第2章 キャッシュの管理の基本からハイパースレッドまで

マルチコア，マルチプロセッサのハードウェア

中森 章

マルチプロセッサを実現するためには，単純にプロセッサを複数搭載すればよいというわけではない．複数プロセッサをバスに接続するだけでは，バスの奪い合いになり，思ったように性能が上がらない．また，共有メモリの排他制御なども問題になる．これらはプロセッサのハードウェアを工夫することで解決することが可能だ．ここでは，マルチコア，マルチプロセッサから，最新技術であるハイパースレッドまで，ハードウェア面の工夫について解説する．（編集部）

1. はじめに

昨今，半導体メーカがマイクロプロセッサやマイクロコントローラを作るために使用するCPUコアの種類は，各企業ごとに淘汰されてきた感があります．あるいは，ARM社やMIPS Technologies社といったCPUコアを提供するIPベンダからCPUコアを買ってきて，そのまま組み込むという場合も多いと思われます．

しかし，今から20年くらい前の1980年代には，各半導体メーカがCPUのさまざまなアーキテクチャを開発し，それを業界標準とするべく躍起になっていました．そのような時代においても，CPU単体の性能向上はいつか限界に達すると予測されており，最終的には「マルチプロセッサが主流になっていくのは必然」と考えられていました．冗談として，一つのプロセッサ内にマルチプロセッサ・システムを構築する「1チップ・マルチプロセッサ」という言葉が話題になったものです．つまり，「マルチプロセッサなのに一つのプロセッサ？」というのが言葉遊びのように矛盾する感覚で興味深かったのです．

しかし，この冗談が今や現実味を帯びてきました．一つのチップに集積する場合は，「マルチプロセッサ」という単語はやはり違和感があるのか，「マルチコア」，「メニイ・コア」という言葉が使われています．マルチコアへの方向性を決定的にしたのはIntel社ですが，最初は性能向上の手段というよりも，性能を維持しつつ消費電力を下げる手段として提案されました．動作周波数が数GHzを超えるCPUコア開発では，製造プロセスを微細化する必要があり，トランジスタのスレッショルド電圧を低く抑えた結果，リーク電力（スタティック電力）が巨大になって使い物にならなくなるという事情があります．また，当然，動作周波数が上がれば，動作電力（ダイナミック電力）も周波数に比例して上昇していくので，その影響も受けます．

これと同じような傾向は，組み込み制御の世界にも起きています．CPUコアを現実的な消費電力（2W～3W程度）で動作させるには，動作周波数を400MHz～600MHz程度に制限せざるを得ません．そこで，それ以上の性能向上を狙うにはマルチコア化が必然となってきます．

とはいえ，CPUコアを二つ持ったからといって，2倍の性能が得られるとは限りません．各CPU間がお互いに協調して動作する仕組みを作り込むことが必要です．それにも増して，OSの手助けが必須となります．

本稿では，マルチコアのハードウェアに求められる最低限の基幹技術を俯瞰（ふかん）していきたいと思います．なお，マルチコアといっても，その実体はマルチプロセッサ・システムを1チップに集積しただけで，必要な基幹技術はマルチプロセッサと同じです．以降では典型的なマルチプロセッサの基礎知識を紹介していきます．

2. マルチプロセッサの構成

● マルチプロセッサの目的は全体的な処理時間の短縮

マルチプロセッサとはプロセッサが複数存在するシステムのことです．ここで各プロセッサにどのように処理させるかによって，論点は微妙に異なります．

現在，OSがマルチタスクで動作するのは当たり前です

が，複数のタスクのそれぞれに一つのプロセッサを固定的に割り当てて動作させるという方法があります．つまりタスクの数だけプロセッサを用意して，タスクとプロセッサを1：1に対応させる方式です．これは，マルチプロセッサのもっとも単純な形態です．一般に，タスクが異なれば，データ領域も異なります．そこでは，特別な場合を除き，同一のメモリ領域をアクセスする場合の競合（順序の保証など）を考慮する必要はありません．その意味で，このような形態は本稿の対象外とします．

私たちが望むのは一つのタスク[注1]が複数のプロセッサで処理されることによる全体的な処理時間の短縮です．それを行うためには，一つのタスクを複数のスレッドに分割し，並列実行可能なスレッドをそれぞれのプロセッサで同時に実行させることです．ここでスレッドをそれぞれのプロセッサに割り付けるのはOSの役割です．

● 共有メモリを使うのが一般的

さて，スレッドは一つのタスクを分割して生成したものなので，スレッド間で共通のメモリ領域（変数領域）をアクセスすることは珍しくありません．というか，ほとんどの場合，メモリ領域は共通といっても過言ではないでしょう．つまり，マルチプロセッサ構成では，複数のプロセッサが互いにアクセス可能な共有メモリを前提とするのが普通です．

この共有メモリの構成方法によって，**図1**のように，マルチプロセッサの構成は2種類に分類できます．一つは集中共有メモリ方式と呼ばれ，一つの共有メモリを複数のプロセッサが一つの共有バスを通してアクセスする方式です．共有メモリが一つしかないのでUMA（Uniform Memory Architecture）とも呼ばれます．もう一つは分散共有メモリ方式と呼ばれ，各プロセッサが抱える局所（ローカル）メモリを共有バスでお互いにアクセス可能にしている方式です．それぞれの局所メモリは物理的にはつながっているので，仮想的に共有していると見なせます．この方式はNUMA（Non-uniform Memory Architecture）とも呼ばれます．このNUMAにおいて，キャッシュのコヒーレンシ（一貫性）を保つように設計されているものをccNUMA（Cache Coherent NUMA）と呼びます．大型計算機やワークステーションではccNUMAの構成を採用するものは珍しくありません．しかし，本稿では私たちが比較的頻繁に目にするUMA，そしてUMAの中でも特に有名なSMPについて説明していきます．

● 対称型マルチプロセッサ「SMP」とは

SMP（Symmetric Multi-processor）は，プロセッサから共有メモリへのアクセス時間が一定の構成のことです．すべてのプロセッサが時間的空間的に対称（Symmetric）であり，それがSMPという単語の由来です．SMPにおいては，複数のプロセッサは対等な関係のものとして扱われます．また，共有メモリはすべてのプロセッサから同一のアドレスでアクセスできます．

SMPの利点としては，各CPUが簡単にデータを共有でき，並列のプログラムが比較的簡単に書けること，および逐次処理のプログラムやプロセスを処理するのに優れていることが挙げられます．逆に弱点は，同時に一つのプロセッサしか共有メモリにアクセスできないので，CPUの数が増えるに従って，共有メモリへのアクセス権の調停が難しくなることがあります．しかし，その取り扱いの簡便さ

注1：UNIXなどでは資源管理の単位をプロセス，実行の単位をスレッドと呼んでいる．μITRONでは実行の単位をタスクと呼ぶことがあるが，本稿では前者の用語を使用し，タスクは「おおまかな処理単位」として使用している．

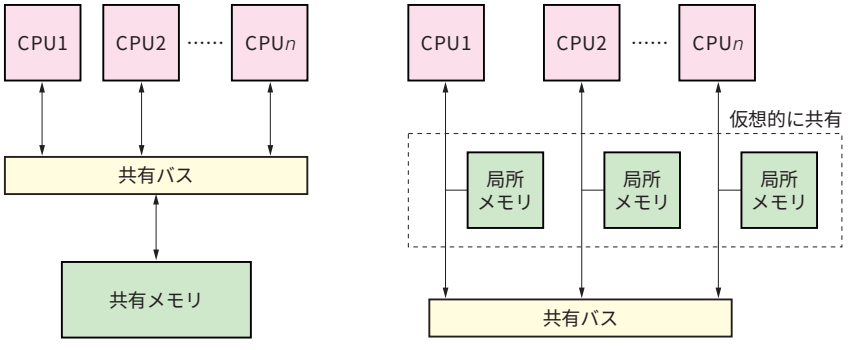

図1 マルチプロセッサの構成方式

（a）集中共有メモリ方式　（b）分散共有メモリ方式