Operating Systems Notes

SubmergedDuck

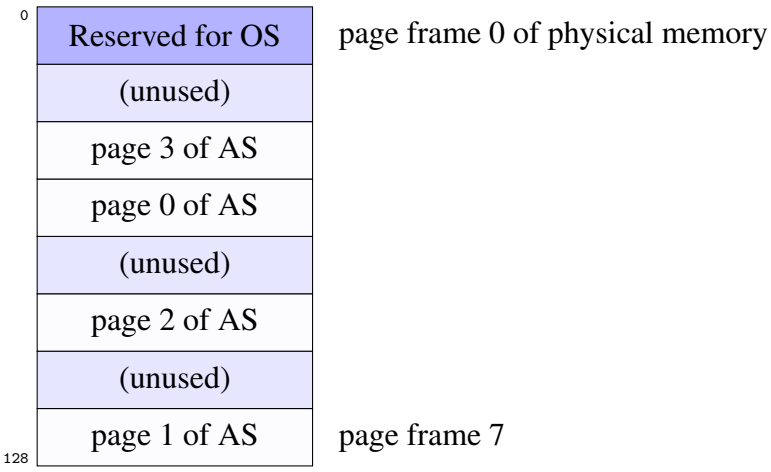
October 29, 2025

1 Page Tables: OSTEP

Paging. When memory is split into fixed-sized pieces. Inside one page, the offset tells "which byte within this page."

Page. A fixed size unit in a process's address space.

Fig 1. 64-Byte Address Space in 128-Byte Physical Memory. TBC.



Notice how each page here is $128 \div 8 = 16$ bytes each. Notice also how the address space is can be scattered throughout physical memory.

Free List. When an OS wishes to place a process's address space into physical memory, it keeps a free list of all free pages/page frames of the physical memory.

Page Table. A data structure to record each where each virtual page of the address space is located in physical memory. Each process has their own page table.

Address Translations. What the page table stores for each virtual page of the address space.

> **Eg.** Virtual Page 1 $\rightarrow$ Physical Frame 7
>
> **Eg.** To translate a virtual address to a physical address, we keep the offset (the right-most bits), but change the VPN to PFN. Using Fig 1, since the virtual page number is 1 (01 in binary) and the physical frame number is 7 (111 in binary) then our physical adress would be 111 | [offset] while our virtual adress would be 01 | [offset].

Offset. Indicates which byte a virtual address is on a virtual page.

2 Page Tables: Other

Offset Bits. How many bits to count all bytes in one page.

> **Eg.** A 4KB page has 4096 bytes. Since $4096 = 2^{12}$, the page offset uses 12 bits.

Virtual Page Number (VPN). The index of a virtual page. These are the high bits of the virtual address.)

> **Eg.** Given a 32-bit virtual address and 4KB pages, there would be 20 bits left for the VPN (cause $4KB = 2^{12}$ bytes). This implies that there are 2^{20} virtual pages if there are 20 VPN bits.

Page Table Entry (PTE). One record in the page table that maps a virtual page to a physical frame and bits like valid/present, R/W/X permissions, dirty, accessed, etc. A single second-level page table holds 1024 PTEs.

> **Note.** Each page table entry has a Present(P) bit saying whether that virtual page is currently in RAM. $P = 1 \Rightarrow$ the mapping is valid and you can use the page table number. $P = 0 \Rightarrow$ otherwise (could be swapped out or not mapped.)

Virtual Page. A fixed-size chunk (eg. 4KB) of a process's virtual address space.

> **Eg.** With 4KB pages, every virtual address splits into a virtual page number (which page) + offset (which byte inside that page)
>
> **Eg 2.** Let virtual address = 0x12345ABC. Then the offset would be 0xABC (low 12 bits) and the VPN would be 0x12345 given 4KB pages.

Single-level Page Table. One big array indexed by the VPN; each slot is a PTE that says where that virtual page lives (which physical frame).

Second-level Page Table. Splits the VPN into PDI | PTI (10 | 10).

> **Eg.** There are 2^{20} page table entries, each entry is 4 bytes. The total size of the single page table would be $2^{20} \times 4$ bytes.

Physical Frame. A 4KB chunck of RAM (or same size as page) where a virtual page's data lives when it's in memory.

Physical Frame Number (PFN). The index of a 4KB physical frame in RAM. After page-table lookup, the PTE gives you the PFN plus status bits. Eg. Given 4KB pages, VA = [PFN | 12-bit Offset]

MiB. A mebibyte. 1 MiB = 2^{20} bytes

Page Directory Index (PDI). The first 10 bits of virtual address (VA). Selects which page table to use.

Page Table Index (PTI). The next 10 bits after PDI of VA. The Selects which PTE inside that page table.

Bit Shifting

```
#define PAGE_SHIFT 12 // 4KB Pages
uint64_t physical_address = ((uint64_t)pfn << PAGE_SHIFT)|
   offset;
```

Translation Lookaside Buffer (TLB). A tiny, very fast cache inside the MMU that stores recent virtual page number (VPN) to physical frame number translations (plus R/W/X/valid bits).

> **Hit.** MMU find the mapping in the TLB so it can form the physical address without reading the page tables from memory. Memory Accesses Needed: 1.
>
> **Miss.** MMU (or OS) must walk the page tables in memory to find the PTE, then it can access the data. Memory Accesses Needed: 3.

Memory Management Unit (MMU). The hardware between the CPU and RAM that translates virtual addresses to physical using TLB and page tables. Also enforces R/W/X permissions and raises page faults if needed and updates accessed/dirty bits.

Accessed Bit (A/R). Set by the hardware on any read or write to the page. OS uses it to tell which pages were recently used (eg. for replacement).

Dirty Bit (D/modified). Set on any write. If a dirty page is evicted, it must be written back to disk; if clean it can be dropped.

Dirty Page. A memory page that's been modified (written to) since it was loaded (its dirty bit is set.) If the OS evicts it, it must write it back to disk; a clean page (non-modified) can be dropped without writing.

Page Fault. The PTE says the page isn't in RAM (invalid/not present), thus trapping to the OS to bring the page in.

3 Page Tables