Lecture 8: Memory Management

Summary

1. Computer system overview (Chapter 1)
2. Basic of virtual memory; i.e. segmentation and paging (Chapter 7 and part of 8)
3. Process (Chapter 3)
4. Mutual Exclusion and Synchronization (Chapter 5 section 1-4)
    - Conditions for race avoidance.
    - Strict alternation.
    - Semaphores.
    - Producers and Consumers problem.
    - Hardware support for mutual exclusion.
    - Monitors.
5. Threading (user mode and kernel mode).
6. Deadlock and Starvation.

Memory Management

1. Physical memory is divided into two parts: kernel and user.
    - The user part is further subdivided to accommodate multiple processes.
    - That task is referred to as memory management.
2. Memory management requirements:
    (a) Relocation: the ability to place a process image in different part of the main memory due to swapping, etc.
    (b) Protection: the ability to protect each process from unwanted/unintentional interference by other processes.
        - Can we check address at compile time?
    (c) Sharing: the ability to allow multiple processes to shared information; e.g. shared memory, interprocess communication mechanisms such as semaphores.
    (d) Logical organization: the ability to support the concept of modules
        i. Modules can be written and compiled independently. References are resolved at runtime.
        ii. Different degree of protection can be given to different modules.
        iii. Modules can be shared.
    (e) Physical organization: the responsibility of moving information between main and secondary memory.
3. Memory partitioning (the old ways):

(a) Fixed partitioning: create multiple regions of fixed size (see Figure 7.2). Can suffer from **internal fragmentation**.

(b) Unequal-size partitions: multiple regions of different sizes (see Figure 7.2). It is designed to combat internal fragmentation. Scheduling requires single or multiple queues (see Figure 7.3).

(c) Dynamic partitioning: create partitions of variable length and number as needed by the system. Can suffer from **external fragmentation** (see Figure 7.4). Compaction can be used to eliminate external fragmentation. The following placement algorithms can be used:

- Best fit
- First fit
- Next fit

(d) Buddy system: (see Figure 7.6).

4. Supporting relocation in memory partitioning.

   - Logical address (e.g. relative address) (see Figure 7.8)
   - Physical address

5. Paging: the memory is divided into fixed size chunks. Similarly, each process is also divided into small fixed size chunks of the same size. The logical address consists of a page number and a page offset. Paging is typically invisible to the programmers.

   - Page and frame: a page is referred to a fixed size chunk of process. A frame is referred to a fixed size chunk of memory (see Figure 7.9).
   - Page table is maintained by the OS for each process. It shows the frame location of each page in the process (see Figure 7.10).
   - Page translation (see Figure 7.12(a)):
     (a) Extract the page number, n from the logical address.
     (b) Use that page number, n as the index to the page table, k.
     (c) Append the offset to k to find the physical address (see Figure 7.11(b)).
   - Do a few examples.

6. Segmentation: a program and its associated data are divided into a number of segments. Segments are not required to be of the same length. The logical address using segmentation consists of a segment number and a segment offset. Segmentation is usually visible to the programmers and provided as a convenience for organizing programs and data. Programmers must be aware of the maximum segment size.

   - An entry in a segment table contains starting address of the segment and the length of the segment.
   - Segment translation (see Figure 7.12(b)):
     (a) Extract the segment number, n from the logical address.
     (b) Use the segment number, n as an index into the process segment table to find the starting address of the segment.
     (c) Compare the offset (right most m bits), to the length of the segment to check the validity of the address.
     (d) The physical address is the sum of the starting address and the offset (see Figure 7.11(c)).

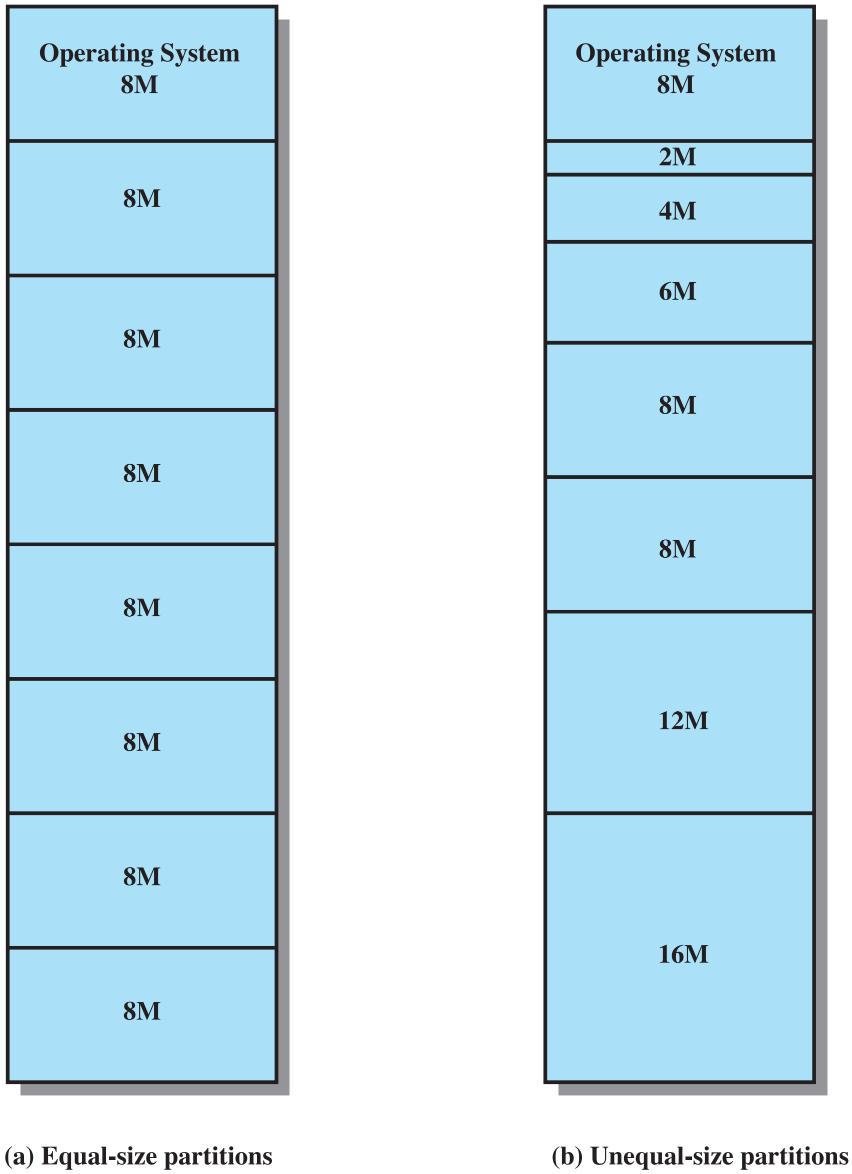

Figure 7.2 Example of Fixed Partitioning of a 64-Mbyte Memory

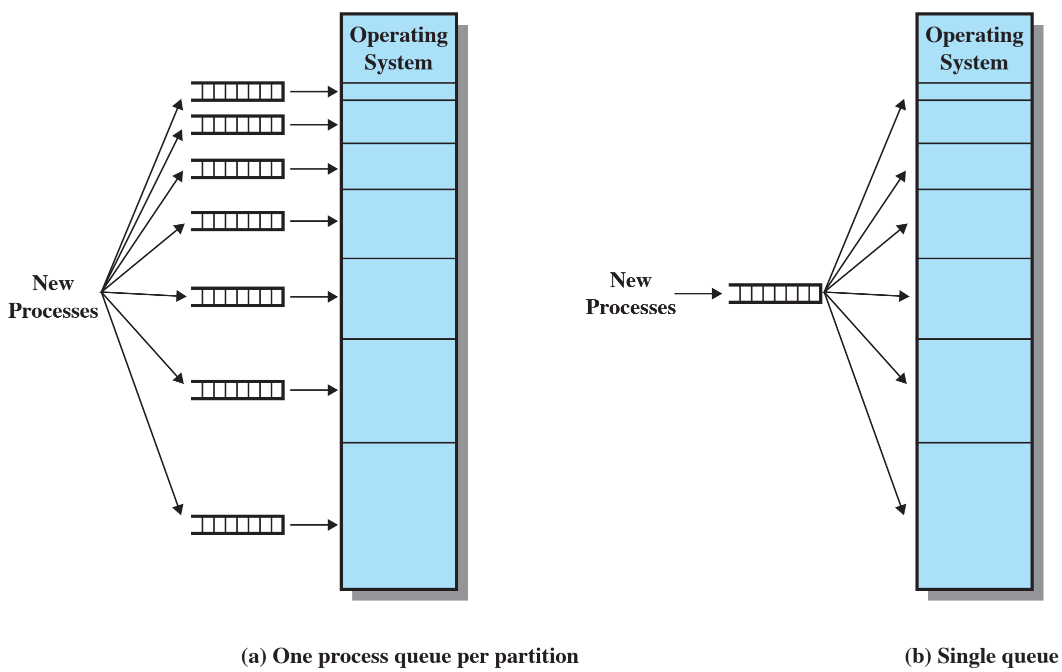

Figure 7.3 Memory Assignment for Fixed Partitioning

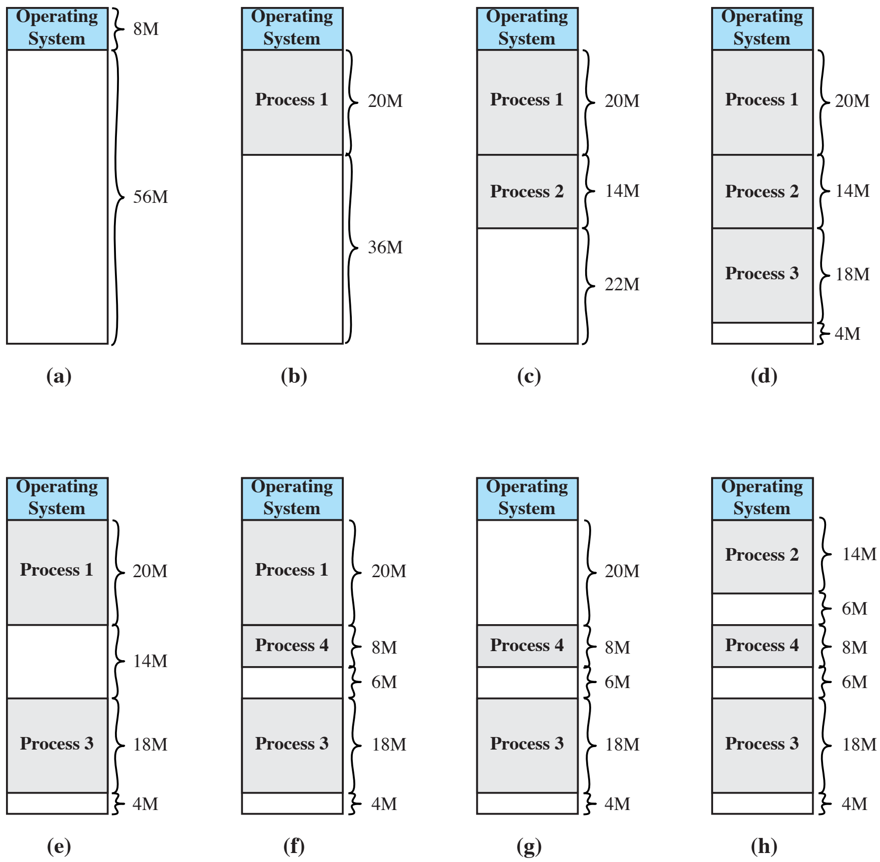

Figure 7.4 The Effect of Dynamic Partitioning

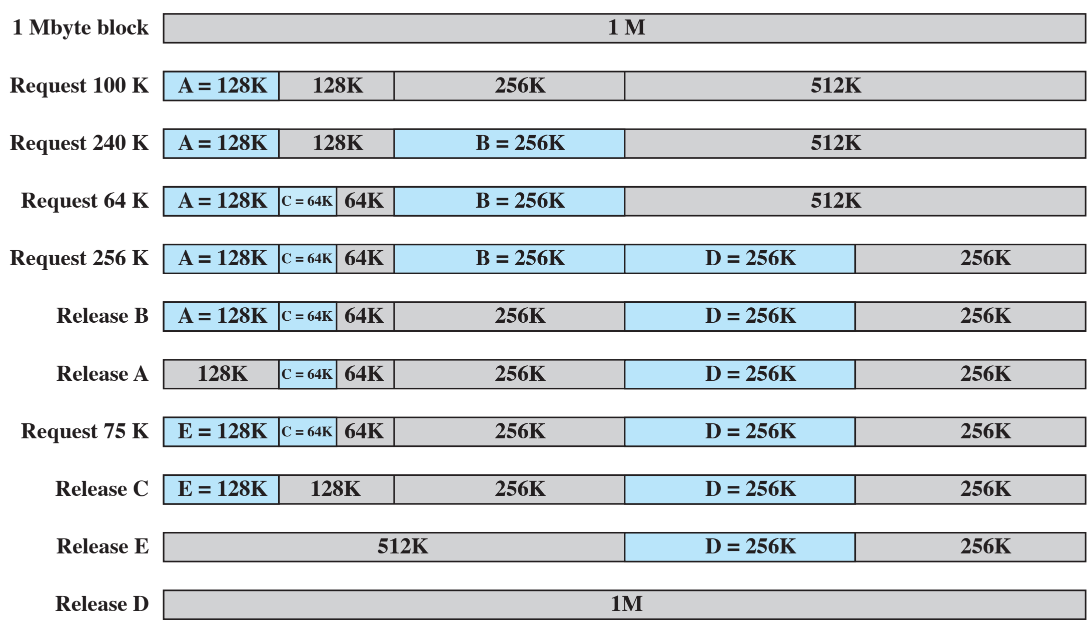

Figure 7.6 Example of Buddy System

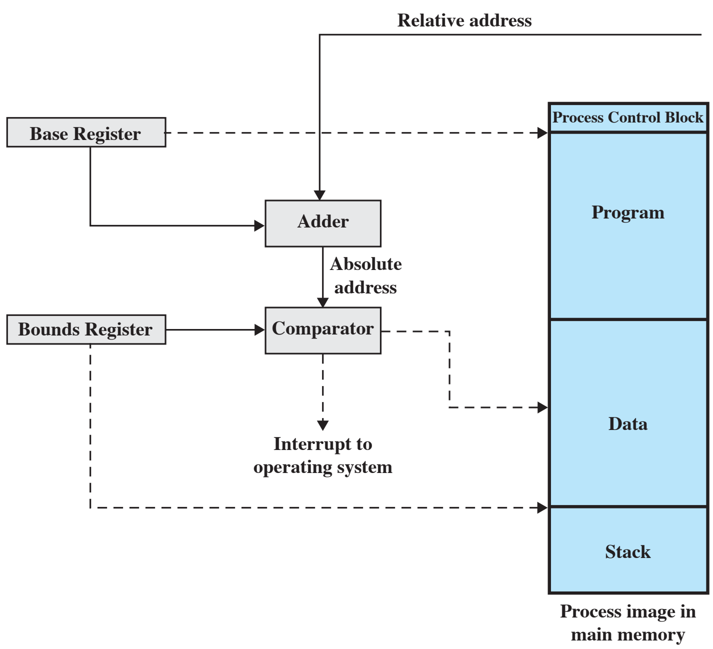

Figure 7.8 Hardware Support for Relocation

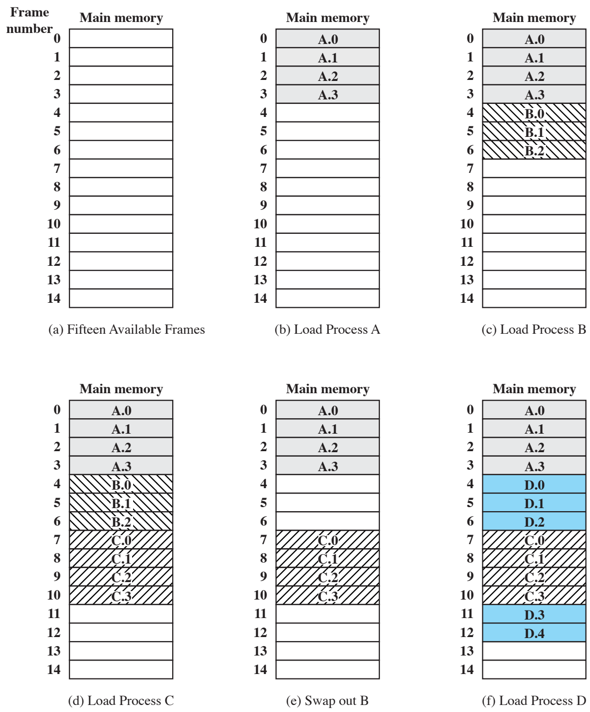

Figure 7.9 Assignment of Process Pages to Free Frames

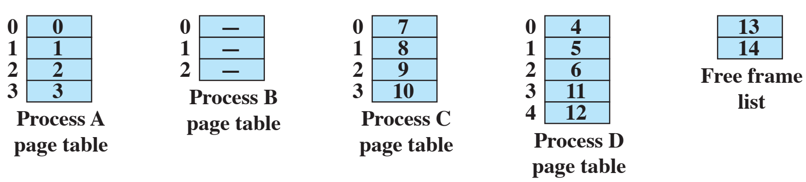

Figure 7.10 Data Structures for the Example of Figure 7.9 at Time Epoch (f)

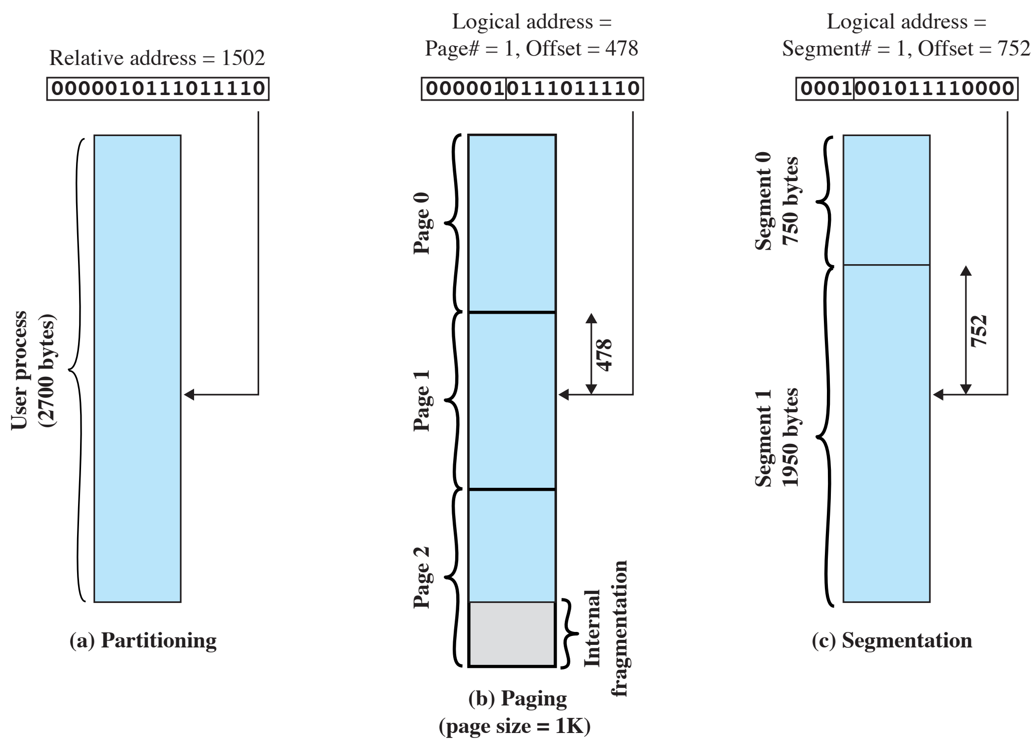

Figure 7.11 Logical Addresses

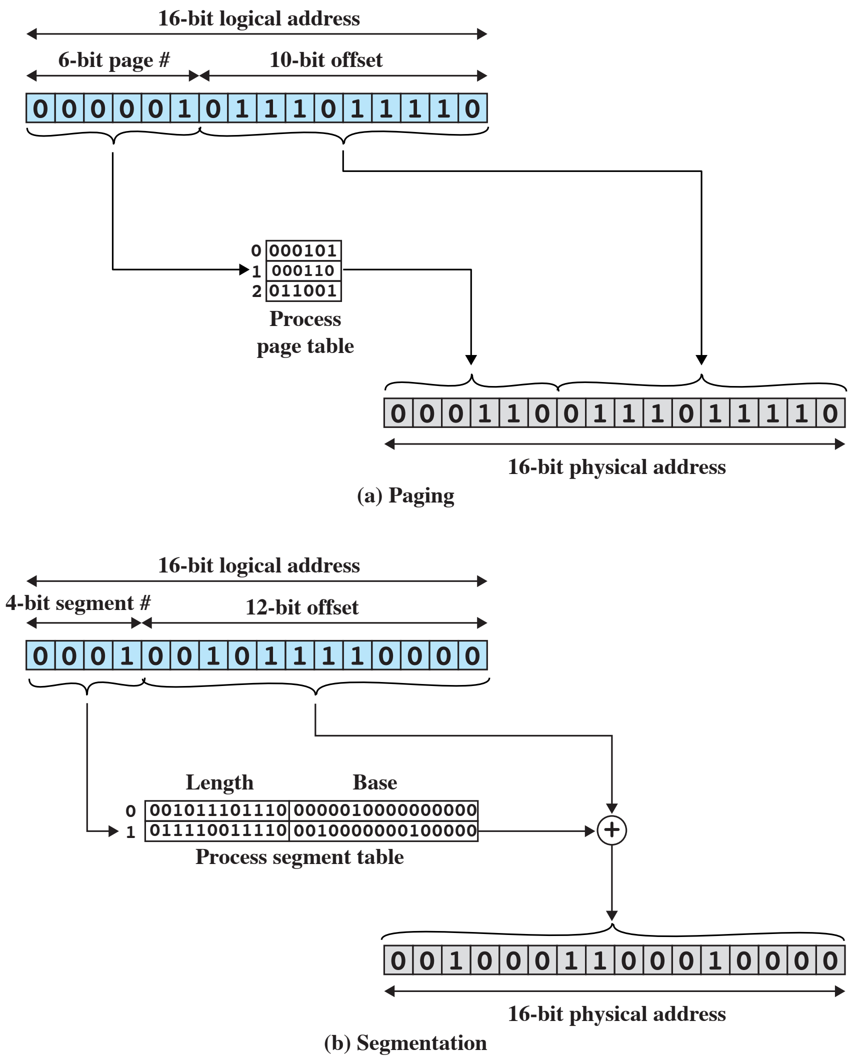

Figure 7.12 Examples of Logical-to-Physical Address Translation