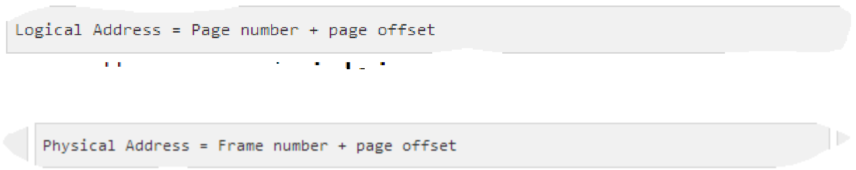
Course: Operating Systems
:
Name: Adrianna Smith and Vivek Bigelow
:
HMW: 5
:
Date Due: 04.10.19

Introduction

The objective of this assignment is to help us to learn the differences between logical/virtual memory as well as physical memory. Students should also learn to understand page faults and how pages work overall in memory. There also should be a stronger grasp of address translation after completing this assignment. Virtual, or logical, memory is memory management that allows an OS to use both software and hardware to compensate for physical memory. The benefit of using virtual memory is that unlike physical memory it allows the user to use more memory than what is tangibly available. Physical memory is typically represented by the actual memory located on the hard disk. It's benefits is that the memory is less likely to become 'lost'. Page faults are when the memory addresses do not show up in the actual memory. Address translation is the process of translating logical memory to a logical address and physical memory to a physical address(see figure 0 bellow). This is needed because it gives optimum utilization of the main memory and to avoid external fragmentation. Overall this assignment should give a glimpse of how these topics can be used in future assignments as well as the real world.



```
Logical Address = Page number + page offset
```

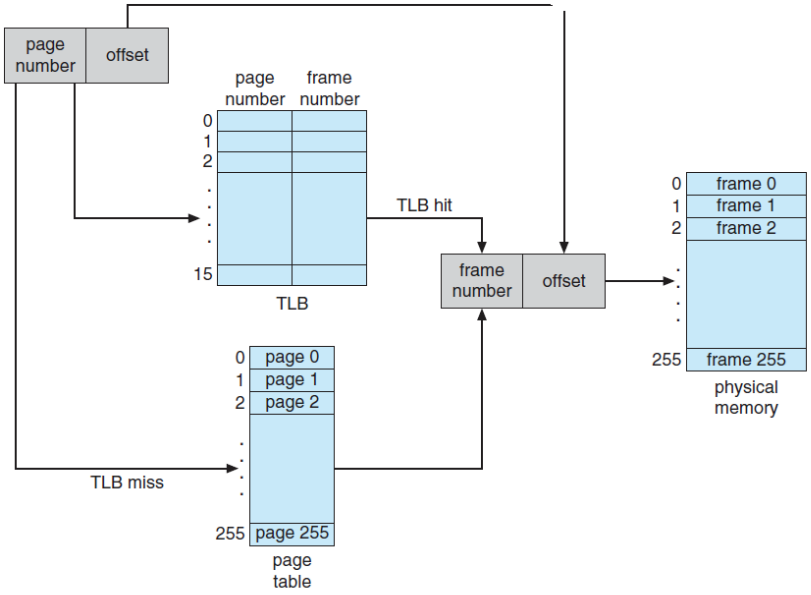
```
Physical Address = Frame number + page offset
```

Figure 0: Types of addresses

Methodology

The general approach that we had for this particular assignment was to meet to discuss design options and ideas for the overall project. We then assigned different parts of the project to each member. After this we tested the code for the project. Finally, we recorded our results which can be seen bellow.

First thing that was tackled was reviewing the material in the book, especially chapter 9, as well as the professor's comments to see how we wanted to approach the assignment. The first thing that needed to be approached was figuring out how to do the address translation(see figure 1 for details). We determined that this would be one of the more challenging aspects of the assignment to complete. The idea here was to figure out the best algorithm to use to extract the page numbers and offset from the given address number in address.txt. We decided to use FIFO in updating the TBL. After figuring out how to do this, everything else became moderately simple to complete. We made it so that the code prints out the corresponding physical address as well as the logical/virtual address and the sign byte. In the testing phase to check the correctness of the code we contrasted the output values on the command prompt to the correct.txt provided in the VM. Finally, we made it so that the program outputs the statistic of the run that was done. These include the page-fault rate/percentage of addresses that resulted in page faults, and the TLB hit rate.



]

Figure 1: Address Translation Process

Results

The first step we did to get the results was compile the code and make sure the correct files were located in the same folder, see figures bellow. The code compiled with no errors.

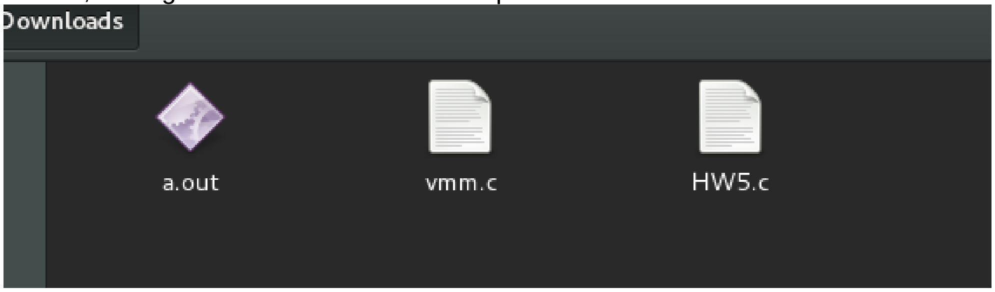


Figure 2: Compiled HW5.c file

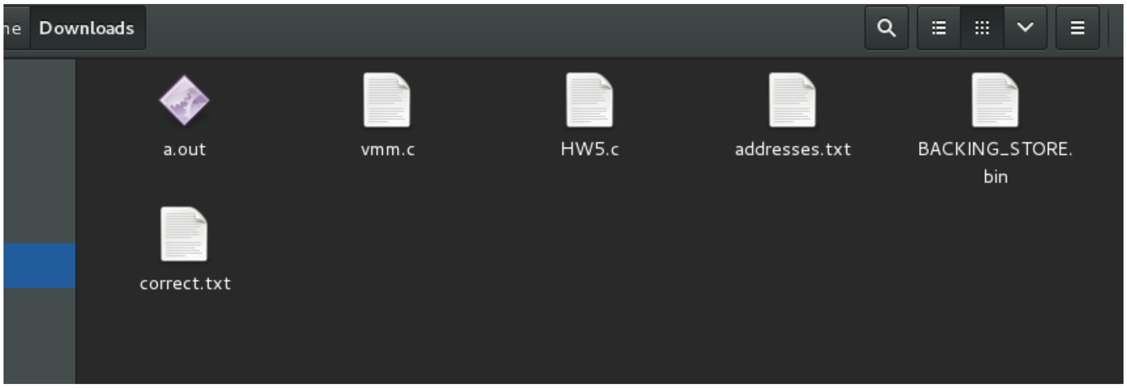
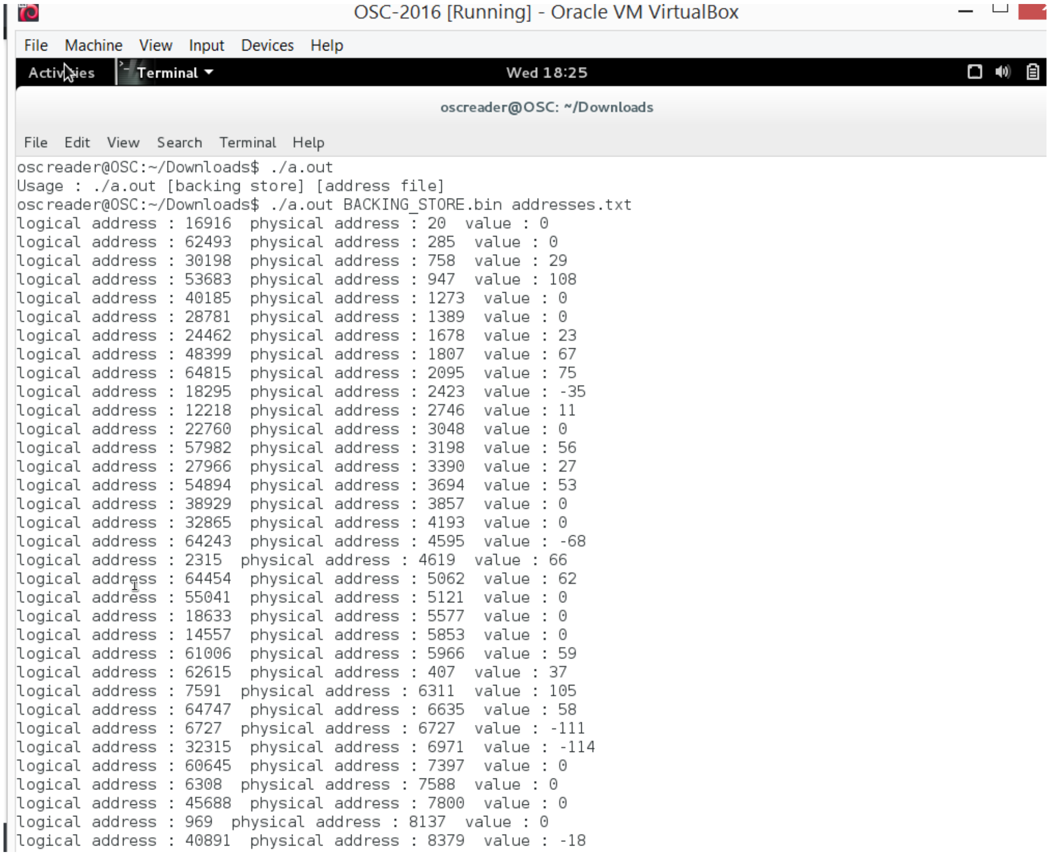


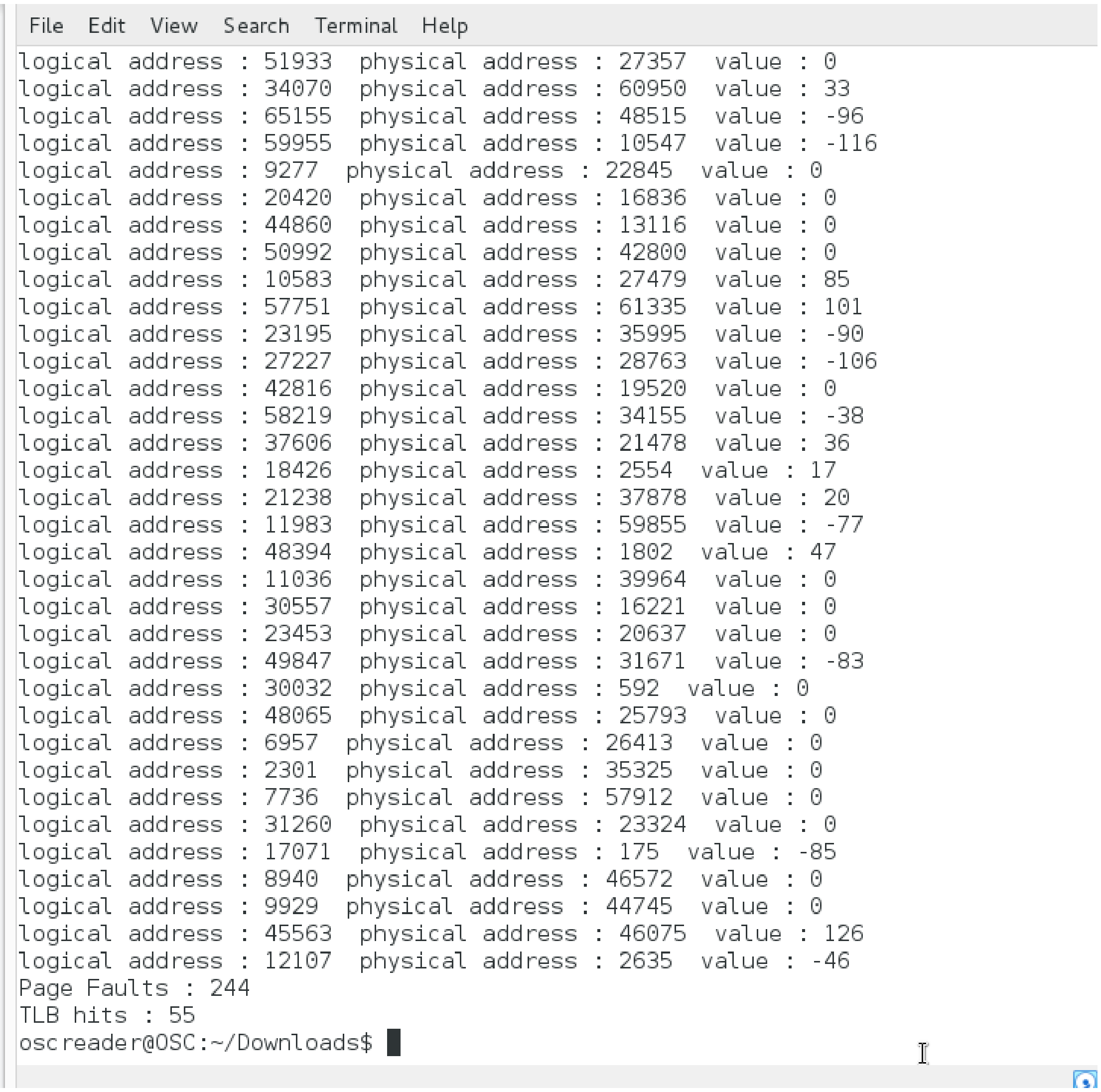
Figure 3: All files in same folder

After this the file was ran with './a.out' on the terminal. As a result the code print off the following, see figure bellow.


OSC-2016 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Wed 18:25
oscreader@OSC: ~/Downloads
File Edit View Search Terminal Help

```
oscreader@OSC:~/Downloads$ ./a.out
Usage : ./a.out [backing store] [address file]
oscreader@OSC:~/Downloads$ ./a.out BACKING_STORE.bin addresses.txt
logical address : 16916  physical address : 20  value : 0
logical address : 62493  physical address : 285  value : 0
logical address : 30198  physical address : 758  value : 29
logical address : 53683  physical address : 947  value : 108
logical address : 40185  physical address : 1273  value : 0
logical address : 28781  physical address : 1389  value : 0
logical address : 24462  physical address : 1678  value : 23
logical address : 48399  physical address : 1807  value : 67
logical address : 64815  physical address : 2095  value : 75
logical address : 18295  physical address : 2423  value : -35
logical address : 12218  physical address : 2746  value : 11
logical address : 22760  physical address : 3048  value : 0
logical address : 57982  physical address : 3198  value : 56
logical address : 27966  physical address : 3390  value : 27
logical address : 54894  physical address : 3694  value : 53
logical address : 38929  physical address : 3857  value : 0
logical address : 32865  physical address : 4193  value : 0
logical address : 64243  physical address : 4595  value : -68
logical address : 2315  physical address : 4619  value : 66
logical address : 64454  physical address : 5062  value : 62
logical address : 55041  physical address : 5121  value : 0
logical address : 18633  physical address : 5577  value : 0
logical address : 14557  physical address : 5853  value : 0
logical address : 61006  physical address : 5966  value : 59
logical address : 62615  physical address : 407  value : 37
logical address : 7591  physical address : 6311  value : 105
logical address : 64747  physical address : 6635  value : 58
logical address : 6727  physical address : 6727  value : -111
logical address : 32315  physical address : 6971  value : -114
logical address : 60645  physical address : 7397  value : 0
logical address : 6308  physical address : 7588  value : 0
logical address : 45688  physical address : 7800  value : 0
logical address : 969  physical address : 8137  value : 0
logical address : 40891  physical address : 8379  value : -18
```

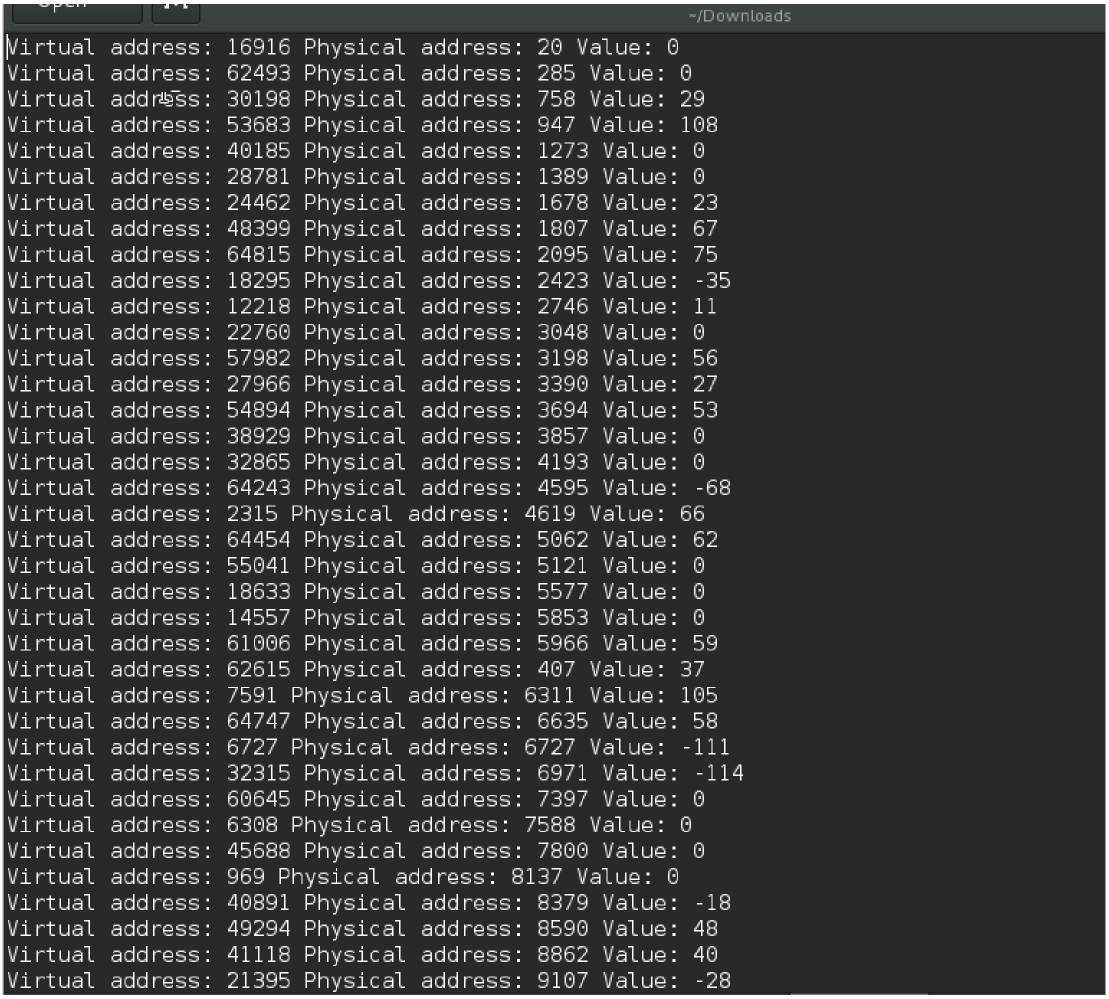
Figure 3: Output of Program



```
logical address : 51933  physical address : 27357  value : 0
logical address : 34070  physical address : 60950  value : 33
logical address : 65155  physical address : 48515  value : -96
logical address : 59955  physical address : 10547  value : -116
logical address : 9277  physical address : 22845  value : 0
logical address : 20420  physical address : 16836  value : 0
logical address : 44860  physical address : 13116  value : 0
logical address : 50992  physical address : 42800  value : 0
logical address : 10583  physical address : 27479  value : 85
logical address : 57751  physical address : 61335  value : 101
logical address : 23195  physical address : 35995  value : -90
logical address : 27227  physical address : 28763  value : -106
logical address : 42816  physical address : 19520  value : 0
logical address : 58219  physical address : 34155  value : -38
logical address : 37606  physical address : 21478  value : 36
logical address : 18426  physical address : 2554  value : 17
logical address : 21238  physical address : 37878  value : 20
logical address : 11983  physical address : 59855  value : -77
logical address : 48394  physical address : 1802  value : 47
logical address : 11036  physical address : 39964  value : 0
logical address : 30557  physical address : 16221  value : 0
logical address : 23453  physical address : 20637  value : 0
logical address : 49847  physical address : 31671  value : -83
logical address : 30032  physical address : 592  value : 0
logical address : 48065  physical address : 25793  value : 0
logical address : 6957  physical address : 26413  value : 0
logical address : 2301  physical address : 35325  value : 0
logical address : 7736  physical address : 57912  value : 0
logical address : 31260  physical address : 23324  value : 0
logical address : 17071  physical address : 175  value : -85
logical address : 8940  physical address : 46572  value : 0
logical address : 9929  physical address : 44745  value : 0
logical address : 45563  physical address : 46075  value : 126
logical address : 12107  physical address : 2635  value : -46
Page Faults : 244
TLB hits : 55
oscreader@OSC:~/Downloads$
```

Figure 4: Statistic results

Finally the results were compared with the 'correctaddresses.txt' file. As you can see bellow, the addresses and values match up for the most part.(see figure 5 bellow)



```
Virtual address: 16916 Physical address: 20 Value: 0
Virtual address: 62493 Physical address: 285 Value: 0
Virtual address: 30198 Physical address: 758 Value: 29
Virtual address: 53683 Physical address: 947 Value: 108
Virtual address: 40185 Physical address: 1273 Value: 0
Virtual address: 28781 Physical address: 1389 Value: 0
Virtual address: 24462 Physical address: 1678 Value: 23
Virtual address: 48399 Physical address: 1807 Value: 67
Virtual address: 64815 Physical address: 2095 Value: 75
Virtual address: 18295 Physical address: 2423 Value: -35
Virtual address: 12218 Physical address: 2746 Value: 11
Virtual address: 22760 Physical address: 3048 Value: 0
Virtual address: 57982 Physical address: 3198 Value: 56
Virtual address: 27966 Physical address: 3390 Value: 27
Virtual address: 54894 Physical address: 3694 Value: 53
Virtual address: 38929 Physical address: 3857 Value: 0
Virtual address: 32865 Physical address: 4193 Value: 0
Virtual address: 64243 Physical address: 4595 Value: -68
Virtual address: 2315 Physical address: 4619 Value: 66
Virtual address: 64454 Physical address: 5062 Value: 62
Virtual address: 55041 Physical address: 5121 Value: 0
Virtual address: 18633 Physical address: 5577 Value: 0
Virtual address: 14557 Physical address: 5853 Value: 0
Virtual address: 61006 Physical address: 5966 Value: 59
Virtual address: 62615 Physical address: 407 Value: 37
Virtual address: 7591 Physical address: 6311 Value: 105
Virtual address: 64747 Physical address: 6635 Value: 58
Virtual address: 6727 Physical address: 6727 Value: -111
Virtual address: 32315 Physical address: 6971 Value: -114
Virtual address: 60645 Physical address: 7397 Value: 0
Virtual address: 6308 Physical address: 7588 Value: 0
Virtual address: 45688 Physical address: 7800 Value: 0
Virtual address: 969 Physical address: 8137 Value: 0
Virtual address: 40891 Physical address: 8379 Value: -18
Virtual address: 49294 Physical address: 8590 Value: 48
Virtual address: 41118 Physical address: 8862 Value: 40
Virtual address: 21395 Physical address: 9107 Value: -28
```

Figure 5: CorrectAddresses.txt

Analysis

What worked for the most part was making sure that there were functions/methods specifically assigned to get the page numbers and offset as well as making sure that the functions/methods were addressing the TLB (seen in figure 6 bellow). The FIFO method seemed to work fairly well on the first go around. Nothing failed during this last test run of the code. Figuring out the overall design first before going too heavy into the coding also helped with making sure that the code delivered what was being asked within the assignment. The only set backs that occurred in this assignment was making sure that things were planned out early enough to have time to double check the assignment. Some due dates that we had set as a team needed to be pushed back a bit, however for the most part the assignment, as well as the code, ran fairly smooth.

Summary

In conclusion, we learned how to track page faults using a backing store. We also learned how to translate addresses from physical and virtual memory. For the most part the assignment went the way that we desired it to go. In the future, we might spend a bit more time in the design phase of the assignment, but for the most part everything went fairly smooth. Overall what was learned in this assignment gave us a stronger grasp of how the memory and pages work in relation to the OS.

Appendix

Figure 0: Types of addresses

Figure 1: Address Translation Process

Figure 2: Compiled HW5.c file

Figure 3: Output of Program

Figure 4: Statistic results

Figure 5: CorrectAddresses.txt