# Sheet7

## MEMORY MANAGEMENT

1) What requirements is memory management intended to satisfy?

Relocation, protection, sharing, logical organization, physical organization.

2) Why is the capability to relocate processes desirable?

Typically, it is not possible for the programmer to know in advance which other programs will be resident in main memory at the time of execution of his or her program. In addition, we would like to be able to swap active processes in and out of main memory to maximize processor utilization by providing a large pool of ready processes to execute. In both these cases, the specific location of the process in main memory is unpredictable.

3) Why is it not possible to enforce memory protection at compile time?

Because the location of a program in main memory is unpredictable, it is impossible to check absolute addresses at compile time to assure protection. Furthermore, most programming languages allow the dynamic calculation of addresses at run time, for example by computing an array subscript or a pointer into a data structure. Hence all memory references generated by a process must be checked at run time to ensure that they refer only to the memory space allocated to that process.

4) What are some reasons to allow two or more processes to all have access to a particular region of memory?

If a number of processes are executing the same program, it is advantageous to allow each process to access the same copy of the program rather than have its own separate copy. Also, processes that are cooperating on some task may need to share access to the same data structure.

5) In a fixed-partitioning scheme, what are the advantages of using unequal-size partitions?

By using unequal-size fixed partitions: 1. It is possible to provide one or two quite large partitions and still have a large number of partitions. The large partitions can allow the entire loading of large programs. 2. Internal fragmentation is reduced because a small program can be put into a small partition.

6) What is the difference between internal and external fragmentation?

Internal fragmentation refers to the wasted space internal to a partition due to the fact that the block of data loaded is smaller than the partition. External fragmentation is a phenomenon associated with dynamic partitioning, and refers to the fact that a large number of small areas of main memory external to any partition accumulates.

7) What are the distinctions among logical, relative, and physical addresses?

- A **logical address** is a reference to a memory location independent of the current assignment of data to memory; a translation must be made to a physical address before the memory access can be achieved.

- A **relative address** is a particular example of logical address, in which the address is expressed as a location relative to some known point, usually the beginning of the program.
- A **physical address**, or absolute address, is an actual location in main memory.

8) What is the difference between a page and a frame?

    In a paging system, programs and data stored on disk or divided into equal, fixed-sized blocks called pages, and main memory is divided into blocks of the same size called frames. Exactly one page can fit in one frame.

9) What is the difference between a page and a segment?

    An alternative way in which the user program can be subdivided is segmentation. In this case, the program and its associated data are divided into a number of segments. It is not required that all segments of all programs be of the same length, although there is a maximum segment length.

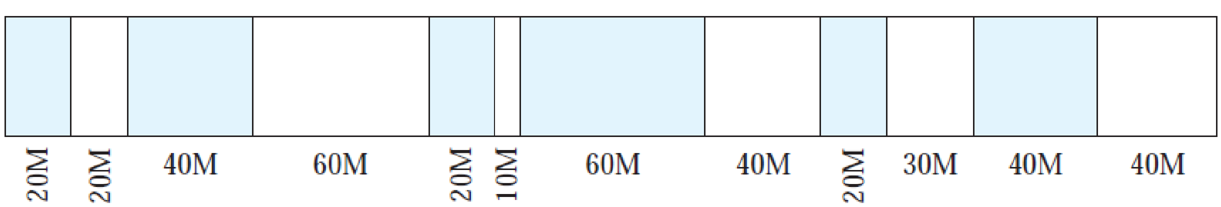
10) Consider a fixed partitioning scheme with equal-size partitions of $2^{16}$ bytes and a total main memory size of $2^{24}$ bytes. A process table is maintained that includes a pointer to a partition for each resident process. How many bits are required for the pointer?

    The number of partitions equals the number of bytes of main memory divided by the number of bytes in each partition: $2^{24}/2^{16} = 2^8$. Eight bits are needed to identify one of the $2^8$ partitions.

11) Another placement algorithm for dynamic partitioning is referred to as worst-fit. In this case, the largest free block of memory is used for bringing in a process. Discuss the pros and cons of this method compared to first-, next-, and best-fit.

    A criticism of the best-fit algorithm is that the space remaining after allocating a block of the required size is so small that in general it is of no real use. The worst fit algorithm maximizes the chance that the free space left after a placement will be large enough to satisfy another request, thus minimizing the frequency of compaction. The disadvantage of this approach is that the largest blocks are allocated first; therefore a request for a large area is more likely to fail.

12) A dynamic partitioning scheme is being used, and the following is the memory configuration at a given point in time. The shaded areas are allocated blocks; the white areas are free blocks. The next three memory requests are for 40M, 20M, and 10M. Indicate the starting address for each of the three blocks using the following placement algorithms:




a) First-fit.

    80, 20 and 120

b) Best-fit.

    230, 20 and 160

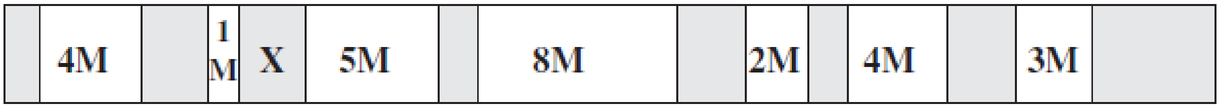
c) Next-fit. Assume the most recently added block is at the beginning of memory.

    80, 120 and 160

d) Worst-fit.

    80, 230 and 360

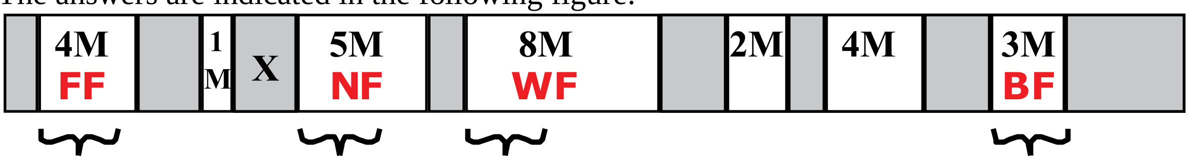
13) This diagram shows an example of memory configuration under dynamic partitioning, after a number of placement and swapping-out operations have been carried out. Addresses go from left to right; gray areas indicate blocks occupied by processes; white areas indicate free memory blocks. The last process placed is 2-Mbyte and is marked with an X. Only one process was swapped out after that.




a) What was the maximum size of the swapped out process?
When the 2-MB process is placed, it fills the leftmost portion of the free block selected for placement. Because the diagram shows an empty block to the left of X, the process swapped out after X was placed must have created that empty block. Therefore, the maximum size of the swapped out process is 1M.

b) What was the size of the free block just before it was partitioned by X?
The free block consisted of the 5M still empty plus the space occupied by X, for a total of 7M.

c) A new 3-Mbyte allocation request must be satisfied next. Indicate the intervals of memory where a partition will be created for the new process under the following four placement algorithms: best-fit, first-fit, next-fit, worst-fit. For each algorithm, draw a horizontal segment under the memory strip and label it clearly.
The answers are indicated in the following figure:

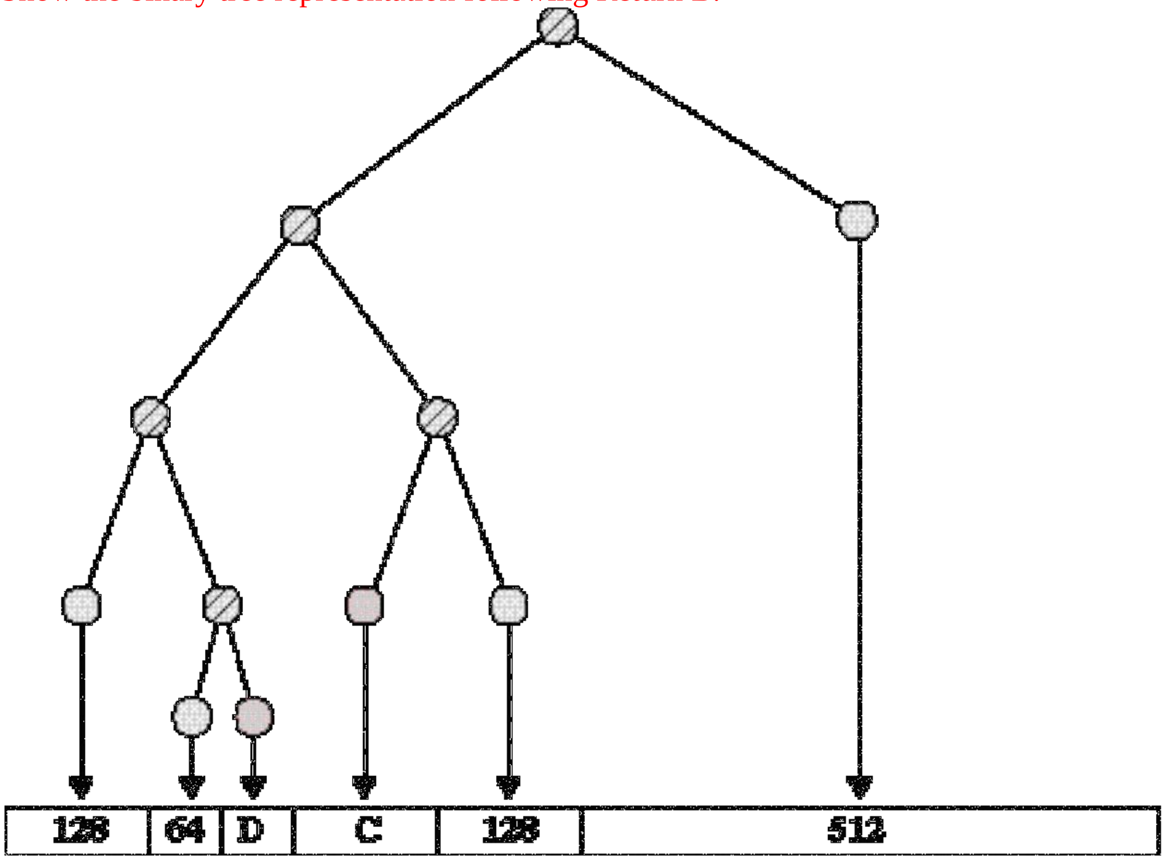



14) A 1-Mbyte block of memory is allocated using the buddy system.

a) Show the results of the following sequence in a figure similar to Figure 7.6 : Request 70; Request 35; Request 80; Return A; Request 60; Return B; Return D; Return C.

| | | | | | | |
|---|---|---|---|---|---|---|
| Request 70 | A | 128 | | 256 | | 512 |
| Request 35 | A | B | 64 | 256 | | 512 |
| Request 80 | A | B | 64 | C | 128 | 512 |
| Return A | 128 | B | 64 | C | 128 | 512 |
| Request 60 | 128 | B | D | C | 128 | 512 |
| Return B | 128 | 64 | D | C | 128 | 512 |
| Return D | 256 | | | C | 128 | 512 |
| Return C | 1024 | | | | | |

b) Show the binary tree representation following Return B.




15) Consider a buddy system in which a particular block under the current allocation has an address of 011011110000.

a) If the block is of size 4, what is the binary address of its buddy?
   011011110100

b) If the block is of size 16, what is the binary address of its buddy?
   011011100000

16) Consider a simple paging system with the following parameters: $2^{32}$ bytes of physical memory; page size of $2^{10}$ bytes; $2^{16}$ pages of logical address space.

a) How many bits are in a logical address?
   The number of bytes in the logical address space is ($2^{16}$ pages) x ($2^{10}$ bytes/page) = $2^{26}$ bytes. Therefore, $2^6$ bits are required for the logical address.

b) How many bytes in a frame?
   A frame is the same size as a page, $2^{10}$ bytes.

c) How many bits in the physical address specify the frame?
   The number of frames in main memory is ($2^{32}$ bytes of main memory)/($2^{10}$ bytes/frame) = $2^{22}$ frames. So 22 bits is needed to specify the frame.

d) How many entries in the page table?
   There is one entry for each page in the logical address space. Therefore there are $2^{16}$ entries.

e) How many bits in each page table entry? Assume each page table entry contains a valid/invalid bit.

   In addition to the valid/invalid bit, 22 bits are needed to specify the frame location in main memory, for a total of 23 bits.

17) Write the binary translation of the logical address 0001010010111010 under the following hypothetical memory management schemes, and explain your answer:

a) A paging system with a 256-address page size, using a page table in which the frame number happens to be four times smaller than the page number

   The page number is in the higher 8 bits: 00010100. We chop it off from the address and replace it with the frame number, which is 4 times less, that is, shifted 2 bits to the right: 00000101. Therefore the result is this frame number concatenated with the offset 10111010:

   Binary physical address = 0000010110111010

b) A segmentation system with a 1K-address maximum segment size, using a segment table in which bases happen to be regularly placed at real addresses: 22 + 4,096 * segment#

   The segment number is in the higher 6 bits: 000101. We chop it off from the address and add the remaining offset 0010111010 to the base of the segment. The base is 22 = 10110 added to the segment number times 4,096, that is, shifted 12 bits to the left: 10110 + 0101000000000000 = 0101000000010110. So adding up the 2 two underlined numbers gives:

   Binary physical address = 0101000011010000

18) Consider a simple segmentation system that has the following segment table:

| Starting Address | Length (bytes) |
|---|---|
| 660 | 248 |
| 1,752 | 422 |
| 222 | 198 |
| 996 | 604 |

For each of the following logical addresses, determine the physical address or indicate if a segment fault occurs:

a) 0, 198

   Segment 0 starts at location 660. With the offset, we have a physical address of 660 + 198 = 858

b) 2, 156

   222 + 156 = 378

c) 1, 530

   Segment 1 has a length of 422 bytes, so this address triggers a segment fault.

d) 3, 444

   996 + 444 = 1440

e) 0, 222

   660 + 222 = 882

19) Consider a system with a 16KB memory. The sequence of processes loaded in and leaving the memory are given in the following.

P1 7K loaded

P2 4K loaded

P1 terminated and returned the memory space

P3 3K loaded

P4 6K loaded

Give the memory map showing allocated portion and free portion after the end of the sequence (if a process cannot be loaded, indicate that) for the following placement algorithms. Also, indicate the internal/external fragmentations.

a) first fit
b) best fit
c) buddy
d) simple paging (assume that each page is of size 2K)