This memory manager program works by reading in a list of logical addresses looping from a text file. Each of the logical addresses read in 32-bit integers. First, the logical address is masked by doing a bitwise and with 0xFFFF, because 0xFFFF has sixteen 1’s, and our physical address is contained in the 16 least significant bits of our 32 bit address. Now we only have the sixteen lowest bits, we right shift by 8 to get the top 8 bits of our address for the page number.

All that’s left to determine now is the offset, which is obtained by taking the low order 8 bits we are left with after anding the 32 bit logical address with 0xFF, which in binary has 8 ones.

First, I read each page to a frame number from the backing store into my physical memory, and I check my TLB for the corresponding page number, and if it’s a TLB Hit, all left to do is find the corresponding frame in the TLB. If it was a miss, then the TLB had to be updated with a corresponding page number and an arbitrary page frame value, which is just a counter that was incremented each time the TLB was written to. If there is a TLB miss, the first step after updating the TLB, was to check my page table. My page table was to be of size 256, and for my page table, I initialized an array with 256 -1’s. If my page table had a -1 at the index of the page number I was looking for, I would then be able to easily determine if it is a page miss! And this is also the time that I increase my page fault counter and page number. If it was a hit, I simply had to index my page table with the desired page number, to get the corresponding frame value. After all of this above code finished executing, there is no question whether the 2d array representing physical memory contained the corresponding value for the page that was desired. All that was left to do was to index the physical memory with the page, and also the corresponding offset. The TLB page table operated with a FIFO structure, and as the TLB page table could only hold 16 pages, after the 16th page filled up and there was a miss, the last page to be accessed was booted and overwritten by the newest page.