Project #4 Design Document

Viraj Sonawane (vsonawa) **Rhushikesh Prabhune (rprabhu)**

Physical memory layout

Draw the physical memory layout, indicating the areas of memory devoted to code (text area), data, page directories and page tables, BSS, etc. For each memory area, indicate its start address and its size. Note that different teams may use different layouts.

FFS	End - 0x63FFFFF Start - 0x2400000
Page Table and Directories area	End – 0x23FFFFF Start - 0x2000000
FREE LIST (HEAP and STACK)	End - 0x1FFFFFF
...	...
...	...
FREE LIST (HEAP and STACK)	Start - 0x142B68
BSS	Start - 0x123020
DATA	Start - 0x121540
TEXT	Start - 0X100000

Initialization of page directories and page tables

Indicate how page directories and page tables should be initialized. You don't need to show the exact content of page directories and page tables (i.e., the content of each entry). But, you need to indicate how many page directory/table entries you need for each memory area that should be mapped to the processes' virtual space. If different types of processes require different mappings, show them all.

In designing the memory layout and defining the initial content of page directories and page tables, you need to consider the following.

- Before paging is enabled, the OS uses only physical addresses. However, after paging is enabled, all memory accesses are through the virtual address space. You need to map the static segments (TEXT, DATA, etc.) in the virtual address space of all processes.
- User processes cannot map the whole FFS area: they can map only the portion of it that they use (that is, FFS area will be mapped only when vmalloc is invoked)
- **If you are taking the course at the graduate level**, in your implementation user processes cannot map the area devoted to page directories and page tables.

For system processes, we need to map the XINU PAGES as well as the Page Table and Page Directory. For the user processes, we need to just map the XINU PAGES. The FFS area need not be mapped on initialization as those mappings will be created when vmalloc is invoked.

System processes Virtual Memory Space	**#Entries of mapping**	**Physical Memory**
FFS	16*1024 PT entries and 16 PD entries	FFS
Page Table and Page Directory	The MAX_PT_SIZE macro(1024) indicates the number of pages reserved in the physical memory for storing PT's and PD's. A system process has to be allowed to modify any table in that region, hence 1024 PTE modifications are needed.	Page Table and Page Directory
XINU PAGES	8192 XINU PAGES will require 8192 PTE modifications. Further 8192/1024=8 PDE modifications will be required.	XINU PAGES

All the PTE modifications above will create a flat mapping, i.e mappings from the same virtual address to physical addresses.

User processes Virtual Memory Space	**#Entries of mapping**	**Physical Memory**
Virtual Heap	No mappings during initialization	FFS
XINU PAGES	8192 XINU PAGES will require 8192 PTE modifications. Further 8192/1024=8 PDE modifications will be required.	XINU PAGES

System Initialization

Where is paging enabled and how? (see hints)

Paging has to be enabled after rest of system initialization is done. So it can be done in the sysinit function in initialize.c . Initialization will mainly involve setting up the page table for the current process and since it is a system process, the mapping to the page table area has to be created as well. Further, after the tables are setup, paging can be enabled by setting the corresponding bit in the CR0 register.

Process Creation

How do you need to modify process creation to support paging?

Firstly, each process will now require its own backup of the Page Directory Base pointer register (PDBR) which has to be saved in the PCB. Also, on creation, as indicated in the question above, the virtual memory space for a process should be setup including adding page directory and page table entries. The XINU Area has to be mapped for all processes whereas the PT Area depending on if it is a system or user process being created. For user processes, we have added a field to track the addresses allocated from the virtual heap area. This field (array) has to be initialized to 0 during process creation.

Process Termination

How do you need to modify process termination to support paging?

Termination would need releasing of the page tables and directories besides the usual tasks that the current kill function performs. Also, if a process had acquired any FFS frames, they need to be released.

Context Switch

What should be done at context switch to support paging?

Context switch will remain the same now, except that the PDBR register of the process being scheduled out has to be saved in the PCB and that of the one being scheduled has to be loaded into the PDBR register.

Heap allocation, deallocation and access

What should be done at heap allocation, deallocation and when the heap is accessed? Remember that you need to implement lazy allocation in your code.

Heap allocation and deallocation will mainly involve finding ‘nbytes’ of free space in the virtual heap using the per process virtual heap tracker array for the process demanding memory. Since, we are using a lazy allocation policy, the mapping for the allocated space to the FFS area need not be done immediately. Once the first access is made, an exception will occur. At this point the necessary mapping can be created instead of raising a segmentation fault. Deallocation will involve freeing up the free list maintenance pointers and also removing the corresponding page table and directory entries.

Page Fault Handler Design

1. In which circumstances will the hardware raise a page fault?

Hardware will raise a page fault due accessing an address not mapped in the page table (i.e present bit==0).

2. What operations should be performed by the page fault handler depending on the circumstances under which it is invoked?

If the page fault handler is invoked because a valid mapping is not found in the page table, then the process should be killed with a segmentation fault.
If the page fault handler is invoked by first access to a virtual heap location allocated with vmalloc, then the required mapping needs to be created.