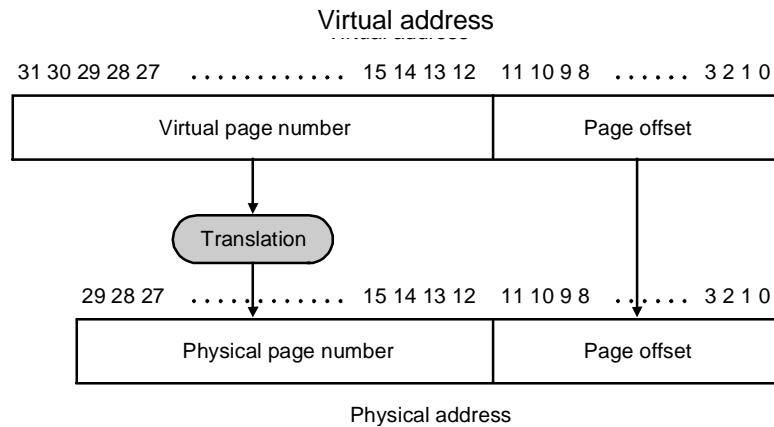
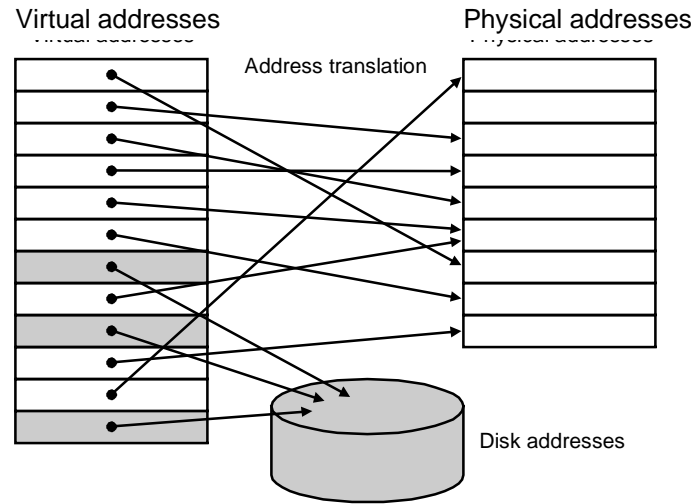


# Virtual Memory



# Virtual memory addressing: page tables

---

## Page table

- Library analogy: the card catalog
- Indexing all pages in virtual memory
- Each program has its own page table
- OS creates disk space for all the pages of every process
- The state of a process is specified by the page table, PC and registers
- Active and inactive processes (saving and loading page table register, PC and the other registers)

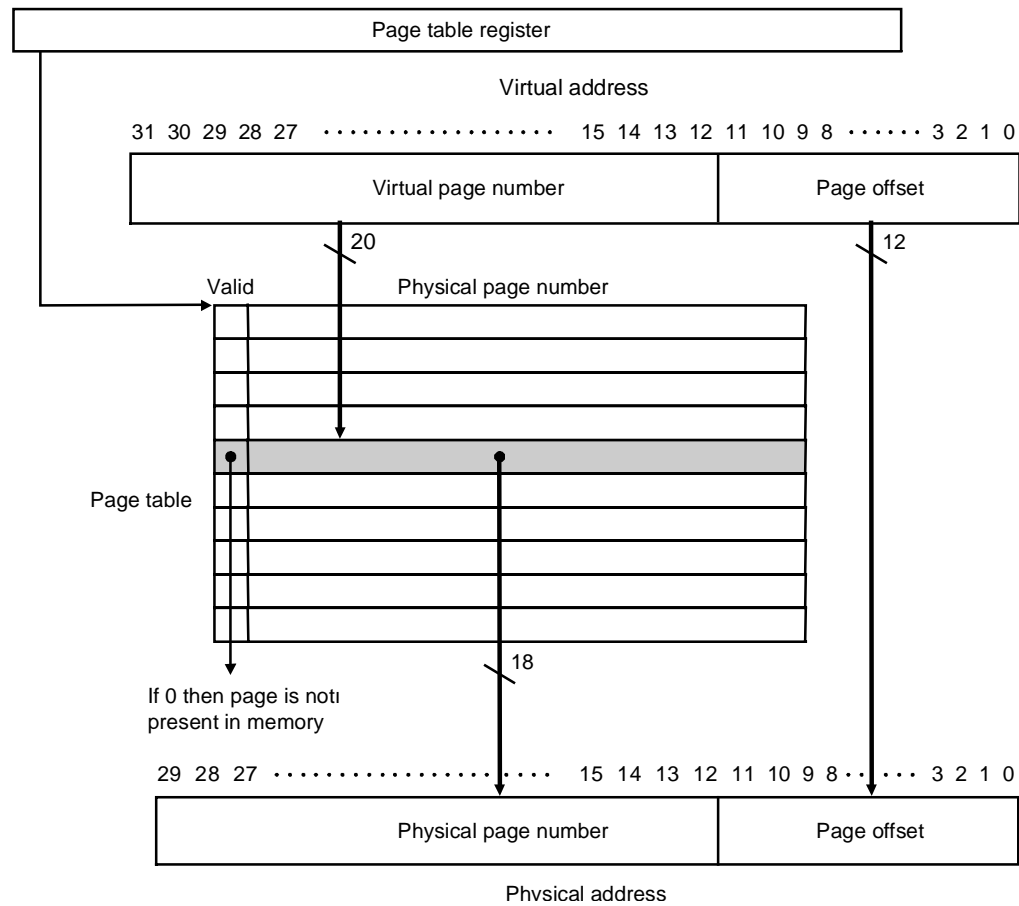
## Page table entries

- Physical page or disk address
- Tag is not necessary (the index specifies the page address)
- Valid bit: 1 - the page is in memory; 2 - the page is on disk
- Reference (use) bit: LRU scheme for replacing pages
- Dirty bit ( the page has been written): write-back scheme
- Information for the process using the page (memory protection)

## Example

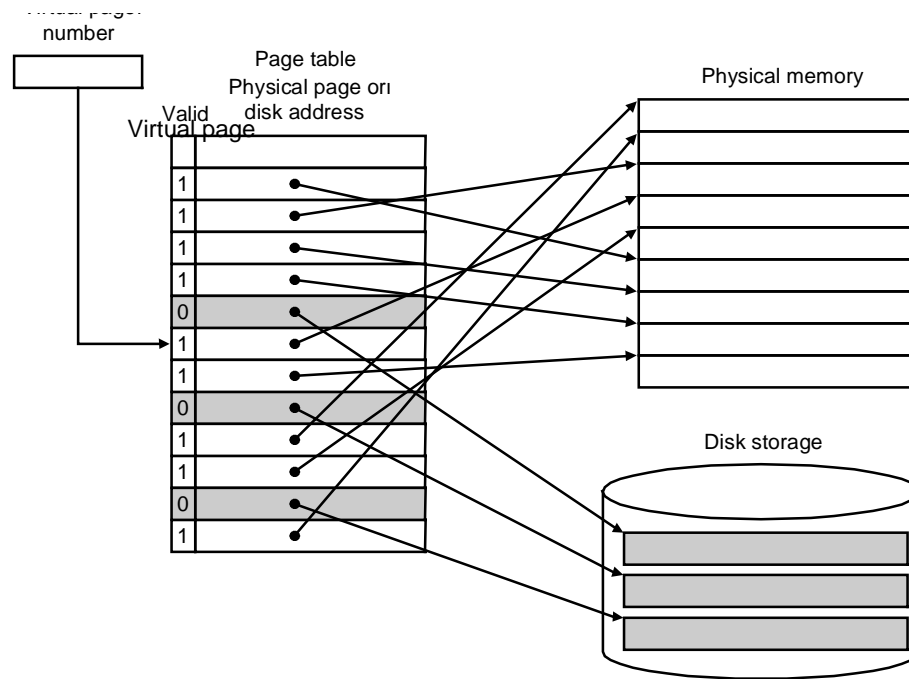
- 32-bit virtual address, 4KB page, 4 byte entry - 4MB for a page table
- 250 processes X 4MB = 1GB !
- Reducing the page table size

# Virtual memory addressing: address translation



# Handling page faults

---



- Transfer the control to OS by using the exception mechanism
- Read the page from disk
- Determine which page in memory to replace (LRU)
- Write back to disk the replaced page (if its dirty bit is on)
- Write the physical address in the page table entry