Componentes Del Computador

Tema 1: Memoria– Conceptos básicos

Definición y jerarquía de memoria:

Según ***(Vaithianathan, 2025)*** la memoria en computación es un componente fundamental que permite guardar datos e instrucciones que el procesador necesita para realizar sus tareas diarias, facilitando así el funcionamiento eficiente del sistema. Es importante entender que su función principal es almacenar información temporalmente para que pueda ser accedida rápidamente y optimizar el procesamiento ya que sin la memoria la ejecución de programas sería mucho más lenta y complicada.[1]

A diferencia del almacenamiento y del disco duro, que guarda datos de forma permanente, la memoria RAM brinda una capacidad limitada, pero de acceso veloz mientras que el disco duro o SSD sirven para guardar la información a largo plazo incluso cuando la máquina está apagada. Por esta razón en la arquitectura de los sistemas de cómputo y es muy esencial distinguir entre la memoria y el que se usa para almacenamiento temporal durante la operación y el almacenamiento además que se encarga de mantener los datos de forma permanente y seguros.

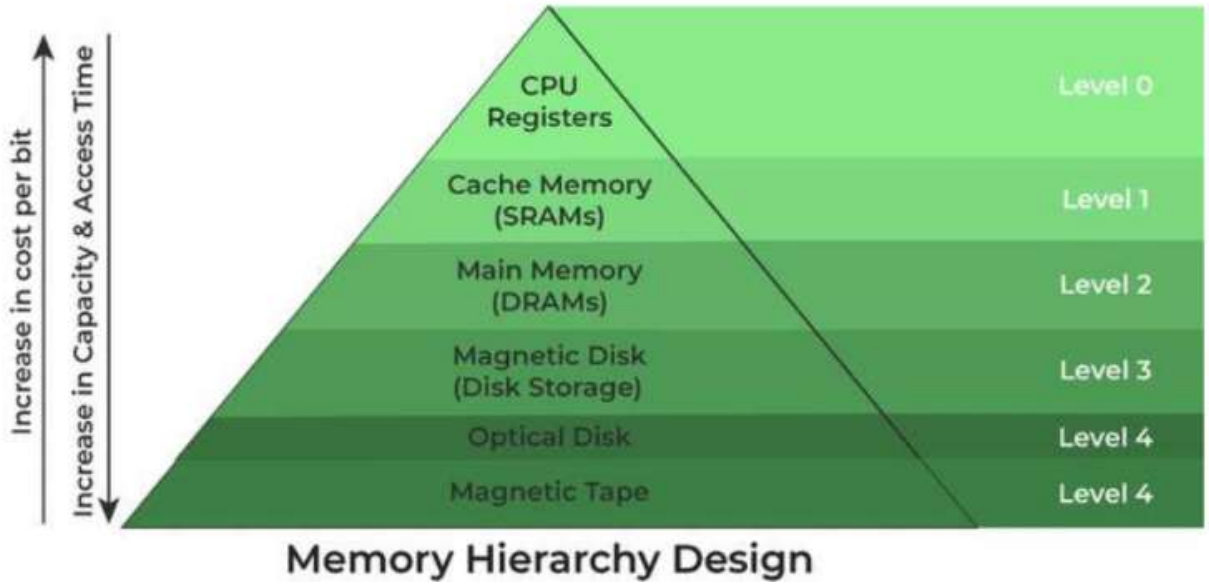


Figura 1. Pirámide de diseño de la jerarquía de memoria

Tipos de memoria (RAM, ROM, volátil vs no volátil):

Según ***(Liu et al. 2021)*** las memorias pueden ser volátiles o no volátiles, donde las RAM (Memoria de Acceso Aleatorio) y la RAM dinámica (DRAM) son ejemplos de memorias volátiles, las cuales requieren energía constante para mantener la información. En cambio, las memorias no volátiles, como las tecnologías NVMM (Memoria Principal No Volátil), conservan los datos incluso cuando se apaga el equipo, ofreciendo ventajas significativas en términos de persistencia y eficiencia energética. Estas memorias no solo permiten reducir los tiempos de recuperación ante fallos, sino que también facilitan una gestión más efectiva en arquitecturas híbridas.[2]

Además las NVMM son más rápidas que los discos duros tradicionales y tienen un menor consumo energético en comparación con las memorias Flash, lo que las convierte en una opción prometedora

para aplicaciones de Big data, seguridad y sistemas de almacenamiento avanzado. En definitiva, las NVMM están revolucionando el campo de la memoria, permitiendo nuevas funciones y diseños que mejoran el rendimiento y la sostenibilidad de los sistemas informáticos.

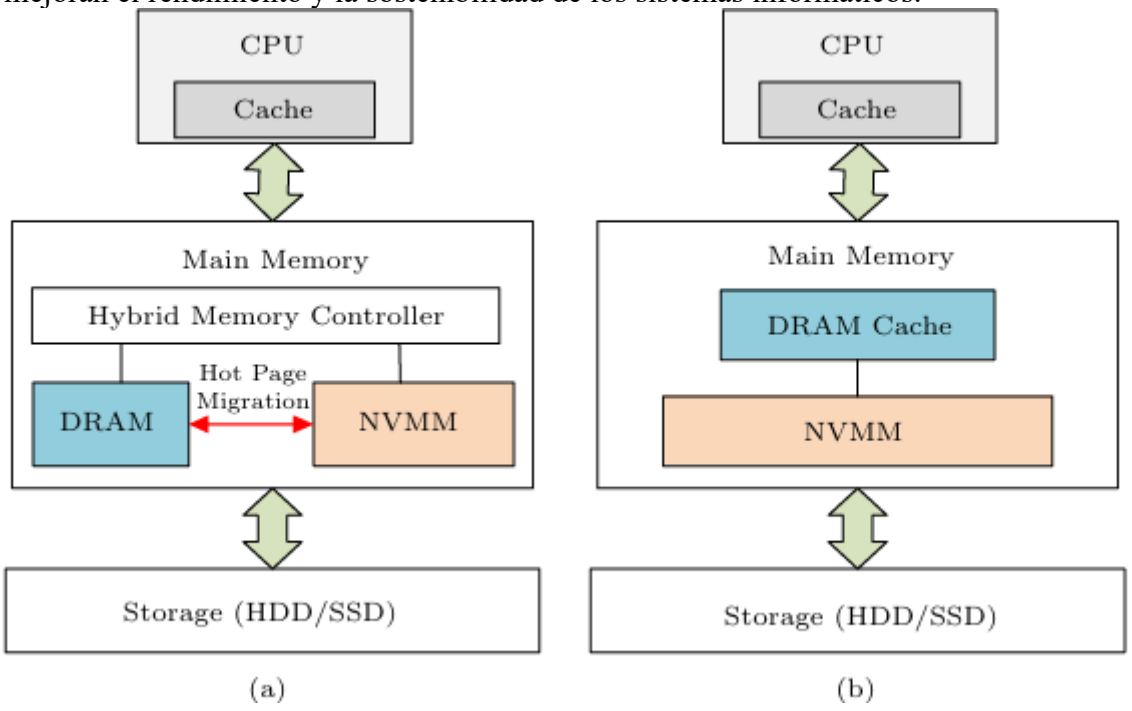


Fig.1. Hybrid memory architectures. (a) Horizontal at-addressable hybrid memory architecture. (b) Hierarchical hybrid memory architecture.

1. **Clasificación de la memoria.**
   - **Memoria principal** (RAM, ROM).
   - **Memoria secundaria** (disco duro, SSD).
   - **Memoria volátil** vs **no volátil**.
2. **Características principals.**
   - Capacidad (medida en bytes, KB, MB, GB, TB).
   - Velocidad de accesso.
   - Costo por bit almacenado.
3. **Tipos de memoria.**
   - RAM (DRAM, SRAM).
   - ROM (PROM, EPROM, EEPROM).
   - Memoria virtual.
4. **Funcionamiento básico.**
   - Cómo la CPU usa la memoria para ejecutar programas.
   - Ciclo de lectura y escritura.

Tema 2: Memoria Caché

Definición de memoria cache:
La memoria caché según ***(Oluwatosin Abayomi et al. 2020*)** es una memoria ultrarrápida que se encuentra entre la CPU y la RAM y su función principal es acelerar el acceso a los datos que la

CPU necesita constantemente. La existencia de la caché surge porque acceder directamente a la RAM sería mucho más lento, así que la caché guarda las instrucciones y datos más utilizados para que la CPU pueda acceder a ellos rápidamente.[3]
Estos datos almacenados en la caché pueden estar en diferentes niveles, como L1, L2 y L3, siendo L1 la más rápida y cercana a la CPU, pero con menor capacidad mientras que L3 es más grande pero un poco más lenta. Cuando la CPU busca un dato primero consulta la caché si lo encuentra, eso se llama un hit y el acceso es muy rápido, pero si no se produce una miss y se busca en la memoria más lenta lo que demora más y reduce el rendimiento.

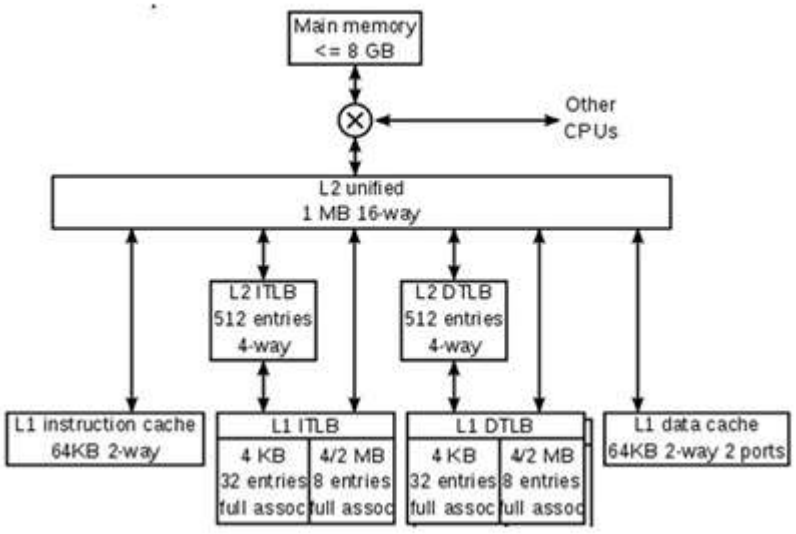


Figura 1. Jerarquía de caché del núcleo K8 de la CPU AMD Athlon 64.

1. **Ubicación y funcionamiento**
    - Dónde está (integrada en la CPU o en la placa base).
    - Cómo guarda datos que la CPU usa con frecuencia.
2. **Tipos de caché**
    - Nivel 1 (L1), Nivel 2 (L2) y Nivel 3 (L3).
    - Diferencias de velocidad, tamaño y función.
3. **Importancia y beneficios**
    - Mejora del rendimiento del sistema.
    - Ejemplos prácticos (cargar más rápido un programa usado).
4. **Relación con la memoria principal**
    - Cómo decide la CPU qué datos guardar en caché.
    - Qué pasa cuando el dato no está en la caché (fallo de caché).

La memoria caché actúa como un puente eficiente que reduce el tiempo de espera en el procesamiento de datos al mantener almacenados los datos más usados cerca del procesador.

Optimización y políticas de caché:
Según ***(Jamet et al. 2020)*** la diferencia en las latencias entre la caché de último nivel (LLC) y la memoria principal ha hecho necesario desarrollar políticas eficientes para gestionar la caché, ya que así se puede mejorar el rendimiento del sistema. Aunque muchos estudios se han enfocado en perfeccionar el método de sustitución LRU, estas investigaciones generalmente evalúan solo ciertas cargas de trabajo, como las de las pruebas SPEC CPU 2006 y 2017, que no representan toda la variedad de trabajos actuales en HPC.[4]

En este artículo, se analizan diferentes cargas de trabajo, incluyendo procesamiento de grafos, ciencia e industria, en políticas modernas de sustitución LLC como MPPP, Glider, Hawkeye, SHiP, DRRIP y SRRIP, y se observa que las políticas superan a LRU en las pruebas tradicionales, no logran captar los patrones complejos de acceso en los trabajos de HPC y Big Data. Por eso, además, presentan dos nuevas políticas derivadas de MPPP, como MS-MPPPB, que emplea múltiples muestreadores para mejorar aún más el rendimiento y mejorar el desempeño general del sistema.

Reducción del movimiento de datos y nuevas arquitecturas:
Las jerarquías tradicionales según el artículo ***(ASPLOS XXV: Twenty-Fifth Inte… 2020)*** de memoria obligan a realizar cálculos en lugares alejados de los datos necesarios, lo que genera ineficiencias en el procesamiento. Además, los enfoques convencionales no aprovechan bien la localidad de los datos, limitando la eficiencia. Por eso, proponemos Memory Services, un modelo de programación flexible que permite centrar los cálculos en los datos a lo largo de toda la jerarquía de memoria, optimizando su uso.[5]

En este contexto se ha diseñamos y evaluamos Livia, una arquitectura que programa dinámicamente las tareas y datos en la ubicación ideal para reducir el movimiento de datos y mejorar el rendimiento. Gracias a Livia, las cargas de trabajo irregulares pueden acelerarse hasta 2,4 veces y reducir el consumo energético entre 1,2 y 4,7 veces, con una mínima sobrecarga de área en sistemas multinúcleo.

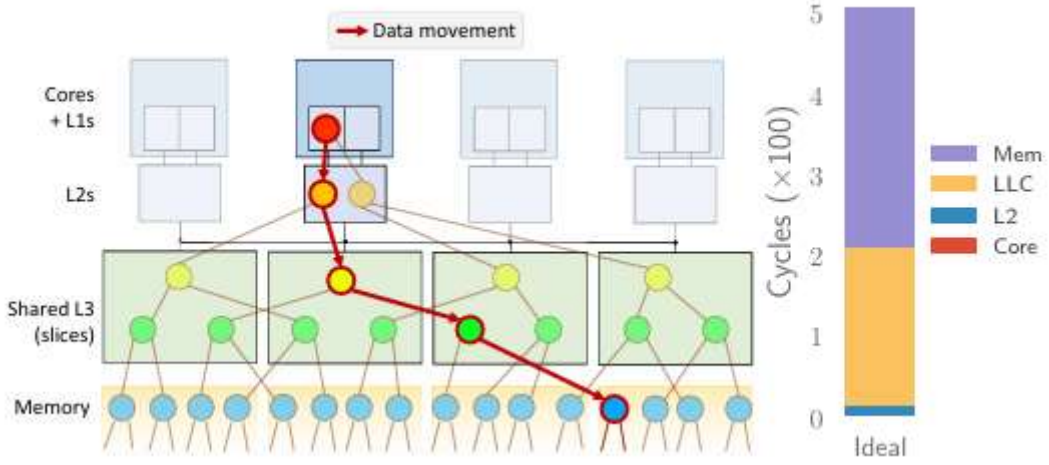


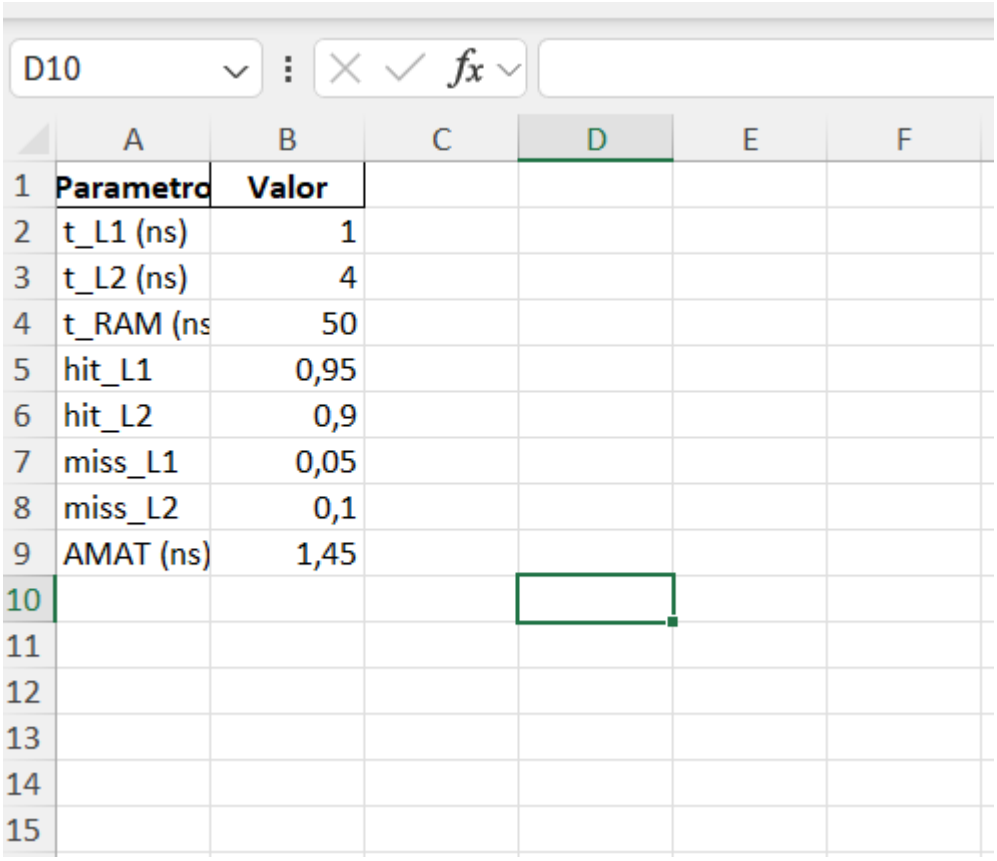
Fig. 3. Movimiento ideal de datos: el árbol recorre la jerarquía de la memoria, desde la raíz en L1 hasta la hoja en la memoria.

Conclusión:
La memoria y la memoria caché son piezas clave para que un computador trabaje rápido y de forma eficiente. La RAM almacena datos temporales para que la CPU los use al instante y mientras que la caché actúa como un acceso aún más veloz a la información más usada. Las nuevas tecnologías buscan no solo aumentar la velocidad sino que también buscan optimizar cómo y dónde se guardan los datos y así logrando que cada proceso sea más ágil y consuma menos recursos del sistema.

Procedimientos:

Estudio de Memoria:

- **Realizar ejercicios prácticos para calcular tiempos de acceso y capacidad en diferentes niveles de memoria.**



	A	B	C	D	E	F
1	Parametro	Valor				
2	t_L1 (ns)	1				
3	t_L2 (ns)	4				
4	t_RAM (ns	50				
5	hit_L1	0,95				
6	hit_L2	0,9				
7	miss_L1	0,05				
8	miss_L2	0,1				
9	AMAT (ns)	1,45				
10						
11						
12						
13						
14						
15						

Figura 1 - AMAT (ejemplo fijo)

Archivo 1 - Capacidades De Caché

	Nivel	Sets	Asociatividad	Tamaño_bloque_Bytes	Capacidad_KB
1	L1	128	4	64	32.0
2	L2	1024	8	64	512.0

Figura 2 - Capacidades de caché

❖ **Análisis del rendimiento de la memoria caché mediante ejemplos teóricos y simulaciones.**

A1 | hit_L1

	A	B	C	D	E	F	G	H
1	hit_L1	hit_L2	t_L1_ns	t_L2_ns	t_RAM_ns	AMAT_ns		
2	0,8	0,9	1	4	50	2,8		
3	0,81	0,9	1	4	50	2,71		
4	0,82	0,9	1	4	50	2,62		
5	0,83	0,9	1	4	50	2,53		
6	0,84	0,9	1	4	50	2,44		
7	0,85	0,9	1	4	50	2,35		
8	0,86	0,9	1	4	50	2,26		
9	0,87	0,9	1	4	50	2,17		
10	0,88	0,9	1	4	50	2,08		
11	0,89	0,9	1	4	50	1,99		
12	0,9	0,9	1	4	50	1,9		
13	0,91	0,9	1	4	50	1,81		
14	0,92	0,9	1	4	50	1,72		
15	0,93	0,9	1	4	50	1,63		
16	0,94	0,9	1	4	50	1,54		
17	0,95	0,9	1	4	50	1,45		
18	0,96	0,9	1	4	50	1,36		
19	0,97	0,9	1	4	50	1,27		
20	0,98	0,9	1	4	50	1,18		
21	0,99	0,9	1	4	50	1,09		
22								
23								

Figura 1 - Simulación AMAT (L1 variable)

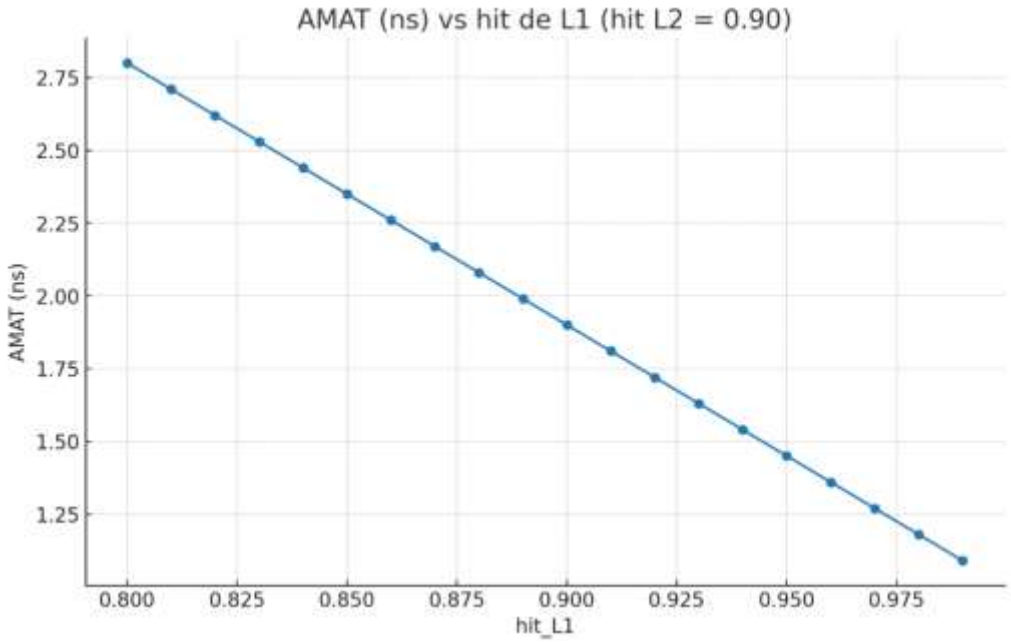


Figura2 - AMAT (ns) vs hit de L1 (hit L2 = 0.90)

Bibliografía:

[1] M. Vaithianathan, “Memory Hierarchy Optimization Strategies for HighPerformance Computing Architectures,” *International Journal of Emerging Trends in Computer Science and Information Technology*, vol. 6, pp. 24–35, 2025, doi: 10.63282/3050-9246.ijetcsit-v6i1p103.

[2] H.-K. Liu *et al.*, “A survey of non-volatile main memory technologies: State-of-the-arts, practices, and future directions,” *J Comput Sci Technol*, vol. 36, no. 1, pp. 4–32, 2021, doi: 10.1007/s11390.

[3] A. Oluwatosin Abayomi, A. Abayomi Olukayode, and G. Oluwole Olakunle, “An Overview of Cache Memory in Memory Management,” *Automation, Control and Intelligent Systems*, vol. 8, no. 3, p. 24, 2020, doi: 10.11648/j.acis.20200803.11.

[4] A. V. Jamet, L. Alvarez, D. A. Jimenez, and M. Casas, “Characterizing the impact of last-level cache replacement policies on big-data workloads,” in *Proceedings - 2020 IEEE International Symposium on Workload Characterization, IISWC 2020*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020, pp. 134–144. doi: 10.1109/IISWC50251.2020.00022.

[5] *ASPLOS XXV : Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems : March 16-20, 2020, Lausanne, Switzerland*. The Association for Computing Machinery, 2020.