Memory Management and Paging

Eric McCreath

Address Binding

For a program to execute it must be copied into main memory at a particular location.

Many instructions use 'fixed' addresses which must be bound to 'fixed' locations in the memory.

This binding of instructions and data to memory addresses may occur at :

- compile time,
- load time, or
- execution time.

Logical/Physical Address Space

Addresses generated by the CPU are referred to as logical addresses. These are the addresses 'seen' by the user's programs.

Addresses seen by the main memory are referred to as physical addresses.

In some systems logical and physical addresses are identical. In these cases address binding must occur at compile-time or load-time.

However, it is useful to separate logical and physical addresses, this permits execution-time address binding schemes. Logical addresses may also be referred to as virtual addresses.

Logical/Physical Address Space

+The set of all logical addresses generated by a program is referred to as the logical address space. These logical addresses map to physical addresses and are referred to as the physical address space.

- The mapping between the logical and physical addresses is performed by the MMU (Memory-Management Unit). This is done in hardware.

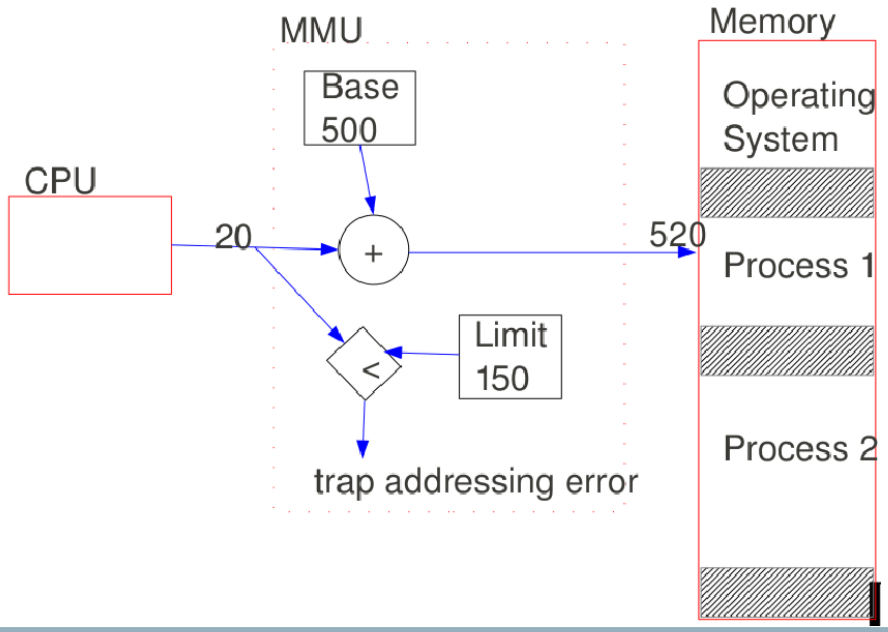
+Hardware MMU can also provide a range of protections.

Logical/Physical Address Spaces

Some advantages of separating logical and physical addresses include:

- execution-time address-binding
- simplifies the swapping of processes in and out of memory
- protection/security of data between processes
- simplifies sharing of data

Simple Base Limit Approach



Paging

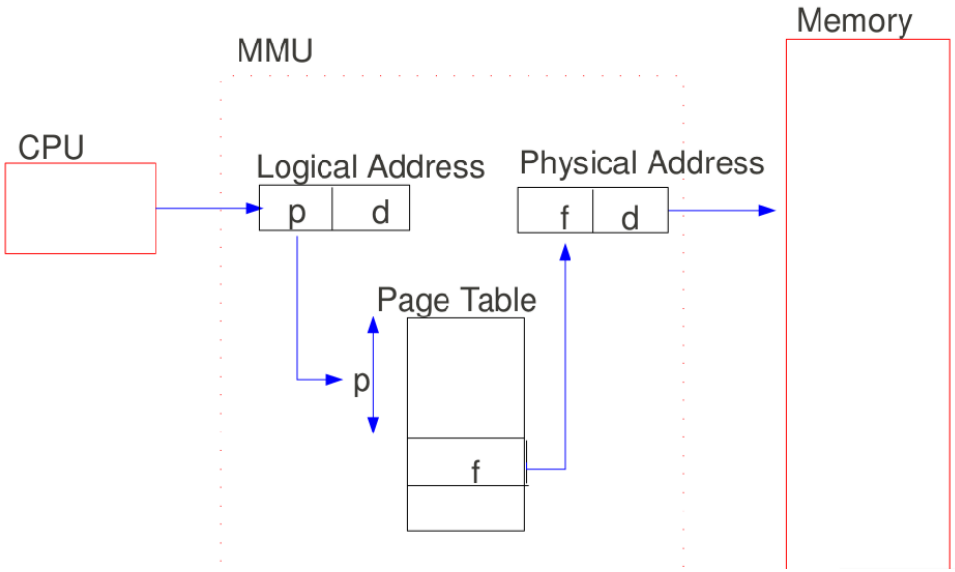
There is considerable overhead in managing variable sized memory-chunks. Paging overcomes many of these problems and solves the external fragmentation problem (memory gets fragmented up into many small unusable sections). Paging is used in many modern operating systems.

Paging

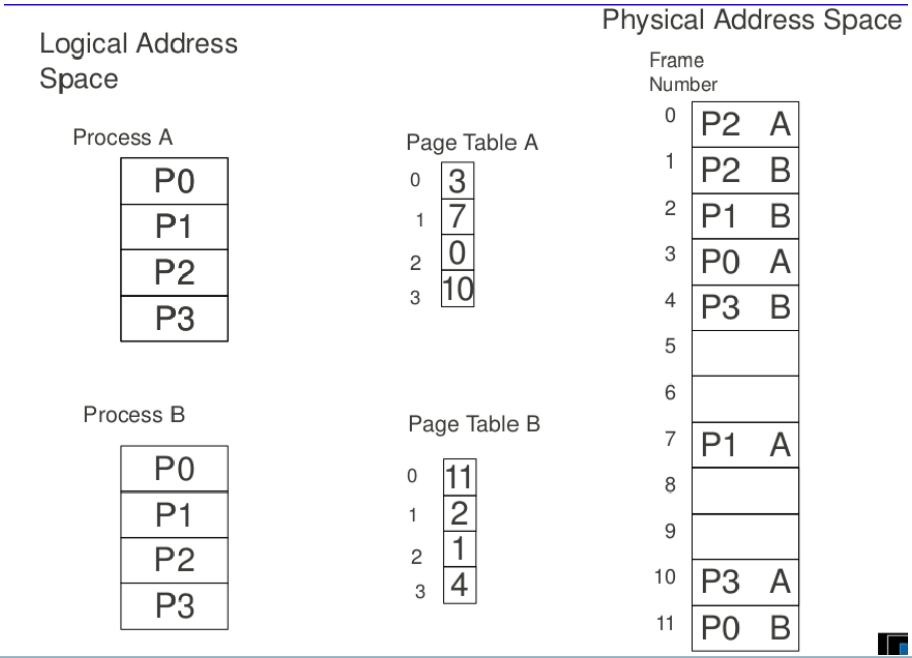
Paging involves the following:

- Physical memory is partitioned into fixed-sized blocks called frames.
- Logical memory is partitioned into blocks of the same size called pages.
- Logical addresses (produced by the CPU) are divided into two parts: the page number and the offset.
- Each process has a page table. The page number indexes the page table for the running process and looks up the frame number for that page. The frame number is combined with the offset to produce the physical address.

Paging

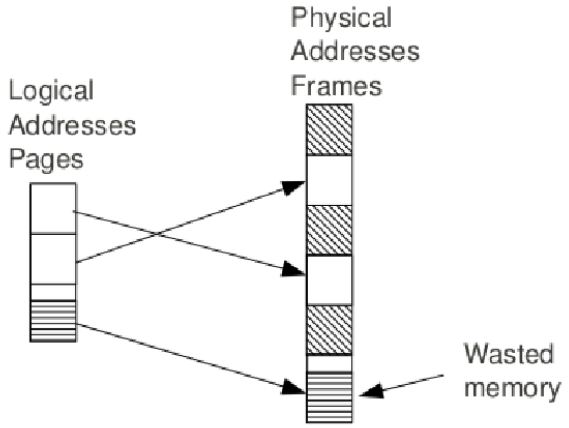


Paging



Internal Fragmentation

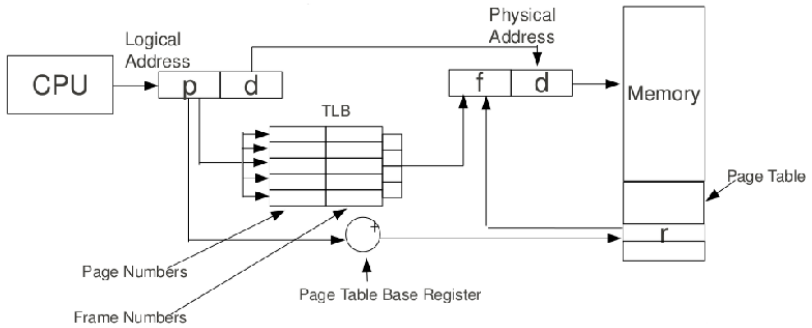
Suppose the offset uses 9 bits. Hence the frames will be 512 bytes long. If a process requires 1025 bytes then three frames must be used by the process. Only 1byte of the third frame is used! This waste is known as internal fragmentation.



Paging

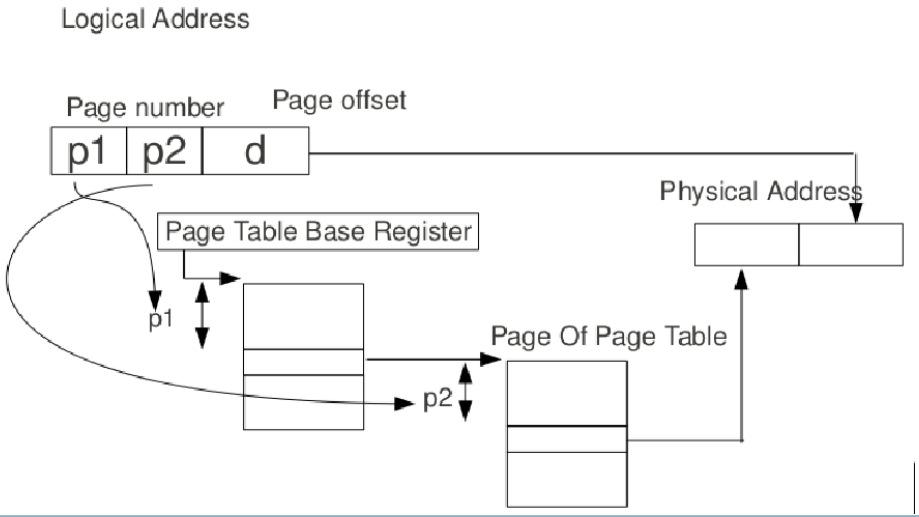
The page tables may be very large, making it not feasible to store the entire page table in registers. So the page table is stored in main memory. A Page

Table Base Register (PTBR) points to the page table. A fast lookup cache may to store the page frame mappings within the CPU. These specially built caches are called translation lookÂ-asidebuffers (TLBs)



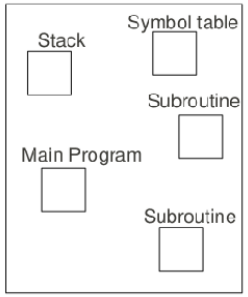
Multilevel Paging

Page tables can get very big. One solution is to divide the page number into smaller pieces and use an outer page table to index a page of the page table. This is known as multilevel paging.



Segmentation

A user or process would like to view memory as a set of variable-sized segments. Each segment has a name and there is no necessary ordering among segments.



Logical Address Space

Segmentation

An address within a segment may be referred to by a segment number and offset. The segment table will consist of base and limit registers for each segment.