

第6章 パイプライン化からマルチコア並列処理まで



GPUのアーキテクチャ研究② CPUの高速化の歴史

GPUのアーキテクチャを議論する前に、ここでは汎用CPUをどのように高性能化してきたかを復習しておきます。

CPUを高性能化するには、単純に動作クロックを上げるのがとっつき早いですが、ここではマイクロ・アーキテクチャ面での工夫に焦点を当てます。

6-1 [黎明期] 命令の実行に何クロックもかかっていた

- 初期のCPUは、命令を直列的に実行
マイコン黎明期の初期のCPUは、図1(a)のように、命令を直列的に実行していました。



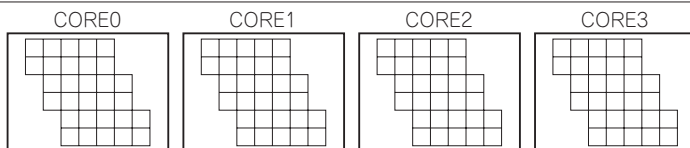
CPUの命令は命令の内容に応じて複数のステージに別れており、例えば下記のステージがあります。

- ▶ **命令フェッチ(F)**
プログラムが格納されたメモリから命令コードをフェッチ(リード)します。
- ▶ **デコード(D)**
命令コードの意味を解釈します。
- ▶ **実行(Eまたはm)**
デコード結果に応じて、CPUレジスタ内のデータを演算器(ALU: Arithmetic Logic Unit)を使って加減算、論理演算、ビット演算したり、乗算器を使って掛け算をしたり、さまざまな演算を実行します。

- 【命令のステージ】
- F: 命令フェッチ
 - D: 命令デコード(解読)
 - E: ALU演算(データ演算, アドレス計算)
 - m: 整数乗算
 - Mr: メモリ・リード
 - Mw: メモリ・ライト
 - W: ライト・バック
 - -: 命令ストール(待ち)

※ALU: Arithmetic Logic Unit

- 【命令の動作】
- LD R2, @(R1, R0)
メモリ・ロード命令: R0とR1を足した値をアドレスとするメモリをリードしてR2に格納する
 - ADD R3, R2, R1
加算命令: R1とR2を加算した値をR3に格納する
 - MUL R6, R5, R4
乗算命令: R4とR5を乗算した値をR6に格納する
 - ST @(R1, R0), R3
メモリ・ストア命令: R0とR1を足した値をアドレスとするメモリにR3をライトする



(e) マルチ・コア

図1 CPUの高性能化手法

【セミナー案内】 [ビギナ向け] [実習セミナー] [KIT付き] 実習・GNU Radioで始めるSDR入門 [教材付き]
— スペアナ, FMラジオ, TVの自作とキーレスエントリの解析で学ぶ
【講師】 小林 真氏, 8/31(土) 25,000円(税込み) <https://seminar.cqpub.co.jp/>