CS4414
Project 3: Designing a Virtual Memory Manager
Wright Kim

This project was to simulate how Operating Systems manage its virtual memory. The logical addresses were given, which can be found from address.txt, and the BACKING_STORE.bin is a binary file where you copy data to the physical memory whenever needed. The size of the page table was 16 and the size of the physical memory (frame) was 8. Each frame has a space of 256 bytes.

In order to run the program,
$make

./memory

The virtual memory manager first consults the TLB.
1) the page number is obtained from the logical address, and the TLB is consulted
2) If TLB hit, the frame number is obtained from the TLB
3) If TLB miss, the page table must be consulted. Either,
    a. frame number is obtained from the page table or,
    b. page fault is occurred
4) when page fault occurs, I programmed to call a function copyBin to read in a 256 byte page from the file BACKING_STORE.bin and store it in an available page frame in physical memory. To copy 256 bytes from the binary file, first needed to get the offset of the BACKING_STORE.bin file which can be obtained by multiplying the page number by 256. For an example, if the page number is 1 and need to load this to the frame, should read from 1 * 256 = 256 of the binary file.

The TLB uses first fit until it is fully filled, then uses FIFO to fill the TLB entry. The frame (physical address) uses first fit until it is filled. Once it is filled, it uses LRU stack algorithm to determine which frame that page number should be placed.

Per recommendation of the author of the text book, I first implemented page table and the TLB was implemented at the very end. Once page table was implemented, I implemented LRU stack algorithm using the deque data structure of C++. The reason of using deque was that it supported pop_front and I only had to implement insertValue function using the provided insert function to simplify the inserting process. Once LRU stack was implemented, I implemented the physical address or frame and updated the values using the information obtained from LRU stack data structure. Lastly, implemented TLB. Every step was carefully tested to confirm the program was functioning as it should.

For all of the process, I carefully programmed to update the entries of TLB, Page Table, and Physical Address properly.

The sample test provided by the professor was used as a templet to print out all of the required information for the project, and confirmed that my program was indeed printing all values as designed.