Assignment 3: Exploring Memory Hierarchy Design in gem5

Pavani Chavali

Department of Computer Science, University of The Cumberlands

MSCS-531-A01: Computer Architecture and Design

Professor. Vanessa Cooper

Date: 9/13/2025

Part1: Understanding Memory Hierarchy

Significance of memory hierarchy in achieving high-performance computing:

Memory hierarchy is crucial for high-performance computing because it solves the core problem of the significant speed difference between the CPU and main memory. Modern CPUs operate at speeds where they can process data far quicker than main memory (DRAM) can deliver it. If a computer had a single, flat memory system, the CPU would spend most of its time waiting for data, drastically slowing down the system. The memory hierarchy overcomes this by using a tiered system. This system consists of multiple levels of memory, each with a different balance of speed, cost, and capacity. The fastest, most expensive, and smallest memory is placed closest to the CPU, while the slower, cheaper, and larger memory is further away. This strategic design ensures that the data most likely to be needed is readily available in the fastest memory, closing the performance gap and boosting overall system speed.

Memory Technologies and their Placement:

Memory hierarchy is build on simple faster and more expensive memory should be closer to CPU, while slower, cheaper memory is used for larger storage and further away from CPU. Memory tier system starts with registers on the CPU chip, which are fastest, smallest and holds CPU's frequently used data. They are made from SRAM, a fast technology which doesn’t need a refresh. Next comes registers in the order which are basically caches also made from SRAM. Caches L1,L2,L3 store frequently accessed data from main memory, with each level being larger but slower than the one above. This ensures CPU can access fata with minimal delay, avoiding performance bottleneck of constantly retrieving information from main memory.

Next comes main memory or RAM, which is made primarily of DRAM. DRAM is slower than SRAM because it requires periodic refreshing, but it is much more affordable and denser, allowing for the large capacities needed for modern computing. This is where active programs and the operating system are stored. The hierarchy extends further to secondary storage like SSDs and HDDs. These are much slower but non-volatile, meaning they retain data when the power is off. While not part of the CPU's immediate access path, they are essential for long-term storage of all system data. This strategic use of different memory technologies, from fast SRAM to large DRAM and persistent secondary storage, is what makes a computer system efficient and balanced.

Advanced Cache Optimization:

Sophisticated techniques are used to build on the basic memory hierarchy and minimize expensive cache misses. Prefetching proactively loads data into the cache that the CPU is predicted to need soon, which works well for predictable workloads. Victim caches are small buffers that temporarily hold data recently evicted from the main cache, allowing for a quick retrieval if the data is needed again, which is especially good at reducing conflict misses. Finally, cache partitioning divides the cache among different processors or applications. This prevents one process from hogging the cache, ensuring better performance and fairness across a multi-core system. While these techniques add complexity, they are essential for maximizing the throughput and efficiency of high-performance systems.

Virtual Memory and Virtual Machines:

Virtual memory is a powerful concept that separates a program's logical address space from the computer's physical memory, providing key benefits for modern systems. The process relies on the Memory Management Unit and page tables to translate a program's virtual addresses into physical addresses. This enables programs to be larger than the available physical RAM, as only the necessary parts called pages are loaded into memory, with the rest stored on a hard drive or SSD. If a program needs a page that isn't currently in RAM, a page fault occurs, prompting the operating system to retrieve it. This system not only manages memory efficiently but also ensures process isolation, preventing one program from interfering with another.

Building on these ideas, virtual machines create a full virtualized hardware environment. This introduces a more complex, nested paging scenario where a hypervisor manages the physical memory and presents a virtualized physical memory to the guest operating system. The hypervisor essentially translates the guest OS's memory requests before the MMU translates them to the actual physical addresses. This layered approach is a significant challenge to manage efficiently for optimal VM performance.

Cross-Cutting Issues:

Cost and Power Consumption

The fundamental challenge in designing a memory hierarchy is balancing cost, speed, and capacity. Faster memory like SRAM is expensive and consumes more power, while slower memory like DRAM is cheaper and denser. The hierarchy solves this by using a small amount of fast, costly memory near the CPU and a large amount of slow, affordable memory further away.

This clever design gives the system the illusion of having a vast, high-speed memory without the prohibitive cost and helps manage power consumption, as a higher cache hit rate reduces the need for frequent, power-intensive access to main memory.

Complexity and Workload impact

The logic required for advanced cache techniques like prefetching and victim caches adds significant design complexity to the processor and the effectiveness is highly dependent on workload. Workload with predictable access patterns like scientific computing will benefit greatly from prefetching, while random access patterns like database transactions may see minimal gains. While designing workloads and target applications must be considered to create balanced and effective hierarchy**.**

Emerging Trends:

The future of memory hierarchies is being shaped by two major trends. First, non-volatile memory (NVM) like Intel Optane is bridging the gap between fast DRAM and slower storage. NVM is nearly as fast as DRAM but retains data like flash memory, allowing for persistent memory that applications can access directly without the overhead of a file system. Second, in-memory computing or processing-in-memory is gaining traction. This approach moves computation directly onto the memory chip to drastically reduce the energy and latency costs associated with moving data, which is a key bottleneck in high-performance computing. These evolving technologies suggest a future where the traditional memory hierarchy becomes more tightly integrated and heterogeneous, fundamentally altering how we design high-performance systems.