第２回　コンピュータ科学基礎論 資料　　2017-09-29　山内

今回は、並列処理の伝統的な Flynn の分類で言うところの SIMD （パイプライン）について、問題点を議論してみたい。

(1)　SIMD と MIMD の構成的な違いは分かったとして、SIMD と MIMD の役割の違い・役立ち方の違いをまとめると、どういうことになるだろうか？　つまり、どういうときに SIMD が有効に使えるのか？　SIMD にできないことは何か？

(2)　SIMD の性能を制約する（ステージ数と同じ並列度が得られない）原因を、ハザードとそれ以外を含めて、列挙し説明してみよう。

(3)　SIMD は、予め行う処理の順序が決まっていて、処理に対応するモジュール（＝ステージ）を並べて、歩調の揃った行進のように処理を進める。 このことの利点と欠点を整理してみよう。

また、この処理と順序が決まっているという制約を取り外すとしたら、（つまり、プログラムとして書く処理手順をパイプラインとして実行するようなことを考えるとしたら）、どういう考え方が考えられるだろうか？ ＜キーワード: データフローマシン ～ ネット等で探してみよ＞

（注） パイプラインは、多データ（SIMD）で使われるだけでなく、1つの命令を多段ステージに分割して命令実行をパイプラインに流す使い方もされる。
多データ（SIMD）のパイプラインは「ベクトルプロセッサ」とも呼ばれ、スーパーコンピュータの主力技術として 1980 年代の Cray-1/2 辺りまで使われたが、今も Intel 86x 系のプロセッサに SSE (Streaming SIMD Extension, 第１世代 Core まで)、AVX /AVX-2 (Advanced Vector Extensions, Sandy Bridge 以降) で搭載されている（AMD の x86 互換にも類似のパイプラインがある）。 ベクトルデータをコンパイラが認識してパイプライン命令用のコードを生成しなければならないので、表面ではあまり取沙汰されないのだろう。
命令のパイプラインは、Intel のマイクロプロセッサ（最新の Core シリーズも含めて）等で広く使われており、20 段程度のパイプラインがコア内部に複数装備されている（ハイパースレッディング）。