第7章 1個のCU（PEの集合体）で実行する演算処理の並列数（ローカル・ワーク・サイズ）の最適化

最適化②…GPUの内部構造に合わせたチューニング

本橋 弘臣

第6章ではカーネル関数での処理ベクタ長をチューニングすることにより，カーネルの性能最適化が可能であることを説明しましたが，GPUにはさらなる性能最適化のための仕組みが用意されています．

本稿ではCPUとGPUを対比させながら，プロセッサの内部構造について説明した上で，それに向けた最適化方法について紹介します．

GPUの内部構造を知るために…シンプルなCPUの仕組みを見てみる

8080Aはインテルが開発して，初めて商業的に成功した8ビットCPUです．8080Aの後継として開発された16ビットCPUが8086/8088で，初代IBM PCに8088が搭載されて爆発的にヒットしました．この8086は，現代のCoreシリーズの遠い祖先です．

本稿ではCPUの内部の仕組みを説明するために，題材として内部構造がシンプルな8080Aを取り上げます．ここでは8080Aの仕組みを説明するための資料として，8080Aの動作をCコードとして実装したソフトウェア・エミュレータのソースコードを用意しました（**リスト1**）．

リスト1　8080Aソフトウェア・エミュレータのソースコード（レジスタ部）

```
 1:// 16bit レジスタを表す共用体
 2:typedef union reg16bit_t {
 3:  uint8_t  b[2];  // 8bit アクセス用
 4:  uint16_t w;     // 16bit アクセス用
 5:} reg16bit;
 6:
 7:// Intel 8080A CPU のレジスタをエミュレートするための構造体
 8:struct cpu_reg_t {  // 8080A レジスタセット
 9:                    //
10:  uint8_t  a;       //  | A |   |  アキュームレータ
11:                    //
12:  reg16bit  bc_reg; //  | B | C |  汎用レジスタ
13:                    //
14:  reg16bit  de_reg; //  | D | E |  汎用レジスタ
15:                    //
16:  reg16bit  hl_reg; //  | H | L |  汎用/間接参照用レジスタ
17:                    //
18:  uint16_t  pc;     //  |  PC   |  プログラムカウンタ
19:                    //
20:  uint16_t  sp;     //  |  SP   |  スタックポインタ
21:} cpu;              //
22:                    //  ←――→
23:                    //  16bit
24:
25:// 16bit レジスタに対するアクセス用のマクロ
26:#define b  bc_reg.b[0]
27:#define c  bc_reg.b[1]
28:#define bc bc_reg.w
```

● 演算対象のデータを置いておくレジスタ

8080Aの内部には各種演算を行う際に演算対象のデータを置いておくためのレジスタが存在しています（**リスト1**の9～21行目のコメント部を参照）．レジスタはCPUのチップ内に存在しているハードウェア機構ですが，とりあえずC言語で言うと変数のようなものだと考えれば大丈夫です．8ビットの大きさのA/B/C/D/E/H/Lレジスタが最も基本となる汎用用途で利用可能なレジスタです．これらのレジスタにはメモリから8ビットのデータを読み込んで，各種の演算を行うことができます．また16ビットのデータ（例えばメモリ・アドレス）を取り扱う際には，例えばHレジスタとLレジスタを連結したHLレジスタとして利用できるようになっています．

リスト1の8～21行目では，レジスタをエミュレートするための構造体を定義しています．BC/DE/HLレジスタは，8ビット単位で分けて使用したり，あるいはセットで16ビットのレジスタとして使用することができるので，2行目で定義した共用体としてbc_regなどの変数を宣言しています．

● 実行する命令のアドレスを保持するプログラム・カウンタ

プログラム・カウンタ（PC）は，メモリ上で次にCPUが実行する命令のアドレスを保持しています．ソフトウェア的な言い方をすると，PCは次に実行する命令を指すポインタだと考えてください．またここで言う命令とは，8080Aが理解して実行することが可能な機械語コードのことを指していて，その実体はメモリ上に存在している1バイトのデータです．

表1に8080Aの命令コード表の一部を掲載しています．00H～FFHの1バイトのデータが，それぞれど