Design Document - Operating Systems - MP3 - Anshul Sharma

==

For this machine problem we are asked to implement the page table manager to handle the page frames of the x86 machine. We implement a multi-level page table(number of levels = 2), the first level has the page directory and the second level has the page table. In case of a page fault we need to find out in which of the above two tables the fault occurred, for this we first check the page directory entry corresponding to the cr2 register value(i.e., the address stored in cr2 register is checked). If we observe a page fault in the PD then we allocate the frame needed to first store the page table and also the pages required from memory.

We use the implementation of ContFramePool.c and ContFramePool.h from MP2. For this problem I have only touched page_table.c file. Here are the list of functions implemented:

1. Constructor: first we handle the directly mapped memory by creating a page table.
2. Init_paging: this api is used to initialize the PageTable. We assign values to the private members of this class.
3. Load: this api is used to set the current_page_table to the current object of the PageTable. After this we write the address of the page directory on the cr3 register.
4. Enable_Paging: we set the enabled_paging variable as true in this api. And also set cr0 register to the specified value.