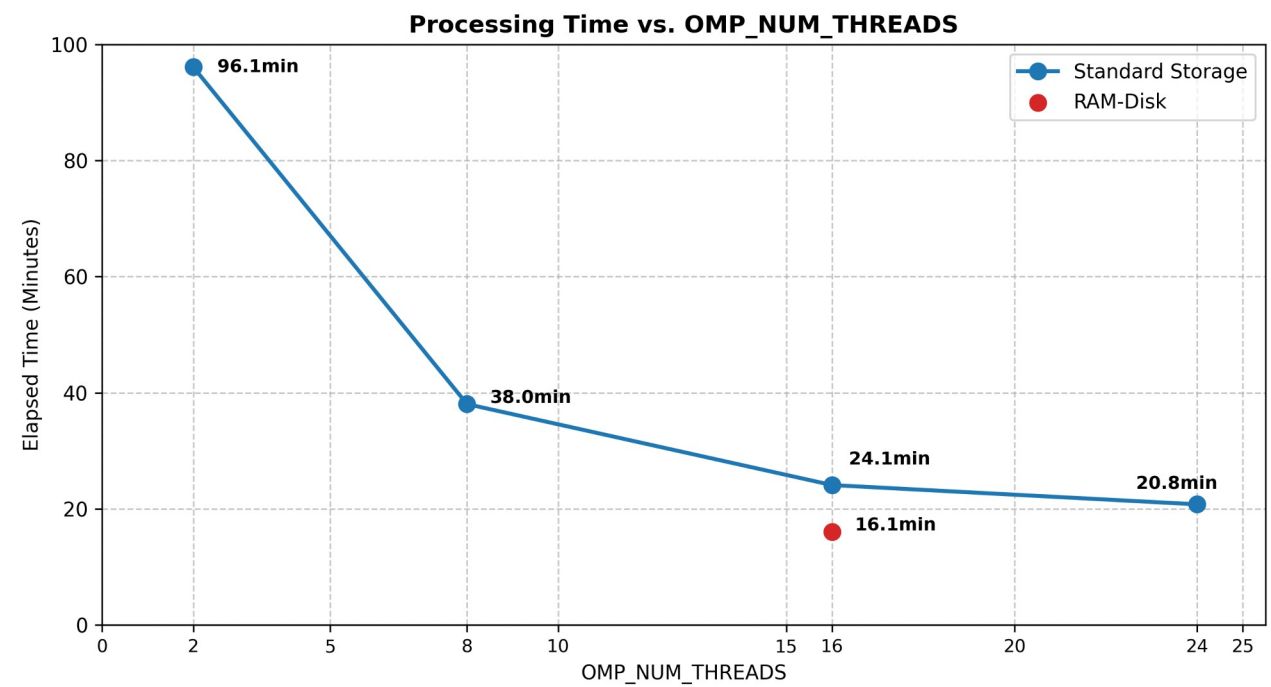


Overview

Four timings were made using the mechanical HDD. One timing was made using a RAM-DISK option. The number of threads were changed for each timing. The RAM-DISK was only checked using 16 threads.

Performance Summary Table

OMP_NUM_THREADS	Elapsed Time (HH:MM:SS)	Total Time (Minutes)	Storage Type
2	01:36:07	96.1 min	HDD
8	00:38:03	38.1 min	HDD
16	00:24:06	24.1 min	HDD
16	00:16:08	16.1 min	RAM-Disk
24	00:20:47	20.8 min	HDD



Thread count check - 2026-10-02

Raw data

```
threads_2_processing_time.txt
export OMP_NUM_THREADS=2
```

```
=====
Process Log
Start Time: 2026-09-30 13:40:32
End Time:   2026-09-30 15:16:39
Total Elapsed: 01:36:07 (HH:MM:SS)
```

```
=====
export OMP_NUM_THREADS=8
```

```
=====
Process Log
Start Time: 2026-10-02 14:26:09
End Time:   2026-10-02 15:04:12
Total Elapsed: 00:38:03 (HH:MM:SS)
```

```
=====
export OMP_NUM_THREADS=16
```

```
=====
Process Log
Start Time: 2026-09-30 15:18:31
End Time:   2026-09-30 15:42:37
Total Elapsed: 00:24:06 (HH:MM:SS)
```

```
=====
export OMP_NUM_THREADS=24
```

```
=====
Process Log
Start Time: 2026-09-30 15:48:01
End Time:   2026-09-30 16:08:48
Total Elapsed: 00:20:47 (HH:MM:SS)
```

Question:

the timing difference between the 16 and 24 core test was not so different. what would account for that? are cores being kept for reserve for other jobs?

Answer:

That flattening of the performance curve between 16 and 24 cores is a classic phenomenon in high-performance computing, and it comes down to a mix of **hardware architecture quirks** and **memory bandwidth saturation**.

No, your computer isn't holding cores in reserve for other jobs—Windows 11 is happily feeding threads to all 24 logical processors. Instead, a few other

bottlenecks are usually at play:

1. The Intel Hybrid Architecture (P-cores vs. E-cores)

Your i9-12900KF uses a “big.LITTLE” design:

- **8 Performance Cores (P-cores):** Feature Hyper-Threading, meaning they handle 16 of your threads. These are heavy hitters with massive cache and high clock speeds.
- **8 Efficient Cores (E-cores):** Do *not* have Hyper-Threading, meaning they handle the remaining 8 threads. These cores are physically smaller, slower, and have much less cache.

When you jumped from 16 to 24 threads, those extra 8 threads were pushed entirely onto the **E-cores**. Because E-cores are significantly slower than P-cores at heavy matrix math and 2D FFTs, they don't contribute nearly as much raw speedup, causing the overall completion time to plateau.

2. Memory Bandwidth Saturation (The Real Wall)

SAR processing routines like SLC_RFI_filtering do immense amounts of heavy floating-point calculation, but they also pull massive blocks of data back and forth from your system RAM.

- Once you hit 16 concurrent threads hammering the memory controller simultaneously, your RAM bandwidth gets saturated.
- Adding more threads (up to 24) means more cores are sitting idle waiting for data to arrive from memory—a classic “memory bottleneck” where throwing more CPU cores at the problem yields diminishing returns.

Summary of Your Scaling:

- **2 Cores:** 01:36:07 (*Slow, but steady*)
- **16 Cores:** 00:24:06 (*Massive jump because it utilizes all 8 fast P-cores + their hyperthreads*)
- **24 Cores:** (*Plateaus slightly because it starts leaning on the slower E-cores and hits memory bandwidth limits*)

Using 16 cores (or locking your threads strictly to the P-cores) is likely the **sweet spot** for maximum efficiency on this particular chip without hitting the architectural wall! How close were the 16 and 24-core times in the end?

Thread count check - 2026-10-02

RAM-DISK check

using a ram disk (osfmount) tool

<https://www.osforensics.com/tools/mount-disk-images.html>

dos command prompt

:: Create a 96GB RAM disk and mount it as drive R:

osfmount -a -t vm -s 96G -m R: -o format:ntfs,volume:RAMDisk

\$ mount

C:/gamma_sar/WIN_202607/Gamma/msys64 on / type ntfs (binary, noacl, auto)

C:/gamma_sar/WIN_202607/Gamma/msys64/usr/bin on /bin type ntfs
(binary, noacl, auto)

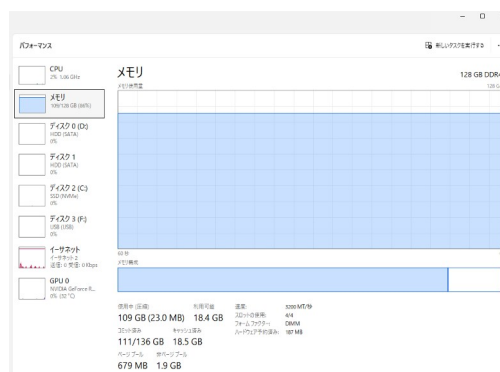
C: on /c type ntfs (binary, noacl, posix=0, user, noumount, auto)

D: on /d type ntfs (binary, noacl, posix=0, user, noumount, auto)

E: on /e type unknown (binary, noacl, posix=0, user, noumount, auto)

F: on /f type ntfs (binary, noacl, posix=0, user, noumount, auto)

R: on /r type ntfs (binary, noacl, posix=0, user, noumount, auto)



\$ pwd ; ls

/r/PALSAR3_demo_ramdisk

orig output_0930a_PALSAR3_ram-disk_demo.txt time_test_README_PALSAR3_demo.sh

input files were copied to the ram-disk

output was written to the ram-disk

gamma_sar system was on

\$ echo \$GAMMA_HOME

/c/gamma_sar/WIN_202607/Gamma/GAMMA_SOFTWARE-20260709

=====

Thread count check - 2026-10-02

number of threads 16

Process Log

Start Time: 2026-09-30 16:33:03

End Time: 2026-09-30 16:49:11

Total Elapsed: 00:16:08 (HH:MM:SS)

=====