CSCE611: MP4

Design Document

Author: Pranav Anand Taukari (433003781)

Problem Statement:

To implement virtual memory manager and a simple virtual memory allocator

Part I: Support for Large Address Spaces:

To extend the usage for larger address spaces, the Process Memory Pool is used to allocate memory to the Page Directory and Page Table. CPU generates random logical addresses; Recursive Page Table is used to map these addresses to the allocated frames.

In Recursive Page Table last entry of Page Directory is made to point to itself. On implementing Recursive Page Table, the Page Directory stored in Virtual Memory can be manipulated as,

| 1023: 10 |1023 : 10 | offset : 12 |

And page table as,

|1023 : 10 | X : 10 | Y : 10 | 0 : 2 |

Functions Used:

PageTable::PageTable() :

This constructor is used for the direct mapping which involves mapping the addresses and setting the bits to be present and the Page directory entries to be not present happens in the constructor. The recursive setting of the Page Table is also implemented in the constructor, where the last entry (1023) is made to point to itself.

PageTable::load() :

Once the page_table is setup, here the current object's page directory index is extracted and stored in the PTBR i.e, CR3 register

PageTable::enable_paging() :

Paging is enabled by setting a specific bit of CR0

PageTable::handle_fault(REGS * _r):

Find the faulty address and map the free frame from process frame pool with Page Directory and Page Table using recursive page table.

PageTable::PDE_address(unsigned long addr):

Used to compute the address of PDE

PageTable::PTE_address(unsigned long addr):

Used to compute the address of PTE

Part II: Preparing Virtual Memory Pools and Legitimacy of Logical address:

FUNCTIONS USED:

PageTable::register_pool(VMPool * _vm_pool)

The Virtual Memory Pool contains a list of regions for virtual memory allocation. The Page table knows about the Pools registered with the Page Table, through the PageTable::register_pool() function. Under this function a Linked list of the VM pools is maintained.

VMPool::is_legitimate(unsigned long _address)

When a Page Fault occurs, it is checked if the fault address is legitimate, by checking if the fault address is within the base address and base address + limit range and if so handle page fault.

PART:III VM_Pools : Allocation And De-Allocation

Here, arrays are used for allocation and de-allocation of these vm regions. Each array element is of a class object type, which contains details regarding the base address and the length of the VM_Pool region allotted. The class is defined as follows:

Data Structure for VM Pool region:

```
class vm_region{
  public:
    unsigned long base_address;
    unsigned long len;
  };
```

Allocation of VM pools:

A simple virtual memory Manager is implemented which allocates and de-allocates memory in multiples of pages in the Virtual Memory. In the allocate function, a regions of virtual memory pool is allotted. The size of the region is number of pages required times the size of each page Once allotted it returns the start address of the regions from which memory was allotted.

De-Allocation Of VM_POOL Region:

FUNTIONS USED:

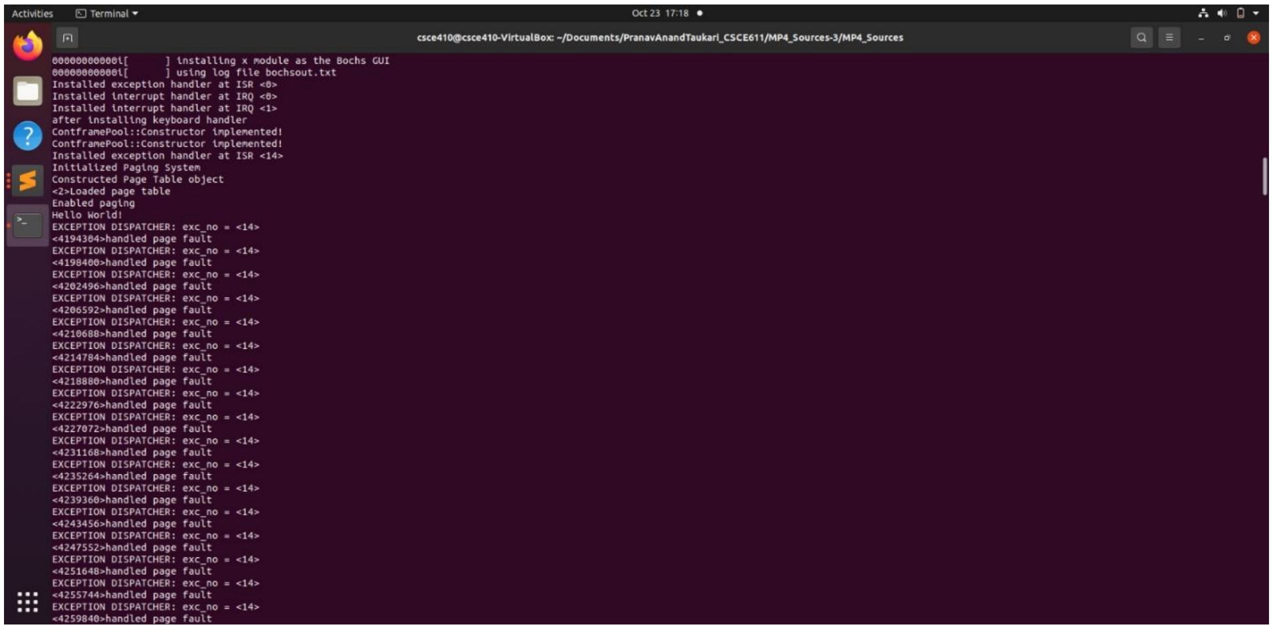
VMPool::release(unsigned long _start_address):

When an address is requested to be deallocated, this function calculates the number of pages to be released and then calls release function for each page

PageTable::free_page(unsigned long _page_no):

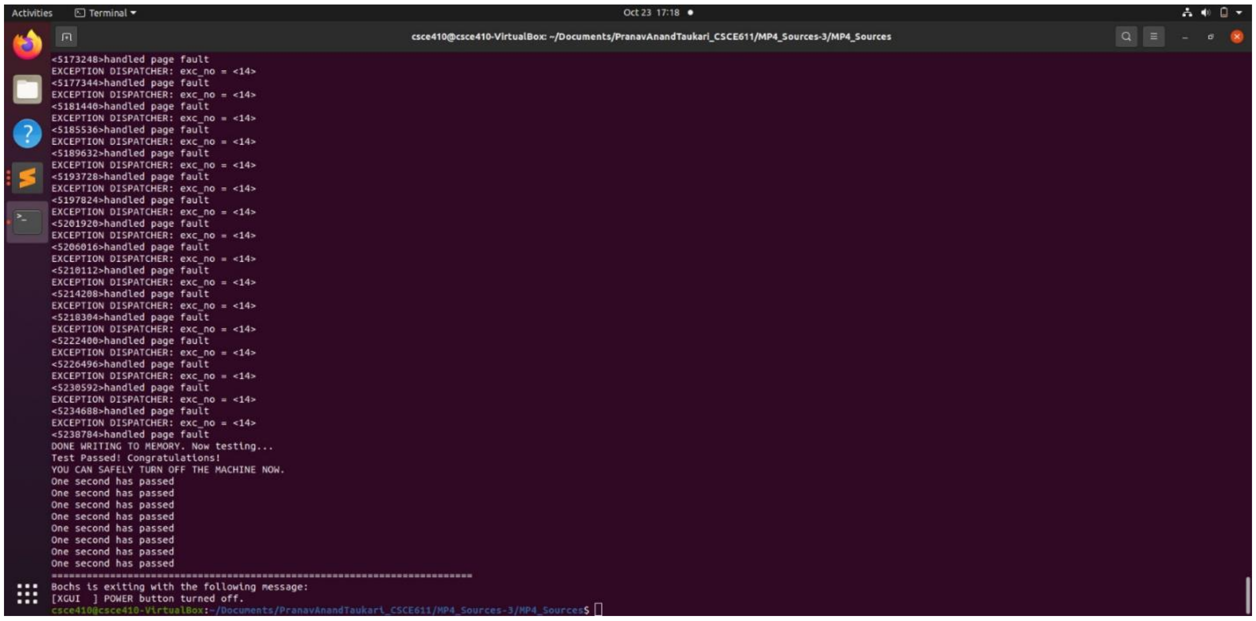
This function, finds the frame number for each page, based on the address, and calls the corresponding release_frame function of the Process pool. The page is marked invalid and the TLB is reflushed using the load () function.

RESULTS:



csce410@csce410-VirtualBox: ~/Documents/PranavAnandTaukari_CSCE611/MP4_Sources-3/MP4_Sources

```
00000000000i[      ] installing x module as the Bochs GUI
00000000000i[      ] using log file bochsout.txt
Installed exception handler at ISR <0>
Installed interrupt handler at IRQ <0>
Installed interrupt handler at IRQ <1>
after installing keyboard handler
ContframePool::Constructor implemented!
ContframePool::Constructor implemented!
Installed exception handler at ISR <14>
Initialized Paging System
Constructed Page Table object
<2>Loaded page table
Enabled paging
Hello World!
EXCEPTION DISPATCHER: exc_no = <14>
<4194304>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4198400>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4202496>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4206592>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4210688>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4214784>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4218880>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4222976>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4227072>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4231168>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4235264>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4239360>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4243456>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4247552>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4251648>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4255744>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<4259840>handled page fault
```



csce410@csce410-VirtualBox: ~/Documents/PranavAnandTaukari_CSCE611/MP4_Sources-3/MP4_Sources

```
<5173248>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5177344>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5181440>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5185536>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5189632>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5193728>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5197824>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5201920>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5206016>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5210112>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5214208>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5218304>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5222400>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5226496>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5230592>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5234688>handled page fault
EXCEPTION DISPATCHER: exc_no = <14>
<5238784>handled page fault
DONE WRITING TO MEMORY. Now testing...
Test Passed! Congratulations!
YOU CAN SAFELY TURN OFF THE MACHINE NOW.
One second has passed
One second has passed
One second has passed
One second has passed
One second has passed
One second has passed
One second has passed
One second has passed
========================================================================
Bochs is exiting with the following message:
[XGUI  ] POWER button turned off.
csce410@csce410-VirtualBox:~/Documents/PranavAnandTaukari_CSCE611/MP4_Sources-3/MP4_Sources$ 
```