# Benchmark et matériel

## CPU

AMD Ryzen 7 3700x 8-core, 2 threads par cœur

3.6 GHz, max 4.4 GHz

## Caches

|  |  |
| --- | --- |
| Cache level | Taille |
| 1 | 32kB |
| 2 | 512kB |
| 3 | 16 mB |

## Benchark des performances

Pour mesurer les performances de la machine utilisée, j’ai utilisé likwid dans sa version 1.18.0.

Les commandes suivantes ont été utilisées :

likwid-bench -t peakflops -W N:256KB

* Selon la topologie obtenue grâce à likwid, le groupe « N » permet d’utiliser tous les cœurs.
* La taille du banc de test est de 256KB car on a 8 cœurs avec chacun un cache L1 de 32KB => 8 \* 32KB = 256KB.

Les valeurs obtenues sont :112006.34 MFlops/s, 56003, MByte/, et 134317.660339 MUops/s.

Ainsi, le *roofline model* est (en terme de MFLops) :

Une image contenant texte, capture d’écran, ligne, diagramme

Description générée automatiquement

et en terme de MUops :

Une image contenant texte, capture d’écran, ligne, Tracé

Description générée automatiquement

## Mesures de base de l’application

Toutes les mesures se feront avec l’image half-life.png.

Pour calculer l’intensité opérationnelle, il nous faut le nombre de MFlops/s, de MUops/s ainsi que la largeur de la bande passante de notre programme. Cependant, seule la première partie du programme (le *greyscaling*) est en opérations à virgule. Pour les filtres gaussiens et de sobel, je me baserai sur le nombre de UOPS. Pour aider aux mesures, j’ai créé un groupe de performance « AI.txt » (cf annexe) permettant de calculer ces mesures ainsi que l’intensité opérationnelle (IO). Il est également nécessaire de séparer ces trois fonctions car le volume de donnée traité n’est pas le même (greyscale traitant des données en couleur, soit 3-4x plus grand que pour gauss et sobel).

J’ai entouré les régions mentionnées avec les marqueurs de likwid et ai mesuré les métriques ci-dessous, qui serviront de baselines.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Région | MUops/s | M/Flops/s | OI | IPC | Temps [ms] |
| greyscale | / | 2675.1780 | 0.181 | 3.6046 | 7.812 |
| gauss | 16518.1684 | / | 10.1171 | 4.0710 | 23.259 |
| sobel | 17378.6076 | / | / | 4.0591 | 41.3043 |

Annexe A – AI.txt

SHORT AI calculations for UOps et FLOPS

EVENTSET

FIXC1 ACTUAL\_CPU\_CLOCK

FIXC2 MAX\_CPU\_CLOCK

PMC0 RETIRED\_INSTRUCTIONS

PMC1 RETIRED\_UOPS

PMC2 RETIRED\_SSE\_AVX\_FLOPS\_ALL

PMC3 MERGE

DFC0 DATA\_FROM\_LOCAL\_DRAM\_CHANNEL

DFC1 DATA\_TO\_LOCAL\_DRAM\_CHANNEL

METRICS

Runtime (RDTSC) [ms] 1000\*time

Runtime unhalted [s] FIXC1\*inverseClock

MFLOP/s 1.0E-06\*(PMC2)/time

Retired MOps/s 1.0E-06\*(PMC1)/time

Memory bandwidth [MBytes/s] 4.0E-06\*(DFC0+DFC1)\*(4.0/num\_numadomains)\*64.0/runtime

AI (Uops) [OP/Byte] (PMC1)/((DFC0+DFC1)\*(4.0/(num\_numadomains))\*64.0)

AI (Flops) [Flop/byte] (PMC2)/((DFC0+DFC1)\*(4.0/(num\_numadomains))\*64.0)

LONG

Truc