CHAPTER 9

Main Memory

Practice Exercises

9.1 Name two differences between logical and physical addresses.

Answer:
A logical address does not refer to an actual physical address; rather, it refers to an abstract address in an abstract address space. A physical address refers to an actual physical address in memory. A logical address is generated by the CPU and is translated into a physical address by the memory management unit(MMU). Therefore, physical addresses are generated by the MMU.

9.2 Why are page sizes always powers of 2?

Answer:
Recall that paging is implemented by breaking up an address into a page and offset number. It is most efficient to break the address into X page bits and Y offset bits, rather than perform arithmetic on the address to calculate the page number and offset. Because each bit position represents a power of 2, splitting an address between bits results in a page size that is a power of 2.

9.3 Consider a system in which a program can be separated into two parts: code and data. The CPU knows whether it wants an instruction (instruction fetch) or data (data fetch or store). Therefore, two base–limit register pairs are provided: one for instructions and one for data. The instruction base–limit register pair is automatically read-only, so programs can be shared among different users. Discuss the advantages and disadvantages of this scheme.

Answer:
The major advantage of this scheme is that it is an effective mechanism for code and data sharing. For example, only one copy of an editor or a compiler needs to be kept in memory, and this code can be shared by all processes needing access to the editor or compiler code. Another advantage is protection of code against erroneous modification. The only

disadvantage is that the code and data must be separated, which is usually adhered to in a compiler-generated code.

9.4 Consider a logical address space of 64 pages of 1,024 words each, mapped onto a physical memory of 32 frames.

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

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

Answer:

a. Logical address: 16 bits

b. Physical address: 15 bits

9.5 What is the effect of allowing two entries in a page table to point to the same page frame in memory? Explain how this effect could be used to decrease the amount of time needed to copy a large amount of memory from one place to another. What effect would updating some byte on one page have on the other page?

Answer:

By allowing two entries in a page table to point to the same page frame in memory, users can share code and data. If the code is reentrant, much memory space can be saved through the shared use of large programs such as text editors, compilers, and database systems. "Copying" large amounts of memory could be effected by having different page tables point to the same memory location.

However, sharing of nonreentrant code or data means that any user having access to the code can modify it, and these modifications would be reflected in the other user's "copy."

9.6 Given six memory partitions of 300 KB, 600 KB, 350 KB, 200 KB, 750 KB, and 125 KB (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of size 115 KB, 500 KB, 358 KB, 200 KB, and 375 KB (in order)?

Answer:

- **First fit**:
  a. 115 KB is put in 300-KB partition, leaving 185 KB, 600 KB, 350 KB, 200 KB, 750 KB, 125 KB
  b. 500 KB is put in 600-KB partition, leaving 185 KB, 100 KB, 350 KB, 200 KB, 750 KB, 125 KB
  c. 358 KB is put in 750-KB partition, leaving 185 KB, 100 KB, 350 KB, 200 KB, 392 KB, 125 KB
  d. 200 KB is put in 350-KB partition, leaving 185 KB, 100 KB, 150 KB, 200 KB, 392 KB, 125 KB
  e. 375 KB is put in 392-KB partition, leaving 185 KB, 100 KB, 150 KB, 200 KB, 17 KB, 125 KB

- **Best fit**:
  a. 115 KB is put in 125-KB partition, leaving 300 KB, 600 KB, 350 KB, 200 KB, 750 KB, 10 KB
  b. 500 KB is put in 600-KB partition, leaving 300 KB, 100 KB, 350 KB, 200 KB, 750 KB, 10 KB
  c. 358 KB is put in 750-KB partition, leaving 300 KB, 100 KB, 350 KB, 200 KB, 392 KB, 10 KB
  d. 200 KB is put in 200-KB partition, leaving 300 KB, 100 KB, 350 KB, 0 KB, 392 KB, 10 KB
  e. 375 KB is put in 392-KB partition, leaving 300 KB, 100 KB, 350 KB, 0 KB, 17 KB, 10 KB
- **Worst fit**:
  a. 115 KB is put in 750-KB partition, leaving 300 KB, 600 KB, 350 KB, 200 KB, 635 KB, 125 KB
  b. 500 KB is put in 635-KB partition, leaving 300 KB, 600 KB, 350 KB, 200 KB, 135 KB, 125 KB
  c. 358 KB is put in 600-KB partition, leaving 300 KB, 242 KB, 350 KB, 200 KB, 135 KB, 125 KB
  d. 200 KB is put in 350-KB partition, leaving 300 KB, 242 KB, 150 KB, 200 KB, 135 KB, 125 KB
  e. 375 KB must wait

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

a. 3085
b. 42095
c. 215201
d. 650000
e. 2000001

Answer:

a. page = 3; offset = 13
b. page = 41; offset = 111
c. page = 210; offset = 161
d. page = 634; offset = 784
e. page = 1953; offset = 129

9.8 The BTV operating system has a 21-bit virtual address, yet on certain embedded devices, it has only a 16-bit physical address. It also has a 2-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 BTV operating system?

Answer:
Conventional, single-level page table will have $2^{10} = 1024$ entries. Inverted page table will have $2^5 = 32$ entries. The maximum amount of physical memory is $2^{16} = 65536$ (or 64 KB.)

9.9 Consider a logical address space of 256 pages with a 4-KB page size, mapped onto a physical memory of 64 frames.

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

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

Answer:

a. $12 + 8 = 20$ bits.

b. $12 + 6 = 18$ bits.

9.10 Consider a computer system with a 32-bit logical address and 4-KB page size. The system supports up to 512 MB 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

Answer:

a. 2^{20} entries.

b. 512 K K/4K = 128K entries.

Exercises

9.11 Explain the difference between internal and external fragmentation.

Answer:

a. Internal fragmentation is memory space within a region or a page that is not used by the job occupying that region or page. This space is unavailable for use by the system until the job is finished and the page or region is released.

b. External fragmentation is unused space between allocated regions of memory. Typically, external fragmentation results in memory regions that are too small to satisfy a memory request, but if we were to combine all the regions of external fragmentation, we would have enough memory to satisfy a memory request.

9.12 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.

Answer:

- **First fit**:
  a. 200 MB is put in 205-MB partition, leaving 100 MB, 170 MB, 40 MB, 5 MB, 300 MB, 185 MB
  b. 15 MB is put in 100-MB partition, leaving 85 MB, 170 MB, 40 MB, 5 MB, 300 MB, 185 MB
  c. 185 MB is put in 300-MB partition, leaving 85 MB, 170 MB, 40 MB, 5 MB, 115 MB, 185 MB
  d. 75 MB is put in 85-MB partition, leaving 10 MB, 170 MB, 40 MB, 5 MB, 115 MB, 185 MB
  e. 175 MB is put in 185-MB partition, leaving 10 MB, 170 MB, 40 MB, 5 MB, 115 MB, 10 MB
  f. 80 MB is put in 170-MB partition, leaving 10 MB, 90 MB, 40 MB, 5 MB, 115 MB, 10 MB
- **Best fit**:
  a. 200 MB is put in 205-MB partition, leaving 100 MB, 170 MB, 40 MB, 5 MB, 300 MB, 185 MB
  b. 15 MB is put in 40-MB partition, leaving 100 MB, 170 MB, 25 MB, 5 MB, 300 MB, 185 MB
  c. 185 MB is put in 185-MB partition, leaving 100 MB, 170 MB, 25 MB, 5 MB, 300 MB, 0 MB
  d. 75 MB is put in 100-MB partition, leaving 25 MB, 170 MB, 25 MB, 5 MB, 300 MB, 0 MB
  e. 175 MB is put in 300-MB partition, leaving 25 MB, 170 MB, 25 MB, 5 MB, 125 MB, 0 MB
  f. 80 MB is put in 125-MB partition, leaving 25 MB, 170 MB, 25 MB, 5 MB, 45 MB, 0 MB
- **Worst fit**:
  a. 200 MB is put in 300-MB partition, leaving 100 MB, 170 MB, 40 MB, 205 MB, 100 MB, 185 MB
  b. 15 MB is put in 205-MB partition, leaving 100 MB, 170 MB, 40 MB, 190 MB, 100 MB, 185 MB
  c. 185 MB is put in 190-MB partition, leaving 100 MB, 170 MB, 40 MB, 5 MB, 100 MB, 185 MB
  d. 75 MB is put in 185-MB partition, leaving 100 MB, 170 MB, 40 MB, 5 MB, 100 MB, 110 MB
  e. 175 MB is denied, as there is no partition large enough to hold the request.
  f. 80 MB is put in 170-MB partition, leaving 100 MB, 90 MB, 40 MB, 5 MB, 100 MB, 110 MB

In this example, only worst fit does not allow a request to be satisfied. An argument could be made that best fit is most efficient, as it leaves the largest holes after allocation.

9.13 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

Answer:

a. Contiguous memory allocation: might require relocation of the entire program, since there is not enough space for the program to grow its allocated memory space.

b. Paging: incremental allocation of new pages is possible in this scheme without relocation of the program's address space.

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

Answer:
There are three reasons: First, these mobile devices typically use flash memory with limited capacity, and swapping is avoided because of this space constraint. Second, flash memory can support a limited number of write operations before it becomes less reliable. Third, there is typically poor throughput between main memory and flash memory.

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

Answer:
ASIDs provide address-space protection in the TLB as well as supporting TLB entries for several different processes at the same time.

9.16 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

Answer:

a. page = 20; offset = 775

b. page = 160; offset = 410

c. page = 118; offset = 525

d. page = 16093; offset = 83

e. page = 26614; offset = 451

9.17 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?

Answer:

a. $11 + 12 = 23$ bits.

b. $9 + 12 = 21$ bits.

9.18 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

Answer:

a. $2^{19} = 524,288$ entries.

b. $2^{17} = 131,072$ entries.

9.19 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?

Answer:

a. The selector is an index into the segment descriptor table. The segment descriptor result plus the original offset is used to produce a linear address with a directory, page, and offset. The dir is an index into a page directory. The entry from the page directory selects the page table, and the page field is an index into the page table. The entry from the page table, plus the offset, is the physical address.

b. Such a translation mechanism offers the flexibility of allowing most operating systems to implement their memory schemes in hardware, instead of having to implement some parts in hardware and some in software. Because translation can be done in hardware, it is more efficient (and the kernel is simpler).

c. Address translation can take longer due to the multiple table lookups it can invoke. Caches help, but there will still be cache misses.