Conceptual Foundations of Computer Memory

Understanding the relationship between Main Memory, Secondary Memory, Virtual Memory, Memory Management, and Replacement Algorithms.

By

Kenneth G. Foster

May 2, 2010

Since computers were first conceived, the number one priority has always been to squeeze the most processing capability possible out of the system within the shortest period of time. This was driven by both cost and processing availability (Stallings, 2009 p. 55). Initially computers only had one type of memory and all instructions were executed in a first in, first out linear format. Then came the advent of disk drives and output devices. It became very clear that these slower input / output (I/O) interfaces caused lost processing capability (Stallings, 2009 p. 62). As computers evolved scientists began to build subsystems to move these time intensive processes off the processor and into secondary memory and systems (figure 1).

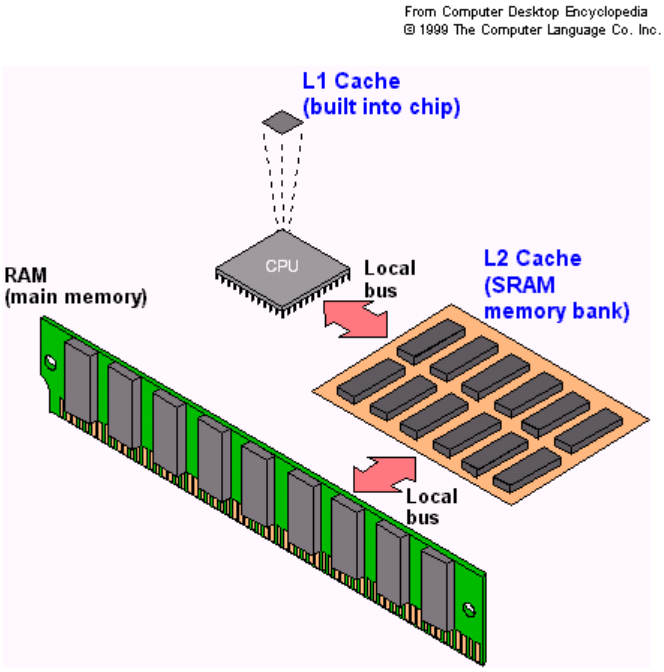

Figure 1. Memory Types Overview

Computers engage in a complex optimization effort to execute the most data possible in the least number of clock cycles. Computers execute these instructions via their processor, which is streamed the instructions from its system memory. There are two types of memory, Main Memory and Secondary Memory (Stallings, 2009 p. 315). Main Memory is the only place in which the processor can receive instruction and data in which to execute (Kozierok, 2001).

When the information the processor requires is not in main memory, this is called a page fault (Red Hat Inc., nd). If a process that requires execution is in secondary memory, the computer must use a system called Memory Management to retrieve the information and move it into main memory so the processor can attempt to re-execute the instruction (Stallings, 2009 p. 315).

In order to improve performance, main memory has been designed with certain considerations in an effort to improve information flow to the processor. Main memory is faster than secondary memory (Stallings, 2009 p. 315). Whenever possible, a system's main memory should operate at or faster than the processor clock speed. These aides in the prevention of system bottlenecks (OEMPCWorld Inc., nd). Because main memory is faster, it is also more expensive than secondary memory. Main memory is also volatile, and does not retain information when power is withdrawn (Stallings, 2009 p. 315). Because of these factors, main memory is never used for long-term storage; this task is relegated to secondary memory.

Secondary memory is less expensive than main memory. Its storage capacities can be as large at terabytes on a single drive (DEW Associates Corp., 2003). It is slower than main memory because of the requirement for Input/Out (I/O) operations across the system bus (OEMPCWorld Inc., nd). Because of its relative cost and size, long-term storage requirements are handled by secondary memory (Kozierok, 2001). Secondary memory also allows for an extension of main memory through a process called Virtual Memory (HowStuffWorks.com, 2001).

One of the earliest problems with computers was the need to have all the programs, processes, and data execute in available memory. This became quite complicated and required strict coding practices to ensure the programmer did not violate the memory space of another process. Today's systems have the luxury of Virtual Memory (HowStuffWorks.com, 2001).

Virtual Memory leverages storage space in the secondary memory (hard drive) to write information not immediately required by the processor for execution. This allows the data to be made available in a more orderly, on demand fashion. Secondary memory does come with a down side; it's slower than main memory so the system must manage its use expeditiously to prevent system bottlenecks (Stallings, 2009 p. 312).

In order for Virtual memory to work, it must take the space allocated on the physical disk and parse the space into manageable chunks (figure 2) through a process called overlaying (Stallings, 2009 p. 315). There are two overlaying techniques used by Virtual Memory, segmentation and paging (Stallings, 2009 p. 315). Segmentation is where a process is divided into a number of smaller parts called segments. These segments are stored on the disk drive and don't require contiguous storage space (Stallings, 2009 p 316). Paging on the other hand divides a set space of secondary memory into frames of equal size. These frames are then loaded into main memory from secondary memory when that process is preparing to execute (Stallings, 2009 p 316).

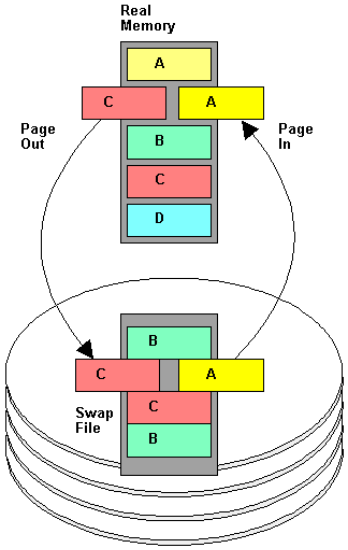

Figure 2. Virtual Memory Disk Swapping Overview

In order for the processor to be able to execute instructions, it must know where to stream those instructions from memory. Main memory is organized in a linear or single dimension address space, referred to as Linear Organization (Stallings, 2009 p. 314). However the process of moving information between main and secondary memory is a complicated process that requires updated memory pointers and a predictive management algorithms to determine when it is perceived safe to move information between the two memories. The process of determining when and where to move information in memory is handled by the Memory Manager (Stallings, 2009 p. 315).

The memory manager is tasked with five requirements, Relocation, Protection, Sharing, Logical Organization, and Physical Organization (Stallings, 2009 p. 312). Relocation is process of moving memory from one area to another and updating it location in the memory reference table (Stallings, 2009 p. 313). Sharing allows a number of processes to access a single copy of a process, rather than spawn multiple copies consuming excess memory. Protection is the process of the hardware preventing accidental or intentional interference by other processes with the memory space owned by another process. Protection can allow reading or a shared memory space, but restricts writing to only authorized processes (Stallings, 2009 p. 313).

The Memory manager uses a replacement algorithm to determine the more judicious method to determine which items can be moved into secondary memory (figure 3). This algorithm is executed by the operating system and the algorithm type used is set by the developer. There are four basic algorithms for determining how page memory is replaced. They are Optimal, First-in-First-Out (FIFO), Least Recently Used (LRU), and Clock (Stallings, 2009 p. 368). The Optimal algorithm is based on forecasted need using the length of last time since the block was last used. The First-in-First-Out (FIFO) algorithm assesses page frames in a

circular fashion, rotating each frame out in a round robin (Stallings, 2009 p. 369). The Least Recently Used (LRU) algorithm determines page replacement based on the last time of reference (Stallings, 2009 p. 368). The clock algorithm uses a circular layout, setting an additional bit for each frame initially set at zero. As the memory manager rotates in a circular fashion, similar to FIFO, it checks the usage bit. If the block is used, the bit is set to one. If after its rotation the block still remains at zero, the page is swapped out. If the bit is set to 1 then the algorithm reset the bit to zero and the process starts again. The replacement algorithm must be properly tuned and implemented if you want to avoid thrashing; the unnecessary delay and writing to and from secondary memory (Stallings, 2009 p. 349).

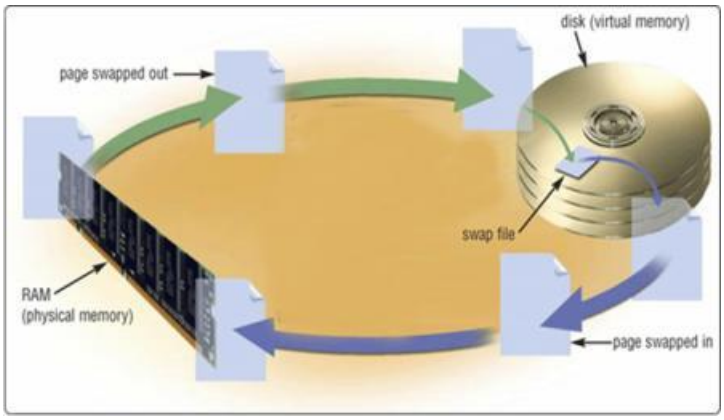

Figure 3. Memory Management Overview

References:

Callari, F. (1996). *Shortest remaining time next (SRTN)*. Retrieved May 2, 2010, from Operating Systems 304: http://www.cim.mcgill.ca/~franco/OpSys-304-427/lecture-notes/node44.html

Carnegie Mellon University. (nd). *Rate Monotonic Analysis.* Retrieved May 2, 2010, from Software Engineering Institute: http://www.cs.fsu.edu/~baker/realtime/restricted/notes/s6-r.pdf

Delis, A. (nd). *WSClock – A Simple and Effective Algorithm for VMM*. Retrieved April 23, 2010, from http://pdc-amd01.poly.edu/~wein/cs6243/ppts/wsclock.ppt

DEW Associates Corp. (2003). *Hard Drive Size Limitations and Barriers*. Retrieved April 23, 2010, from http://www.dewassoc.com/kbase/index.html

HowStuffWorks.com. (2001, July 20). *What is virtual memory*. Retrieved April 2010, 2010, from http://computer.howstuffworks.com/question684.htm

Kozierok, C. M. (2001, April 17). *System Memory*. Retrieved April 24, 2010, from http://www.pcguide.com/ref/ram/index.htm

McCraw, T. (nd). *Virtual Memory Page Replacement Algorithms*. Retrieved April 23, 2010, from http://people.msoe.edu/~mccrawt/resume/papers/CS384/mccrawt_cs384_virtual.pdf

OEMPCWorld Inc. (nd). *Memory Access vs CPU Speed*. Retrieved April 24, 2010, from http://www.oempcworld.com/support/Memory_Access_vs_CPU_Speed.htm

Red Hat Inc. (nd). *4.4.1. Page Faults*. Retrieved April 24, 2010, from http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/en-US/Introduction_To_System_Administration_/s2-memory-concepts-paging.html

Sha, L., & Sathaye, S. S. (1995, September). *Distributed System Design Using Generalized Rate Monotonic Theory*. Retrieved May 2, 2010, from Defense Technical Information Center: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.9376&rep=rep1&type=pdf

Stallings, W. (2009). *Operating Systems.* Upper Saddle River, NJ: Pearson Prentice Hall.

University of Bridgeport. (nd). *Operating Systems (CS/CPE 408) : CPU Scheduling* . Retrieved May 2, 2010, from http://www1bpt.bridgeport.edu/sed/projects/cs503/Spring_2001/kode/os/scheduling.htm

Virginia Tech. (nd). *Virtual Memory*. Retrieved April 24, 2010, from http://courses.cs.vt.edu/~csonline/OS/Lessons/VirtualMemory/index.html

Yaashuwanth, C., & Ramesh, R. (2010). Intelligent Time Slice for Round Robin in Real Time Operating Systems. *International Journal of Robotics Research*, 126-131.