第6章 足し算関数の実装からシンプルなシミュレーション・ゲームの作成，高速化まで

# C++で本格 CUDAプログラミング



鈴木 量三朗

リスト1　GPUプログラミングのHello World的存在である足し算の実装（add-main.cu）

```
#include <iostream>
#include <cassert>
#include <ctime>
#include <chrono>

__global__
void
add(const float *const a, const float *const b,
                                    float *const c)
{
    unsigned int index = threadIdx.x;

    c[index] = a[index] + b[index];
}


int
main(int argc, char **argv)
{
    cudaError_t cu_err;
    const unsigned int mem_size = sizeof(float) *
                                                1024;
    float *managed_a, *managed_b, *managed_c;

    cu_err = cudaMallocManaged(&managed_a, mem_size);
    assert(cu_err == cudaSuccess);

    cu_err = cudaMallocManaged(&managed_b, mem_size);
    assert(cu_err == cudaSuccess);

    cu_err = cudaMallocManaged(&managed_c, mem_size);
    assert(cu_err == cudaSuccess);

    for( unsigned i = 0 ; i < 1024 ; i++ ) {
        managed_a[i] = (float)i;
        managed_b[i] = (1024 - i);
    }

    auto start_time = std::chrono::
                        high_resolution_clock::now();
    for (int i = 0; i < 100000; ++i) {
        add<<<1, 1024>>>(managed_a, managed_b,
                                            managed_c);
    }
    cudaDeviceSynchronize();

    auto end_time = std::chrono::
                        high_resolution_clock::now();
    auto ms = std::chrono::duration_cast<std::chrono::
                milliseconds>(end_time - start_time);

    std::cout << ms.count() << "ms" << std::endl;

    for( unsigned i = 0 ; i < 1024 ; i++ ) {
        if ( managed_c[i] != (1024.0 + i) ) {
            std::cerr << "error:" << i << " " <<
                                managed_c[i] << std::endl;
        }
    }

    cudaFree(managed_a);
    cudaFree(managed_b);
    cudaFree(managed_c);
}
```

（GPU側で実行されるカーネル関数）

（カーネル関数が自分のスレッドIDを特定するための計算を行う）

（カーネルで使うユニファイド・メモリ領域を確保）

（ブロック数とスレッド数で並列処理の大きさを指定）

（GPUから計算結果を受け取るために処理を待って同期を取る）

## CUDA版Hello World…GPUで足し算

ここからは直接CUDAを使います．まずはGPUプログラミングのHello Worldとも言えるadd関数（GPUで並列に実行可能な足し算をする関数）を実装します[注1]．

### ● CUDAのコード

#### ▶CUDA部分は__global__で修飾する

リスト1にソースコードを示します．ファイル名はadd-main.cuとしました．ファイルの中身はC++と非常に似ているのですが，実際にはCUDAであり，通常のC++コンパイラではコンパイルできません．

CUDAではGPU側で実行されるカーネル関数に__global__という記述を付けて修飾します．

#### ▶関数の内容

add関数は，スレッドと呼ばれる単位で並列に実行されると考えることができます．add関数の中では，自分の担当場所を特定するために，threadIdx.xを使いindexを計算します．indexが特定できれば，aとbの配列の中のどの値をアクセスして足し算し，cの配列のどこに格納すればよいかが分かります．

注1：エヌビディアは，CUDAのサンプルをGitHub（https://github.com/NVIDIA/cuda-samples）で公開している．