The Page Table

Once a page has been committed using `VirtualAlloc`, it's only when it's accessed for the first time that anything actually happens. When a thread accesses the committed virtual memory, physical RAM memory is allocated and the corresponding virtual and physical addresses are recorded as an entry in a new structure called the **page table** (see Figure 7.2).

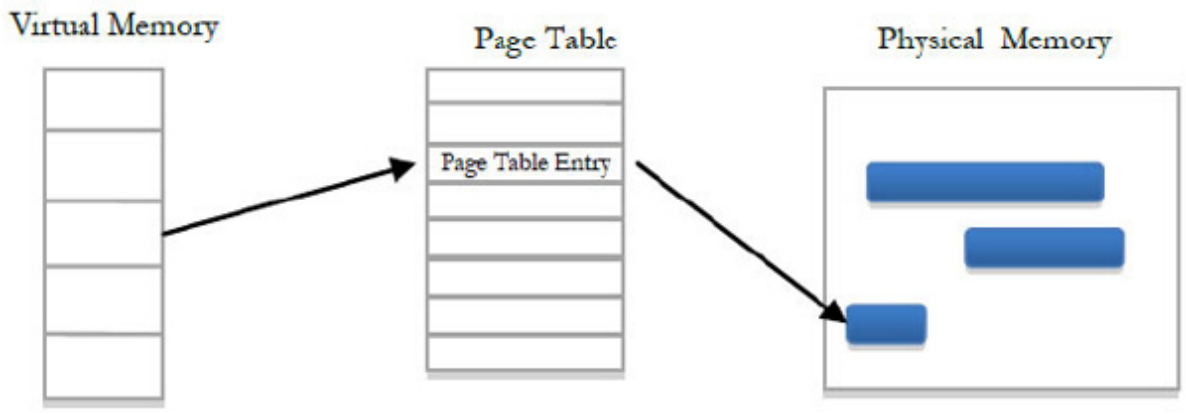

Figure 7.2: Conceptual page table design.

So the page table stores the physical memory address and its corresponding virtual address, and is key to address translation.

Each page table entry records when the page was changed and if it's still loaded in physical memory.

Committed pages are backed up to the page file on disk. When a page hasn't been used for a while, the memory manager determines if it can be removed from physical memory. When this removal or unused pages happen, the MM just leaves a copy on disk in the page file.

Using a set of complex algorithms, the VMM can swap pages in and out of memory based on their usage patterns and demand. This gives applications access to a large memory space, protected memory, and efficient use of memory, all at the same time.

We will look at page swapping in more detail later on, but let's now look a little deeper into address translation.