは、Machにおけるそれと同じである(2)。プロセッサの割当てはまずタスク単位で行い、さらにそのタスク内の複数のスレッドに割当てる。

本OSの特徴は、ハードウェアが提供するメモリ・アーキテクチャの可変性を活かして、タスク、スレッドを実現するところにある。タスク間のスレッド通信機構として、共有メモリ、メッセージの2つを提供する。

3. カーネルの構成

主要な機構を以下に述べる。

(1)実メモリ管理機構

実メモリはページフレーム単位で管理する。本機構はコアマップを保持する。ページ単位の実メモリの割当て、解放などの操作を提供する。

(2)実プロセッサ管理機構

スレッドの実プロセッサへの割当てを管理する。ディスパッチなどの操作を提供する。

(3)タスク管理機構

タスク・コントロール・ブロックを保持する。タスクの生成、起動、励起、中断、終結、消滅などの操作を提供する。

(4)スレッド管理機構

スレッド・コントロール・ブロックを保持する。スレッドの生成、起動、励起、中断、終結、消滅などの操作を提供する。

(5)仮想メモリ管理機構

仮想アドレスから実アドレスへの変換を行うマッピング・テーブルを保持する。仮想メモリの生成、削除、割当て、解放などの操作を提供する。

(6)共有メモリ管理機構

共有メモリに関するマッピング・テーブルを保持する。共有メモリの生成、削除、割当て、解放などの操作を提供する。

(7)レディキュー管理機構

実行可能なタスク、スレッドを管理するもので、レディキューを保持する。スケジューリングは2段階の優先度方式で行う。すなわち、優先度の最も高いタスクの中でスレッドの優先度に従って実行すべきスレッドを選択する。スレッドの優先度だけでスケジューリングすると、TLBおよびキャッシュのヒット率が低下する。さらにページ・フォルト頻度も増加する。上記の2段階優先度方式を採用することにより、コンテキスト・スイッチの際に、アドレス空間の切り換えに伴うオーバヘッドを抑えることができる。本機構は、エンキュー、デキュー、スケジューリングなどの操作を提供する。

(8)セマフォ管理機構

セマフォリスト、セマフォ値を保持する。セマフォの生成、消滅、P操作、V操作を提供する。

(9)メイルボックス管理機構

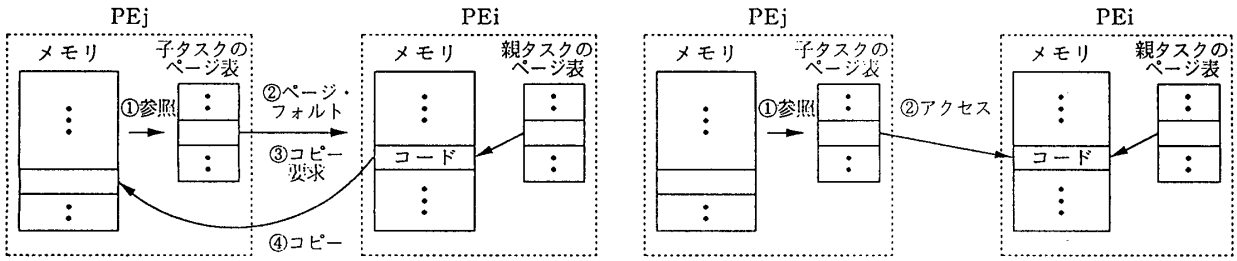
メイルボックス用のバッファを保持する。メイルボックスの生成、消滅、書きこみ、読みだしなどの操作を提供する。

4. タスクの生成

(1)自PE内に生成する場合

タスク管理機構にタスク生成を依頼する。アドレス空間生成の際には、効率を考えてコピーオン

A Distributed Operating System for a Reconfigurable Parallel Processor
Akira FUKUDA, Kazuaki MURAKAMI, Toshinori SUEYOSHI and Shinji TOMITA
Kyushu University



(a) 方法①　　(b) 方法②

図1 コードのアクセス方法

ライトの手法を用いる。すなわち、子タスクは親タスクのマッピング･テーブルのコピーをもち、各データ領域のマッピング･テーブルにはコピーオンライトビットをたてる。具体的には、当該ページをリード･オンリにしておき、書きこみがおこると割り込みを生じさせ、仮想メモリ管理機構に当該ページのコピーを依頼する。仮想メモリ管理機構はページのコピーと、マッピング･テーブルの再設定などを行う。

(2)他PE上に生成する場合

PE*i*上で実行中のスレッドがforkシステム･コールを発すると、カーネルはプロセッサ間通信機構を介して、PE*j*上のタスク管理機構にタスク生成を依頼する。子タスクは親タスクのコードを共有するが、この実現方法には以下の2つが考えられる。

①コードをPE*j*上にコピーする。このとき、効率を考えてオンデマンドで行う(図1(a))。

②PE*i*上に存在する親タスクのコードをネットワークを介して直接アクセスする(図1(b))。

可変構造型並列計算機は、メモリ･アーキテクチャの可変性、すなわち各PE上のローカル･メモリを動的に共有メモリとしてみせる機構をハードウェアで提供しているので②も可能である。

①の場合は、一旦PE*j*のメモリ上にもってくると、それ以後のアクセスは速い。しかし、コードの転送はページ(4KB)単位で行わなければならないので、このときのオーバヘッドが問題となる。②の場合は、共有アクセス可能とするための各種マッピング･テーブルの設定と、ネットワークを介するアクセス遅延のオーバヘッドが問題となる。

ここで、forkシステム･コールの後にすぐexecveシステム･コールがくる場合を考える。execveは元のアドレス空間の上に新たなイメージを上書する。従って、①の場合のオーバヘッドは1ページを転送する時間である。②の場合は、マッピング･テーブル設定の時間とネットワーク経由で数回アクセスする場合の遅延である。この場合、オーバヘッドは通常①の方が大きい。forkはexecveを伴うことが多いので、我々は方法②を採用する。

5. スレッドの生成

PE*i*上で実行中のスレッドがスレッド生成システム･コールを発し、それをPE*j*上に生成する場合について述べる。親スレッドと子スレッドは同一のアドレス空間をもつので、コード、データは共有する。この実現方法にも4.(2)と同様に①コピーと、②直接アクセスする2つが考えられる。コピーの場合は、明らかにデータの一貫性を保つためのオーバヘッドが大きい。従って、直接アクセスの方法をとる。概要を以下に述べる。

スレッドがスレッド生成システム･コールを発すると、カーネルは、プロセッサ間通信機構を介してPE*j*上のタスク管理機構にタスクのコピーを依頼する。このとき、PE*i*上のメモリ内のコード、データを直接アクセスできるように各種マッピング･テーブルを作成する。スタック領域はPE*j*のメモリ内にマッピングされる。生成されたスレッドは、このマッピング･テーブルを介してPE*i*上のコード、データに直接アクセスする。

6. おわりに

プロセス管理の概要を述べた。本OSの特徴は、ハードウェアが提供するメモリ･アーキテクチャの可変性を活かしてタスク、スレッドを実現するところにある。現在、詳細設計を進めている。

参考文献: (1)村上ほか:"可変構造型並列計算機のシステム･アーキテクチャ",情報処理学会「コンピュータアーキテクチャ」シンポジウム, pp.165-174(1988). (2)Rashid, R. F. et al.: "Machine-Independent Virtual Memory Management for Paged Uniprocessor and Multiprocessor Architectures", Proceedings of the 2nd Symposium on Architectural Support for Programming Language and Operating Systems, ACM, pp.31－39(1987).