International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 03 | March - 2024 SJIF Rating: 8.176 ISSN: 2582-3930

Cache Memory Organization and Virtual Memory

Rupam Sardar

Budge Budge Institute of Technology

Abstract:

Cache memory is mainly inculcated in systems to overcome the gap created in-between the main memory and CPUs due to their performance issues. Since, the speed of the processors is ever-increasing, so a need arises for a faster speed cache memory that can definitely assist in bridging the gap between the speed of processor and memory. Therefore, this paper proposes architecture circumscribed with three improvement techniques namely victim cache, sub-blocks, and memory bank. These three techniques will be implemented one after other to improve and make the speed and performance of cache comparative to main memory.

Introduction

Due to its faster access time than other memories, cache memory is a crucial component of the computer that may have an impact on how well a program runs. It is the quickest member of the memory hierarchy and gets close to CPU speed. It operates on the "locality of reference" characteristic, which states that references to memory at any given time are often limited to a small number of localized regions of memory. The program's variables and active instructions will both be kept in the cache memory;

Hierarchy Memory Technology

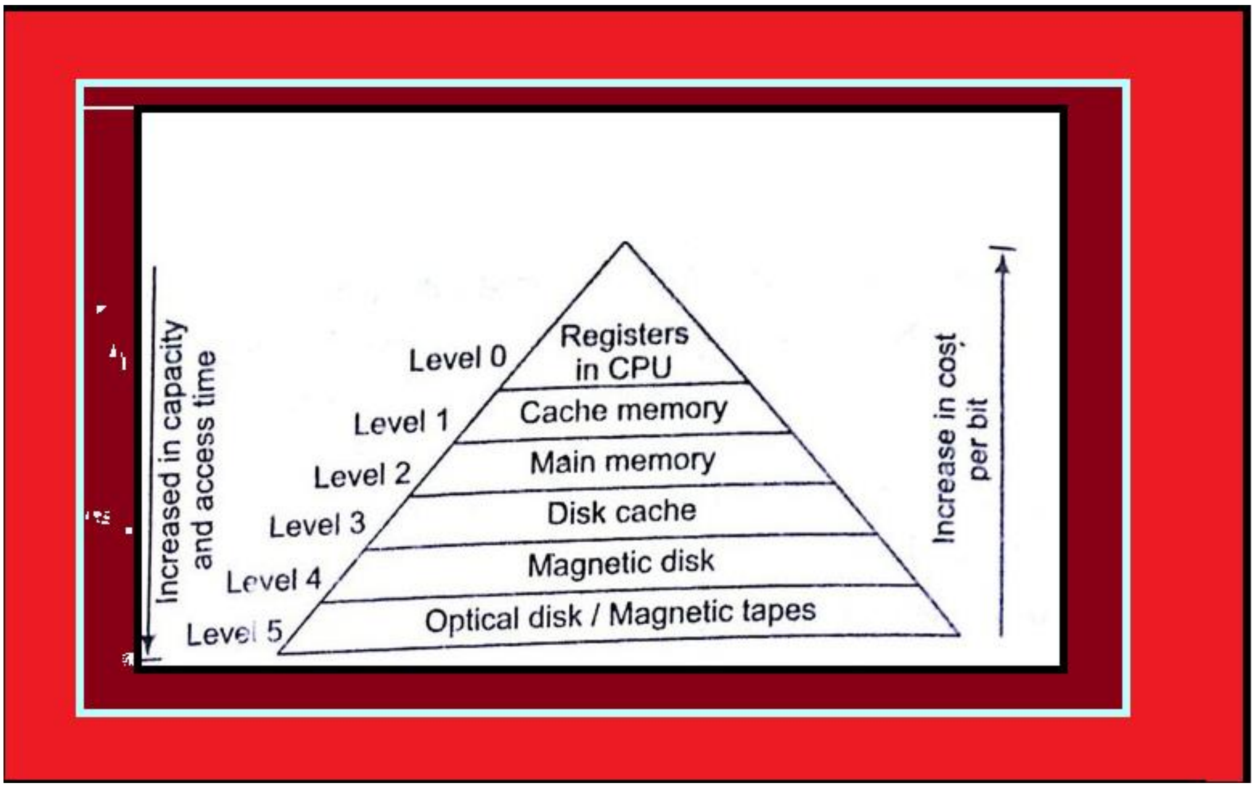
Why Does the System Need Memory Hierarchy?

One of the most important components of computer memory is memory hierarchy, which maximizes the amount of memory that is available. The memory is divided into several tiers, each of which has a distinct size, price, etc. While some memory has a slightly higher storage value but is a little slower, other memory kinds, such as cache and main memory, are faster than other types of memory but are also a little smaller and more expensive. Different forms of memory have different speeds at which data may be accessed; some have faster access than others.

Memory Hierarchy Types
There are two primary categories for this memory hierarchy design:

Secondary memory, also known as external memory, is made up of magnetic tape, optical disk, and magnetic disk—peripheral storage devices that the processor can access through an I/O module.

Main memory, cache memory, and CPU registers make up internal memory, often known as primary memory. The processor can access this directly.



Organization:

Registers
The CPU contains tiny, fast memory modules called registers. The most often utilized information and instructions are kept in them. With a typical capacity of 16–64 bits, registers have the smallest storage capacity and the fastest access time.

1. 2. Memory Cache
Situated near the CPU, cache memory is a compact, quick memory module. It holds recently accessible instructions and often utilized data from the main memory. By giving the CPU instant access to frequently used data, cache memory is intended to reduce the amount of time it takes to retrieve data.

2. 3. Primary Memory
The primary memory of a computer system is called main memory, or RAM (Random Access Memory). Although it is slower than cache memory, it can store more data. The data and instructions that the CPU is now using are kept in main memory.

Main Memory Types
Static RAM: Static RAM uses flip flops to store binary data, which is only valid while power is applied. It

implements cache memory and offers a faster access time.
Dynamic RAM: It uses a capacitor to store binary data as a charge.

3. 4. Backup Storage

Secondary storage is a non-volatile memory type with a higher storage capacity than main memory. Examples of this type of storage are hard disk drives (HDD) and solid-state drives (SSD). When the CPU is not using the data or instructions, they are stored there. In the memory hierarchy, secondary storage has the slowest access speed and is usually the most affordable type of memory.

5. Magnetic Record

Simply put, magnetic disks are circular plates made of plastic, metal, or a magnetized substance. Within the computer, magnetic disks are often used and operate at a high speed.
Dynamic RAM: It uses a capacitor to store binary data as a charge.

4. 6. Adhesive

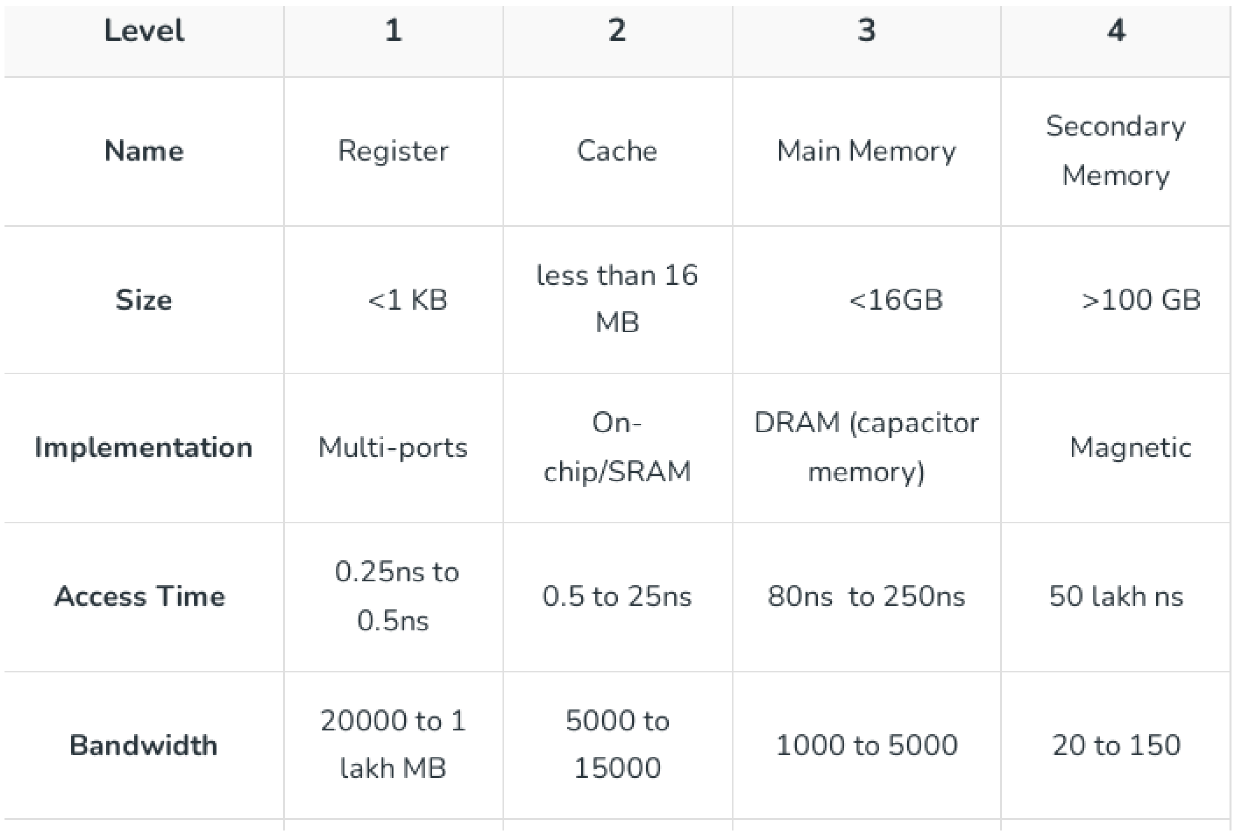
All that magnetic tape is is a recording medium that is magnetic and wrapped in a plastic film. It is typically utilized for data backups. It takes some time for a computer to access a magnetic tape since the computer's access time is a little slower in this scenario.

5. Features of the Memory Hierarchy

Capacity: The total amount of data that a memory device is capable of storing. The capacity rises as we proceed down the Hierarchy from top to bottom.
Access time is the amount of time that passes between a read or write request and the data becoming available. The access time rises as we descend the Hierarchy from top to bottom.
Achievement: The performance differential between the CPU registers and Main Memory grew earlier when the computer system was constructed without a Memory Hierarchy design because of the significant access time difference. This causes the system to perform worse, necessitating improvement. Memory Hierarchy Design was implemented as part of this improvement, which boosts system performance.

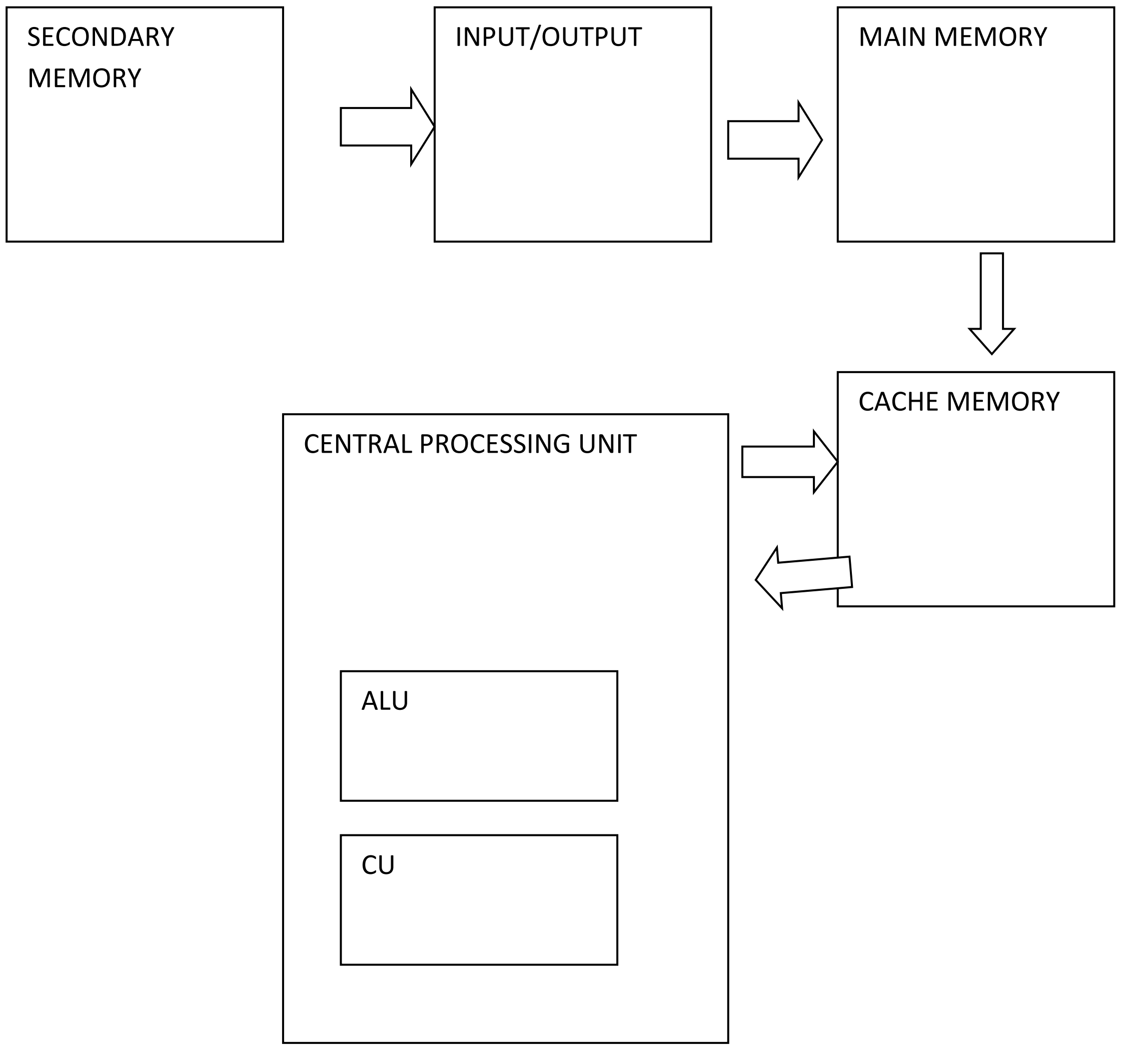
6. The Memory Hierarchy's Benefits

It facilitates improved memory management and the removal of some destruction.
It facilitates the data's distribution throughout the computer system.
Both money and time are saved for the customer.



Level	1	2	3	4
Name	Register	Cache	Main Memory	Secondary Memory
Size	<1 KB	less than 16 MB	<16GB	>100 GB
Implementation	Multi-ports	On-chip/SRAM	DRAM (capacitor memory)	Magnetic
Access Time	0.25ns to 0.5ns	0.5 to 25ns	80ns to 250ns	50 lakh ns
Bandwidth	20000 to 1 lakh MB	5000 to 15000	1000 to 5000	20 to 150

A phenomenon known as "locality of reference" describes how a computer program frequently accesses the same set of memory locations for a predetermined amount of time. Stated differently, Locality of Reference describes the computer program's inclination to access instructions whose addresses are close to each other. The program's loops and subroutine calls are the primary examples of the locality of reference attribute.

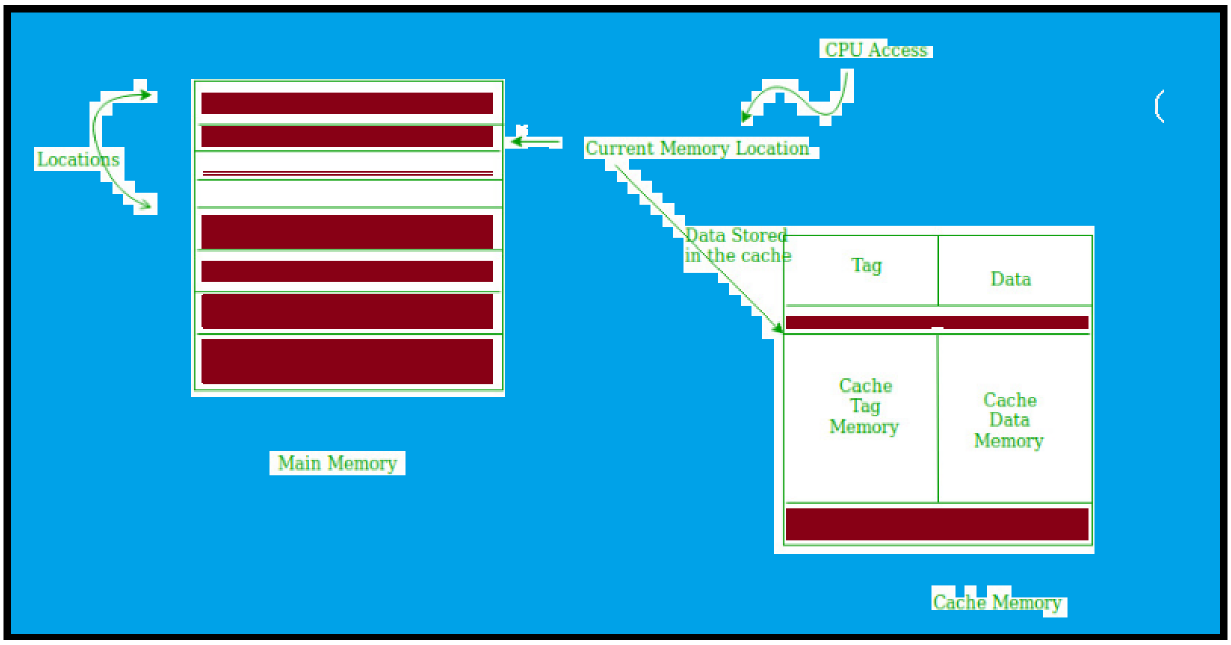


When a program control processing unit encounters a loop, it will continually refer to the collection of instructions that make up the loop.

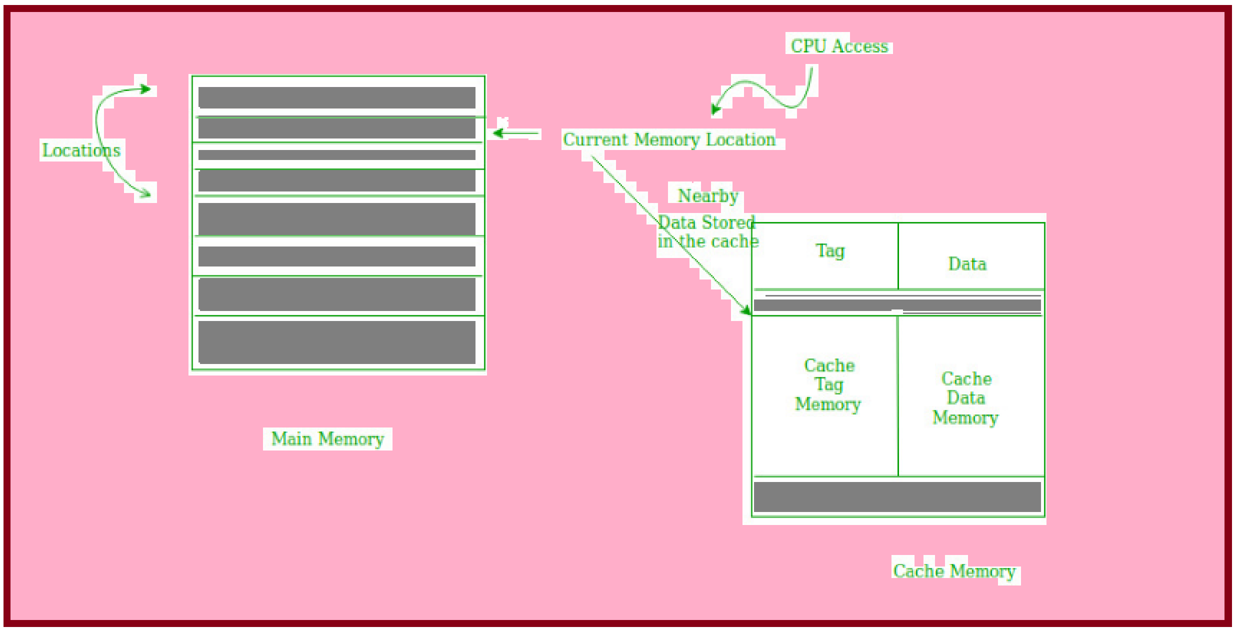
When a subroutine is called, the set of instructions is retrieved from memory each time. The localization of references to data items results in repeated references to the same data item.

The CPU should retrieve the necessary data or instruction, use it, and stop. However, what happens if the same data or instruction is needed again?Since we already know that main memory is the memory that accesses the slowest, the CPU must once more reach the same address in main memory. The second option is to save the information or instruction in the cache memory, which will enable a significantly faster fetch if it becomes necessary again shortly.

The locality of reference principle serves as the foundation for the cache operation. There are two methods for retrieving information or commands from main memory and storing them in cache memory. These two approaches are as follows:



The term "spatial locality" refers to the possibility that an instruction or piece of data that is currently being fetched from memory may be needed shortly. This is not the same as the temporal locality. While we were discussing the actual memory location that was being fetched in temporal locality, now we are discussing memory locations that are almost located.



Cache Performance:

The hit ratio is used to gauge the cache's performance. Cache hits occur when a CPU searches memory for information and locates it in the cache memory. Cache miss occurs when the CPU accesses main memory to locate the desired data or instruction if it cannot be found in cache memory.

Hit + Miss = Total CPU Reference

Hit Ratio(h) = Hit / (Hit+Miss)

Miss Ratio = 1 - Hit Ratio(h)

Miss Ratio = Miss / (Hit+Miss)

Average access time of any memory system consists of two levels: Cache and Main Memory. If *Tc* is time to access cache memory and *Tm* is the time to access main memory then we can write:
Tavg = Average time to access memory

For simultaneous access

Tavg = h * Tc + (1-h)*Tm

For hierarchial access

Tavg = h * Tc + (1-h)*(Tm + Tc)

Three Crucial Pointers for Reducing Cache Misses
Determine the Cache Lifespan's Expiration Date. The information in your cache must be written into memory following the initial request each time it is cleared.
Boost the Capacity of Your Random Access Memory (RAM) or Cache...
Choose the Best Cache Policies for Your Unique Situations.

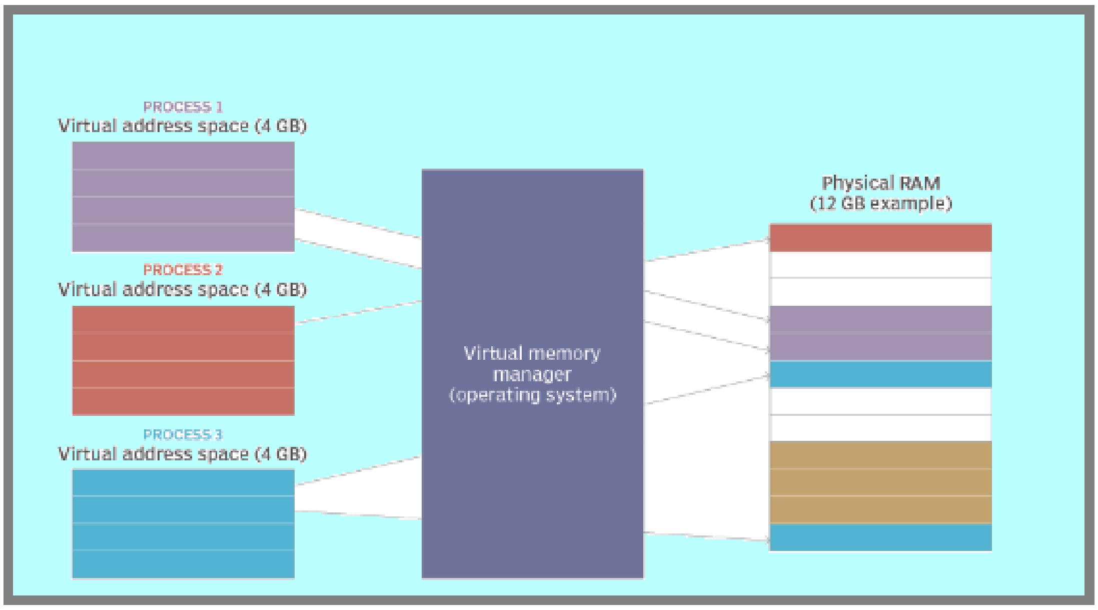
Virtual memory: what is it?
Secondary memory can be used as though it were a part of the main memory thanks to a memory management approach called virtual memory. An operating system (OS) on a computer frequently uses a method called virtual memory.

In order to make up for physical memory deficits, a computer can temporarily move data from random access memory (RAM) to disk storage through the use of virtual memory, which combines hardware and software. A computer can use secondary memory as if it were main memory by mapping memory chunks to disk files.

How virtual memory works Virtual memory uses both hardware and software to operate. When an application is in use, data from that program is stored in a physical address using RAM. A memory management unit (MMU) maps the address to RAM and automatically translates addresses. The MMU can, for example, map a logical address space to a corresponding physical address. If, at any point, the RAM space is needed for something more urgent, data can be swapped out of RAM and into virtual memory. The computer's memory manager is in charge of keeping track of the shifts between physical and virtual memory. If that data is needed again, the computer's MMU will use a context switch to resume execution.

What advantages does virtual memory offer?
The following are some benefits of virtual memory use:

Compared to main memory, it can manage twice as many addresses.
It makes it possible to use multiple applications at once.
It spares users from needing to install memory modules when RAM runs out and frees apps from maintaining shared memory.
When only a portion of a program is required for execution, its speed increases.
The memory isolation feature provides improved security.
It makes it possible for several larger apps to operate at once.


PROCESS 1
Virtual address space (4 GB)
PROCESS 2
Virtual address space (4 GB)
PROCESS 3
Virtual address space (4 GB)
Virtual memory manager (operating system)
Physical RAM (12 GB example)

What restrictions apply while utilizing virtual memory?
While virtual memory provides advantages, there are certain trade-offs to be aware of, including:

If an application is using virtual memory, it will operate more slowly.
A computer must map data between virtual and real memory, which slows it down even more and necessitates additional hardware support for address translations.
The quantity of secondary storage and the computer system's addressing architecture both place restrictions on the extent of virtual storage.

REFERENCES

[1] V. Chaplot, “Virtual Memory Benefits and Uses,” International Journal of Advance Research in Computer Science and Management Studies, vol. 4, no. 9, 2016.

[2] C. Zhang, F. Vahid, and R. Lysecky, “A Self-Tuning Cache Architecture for Embedded Systems,” ACM Transactions on Embedded Computing Systems, vol. 3, no. 2, pp. 407-425, 2004.

[3] A. Agrawal and S. D. Pudar, “Column-associative Caches: a Technique for Reducing the Miss Rate of Direct-mapped Caches,” in Proc. of the 20th Annual International Symposium on Computer Architecture, 1993, pp. 179-190.

[4] S. Kumar and P. K. Singh, “An overview of modern cache memory and performance analysis of replacement policies,” in Proc. of the IEEE International Conference on Engineering and Technology (ICETECH), 2016, Coimbatore, India, pp. 210-214.

[5] S. S. Omran and I. A. Amory, “Implementation of LRU Replacement Policy for Reconfigurable Cache Memory Using FPGA,” in Proc. Of the International Conference on Advanced Science and Engineering, 2018, pp. 13-18.

[6] K. Westerholz, S. Honal, J. Plankl, and C. Hafer, “Improving performance by cache driven memory management,” in Proc. of the 1st IEEE Symposium on High Performance Computer Architecture, 1995, Raleigh, USA, pp. 234-242.

[7] Y. A. Divya, “An Efficient Virtual Memory using Graceful Code,” International Journal of Trend in Scientific Research and Development vol. 3 no. 4, pp. 623-626, 2019.

[8] M. Biglari, K. Barijough, M. Goudarzi, and B. Pourmohseni, “A Fine- Grained Configurable Cache Architecture for Soft Processors,” in Proc. of the 18th CSI International Symposium on Computer Architecture and Digital Systems (CADS), 2015.

[9] A. Q. Ahmad, M. Masoud, and S. S. Ismail, “Average Memory Access Time ReductionVia Adding Victim Cache,” International Journal of Applied Engineering Research, vol. 11, no. 19, pp. 9767-9771, 2016.

[10] Y. Kim and Y. H. Song, “Impact of processor cache memory on storage performance,” in Proc., of the International SoC Design Conference (ISOCC), 2017, Seoul, pp. 304-305.

[11] W. P. Dias and E. Colonese, “Performance Analysis of Cache and Scratchpad Memory in an Embedded High Performance Processor,” in Proc. of the 5th International Conference on Information Technology: New Generations, 2008, Las Vegas, USA, pp. 657-661.

[12] K. Singh and S. Khanna, “Split Memory Based Memory Architecture with Single-ended High Speed Sensing Circuit to Improve Cache Memory Performance,” in Proc. of the 6th International Conference on Signal Processing and Communication (ICSC), 2020, Noida, India, pp. 188-193.

[13] R. Kolanski, “A logic for virtual memory, “Electronic Notes in Theoretical Computer Science, vol. 217, pp. 61-77, 2008.

[14] S. P. Smith and J. Kuban, “Modelling and Enhancing Virtual Memory Performance in Logic Simulation,” in Proc. of the IEEE International Conference on Computer-Aided Design, January 1988, pp. 264-265.

[15] M. T. Banday and M. Khan, “A study of recent advances in cache memories,” in Proc. of the International Conference on Contemporary Computing and Informatics (IC3I), 2014, Mysore, India, pp. 398-403.