第6章 OSが軽くてなんとタブレット越え!

Windows 10 IoT Core コンピュータの処理性能

山本 隆一郎

ベンチマーク①…シングル・スレッド

DragonBoardの実力の評価を兼ねて，簡単なWindows 10デバイス共通UWPアプリを作成してベンチマーク計測を行います．ここではモンテカルロ法を使って円周率を求めてみます．

実験は前章と同じ構成で行います．

● させる処理…モンテカルロ法による円周率の計算時間を計る

モンテカルロ法は，広く知られている古典的な円周率を求める手法です．乱数で複数の座標点を指定し，それが円の面積に入っているかどうかをカウントして，入っている数を元に確率的に円周率を算出するものです．乱数の呼び出しや加減乗除算の複数回の繰り返しを行いますので，初歩的なベンチマークとしては適していると思います．ここでは，1,000万回の計算を行い，完了までの時間を計測します．

プログラムは先ほど作成したHello World!を修正します．ボタンを押した際に実行される関数button_Click()を**リスト1**のように書き換えます．Hello World!ではボタンを押したら，テキスト・ボックスに文字列を表示するだけでしたが，今回はボタンを押してからモンテカルロ法の計算を開始して，求めた円周率と計算にかかった時間をテキスト・ボックスに表示します．

● 条件：リリース・モードでビルド

Hello World!では，デバッグ・モードでビルドしていました．今回はリリース・モードでビルドして実行します．

デバッグ・モードは，Visual Studio 2015とビルド情報をやりとりするため，実行速度が本来より低下します．今回はベンチマークが目的ですので，余計な処理を除けるリリース・モードでビルドして実行します．

ベンチマーク①の実行結果

● 64ビットARMボードDragonBoardの処理性能

実行結果を**図1**と**写真1**に示します．DragonBoardでは，求めた円周率は3.14151でした．アルゴリズム

リスト1　ベンチマーク①…シングル・スレッド用モンテカルロ法による円周率計算のプログラム

```
#define MONTECARIOLOOPMAX 10000000
                              // モンテカルロ法でループを行う数

void IoTCoreTest::MainPage::button_
Click(Platform::Object^ sender,
                Windows::UI::Xaml::RoutedEventArgs^ e)
{
    std::chrono::system_clock::time_point  tmstart,
                                                    tmend;
    tmstart = std::chrono::system_clock::now();
                                                // 計測開始

    double innum = 0;
    double x, y, pi;

    // 指定回数ループを行う
    for (int i = 0; i < MONTECARIOLOOPMAX; i++) {

        /// 0～1の範囲の乱数を生成する
        x = rand() / (RAND_MAX + 1.0);
        y = rand() / (RAND_MAX + 1.0);

        // 半径1の円の内部にプロットされた場合
        if ((x*x + y*y) <= 1.0) {
            innum++;    // 円の内部にプロットされた数を数える
        }
    }

    pi = (double)innum / MONTECARIOLOOPMAX * 4.0;
                // 4倍して完全な円の面積に相当するプロット数に変換

    tmend = std::chrono::system_clock::now();
                                                // 計測終了
    int tmelapsed = std::chrono::duration_cast<std::c
        hrono::milliseconds>(tmend - tmstart).count();
                                // 処理時間をmsに変換

    // 結果をテキスト・ボックスに表示
    textBox->Text = "π=" + pi.ToString() + ", Time="
                        + tmelapsed.ToString() + "msec";
}
```