|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 🔑 **Essential** 🔑 | | | |  | 🏆 **Enhanced** 🏆 | | | |
|  |  |  |  |  |  |  |  |  |
| \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ |  | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ | \_\_\_\_\_ |

**Score: \_\_\_\_\_**

**24 – Paged Virtual Memory**

**Activities**

COMP256 – Computing Abstractions

Dickinson College

Spring 2022

Prof. Grant Braught

**Name:**

**Introduction:**

In the prior class we learned about logical memory and how processes interact with it. We saw that each process has its own logical memory and that its logical memory could be significantly larger than the physical memory in the machine. In today’s class we saw how this seemingly impossible feat can be accomplished through a combination of software (OS) and hardware (memory management unit and page table). Finally, we saw how by handling page faults and interrupts the OS implements the logical memory abstraction so that it is transparent to the running processes (and mostly to the user). These activities will reinforce and extend this understanding of virtual memory.

**Some Terminology:**

🔑 1. Using your own words, give a brief explanation or definition of the following terms, as they relate to paged virtual memory:

a. Page

b. Page Frame

c. Page Fault

🔑 2. Briefly explain in a few sentences of your own words how a paged virtual memory system facilitates the sharing of the physical memory of a computer among many processes. Your answer must correctly use the terms page and page frame.

**Address Translation in Paged Virtual Memory:**

The process by which logical addresses are translated to physical addresses in a paged virtual memory system was explained in today’s class and several examples were done. The next few questions will ask about this type of address translation.

It is not required viewing, but if you would like another explanation of address translation in a paged virtual memory system the video Virtual To Physical Translation by Milos Prvulovic at Georgia Tech does a nice job:

* <https://www.youtube.com/watch?v=l7HoguhFVQ4> (2:25)

🔑 3. In class our examples used 1KB (i.e. 1024 byte) pages. For this question, consider a paged virtual memory system that uses 2KB (i.e. 2048 byte) pages.

a. Give the page number and offset corresponding to the following logical addresses:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  | **Logical Address** | **Page** | **Offset** |  |
|  | 1786 |  |  |  |
|  | 2047 |  |  |  |
|  | 3072 |  |  |  |
|  | 3924 |  |  |  |
|  | 5270 |  |  |  |
|  |  |  |  |  |

b. Give the full logical address corresponding to the following page/offset values:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  | Page | Offset | Logical Address |  |
|  | 0 | 1398 |  |  |
|  | 1 | 422 |  |  |
|  | 4 | 0 |  |  |
|  | 5 | 1398 |  |  |
|  |  |  |  |  |

🔑 4. Consider the logical and physical memories shown below:



a. In a paged virtual memory system, the OS maintains a page table for each process. The MMU uses these page tables to translate logical page numbers to physical page frame numbers. Complete the page tables below for process 1 and process 2 as shown above.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  | **Process 1** |  |  | **Process 2** |  |  |
|  | **Page** | **Page Frame** |  | **Page** | **Page Frame** |  |
|  | 0 |  |  | 0 |  |  |
|  | 1 |  |  | 1 |  |  |
|  | 2 |  |  | 2 |  |  |
|  | 3 |  |  | 3 |  |  |
|  | 4 |  |  | 4 |  |  |
|  | 5 |  |  | 5 |  |  |
|  | 6 |  |  | 6 |  |  |
|  | 7 |  |  | 7 |  |  |
|  | 8 |  |  | 8 |  |  |
|  | 9 |  |  | 9 |  |  |
|  | 10 |  |  | 10 |  |  |
|  |  |  |  |  |  |  |

b. Using your page tables from part a, complete the logical to physical address translations shown in the table below. For these translations, assume that the virtual memory system uses 1kb pages.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  | **Logical** | **Logical** | **Physical** | **Physical** |  |
|  | **Process** | **Page / Offset** | **Address** | **Frame / Offset** | **Address** |  |
|  | 1 | 0 / 120 | 120 | 6 / 120 | 6264 |  |
|  | 2 | 0 / 120 |  |  |  |  |
|  | 1 | 3 / 950 |  |  |  |  |
|  | 2 | 5 / 950 |  |  |  |  |
|  |  |  |  |  |  |  |

c. Give a logical address as a page/offset that meets each of the following criteria:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  | **Criteria** | **Page** | **Offset** |  |
|  | Causes a page fault if accessed by process 1. |  |  |  |
|  | Causes a page fault if accessed by process 2. |  |  |  |
|  | Causes a page fault if accessed by process 1 but not if accessed by process 2. |  |  |  |
|  | Causes a page fault if accessed by process 2 but not if accessed by process 1. |  |  |  |
|  |  |  |  |  |

**Virtual Memory Tradeoffs:**

Providing the logical memory abstraction has many benefits. It facilitates the sharing of the physical memory among multiple processes, it simplifies processes by allowing them to interact with their logical memory rather than worrying about where they are in physical memory and it allows processes to be much larger than they could be otherwise. However, all that doesn’t come for free, there are tradeoffs. The following questions explore these tradeoffs.

🔑 5. In class, the campus library was used as a metaphor to help understand the operation of virtual memory.

a. Fill in the cells in table below with the terms listed to indicate what corresponds to the physical and virtual memory in the library metaphor and in an actual computer system.

* + Library Building
  + Hard Disk Drive
  + Main Memory (RAM)
  + Interlibrary Loan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  | **Library Metaphor** | **Computing System** |  |
|  | Physical Memory |  |  |  |
|  | Virtual Memory |  |  |  |
|  |  |  |  |  |

b. What event in the campus library metaphor would correspond to a page fault?

c. Use the library metaphor to briefly explain why it takes longer to access information stored in the virtual memory than in the physical memory.

d. Briefly explain why it takes longer to access information stored in the virtual memory than in the physical memory on a computing system.

🔑 6. You may have heard the advice that if your system is getting slow adding more RAM will make it faster. This is almost entirely due to the functioning of the virtual memory system and the page fault rate. This question explores that a little further.

Consider a system that has a fixed amount of physical memory (RAM) installed and is running an operating system that uses paged virtual memory as described in class.

a. Imagine that there are 5 user processes running. At any given time some of the pages of each of these processes will be in the physical memory, and the remaining ones will be on the disk. Now imagine that we launch 5 more user processes, for a total of 10 running processes. Will each process be able to have more or less of its pages in the physical memory on average? Briefly explain your reasoning.

b. Considering your answer to part a, would you expect the number of page faults that occur to increase or decrease as the number of running processes is increased? Briefly explain your reasoning.

c. What effect does increasing the total amount of RAM in a system have on the number of page frames that exist in the physical memory? Briefly explain your reasoning.

d. Based on your answers to parts a-c, briefly explain why adding more physical memory (RAM) to a system that uses a paged virtual memory system can greatly improve its performance?

🔑 7. Briefly describe in a few sentences of your own words the advantages and disadvantages of virtual memory. That is, what does virtual memory do for us (advantages) and what do we give up (disadvantages) in order to gain the advantages?

**Memory Management Implementation:**

🔑 8. Briefly explain why when a new process is created the OS does not immediately allocate all of the pages in the logical memory.

🔑 9. Briefly explain how the OS keeps track of which pages in the logical address space have been allocated.

🔑 10. Briefly explain what causes a segmentation fault.

🏆 11. We know that because the logical memory space is so large the OS will allocate only a small part of it when a new process is created. This includes a small amount of heap space. But how much? Study the C program at: <https://repl.it/@braughtg/SegFault>.

a. What happens when you run this program? Briefly explain why that happened in terms of the process’ memory.

c. Fork the repl for this program and experiment with it to determine approximately how many bytes of heap space are initially allocated to the program. State your answer and briefly explain how you determined it.

It is not required viewing but if you are interested in learning about the heap and the malloc function in more detail you can watch the video *The Heap: what does malloc() do?* by LiveOverflow.

* <https://www.youtube.com/watch?v=HPDBOhiKaD8>

**But That Page Table…:**

🏆 12. Consider a system with the following characteristics:

* A paged virtual memory system.
* 48 bit logical addresses with 8KB pages.
* 16GB of physical memory (RAM)
* 44 bit physical addresses.

a. How many pages are there in the virtual address space?

b. How many rows would there be in the page table for a process?

c. How page frames of physical memory are there?

d. How many bits would be required to specify the offset within a page or page frame (i.e. to uniquely address each of the 8kB addresses in a page)?

e. Given your answer to d, how many bits of the physical memory address would be left to represent the page frame?

f. Recall that each row in the page table must store a page frame number. Given your answers to b and e, how many bits would the page table need to contain? Express that result as a number of MB.

g. Would the amount of memory you found in part f be reasonable in a real system running multiple processes each of which needs its own page table? Why or why not?

Recognizing that most of the page table entries for most processes will be invalid would suggest that there is probably a better way to represent this information. Not surprisingly, there are a few options. If you are interested in what they might be, you can read about inverted or multi-level page tables or wait and keep your eyes open for them in the operating systems course where these topics are covered in some detail.

Optional: To help me improve and scope these activities for future semesters please consider providing the following feedback.

a. Approximately how much time did you spend on this activity outside of class time?

b. Please comment on any particular challenges you faced in completing this activity.