Assignment 3: Part 1 Understanding the Memory Hierarchy

Nischal Joshi

University of Cumberlands

MSCS - 531: Computer Architecture and Design

Dr. Charles Lively

October 6, 2024

Memory Hierarchy & its Significance

Memory hierarchy is a technique implemented in modern computer architecture and design to establish a faster and better flow of data to and from the processor by focusing on a core concept of maintaining a perfect balance between various aspects like cost, power consumption, and design technique complexity.

Memory Technologies

Each level of the memory hierarchy includes a unique memory component with its specialty and weaknesses and would require a trade-off among each other. Generally, these components are split into two categories: primary memory and secondary memory. The central memory components used in modern computing systems are given below.

- **Registers:** Registers are the fastest unit in the memory hierarchy and are kept closest to the processors. Although fast, they have the smallest storage capacity. Also, the manufacturing cost is very high, and because of the limited storage, it often holds a limited amount of used data.
- **SRAM (Cache Memory):** Static Random Access Memory(SRAM) forms the cache components of the memory hierarchy because of the high speed and minimal lags. It holds the data bits in a flip-flop structure as long as the power is supplied. This causes it to consume more energy. Additionally, it takes more space than the DRAM. Hence, SRAMs are used to form the cache component in the memory unit and are kept closest to the processor. Although they possess high throughput and low latency, they are expensive and more extensive, so they must be adjusted accordingly (Jacob et al., 2010).

- **DRAM**: Dynamic Random Access Memory(DRAM) is used mainly as the main memory in most modern computing systems. It is more cost-effective and denser per bit than the SRAM and stores active data that does not fit in the cache memory (Jacob et al., 2010). It is faster than other secondary storage like hard drives and magnetic disks but slower than the SRAM, which makes it an essential middle layer in the hierarchy connecting cache to secondary storage.
- **Secondary Storage:** Secondary storage falls at the lower level of the memory hierarchy and is used to retain data permanently even after the power supply is off. Hard Disk Drives (HDDs) and solid-state Drives (SSD) are the most common examples of secondary storage in modern computing devices. In contrast to main memory and cache, they are slow but have more storage space and are cheaper.

Additionally, new emerging technologies like non-volatile memory NVM and high-bandwidth memory HBM are being experimented on. In general, this hierarchy ensures that frequently accessed data by the processor is retrieved quickly from faster cache memories, and less frequently accessed data is fetched from more extensive, slower memory.

Advanced Cache Optimization

Caches are frequently used data that help optimize the CPU's performance. Given below are a few of the advanced cache optimization techniques available:

- Prefetching: Prefetching is a technique in which frequently used data are fetched into the cache before the processor requests it. It can be hardware- or software-based. Dedicated logic analyzes the access pattern in hardware-based prefetching, whereas the software-based prefetching compiler provides the respective instruction. Prefetching

reduces the number of cache misses but increases the risk of fetching unnecessary data, leading to cache pollution and wasted bandwidth (Wulf & McKee, 1995).

- **Victim Cache:** A victim cache is a small associative cache usually kept between the primary cache to store the recently removed cache line from the primary cache. In case of any cache miss, data is checked in the victim cache before moving to the main memory (Smith, 1982).
- **Cache Partitioning:** Cache partitioning is a technique in which the cache memory is divided into different units, which allows each thread to access the respective cache data without affecting the other thread.

Virtual Memory and Virtual Machines

Virtual memory is an abstraction of physical memory that allows secondary storage to mimic a part of the main memory. The core idea is to free up the RAM by swapping data that has not been used recently over to secondary storage. This facilitates process isolation, memory overcommitment, and efficient page management through techniques like paging and swapping. Virtual machines work with it to allow multiple OS to run concurrently on the same hardware using nested address translation.

Page Table and Address Translation.

The addresses in the virtual memory need to be mapped to the physical addresses. This translation is carried out by the Main Memory Unit (MMU) and is managed using the page table that maps the virtual addresses to physical addresses. Each page is generally 4KB or more extensive and is managed by the OS. This process of converting the virtual address into an actual location in the memory address is known as the address transition. Translation Lookaside Buffer

(TLB) is a hardware cache designed to store the recent address translation, and it helps the translation process by preventing frequent lookups in the page table.

Page Replacement Algorithm

Place Replacements Algorithms are the means the system uses to decide which page needs to be removed for physical memory when it is complete. Some standard replacement algorithms include Least Recently Used (LRU), First-in-First-Out(FIFO) and Clock Algorithm Jacob, Ng, & Wang, 2010). LRU removes the page that has not been used for the longest time. FIFO, on the other hand, removes the pages in the order they are loaded into the memory regardless of the frequency of their use.

Similarly, the clock algorithm is an advanced form of LRU in which pages are arranged in a circular list, and the decision to keep or remove the page is made based on the reference bit.

Cross-Cutting Issue and Trade-offs: A fundamental assumption of a memory unit is fast, cost-efficient, and less complicated. Designing a memory hierarchy involves balancing these components:

- **Cost and Performance:** As we move to the top of the memory hierarchy, the cost of the technologies being used increases. Components like SRAM are speedy and have better performance but are costly, so increasing the size will significantly increase the system cost(Burr et al., 2016).
- **Power Consumption:** Memory units may consume a significant proportion of system power. Fast and efficient memory components like SRAM and registers consume more power than lower-level units.
- **Workload Diversity:** Different types of workloads have different requirements for memory access. A particular hierarchy might only be effective for some kinds of

workloads. Memory hierarchy design must consider the nature of the workloads to optimize system performance (Wulf & McKee, 1995).

Emerging Technologies

Various technologies like High Bandwidth and Non-Volatile memory are being explored to optimize the memory unit. Non Volatile Memories possess lower latency but provide more persistent storage than traditional SSD and HDDs (Burr et al., 2016). On the other hand, HBM offers both high bandwidth and low power consumption. It is based on 3D stacking technology, which allows many chips to be stacked in a small space.

References

Burr, G. W., Kurdi, G., Scott, J., & Shenoy, R. (2016). Overview of candidate device technologies for storage-class memory. *IBM Journal of Research and Development*,

Jacob, B., Ng, S. W., & Wang, D. T. (2010). *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann.

Wulf, W. A., & McKee, S. A. (1995). Hitting the memory wall: Implications of the obvious. *ACM SIGARCH Computer Architecture News*, 23(1), 20–24.

Smith, A. J. (1982). Cache memories. *ACM Computing Surveys*, 14(3), 473–530.