University of Tlemcen

Computer Science Department

1st Year Engineer

"Introduction to the operating systems 2"

Session 11: Management memory

Exercises

Academic year: 2023-2024

Exercise n° 1

- We assume that we have:
  - *16-bit address bus*
  - *Page size* $4\ KB = 4096\ bytes = 2^{12}\ bytes$*(12-bit address)*
  - *Table of pages as presented in the following figure*

15	**000**
14	000
13	000
12	000
11	111
10	000
9	101
8	000
7	000
6	000
5	011
4	100
3	000
2	110
1	001
0	010

Calculate the physical address corresponding to logical address address 8196?

$16\text{-}bit\ address\ bus \rightarrow 16\text{-}bit\ physical\ address$

We work in decimal format:

$logical\ address = (page\ no., offset)$

$page\ no. = \mathrm{div}(logical\ @, page\ size) = \mathrm{div}(8196,4096) = 2$

offset= $\%(logical\ @, page\ size) = \%(8196,4096) = 4$

$2\ logical\ @ \rightarrow 6\ physical\ @$

$offset\ remains\ the\ same\ in\ both\ addresses$ (physical and logical)

Physical @ $= 6 * 4096 + 4 = 24\,580$

Exercise n° 2

- A computer with the following characteristics:
  - *A physical memory with a capacity of 128 MB,*
  - *32-bit architecture,*
  - *The size of a page is 4 KB,*
  - *A virtual address indexes one byte.*

Give the maximum size of the page table where in this table each entry (frame reference + presence bit)?

Solution: Exercise n° 2

Page table size= Number of logical pages $*$
$(Number\ of\ bits\ to\ encode\ a\ frame\ no. + 1\ bit)$

Number of logical pages represents the number of pages that a process can contain.

$$\text{Number of logical pages} = \frac{Max\ process\ size}{Page\ size} = \frac{2^{32}}{2^{12}} = 2^{20}$$

$$Number\ of\ frames = \frac{Physical\ memory\ size}{Frame\ size} = \frac{2^{27}}{2^{12}} = 2^{15}$$

$To\ represent\ these\ frames\ we\ need\ 15\ bits$

$$\text{Page table size} = 2^{20} * (15\ bits + 1\ bit) = 2^{24}\ bits = 2\ Mo$$

Table of pages

	frame no.	PB
0		1
1		

Exercise n° 3

Consider a program that occupies 3024 bytes of virtual memory. This program is executed in a system that uses memory paging. The size of the physical memory is 1 MB, the size of a page is 512 bytes and the instructions in this program are referenced by a 24-bit virtual address.

1) Calculate :
   - *a) the size of the virtual address space (virtual memory size),*
   - *b) the number of bits of offset,*
   - *c) the number of bits of the page number,*
   - *d) the number of bits of a physical address,*
   - *e) the number of bits of the frame number,*
   - *f) the number of entries in the page table.*
2) Does loading this program cause internal fragmentation? Justify your answer.

Solution : Exercise n° 3

- We have:
  - *Program size = 3024 bytes*
  - *The size of the main memory = 1MO*
  - *Page size = 512 bytes*
  - *Logical address length = 24 bits*

1) Let's calculate :

a) The size of the logical address space (virtual memory) = 2^24=16 Mo

b) The number of offset bits corresponds to the page size

$512\ Bytes = 2^9\ Bytes$

Therefore: the number of offest bits = 9 bits

c) The number of bits in the page number: $24 - 9 = 15\ bits \implies 2^{15}\ pages$

d) The length of a physical address : $1\ Mo = 2^{20} \implies 20\ bits\ for\ physical\ @$

e) The number of frame no. bits: $20 - 9 = 11\ bits.$

f) The number of entries (rows) in the page table = the number of pages = 2^{15} *rows*

2) Yes :

The program needs 3024 bytes of memory.

The number of pages occupied is :

$$\frac{Size\ of\ the\ program}{Size\ of\ the\ page} = \frac{3024}{512} = 6\ pages$$

In the last page, it remains :

$$512 * 6 - 3024 = 48\ free\ bytes$$

What causes internal fragmentation.

Exercise n°4

We consider a 16 KB memory management system, managed in a segmented and paged way with a single paging level. A process can have a maximum of 4 segments. Each segment can address a maximum of 1 KB of data or code. Finally, the size of page frames is fixed at 256 bytes.

1) What is the size (in number of bits) and composition of the physical address?
2) What is the size (in number of bits) of the logical address?
3) What is the size (in bits) of the linear address?
4) What is the size of each page table and the segment table?
5) Consider the logical address 9A1. Explain how the physical address is calculated and give its value, clearly indicating all the intermediate calculations.

Assume that the process uses 4 segments and that the segment table contains the following values (all given in hexadecimal):

Segment:	Limite	Base
0	24A	D29
1	0B7	127
2	1C8	B32
3	037	086

It is also assumed that the process page table is as follows:

Page:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Cadre:	0E	17	01	2A	20	18	30	23	00	00	12	3E	05	00	04	37
Valide:	1	1	1	1	0	1	0	1	1	0	0	1	1	0	0	1

Solution de l'exercice n° 4

1) If you have 16KB of RAM, the physical address is necessarily on 14 bits ($16\ KB = 2^{14}\ Bytes$), with 8 bits for the offset ($2^8 = 256$ bytes per page frame) and 6 bits for the frame number ($2^6 = 64$ frames in all).

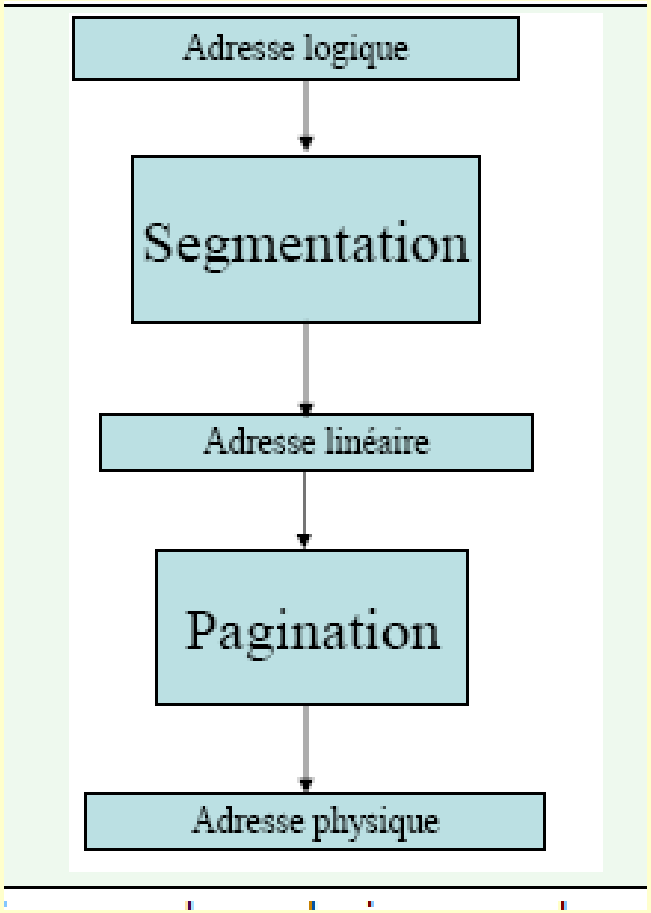
2) To address 4 segments per process, 2 bits are needed.

   So the selector in the logical address is on 2 bits.

   If each segment can address a maximum of 1 KB of data, the offset is 10 bits. The logical address is therefore 12 bits long (each process can address a maximum of 4 KB).

3) As with the physical address, the offset is 8 bits. On the other hand, we don't need 2^6 pages because each process can address a maximum of 4 KB, we just need 16 pages of 256 bytes, so the page number is 4 bits long.

   The linear address is therefore 12 bits long, 4 for the page number and 8 for the offset.


Adresse logique
Segmentation
Adresse linéaire
Pagination
Adresse physique

4) Each page table contains 2^4 rows (entries) of 6 bits (the number of bits used to code a frame) + 1 validity bit. This gives a total of 16*7=112 bits (14 bytes).

The segment table contains a maximum of 4 lines of 10 (limit) + 12 bits (base) each, giving a total of 88 bits (11 bytes).

5) Logical address: 9A1 (1001 1010 0001), divided into segment no. (2) and offset (1A1)

$$\underbrace{10}_{segment\ no.} \qquad \underbrace{01\ 1010\ 0001}_{offset}$$

$$\underbrace{2}_{segment\ no.} \qquad \underbrace{1A1}_{offset}$$

The segment table shows that segment 2 has a limit of 1C8, which is well above the offset.

The linear (virtual) address is therefore :

$$B32\ (base) + 1A1\ (offset) = CD3\ (1100\ 1101\ 0011\ on\ 12\ bits)$$

This consists of C for the page number and D3 for the offset.

According to the page table, page C is located in frame 5 (00 0101).

The physical address is therefore 00 0101 1101 0011, i.e. 05 D3

Exercise n° 5

Consider a memory managed using the "pagination on demand" technique with 3 frames and a sequence of page references in this order: 0, 4, 2, 3, 4, 0, 5, 6, 4, 0, 4, 2, 7, 6, 2, 4, 0, 4, 2, 6. The frames are initially empty.

1) Calculate the number of page faults generated by the FIFO, OPTIMAL, LRU, LFU and SECOND CHANCE replacement algorithms.
2) By reconsidering the memory with 4 frames, test the "Belady anomaly" for the FIFO algorithm.

- FIFO algorithm

  This is the simplest page replacement algorithm. In this algorithm, the S.E keeps track of all the pages in memory in a queue, with the oldest page at the head of the queue. When a page needs replacing, the page at the top of the queue is selected for deletion.

	0	4	2	3	4	0	5	6	4	0	4	2	7	6	2	4	0	4	2	6
p0	**0**	0	0	**3**	3	3	3	**6**	6	6	6	**2**	2	2	2	**4**	4	4	4	**6**
p1		**4**	4	4	4	**0**	0	0	**4**	4	4	4	**7**	7	7	7	**0**	0	0	0
p2			**2**	2	2	2	**5**	5	5	**0**	0	0	0	**6**	6	6	6	6	**2**	2
Page default	x	x	x	x		x	x	x	x	x		x	x	x		x	x		x	x

$$Number\ of\ page\ defaults = 16$$

$$Page\ fault\ rate = \frac{Number\ of\ page\ defaults}{Number\ of\ the\ pages} = \frac{16}{20} * 100 = 80\%$$

- LRU algorithm
  - *In this algorithm, the page will be replaced by the one that has been used least recently.*

	0	4	2	3	4	0	5	6	4	0	4	2	7	6	2	4	0	4	2	6
p0	**0**	0	0	**3**	3	3	**5**	5	5	**0**	0	0	**7**	7	7	**4**	4	4	4	4
p1		**4**	4	4	4	4	4	**6**	6	6	6	**2**	2	2	2	2	2	2	2	2
p2			**2**	2	2	**0**	0	0	**4**	4	4	4	4	**6**	6	6	**0**	0	0	**6**
Page default	x	x	x	x		x	x	x	x	x		x	x	x		x	x			x

Number of page defaults =15

Page default rate =15/20*100 =75%

- OPTIMAL algorithm
  - *In this algorithm, pages that are not used for the longest time in the future will be replaced.*

	0	4	2	3	4	0	5	6	4	0	4	2	7	6	2	4	0	4	2	6
p0	**0**	0	0	0	0	0	0	0	0	0	0	**2**	2	2	2	2	2	2	2	**6**
p1		**4**	4	4	4	4	4	4	4	4	4	4	**7**	7	7	**4**	4	4	4	4
p2			**2**	**3**	3	3	**5**	**6**	6	6	6	6	6	6	6	6	**0**	0	0	2
Défaut page	x	x	x	x			x	x				x	x			x	x			x

Number of page defaults =11

Page default rate = 11/20*100=55%

LFU algorithm

- *In this algorithm, the page with the minimum number of frequencies is selected to be replaced by the page to be entered into the system.*

	0	4	2	3	4	0	5	6	4	0	4	2	7	6	2	4	0	4	2	6
p0	**0**	0	0	**3**	3	3	**5**	5	5	**0**	0	0	**7**	7	**2**	2	2	2	2	2
p1		**4**	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
p2			**2**	2	2	**0**	0	**6**	6	6	6	**2**	2	**6**	6	6	**0**	0	0	**6**
Page default	x	x	x	x		x	x	x		x		x	x	x	x		x			x

Number of page defaults =14

Page default rate = 14/20*100=70%

Second Chance Algorithm

- *A modified form of the FIFO page replacement algorithm, known as the second chance page replacement algorithm, performs relatively better than FIFO.*
- *It works by looking at the start of the queue as FIFO does, but instead of immediately replacing that page, it checks to see if its referenced bit is 1. If it is not 1, the page is replaced otherwise, the referenced bit is reset to 0 and the page will not be replaced.*

	0	4	2	3	4	0	5	6	4	0	4	2	7	6	2	4	0	4	2	6
p0	0/0	0/0	0/0	3/0	3/0	3/0	3/0	6/0	6/0	6/0	6/0	2/0	2/0	2/0	2/1	2/0	0/0	0/0	0/0	6/0
p1		4/0	4/0	4/0	4/1	4/0	5/0	5/0	5/0	0/0	0/0	0/0	7/0	7/0	7/0	4/0	4/0	4/1	4/0	4/0
p2			2/0	2/0	2/0	0/0	0/0	0/0	4/0	4/0	4/1	4/1	4/0	6/0	6/0	6/0	6/0	6/0	2/0	2/0
Pagrs default	x	x	x	x		x	x	x	x	x		x	x	x		x	x		x	x

Number of page defaults =16

Page default rate = 16/20*100=80%

- Belady anomaly with 4 frames

	0	4	2	3	4	0	5	6	4	0	4	2	7	6	2	4	0	4	2	6
p0	**0**	0	0	0	0	0	**5**	5	5	5	5	**2**	2	2	2	2	**0**	0	0	0
p1		**4**	4	4	4	4	4	**6**	6	6	6	6	**7**	7	7	7	7	7	**2**	2
p2			**2**	2	2	2	2	2	**4**	4	4	4	4	**6**	6	6	6	6	6	6
p3				**3**	3	3	3	3	3	**0**	0	0	0	0	0	**4**	4	4	4	4
Page default	x	x	x	x			x	x	x	x		x	x	x		x	x		x	

Number of page defaults =14

Page default rate = 14/20*100=70%