Course: Operating Systems CSCE 611

File: MP4 Design document

Name: Kaushal Prudhvi Raj

UIN : 630009512

Procedure:

1) The handout is clearly studied and the recursive directory lookup is implemented
2) Separate functions like **PDE_address() and PTE_address()** for recursive page table lookup as assigned in the handout were defined in .H file and were implemented in .C file.
3) Everything is now freely mapped. The implementation from MP3 is changed to reflect those changes of freely mapping rather than direct mapping. Hence Frames are now allocated from process pool.
4) Page directories, tables are setup correctly and are allocated frames using process mem pools
5) Paging is enabled
6) The CR0, CR2, CR3 registers are properly utilised.
7) The Virtual Pools are tracked using array of pointers. A pool count of 500 is set for simplifying the allocation process and not worrying about internal fragmentation as given in the handout.
8) Every time an address generates an exception it is first checked to see if the address is legitimate using islegitimate function.. then only the exception is allowed to go forward else it is caught in an assert statement and will trap to the OS.
9) The Kernel. C has got few lines commented out as directed in the file to test code and heap pools.
10) All the variables, functionalities of several loops have been clearly commented and are detailed.
11) The testing is done in new development environment using the following commands
12) make, ./copykernel.sh, bochs -f bochsrc.bxrc
13) The following screenshots shows the result of the output..
14) The github link is https://github.tamu.edu/kausht14/OS_MP4
15) The results for MP4 are shown below.

Result:

```
(0) [0x0000fffffff0] f000:fff0 (unk. ctxt): jmpf 0xf000:e05b          ; ea5be
000f0
<bochs:1> c
Installed exception handler at ISR <0>
Installed interrupt handler at IRQ <0>
Installed interrupt handler at IRQ <1>
after installing keyboard handler
Frame Pool initialized
Frame Pool initialized
Installed exception handler at ISR <14>
Initialized Paging System
Constructed Page Table object
Loaded page table
Enabled paging
Hello World!
registered VM pool
Constructed VMPool object.
registered VM pool
Constructed VMPool object.
VM Pools successfully created!
I am starting with an extensive test
of the VM Pool memory allocator.
```

```
Please be patient...
Testing the memory allocation on code_pool...
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
Checked whether address is part of an allocated region.
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
Checked whether address is part of an allocated region.
handled page fault
handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
Checked whether address is part of an allocated region.
handled page fault
Allocated region of memory.
EXCEPTION DISPATCHER: exc_no = <14>
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
Checked whether address is part of an allocated region.
handled page fault
handled page fault
Loaded page table
Released region of memory.
EXCEPTION DISPATCHER: exc_no = <14>
```

```
freed page
freed page
freed page
freed page
Loaded page table
Released region of memory.
Testing the memory allocation on heap_pool...
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
Checked whether address is part of an allocated region.
handled page fault
Allocated region of memory.
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
Checked whether address is part of an allocated region.
```

```
Released region of memory.
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
Checked whether address is part of an allocated region.
handled page fault
Allocated region of memory.
EXCEPTION DISPATCHER: exc_no = <14>
Checked whether address is part of an allocated region.
handled page fault
freed page
freed page
freed page
freed page
Loaded page table
Released region of memory.
Test Passed! Congratulations!
YOU CAN SAFELY TURN OFF THE MACHINE NOW.
One second has passed
One second has passed
```