Main Memory

Exercises

9.9 Explain the difference between internal and external fragmentation.

9.10 Consider the following process for generating binaries. A compiler is used to generate the object code for individual modules, and a linker is used to combine multiple object modules into a single program binary. How does the linker change the binding of instructions and data to memory addresses? What information needs to be passed from the compiler to the linker to facilitate the memory-binding tasks of the linker?

9.11 Given six memory partitions of 100 MB, 170 MB, 40 MB, 205 MB, 300 MB, and 185 MB (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of size 200 MB, 15 MB, 185 MB, 75 MB, 175 MB, and 80 MB (in order)? Indicate which—if any—requests cannot be satisfied. Comment on how efficiently each of the algorithms manages memory.

9.12 Most systems allow a program to allocate more memory to its address space during execution. Allocation of data in the heap segments of programs is an example of such allocated memory. What is required to support dynamic memory allocation in the following schemes?

a. Contiguous memory allocation

b. Paging

9.13 Compare the memory organization schemes of contiguous memory allocation and paging with respect to the following issues:

a. External fragmentation

b. Internal fragmentation

c. Ability to share code across processes

9.14 On a system with paging, a process cannot access memory that it does not own. Why? How could the operating system allow access to additional memory? Why should it or should it not?

9.15 Explain why mobile operating systems such as iOS and Android do not support swapping.

9.16 Although Android does not support swapping on its boot disk, it is possible to set up a swap space using a separate SD nonvolatile memory card. Why would Android disallow swapping on its boot disk yet allow it on a secondary disk?

9.17 Explain why address-space identifiers (ASIDs) are used in TLBs.

9.18 Program binaries in many systems are typically structured as follows. Code is stored starting with a small, fixed virtual address, such as 0. The code segment is followed by the data segment, which is used for storing the program variables. When the program starts executing, the stack is allocated at the other end of the virtual address space and is allowed to grow toward lower virtual addresses. What is the significance of this structure for the following schemes?

a. Contiguous memory allocation

b. Paging

9.19 Assuming a 1-KB page size, what are the page numbers and offsets for the following address references (provided as decimal numbers)?

a. 21205

b. 164250

c. 121357

d. 16479315

e. 27253187

9.20 The MPV operating system is designed for embedded systems and has a 24-bit virtual address, a 20-bit physical address, and a 4-KB page size. How many entries are there in each of the following?

a. A conventional, single-level page table

b. An inverted page table

What is the maximum amount of physical memory in the MPV operating system?

9.21 Consider a logical address space of 2,048 pages with a 4-KB page size, mapped onto a physical memory of 512 frames.

a. How many bits are required in the logical address?

b. How many bits are required in the physical address?

9.22 Consider a computer system with a 32-bit logical address and 8-KB page size. The system supports up to 1 GB of physical memory. How many entries are there in each of the following?

a. A conventional, single-level page table

b. An inverted page table

9.23 Consider a paging system with the page table stored in memory.

a. If a memory reference takes 50 nanoseconds, how long does a paged memory reference take?

b. If we add TLBs, and if 75 percent of all page-table references are found in the TLBs, what is the effective memory reference time? (Assume that finding a page-table entry in the TLBs takes 2 nanoseconds, if the entry is present.)

9.24 What is the purpose of paging the page tables?

9.25 Consider the IA-32 address-translation scheme shown in Figure 9.22.

a. Describe all the steps taken by the IA-32 in translating a logical address into a physical address.

b. What are the advantages to the operating system of hardware that provides such complicated memory translation?

c. Are there any disadvantages to this address-translation system? If so, what are they? If not, why is this scheme not used by every manufacturer?