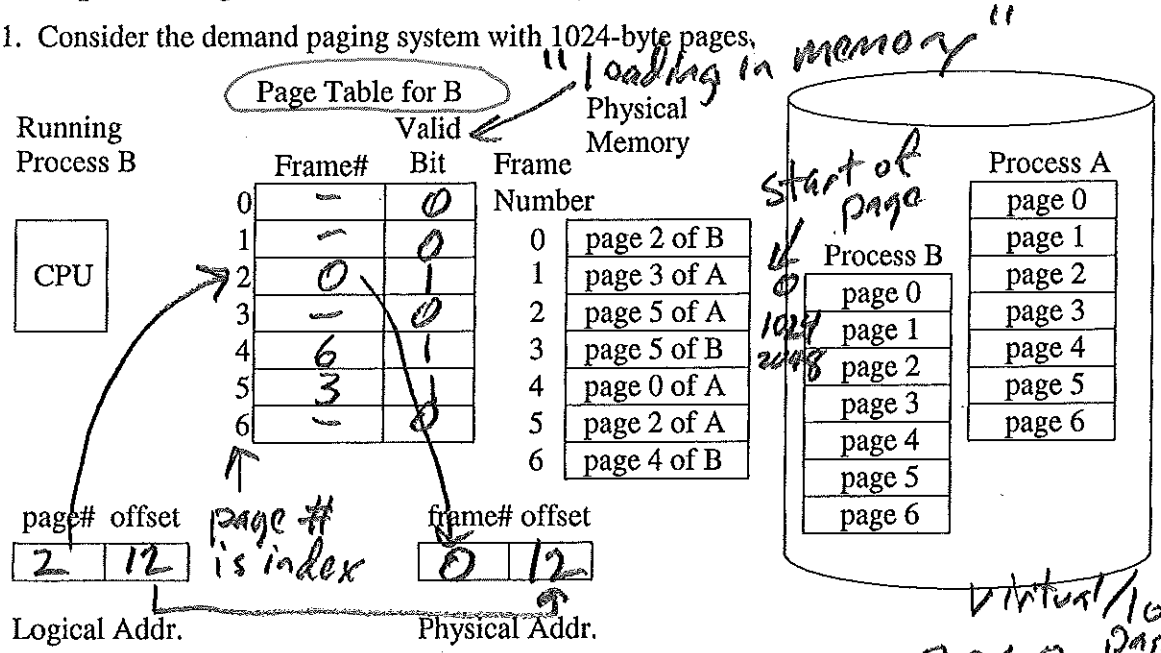
Computer Org. Lecture 29 Name :_____

1. Consider the demand paging system with 1024-byte pages.



"loading in memory"

start of page

1024

2048

virtual/logical

$\frac{2060}{1024} = 2$ page

2060 % 1024 = 12 page offset

a) Complete the above page table for **Process B**.

b) If process B is currently running and the CPU generates a logical/virtual address of 2060_{10}, then what would be the corresponding physical address?

Each page is 1024 bytes, so page 0 starts at addr. 0, Page 1 starts at 1024 and Page 2 starts at 2048. See

2. For a 32-bit address machine with 4 KB (2^{12} bytes) pages and 4 byte page-table entries, how big would the page table be?

32-12=20-bits | 12-bit page offset since each page is 4KB (2^{12})

page #

page #'s 0 to $(2^{20}-1)$

for 2^{20} page table entries, so total page table size of $2^{20} \times 2^2$ bytes

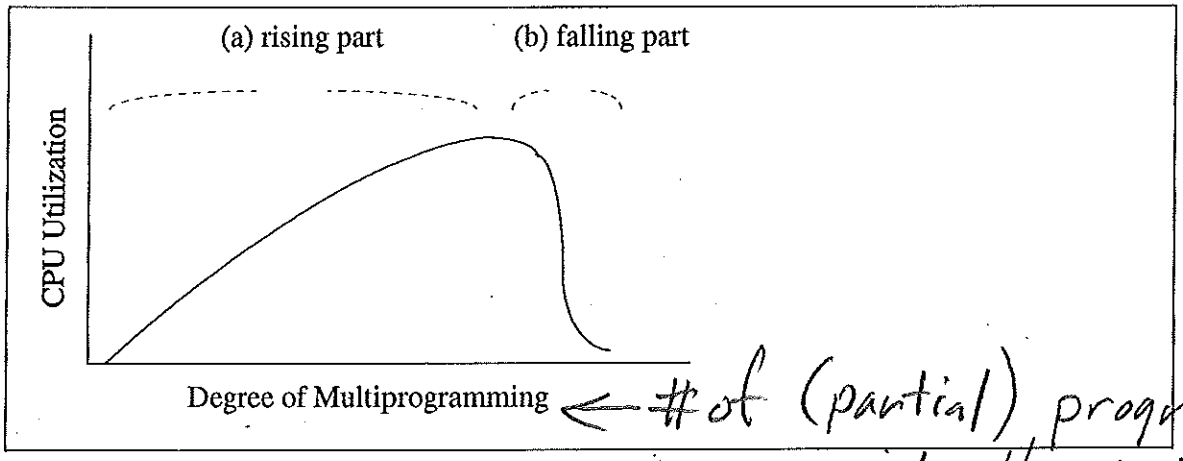
3. How does a TLB (translation-lookaside buffer) speed the process of address translation?

TLB is a cache in CPU of recent page table entries so the memory-management unit (MMU) can translate from a logical address to a physical address without accessing the actual page table.

Design issues for Paging Systems

Conflicting Goals:

- Want as many (partial) processes in memory (high degree of multiprogramming) as possible so we have better CPU & I/O utilization ⇒ allocate as few page frames as possible to each process
- Want as low of page-fault rate as possible ⇒ allocate enough page frames to hold all of a process' current working set (which is dynamic as a process changes locality)



4. Explain the shape of each section indicated on the above curve:

a) (rising part of the curve)

When pgm A does I/O (which is very slow), the CPU can be switched to execute pgm B if it is in memory, when pgm B now does I/O, the CPU can run pgm C, if it is in memory, etc.

b) (falling part of the curve)

The more partial programs in memory the less memory frame each can use. Since each pgm needs a minimum # to hold the current "loop" and data, each eventually page faults each iteration of loop, so poor performance.

5. There are many similarities between the cache-memory level and memory-disk level of the memory hierarchy, but there are also important differences. For example, a cache miss stalls the running program temporarily, but a page fault causes the running program to turnover the CPU to another program. Why are these cases treated differently by the computer system?