Architektura Komputerów 2 Laboratorium

Sprawozdanie nr 2

Zadanie w języku C++ — SIMD / SISD

Autor
Damian Łukasiewicz 259186

Laboratorium nr 3

Prowadzący – Mgr inż. Tomasz Serafin

Grupa – K02-17d, Poniedziałek TP 14:30

Spis treści

1 Cel ćwiczenia

Celem ćwiczenia było zaimplementowanie, zmierzenie czasu oraz porównanie operacji dodawania, odejmowania, mnożenia i dzielenia za pomocą SIMD oraz SISD.

2 Kod programu

2.1 Struktury

```
struct Vector{
    float a, b, c, d;
};

Vector Arr1[SIZE];
Vector Arr2[SIZE];
```

2.2 Dodawanie za pomocą SIMD

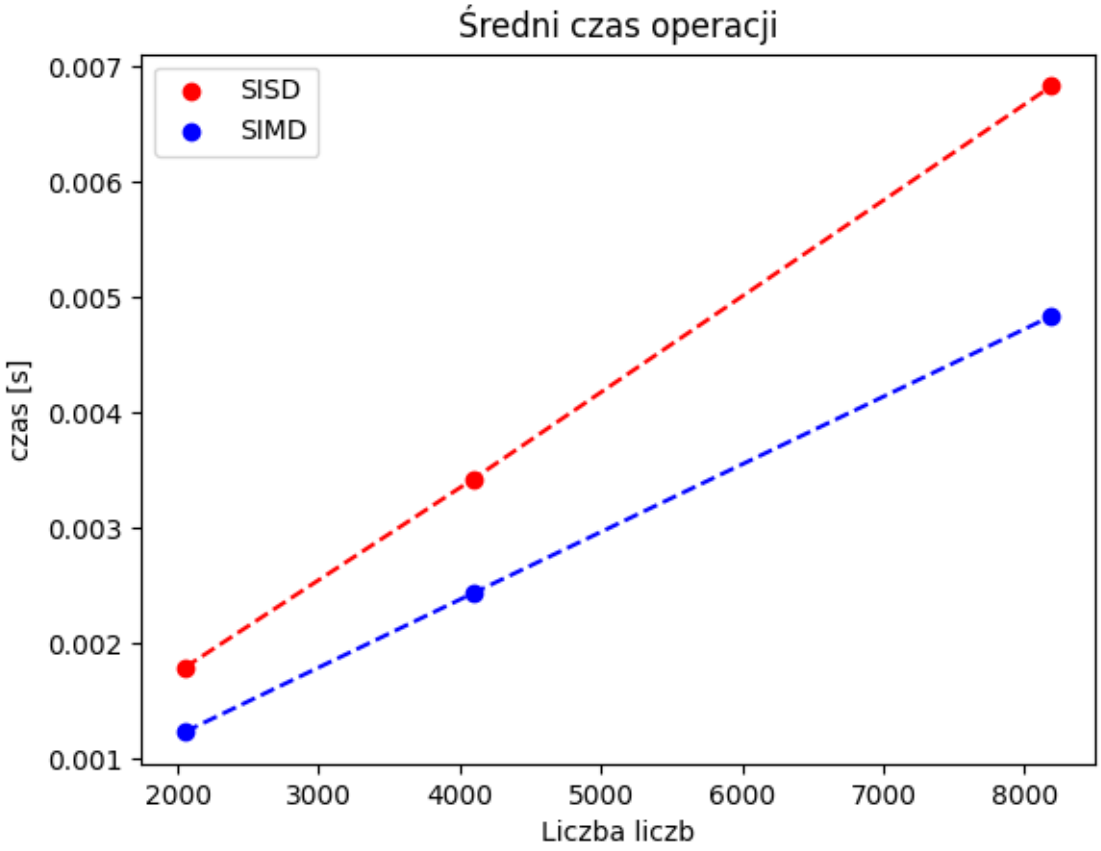
```
double add_SIMD(int i) {
    // start mierzenia czasu za pomocą <time.h>
    start = clock();
    asm(R"(
        movaps %1, %%xmm0
        movaps %2, %%xmm1
        addps  %%xmm1, %%xmm0
        movaps %%xmm0, %0
    )"
    :"=m" (Results[i]) // 0 -> output
    :
    "m" (Arr1[i]),       // 1 -> input
    "m" (Arr2[i])        // 2 -> input
    );
    // zakończenie mierzenia czasu
    stop = clock();
    return getClocks(start, stop);
}
```

2.3 Dodawanie za pomocą SISD

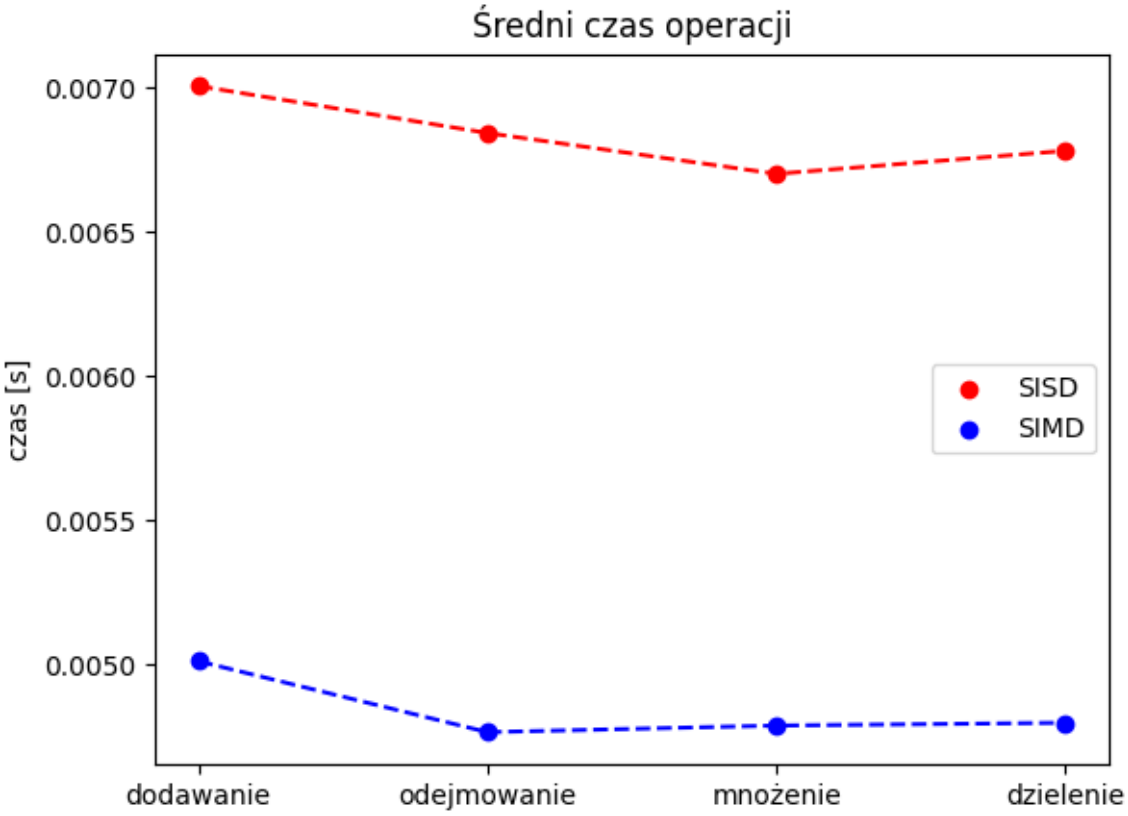
```
double add_SISD(int i) {
    start = clock();
    asm(R"(
        fld %4
        fld %8
        faddp
        fstp %0
        fld %5
        fld %9
        faddp
        fstp %1
        fld %6
        fld %10
        faddp
        fstp %2
        fld %7
        fld %11
        faddp
        fst %3
    )"
    :  // outputs
    "=m" (Results[i].a),    // 0
    "=m" (Results[i].b),    // 1
    "=m" (Results[i].c),    // 2
    "=m" (Results[i].d)     // 3
    :  // inputs
    "g" (Arr1[i].a),        // 4
    "g" (Arr1[i].b),        // 5
    "g" (Arr1[i].c),        // 6
    "g" (Arr1[i].d),        // 7
    "g" (Arr2[i].a),        // 8
    "g" (Arr2[i].b),        // 9
    "g" (Arr2[i].c),        // 10
    "g" (Arr2[i].d)         // 11
    );
    stop = clock();
    return getClocks(start, stop);
}
```

3 Wykresy

3.1 Zmienność średniego czasu w zależności od liczby liczb



3.2 Zmienność średniego czasu w zależności od typu działania dla 8192 liczb



3.3 Zysk z zastosowania mechanizmów SIMD

4 Napotkane problemy

Podczas testowania kodu zauważyłem, że mierzenie czasu SISD zwracało -NaN. Debugowanie za pomocą GDB nie pomogło znaleźć przyczyny tego błędu. Podczas szukania błędów w kodzie zauważyłem jednak, że wprowadzenie zmian w kodzie nieodpowiedzialnym za mierzenie czasu sprawia, naprawiało błąd. Założyłem więc, że przyczyną było optymalizowanie kompilacji kodu przez g++. Aby uniknąć tego błędu zastosowałem flagę -O2.

5 Uruchomienie programu

```
1 all: runnable
2
3 runnable: main.cpp
4     g++ -Wall -m32 -O2 -o Runnable main.cpp
```

Flagi:

- -Wall — wyświetl wszystkie ostrzeżenia
- -m32 — kompiluj w wersji 32-bitowej
- -O2 — optymalizuj szybkość wykonywania kodu (w porównaniu do domyślnego trybu -O0 optymalizującego czas kompilacji[1])

[1] Optimize options

6 Zarys historyczny

Od lat 2000 producenci procesorów napotkali wiele przeszkód stojących na drodze zwiększenia taktowania CPU. Musieli znaleźć inne spodoby zwiększania wydajności swoich procesorów. Jednym z nich jest okazało się wynalezienie SIMD. Pierszym procesorem z instrukcjami SIMD (Streaming SIMD Extensions) był wydany w roku 1999 Intel Pentium III.

Dzięki SIMD możliwe jest wykonywanie operacji wielu liczbach jednocześnie. Jest to szczególnie przydatne w przetwarzaniu grafiki 3D.

7 Wnioski

Zarówno w przypadku SISD jak i SIMD zależność czasu operacji od rozmiaru danych jest liniowa, jednakże czas dla SISD wzrasta gwałtowniej.

Zysk w czasie z zastosowania mechanizmów SIMD jest podobny dla wszystkich testowanych operacji i wynosi on około 30%.