Gebze Technical University
Department Of Computer Engineering
CSE 312 / CSE 504
Operating Systems

Homework #03
Due Date: May 30th 2017
(No late submissions will be accepted for this homework)
Paging

In this homework, you will simulate the virtual memory technique paging. You will write a new memory management unit (MMU) that derives from memoryBase and handles all paging related issues. Your new MMU supports page faults and performs page replacement algorithms.

Your MMU and the general computer will have the following

- Your computer has 8 KBytes of physical main memory
- Each process has a virtual address space of 16 KBytes. Therefore, even if you have one process running, you will need many page replacements because your physical memory is not large enough to hold one process.
- If an instruction does not find its operand in memory or it is not in memory, it causes a page fault
- Your programs will use virtual addresses, so each address needs to be translated

Your paging system will have the following features

- The page size will be 1 KBytes
- The page table will hold the following information for each page
  - Modified bit
  - Referenced bit
  - Present/absent
  - Page frame number
  - Any other information needed by your system
- You will use the FIFO method for the page replacement algoritm. This algorithm is not very efficient but it is easy to implement.

For this homework, you will do the following

- Update your OS class to support both multiprocessing and paging.
- Derive a new memory class from the MemoryBase class that can be used with the new 8080 CPU to support paging, page tables, etc.

You will write two ASM programs for your new system

- You will write an ASM program (sortv3.asm) that sorts 10.000 numbers and prints them similar to the sort program that you wrote in HW1. You will choose random 10.000 numbers and put them in your ASM file using DW directives.
- You will also write another small ASM program (loadsort.asm) that uses fork system class to run three sortv3.asm programs at the same time.

.

Write a simulation program that runs your systems with some command line parameters. There should be parameters for the program name and debug flag.

- **`sim8080 exe.com 1`** : will read the program from exe.com. In debug mode 1, the status of the CPU and the page table will be printed to the screen after each instruction execution. At the end of the simulation, the whole memory will be saved to exe.mem as a text file of hexedecimal numbers. Each line of the file will start with the memory addres, then it will show 16 bytes of hexadecimal numbers separated with spaces.
- In Debug mode 0, the program will be run and the contents of the memory will be saved as in Debug mode 1.
- In Debug mode 2, after each CPU emulation, the contents of the page table will be displayed. Your simulation will wait for an entry from the keyboard and it will continue for the next tick.
- In Debug mode 3, the contents of the page table will be printed after each page fault. You should also print information about which pages were replaced and the contents of the page FIFO.

At the end of the program run, you will print statistics about your page replacements such as number of pages replaces, number of page faults, etc.

We will provide the submission instructions in a separate document. You should strictly follow these instructions otherwise your submission may not get graded.
You will submit the following files for this homework

- memory.h and memory.cpp: the MMU files to handle paging
- main.cpp
- gtuos.cpp and gtuos.h
- sortv3.asm and loadsort.asm as described above
- Any other cpp, header, make files that we need to compile your submission.