P3 – Design Document

Cont Frame Pools

I did have to make a few changes in the .C file because I was not doing the correct things in the constructor (since the first get\_frames(1) would give me frame 515 instead of 512). I had to correct the part where I would update parts of the bitmap to HEAD/ALLOCATED and changed it to so that it only occurs when info\_frame\_no is 0 (before it was every time, which was a problem later for get\_frames()).

Page Table

init\_paging

This was just initializing the variables sent into init\_paging as parameters

constructor

I initialized the page directory and got a new frame for it from the kernel\_mem\_pool using get\_frames(). I then set up the page table and put the correct attributes for each spot in the page table. I added that page table to the first index of the page directory. Lastly, I set all the attributes for the spots in the page directory.

load()

This function was also simple. I just had to write the page directory to the CR3 register.

enable\_paging()

This followed what was in the Implementing Basic Paging Tutorial given in the P3 pdf.

handle\_fault()

I used the parameter \_r to determine what should happen. If it was one, that means a protection fault, and that is not handled in this function, so I just returned. After that, I used read\_cr3() to get the page directory and read\_cr2() to get the logical address. I did some bit shifting and other conversions to get the actual indices for the page directory and page table. Then, I determined which situation the error is using an if statement. If that statement was true, then that meant that it was the page directory error and that a new page table needed to be made and set up. The last part is just putting a frame in the page table at the index given since that happens for both of the types of errors that handle\_fault resolves.