# University of Newcastle

**School of Electrical Engineering and Computing**

**COMP2240 - Operating Systems Workshop 7 - Solution**

**Topics: Memory Management**

1. Consider a memory management system based on paging (without virtual memory). The total size of the physical memory is 2GB, laid out over frame of size 8 KB. The logical address space of each process has been limited to 128 MB.
   1. Determine the total number of bits in the physical address.
   2. Determine the number of bits specifying page replacement and the number of bits for page frame number.
   3. Determine the number of page frames.
   4. Determine the logical address layout.

## Answer:

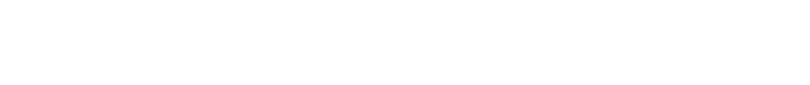
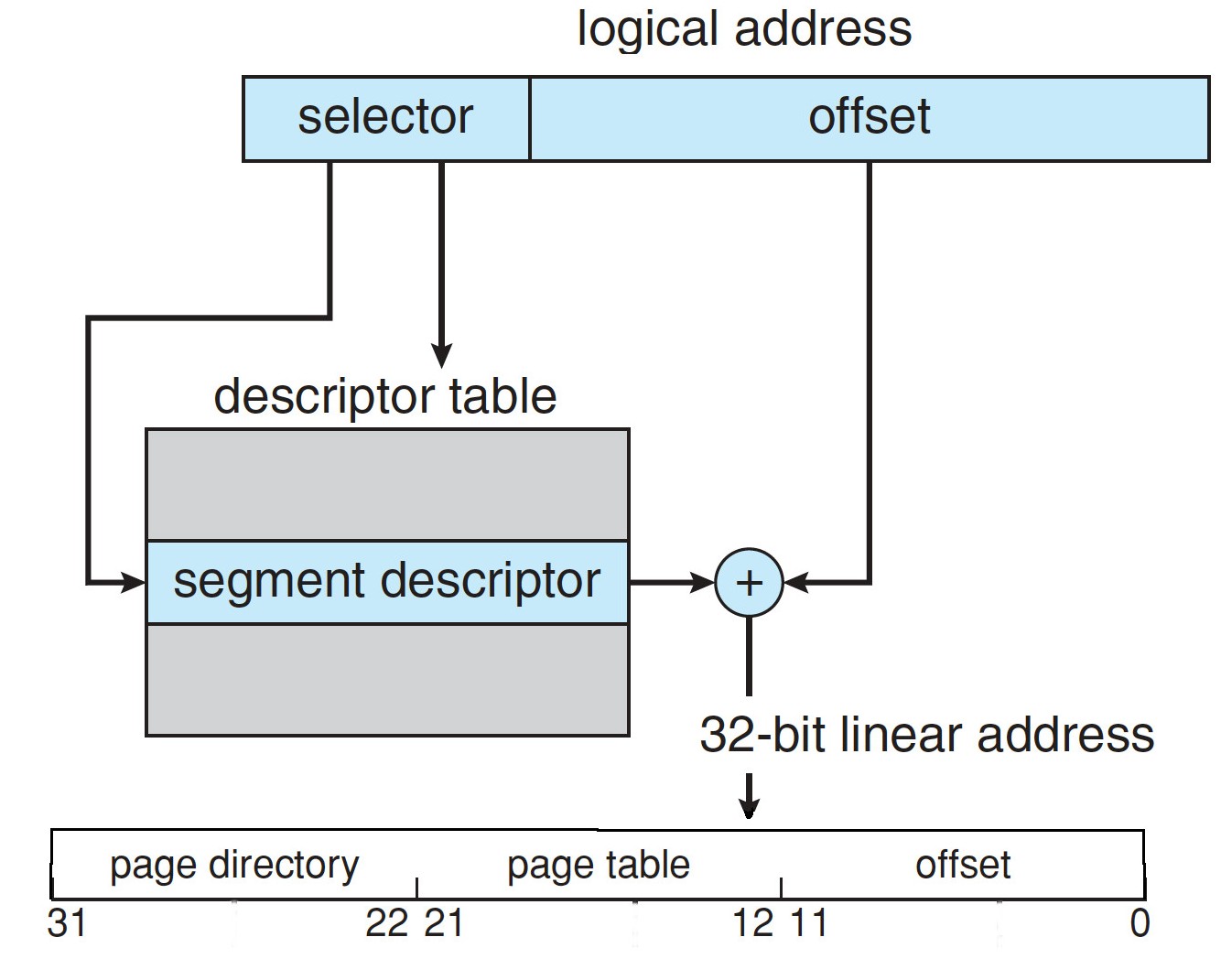
1. The total physical memory size is 2GB. The number of bits in the physical address =

bits.

1. The page size is 8KB.

The number of bits specifying page displacement = bits Thus, the number of bits for the page frame number = 31 − 13 = 18 bits

1. The number of physical frames = 218 = 262144
2. The logical address space for each process is 128MB, requiring a total of 27 bits. The page size is the same as that of the physical pages (i.e., 8KB). Therefore, the logical address layout is 27 bits, with 14 bits for the page number and 13 bits for displacement.
3. Consider the Intel address-translation scheme shown in Figure 1.



*Figure 1: IA-32 segmentation.*

* 1. Describe all the steps taken by the Intel Pentium in translating a logical address into a physical address.
  2. What are the advantages to the operating system of hardware that provides such complicated memory translation?
  3. 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:

1. 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 *dir*, *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.
2. Such a page-translation mechanism offers the flexibility to allow most operating systems to implement their memory scheme in hardware, instead of having to implement some

parts in hardware and some in software. Because it can be done in hardware, it is more efficient (and the kernel is simpler).

1. Address translation can take longer due to the multiple table lookups it can invoke. Caches help, but there will still be cache misses.
2. Consider a simple segmentation system that has the following segment table:

|  |  |
| --- | --- |
| **Base Address** | **Length (bytes)** |
| 0010 1001 0100 | 0000 1111 1000 |
| 0110 1101 1000 | 0001 1010 0110 |
| 0000 1101 1110 | 0000 1100 0110 |
| 0011 1110 0100 | 0010 0101 1100 |

For each of the following logical addresses, determine the physical address or indicate if a segment fault occurs. Assume an offset field of 12 bits, leaving 4 left-most bits for the segment number:

a) 0000 0000 1100 0110

b) 0010 0000 1001 1100

c) 0001 0010 0001 0010

d) 0011 0001 1011 1100

## Answer:

* 1. The left-most 4 bits indicates segment number so ‘0000’ indicates segment 0.

And the remaining 12bits indicate the offset 0000 1100 0110 which is less than the segment length ‘0000 1111 1000’

Now the base address of Segment 0 is 0010 1001 0100. With the offset, we have a physical address of

0010 1001 0100 + 0000 1100 0110 = 0011 0101 1010

* 1. Here, the left-most 4 bits ‘0010’ indicate segment 2 and the offset is ‘0000 1001 1100’

Since the offset is less than the segment length, the physical address is

0000 1101 1110 + 0000 1001 1100 = 0001 0111 1010

c) Here the left-most 4 bits ‘0001’ indicate segment 1 and the offset is ‘0010 0001 0010’

Segment 1 has a length of 0001 1010 0110 which is less than 0010 0001 0010 bytes, so this address triggers a segment fault.

d) Here, the left-most 4 bits ‘0011’ indicate segment 3 and the offset is ‘0001 1011 1100’

Since the offset is less than the segment length, the physical address is

0011 1110 0100 + 0001 1011 1100 = 0101 1010 0000

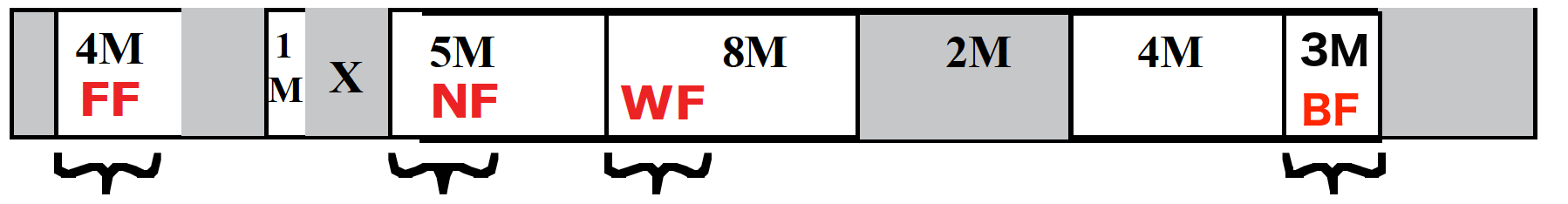
1. This diagram shows an example of memory configuration under dynamic partitioning, after a number of placement and swapping-out operations have been carried out. Addresses go from left to right; gray areas indicate blocks occupied by processes; white areas indicate free memory blocks. The last process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



* 1. What was the maximum size of the swapped-out process?
  2. What was the size of the free block just before it was partitioned by X?
  3. 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, and worst-fit. For each algorithm, draw a horizontal segment under the memory strip and label it clearly.

## Answer:

1. 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.
2. The free block consisted of the 5M still empty plus the space occupied by X, for a total of 7M.
3. The answers are indicated in the following figure:



1. 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.
   1. Discuss the pros and cons of this method compared to first-, next-, and best-fit.
   2. What is the average length of the search for worst-fit?

# Answer:

* + 1. 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.
    2. Same as best fit.

## Supplementary problems:

**S1.** The Intel 8086 processor did not have an MMU or support virtual memory. Nevertheless, some companies sold systems that contained an unmodified 8086 CPU and did paging. Make an educated guess as to how they did it. (Hint: Think about the logical location of the MMU.)

## Answer:

They built an MMU and inserted it between the 8086 and the bus. Thus all 8086 physical addresses went into the MMU as virtual addresses. The MMU then mapped them onto physical addresses, which went to the bus.

**S2.** The IBM System/370 architecture uses a two-level memory structure and refers to the two levels as segments and pages, although the segmentation approach lacks many

of the features described in the lecture. For the basic 370 architecture, the

page size may be either 2 Kbytes or 4 Kbytes, and the segment size is fixed at either 64 Kbytes or 1 Mbyte. For the 370/XA and 370/ESA architectures, the page size is 4 Kbytes and the segment size is 1 Mbyte. Which advantages of segmentation does this scheme lack? What is the benefit of segmentation for the 370?

## Answer:

The S/370 segments are fixed in size and not visible to the programmer. Thus, none of the benefits listed for segmentation are realized on the S/370, with the exception of protection. The P bit in each segment table entry provides protection for the entire segment.

**S3.** 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?

## Answer: There are three reasons why Android does not support swapping: First is that 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. Lastly, there is typically poor throughput between main memory and flash memory.

## Primarily because Android does not wish for its boot disk to be used as swap space- because the boot disk has limited storage capacity. However, Android does support swapping, it is just that users must provide their own separate SD card for swap space.