

# Quali sono le caratteristiche della nostra CPU?

## CORE:

Thread(s) per core:	1
Core(s) per socket:	12

## CACHE:

L1d cache:	288 KiB
L1i cache:	192 KiB
L2 cache	7.5 MiB

```
File Edit View Search Terminal Help
dpcpp-ct -- latest
dpl -- latest
inspector -- latest
ipp -- latest
ippcp -- latest
ipp -- latest
itac -- latest
mkl -- latest
mpi -- latest
tbb -- latest
vtune -- latest
oneAPI environment initialized ::

5680500@sw120:~$ lstopo --of txt > cpu.txt
5680500@sw120:~$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
Address sizes:         46 bits physical, 48 bits virtual
CPU(s):                20
On-line CPU(s) list:  0-19
Thread(s) per core:   1
Core(s) per socket:   12
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 151
Model name:            12th Gen Intel(R) Core(TM) i7-12700
Stepping:               2
CPU MHz:                2100.000
CPU max MHz:           4900.0000
CPU min MHz:           800.0000
BogoMIPS:              4224.00
Virtualization:        VT-x
L1d cache:              288 KiB
L1i cache:              192 KiB
L2 cache:                7.5 MiB
NUMA node0 CPU(s):     0-19
Vulnerability Itlb multihit: Not affected
Vulnerability L1tf:      Not affected
Vulnerability Mds:       Not affected
Vulnerability Meltdown: Not affected
Vulnerability Mmio stale data: Not affected
Vulnerability Retbleed:  Not affected
Vulnerability Spec store bypass: Mitigation: Speculative Store Bypass disabled via prctl
Vulnerability Spectre v1: Mitigation: usercopy/swaps barriers and __user pointer sanitization
Vulnerability Spectre v2: Mitigation: Enhanced IBRS, IBPB conditional, RS B filling, PBRSB-eIBRS SW sequence
Vulnerability Srbds:     Not affected
Vulnerability Tsx async abort: Not affected
Flags:                  fpu vme de pse tsc msr pae mce cx8 apic sep mtr r pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rd tscp lm constant_tsc art arch_perfmon pebs bts
```

**Nome architettura:** intel SSE4.1, intel SSE4.2, intel AVX2

Specifications > Processors		Search specifications
Intel® Deep Learning Boost (Intel® DL Boost) on CPU	?	Yes
Intel® Optane™ Memory Supported	?	Yes
Intel® Speed Shift Technology	?	Yes
Intel® Turbo Boost Max Technology 3.0	?	Yes
Intel® Turbo Boost Technology	?	2.0
Intel® Hyper-Threading Technology	?	Yes
Intel® 64	?	Yes
Instruction Set	?	64-bit
Instruction Set Extensions	?	Intel® SSE4.1, Intel® SSE4.2, Intel® AVX2
Idle States	?	Yes
Enhanced Intel SpeedStep® Technology	?	Yes
Thermal Monitoring Technologies	?	Yes
Intel® Volume Management Device (VMD)	?	Yes
<b>Security &amp; Reliability</b>		
Intel vPro® Eligibility	?	Intel vPro® Enterprise, Intel vPro® Essentials
Intel® Threat Detection Technology (TDT)		Yes
Intel® Active Management Technology		

## I flag di ottimizzazione:

**Con:**

```
gcc Driver.c Multiply.c -o Mult -lm
```

**Execution time = 80.866 secondi**

**GFlops: 0.999206**

**Con l'ottimizzazione ottengo:**

```
gcc -O3 Driver.c Multiply.c -o Mult -lm
```

**Execution time = 18.676 secondi**

**GFlops: 4.326590**

**Con:**

```
icc -diag-disable=10441 Driver.c Multiply.c -o Mult
```

**Execution time = 16.856 secondi**

**GFlops: 4.793709**

**Il compilatore fa diverse ottimizzazioni legate all'architettura, La vettorizzazione in questo caso non riesce.**

## **Analisi con i tool intel:**

Il programma usa solo un thread sui 20 disponibili e per questo motivo non è efficiente per quanto riguarda il parallelismo.

Il compilatore non è riuscito a fare la vettorizzazione.

Durante l'esecuzione possiamo osservare non vengono utilizzati gli E-core ma solamente i Pcore.  
Per la speculazione utilizziamo soltanto il risultato del 55% delle istruzioni eseguite in quanto le altre 45 in quanto le altre non dovevano essere eseguire secondo la logica del programma.

JRE  
Access

YSES  
 tendering review)

ber of processed samples

⌚ Elapsed Time **17.357s**

⌚ IPC: 3.377  
SP GFLOPS: 0.000  
DP GFLOPS: 4.550  
x87 GFLOPS: 0.000  
Average CPU Frequency: 3.8 GHz

⌚ Logical Core Utilization **5.5% (1.108 out of 20) ↘**  
Physical Core Utilization: 9.1% (1.094 out of 12) ↗

⌚ Microarchitecture Usage **26.2% ↘ of Pipeline Slots**

⌚ P-Core:  
⌚ Retiring: 26.3% of Pipeline Slots  
⌚ Front-End Bound: 1.4% of Pipeline Slots  
⌚ Bad Speculation: 50.1% ↗ of Pipeline Slots  
⌚ Back-End Bound: 22.2% ↗ of Pipeline Slots  
⌚ Memory Bound: 1.4% of Pipeline Slots  
⌚ Core Bound: 20.8% ↗ of Pipeline Slots

⌚ E-Core:  
⌚ Retiring: 12.1% of Pipeline Slots  
⌚ Front-End Bound: 41.7% ↗ of Pipeline Slots  
⌚ Bad Speculation: 21.6% ↗ of Pipeline Slots  
⌚ Back-End Bound: 24.7% ↗ of Pipeline Slots  
⌚ Core Bound: 0.6% of Clockticks  
⌚ Memory Bound: 24.1% ↗ of Clockticks  
⌚ Back-End Bound Auxiliary: 24.7% ↗ of Pipeline Slots  
⌚ Resource Bound: 24.7% ↗ of Pipeline Slots

\*N/A is applied to metrics with undefined value. There is no data to calculate the metric.

⌚ Memory Bound **1.4% of Pipeline Slots**

⌚ P-Core:  
⌚ Memory Bound: 1.4% of Pipeline Slots

⌚ E-Core:  
⌚ Memory Bound: 24.1% ↗ of Clockticks

\*N/A is applied to metrics with undefined value. There is no data to calculate the metric.

- ⌚ Memory Bound ⓘ: **24.1% ↘** of Clockticks
- ⌚ Back-End Bound Auxiliary ⓘ: **24.7% ↘** of Pipeline Slots
- ⌚ Resource Bound ⓘ: **24.7% ↘** of Pipeline Slots

\*N/A is applied to metrics with undefined value. There is no data to calculate the metric.

## ⌚ Memory Bound ⓘ: 1.4% of Pipeline Slots

- ⌚ P-Core:
  - ⌚ Memory Bound ⓘ: **1.4%** of Pipeline Slots

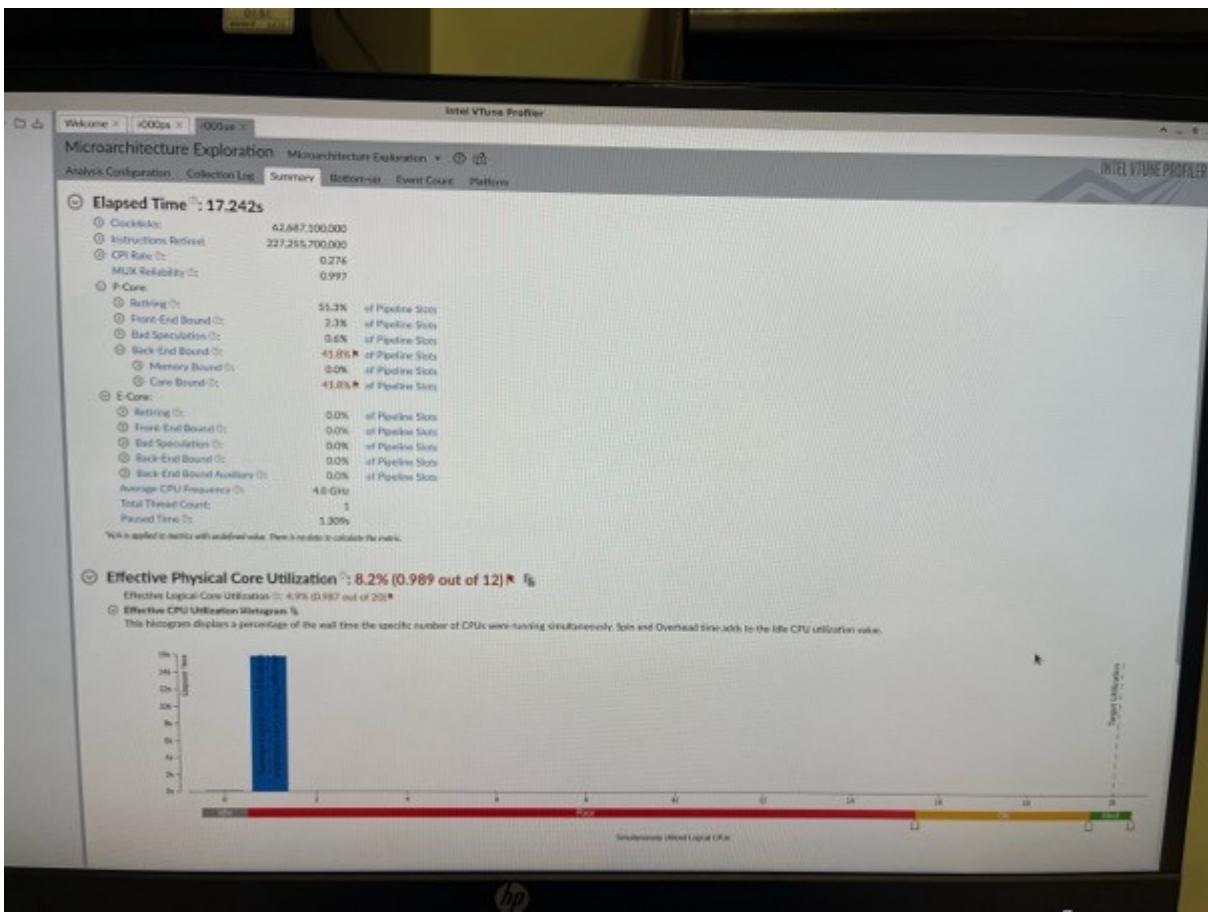
- ⌚ E-Core:
  - ⌚ Memory Bound ⓘ: **24.1% ↘** of Clockticks

\*N/A is applied to metrics with undefined value. There is no data to calculate the metric.

## ⌚ Vectorization ⓘ: 0.0% ↗ of Packed FP Operations

- ⌚ Instruction Mix:
  - ⌚ SP FLOPs ⓘ: 0.0% of uOps
  - ⌚ Packed ⓘ: 16.4% from SP FP
    - ⌚ 128-bit ⓘ: **16.4% ↗** from SP FP
    - ⌚ 256-bit ⓘ: 0.0% from SP FP
    - ⌚ Scalar ⓘ: **83.6% ↘** from SP FP
  - ⌚ DP FLOPs ⓘ: 35.4% of uOps
    - ⌚ Packed ⓘ: 0.0% from DP FP
      - ⌚ Scalar ⓘ: **100.0% ↗** from DP FP
    - ⌚ x87 FLOPs ⓘ: 0.0% of uOps
    - ⌚ Non-FP ⓘ: 64.6% of uOps

Metrics were collected from Big Cores only.

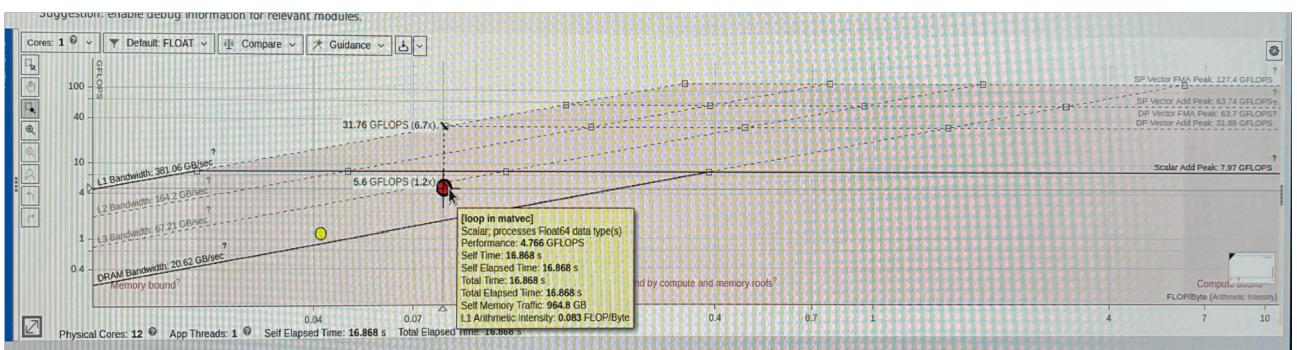


La memoryBound è pari allo 0% in quanto non ha carichi di istruzioni.

## analisi del modello roofline

Abbiamo un programma non vettoriale ma scalare che sarebbero una somma di numeri in una matrice 201 x 201 .

Stiamo usando la memoria L3 con una performance di 4.766 Gflops in 16 secondi . Non stiamo usando tutte le CPU e stiamo utilizzando un solo core.



Vettorizziamo

Possiamo osservare che con la vettorizzazione il loop del file driver.c era già stato vettorizzato mentre quello delle moltiplicazioni(Multiply.c) ce una dipendenza RAW che non permette la vettorizzazione.

**Con:**

```
icc -diag-disable=10441 -qopt-report=3 -qopt-report-phase=vec Driver.c Multiply.c
```

```
il tempo di esecuzione è 16.830 secondi  
gigaFlops = 4.801001
```

```
Multiply.optrpt Driver.optrpt
Intel(R) C Intel(R) 64 Compiler Classic for applications running on Intel(R) 64, Version 2021.10.0 Build
Compiler options: -diag-disable=10441 -qopt-report=3 -qopt-report-phase=vec -o Mult
Begin optimization report for: main()
Report from: Vector optimizations [vec]

LOOP BEGIN at Driver.c(54,2) inlined into Driver.c(141,2)
  remark #15542: loop was not vectorized: inner loop was already vectorized

LOOP BEGIN at Driver.c(55,3) inlined into Driver.c(141,2)
<Peeled loop for vectorization>
  LOOP END

  LOOP BEGIN at Driver.c(55,3) inlined into Driver.c(141,2)
    remark #15300: LOOP WAS VECTORIZED
    remark #15449: unmasked aligned unit stride stores: 1
    remark #15475: --- begin vector cost summary ---
    remark #15476: scalar cost: 113
    remark #15477: vector cost: 41.500
    remark #15478: estimated potential speedup: 2.690
    remark #15482: vectorized math library calls: 1
    remark #15487: type converts: 1
    remark #15488: --- end vector cost summary ---
  LOOP END

  LOOP BEGIN at Driver.c(55,3) inlined into Driver.c(141,2)
  <Remainder loop for vectorization>
  LOOP END
  LOOP END
  Driver.c(60,2) inlined into Driver.c(142,2)
```

Guardando il Driver.optrpt osserviamo che tutti i loop sono stati vettorizzati

**Con:**

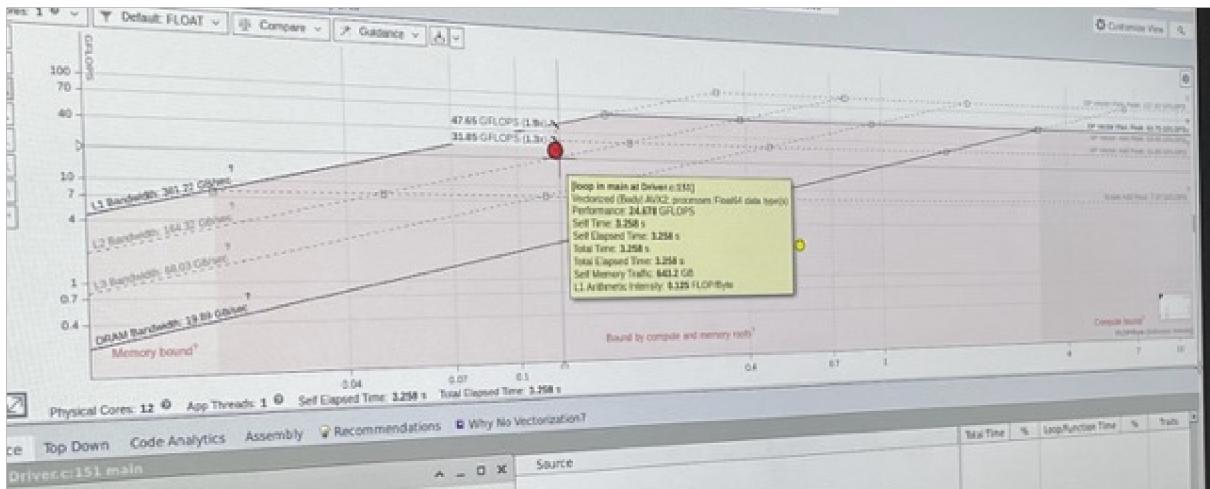
```
icc -diag-disable=10441 -qopt-report=3 -qopt-report-phase=vec -DNOfUNCCALL
Driver.c Multiply.c -o Mult
```

```
il tempo di esecuzione è 5.766 secondi  
gigaFlops = 14.014247
```

**Con:**

```
icc -diag-disable=10441 -qopt-report=3 -qopt-report-phase=vec -DNOfUNCCALL  
march=alderlake Driver.c Multiply.c -o Mult
```

```
il tempo di esecuzione è 4.056 secondi  
gigaFlops = 19.922084
```



## Parallelizzazione

**Il tempo di esecuzione è di 77.782**  
**GigaFlops = 16.604048**

```
ROW:801 COL: 801
Execution time is 77.282 seconds
GigaFlops = 16.604048
Sum of result = 12486803.999199
S6463898@sw118:~/Desktop/uegia$ icc -diag-disable=1044
```

Dopo la modifica:

```
#ifdef NOFUNCCALL
int i, j;
#pragma omp parallel for private(j)
for (i = 0; i < size1; i++) {
```

**Il tempo di esecuzione è di 24.630**  
**GigaFlops = 52.098944**

