

Lecture #13 – Virtual Memory

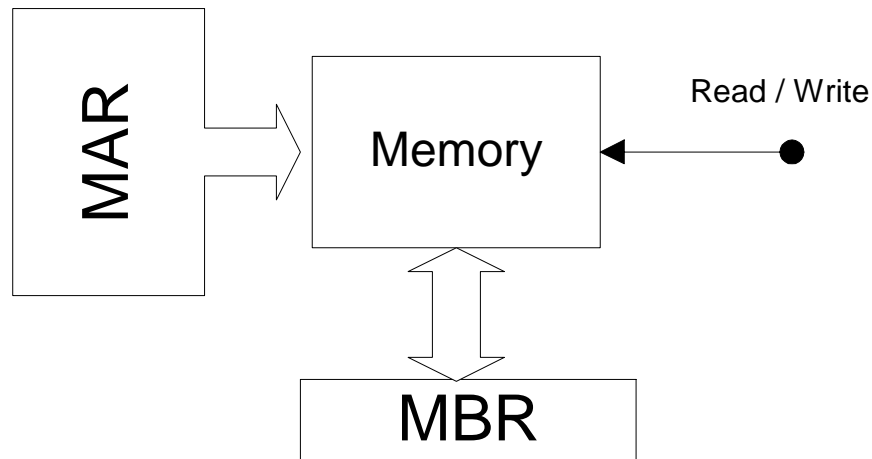
- Virtual Memory Access

Remember in a simple system, memory is accessed as follows:

MAR = Memory Address Register

MBR = Memory Buffer Register

In general these are logically used as follows:



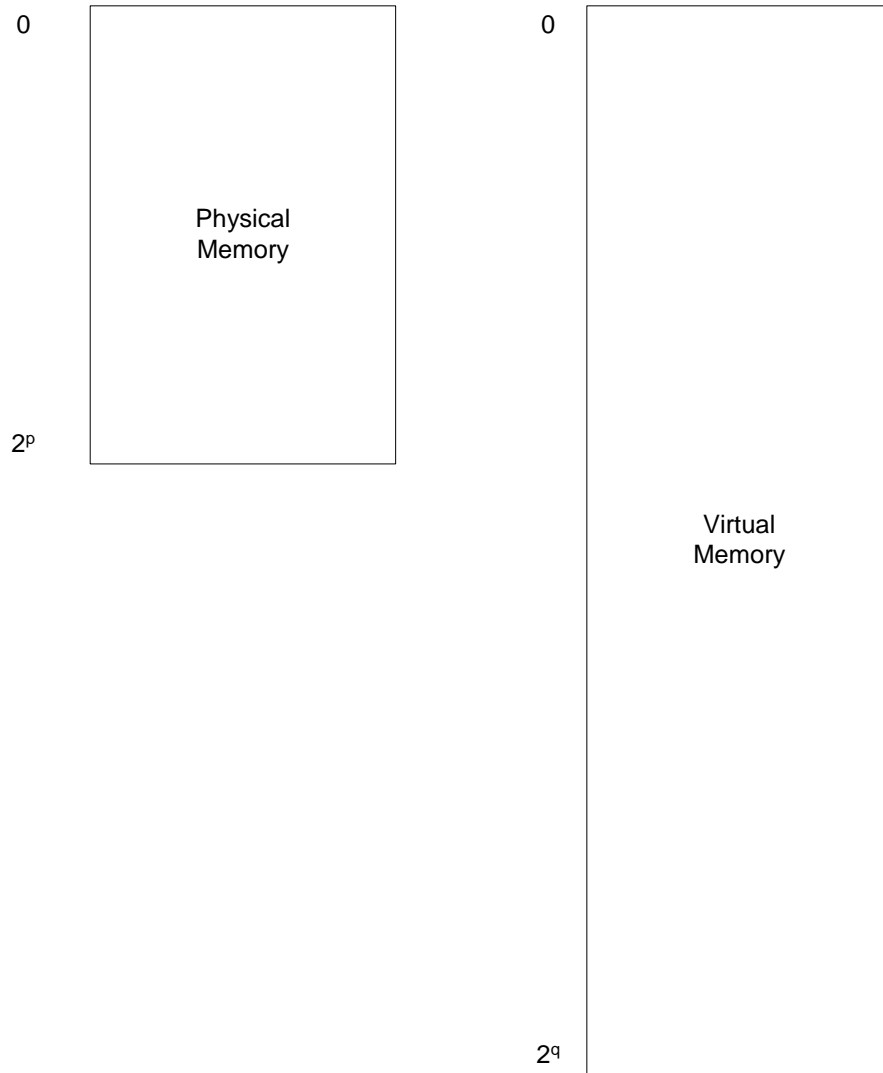
During the fetch cycle, the PC is loaded into the MAR, and the MAR determines the location of the next instruction to execute.

In Virtual memory, we want to run a program that is too big to fit into physical memory.

Now the PC is loaded into a Virtual MAR. The Virtual MAR is translated into a Physical MAR. The Physical MAR determines the location in memory of the next instruction.

It is possible that a virtual address does not exist in physical memory. In this case, the memory associated with the virtual address is on disk (or secondary storage) someplace. The operating system must load the virtual memory into physical memory before the next instruction can be executed.

The process of stopping execution and loading physical memory from disk is called a “page fault”.



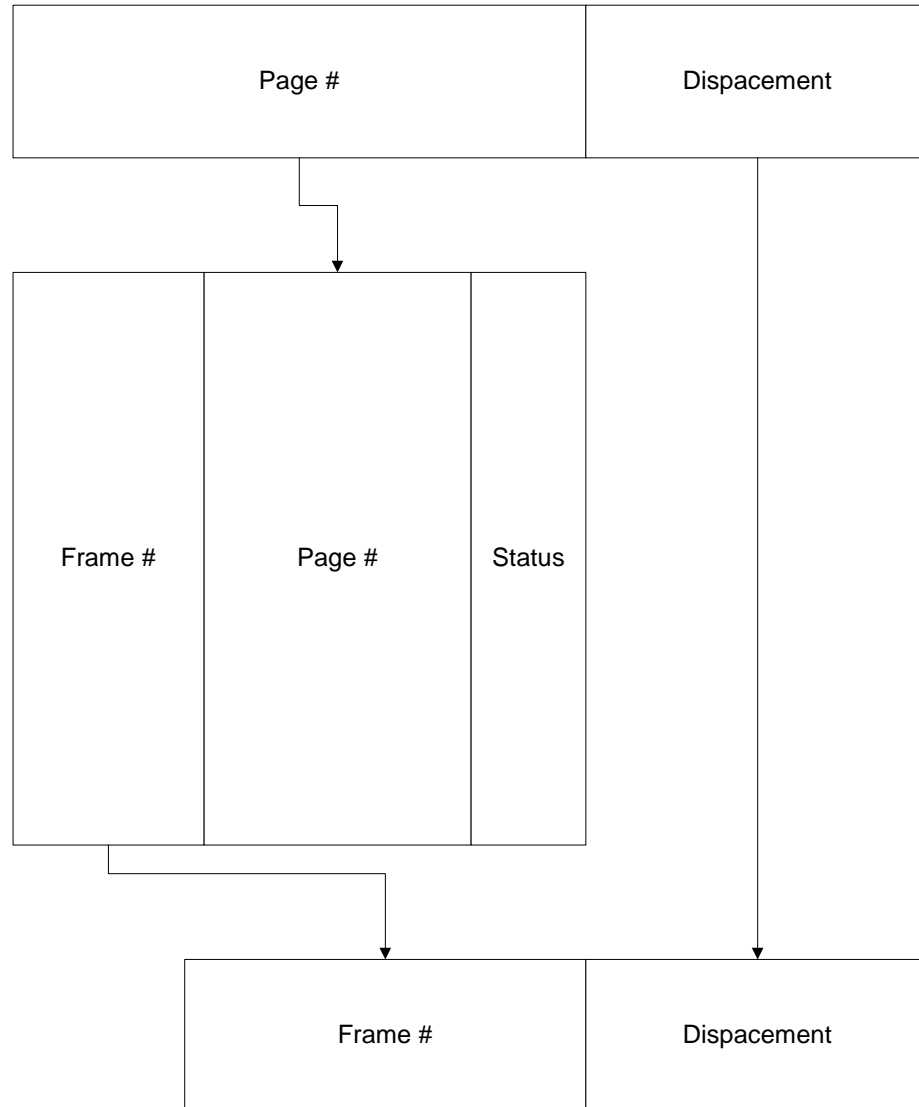
- Virtual Address Translation

It would be too much to ask for the CPU to translate each individual word into separate physical addresses. Instead memory is broken into pages, and each page is translated from virtual to physical.

To translate from virtual address to physical address, the CPU uses a “translation table” consisting of:

PAGE #	-	Virtual block of memory
FRAME #	-	Physical block of memory
STATUS	-	Present, Dirty, Locked, counters, etc.

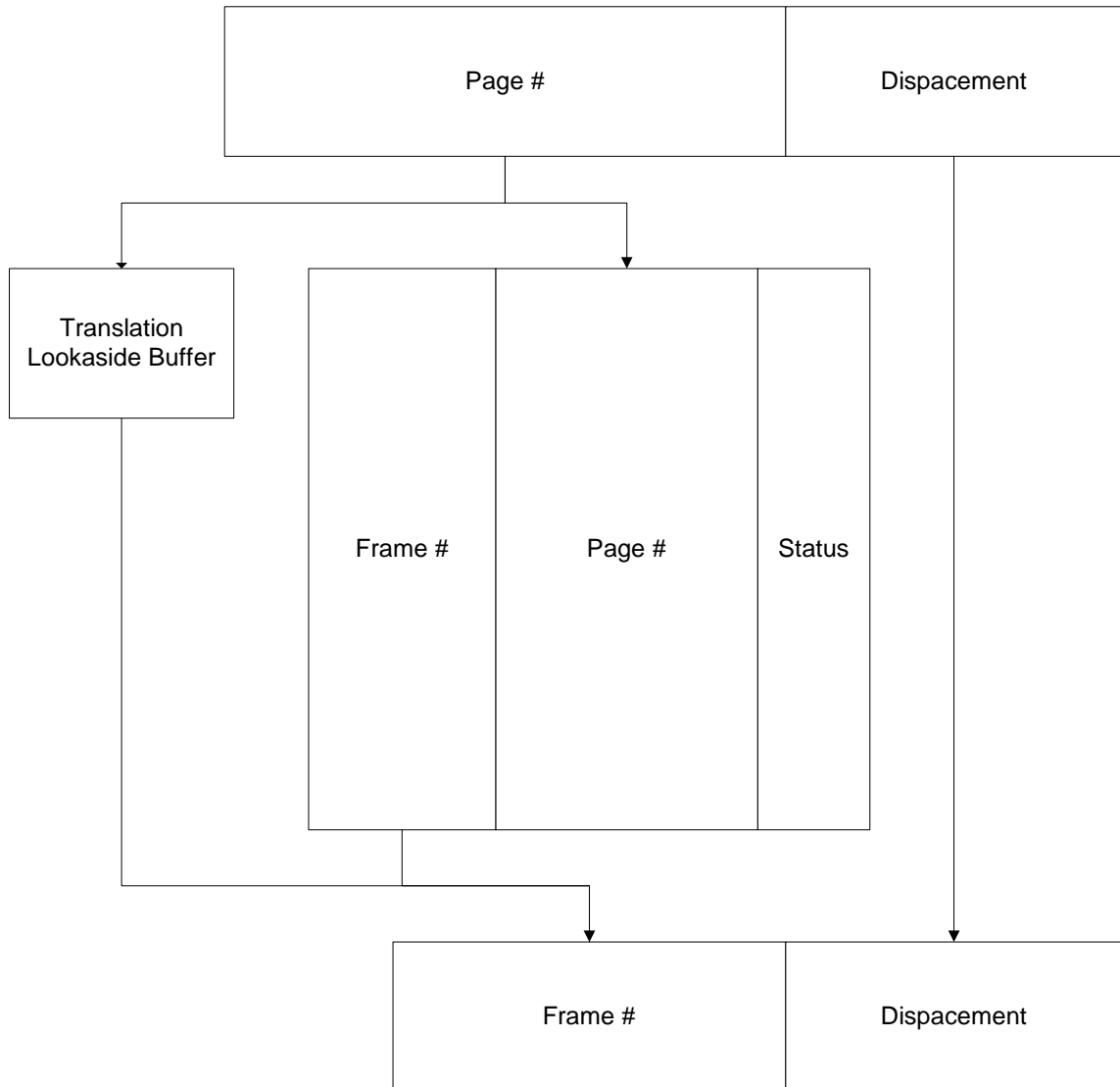
The translation process can be thought of as follows:



This virtual memory system requires the CPU to search the translation table on each memory access. Even this very fast memory, this can significantly slow down the CPU.

- Translation Lookaside Buffer (TLB)

To improve performance, a small translation table of the most recently used pages is maintained by some systems. This small table is called a translation lookaside buffer (TLB).



The TLB speeds the search for the most common pages.

Now you have a hierarchy of virtual addresses:

- Virtual addresses in TLB (FASTEST)
- Virtual addresses in Main Memory
- Virtual addresses on Disk (SLOWEST)

Combine this with several levels of caching and the access time for any given address can become quite complex.