COMP 530 Introduction to Operating Systems

Fall 2017  
Kevin Jeffay

Worksheet 14, October 25

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Your Name: |  | You worked with: |  | +1/blank/-1: |  |
|  | Aaron Zhang |  | John Espenhahn |  | +1 |  |
|  |  |  | Brennan Proudfoot |  | +1 |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. A certain computer provides each process with a 4 gigabyte virtual address space. Virtual memory is implemented by paging with a single level page table with a page size of 4K bytes. The computer has 1M bytes of physical memory.

A user process generates the (base 10) virtual address 28,718. Describe how the computer system establishes the corresponding physical address. Give as much detail as you can. (Note that in providing your answer you are free to draw a figure, however, if this were an exam your score for this problem would largely be based on the text that you write [*i.e.*, your explanation] rather than any figure you draw.)

Virtual address:

|Page| = |page fame|

Virtual address = omaxp+o

Take log base 2 of o max (which will always be a power of two), then all the remaining bits are the page number. Figure out how many bits are remaining, then two to that number of bits will tell you how much space you have.

Physical Adress

Conceptual the same of virtual address.

32 bits in a virtual address. 20 bits in a physical address. 232 220

Partition the address, how many bits are required to represents the offset? (which is 4k)

Page size is 4k bytes = 212 which requires 12 bits.

12 bits required for offset, leaving 20 bits.

220 pages in the virtual address space (roughly a million)

20->number of bits in physical address, 12->required for offset

20-12 = 8 bits for the frame

28 = 256 frames

This is virtual page index 7…

Virtual page number goes into TLB (cache of page #’s to frame #’s)

If miss, index into page table… go to seventh entry starting with 0

Frame number combined with offset->physical address

2. In the discussion of demand paged virtual memory in lecture, we only ever considered examples wherein processes had a larger virtual address space than the amount of physical memory present on the computer. Does it have to be this way? That is, can a process have a *smaller* virtual address space than the amount of physical memory present in the computer?

*a*) For any of implementations of paged virtual memory that we considered in class, either:

• Explain what “breaks” in the implementation if processes have a smaller virtual address space than the amount of physical memory present, or

• Explain how the implementation can be extended/modified to accommodate processes with a smaller virtual address space than the amount of physical memory present.

*b*) In either case, would virtual memory still be useful as a memory management discipline if processes had smaller address spaces than the amount of physical memory present on the computer? Explain.

(a)

Page table will be smaller (number of rows in page table is still number of virtual pages). Number of physical frames a process can occupy will be smaller than total number of frames. Every virtual address can be mapped to a physical address.

Mapping is still good so nothing breaks

(b)

Security – can’t see memory allocated to other processes. Indirection provides memory protection

3. Consider the impact of increasingly faster CPUs on the design of page-based virtual memory systems. That is, as processors get faster, are page-based virtual memory systems as we have discussed them in class likely to yield performance that will scale with processor speeds? For example, the speed of CPUs (the clock rate) has doubled within the last 5 years and memory access times have halved. In the same time period disk densities have also doubled but access times have remained relatively constant. Given these facts, ignoring the time spent performing user/program-requested I/O, if you were to take a computer system and double the CPU speed, will the response time or turnaround time of jobs be cut in half if the OS supports page-based virtual memory? Explain.

What are we to do about this problem?

SOMETHING ON SHARED MEMORY

Increase TLB (translation lookup buffer) cache size