Yang Wang 321004893

Implement details of MP3
Yang Wang

Virtual Memory

1 Virtual Memory Pool

The data member of the VMPool class contains:

1. base_address : The starting base logical address of the virtual memory pool
2. size: the size of contiguous virtual memory
3. FramePool* frame_pool: A pointer to the frame pool which current virtual memory pool is mapped to
4. PageTable* page_table: a pointer to the page table which map the virtual memory pages to physical frames
5. used_block: a doubly-linked-list that manage the allocated virtual memory
6. unused_block: manage the free virtual memory pool

2 VMPool::VMPool()

The constructor of VMPool will set the base_address, size, page_table and frame_pool. Also it will allocate the first free virtual memory to the unused_block. Set the base_address as its start address. Leaving unused_block as NULL.

The management information of the new blocks are located in kernel memory. When a new memory region is created, the pointer to the new memory block will get the address by adding the size of the block to the frame boundary. If there are too many blocks, new frame will be allocated from the kernel frame pool.

3 Virtual memory allocator

It will traverse the list find a block that is no less than required size. If the block's size is the same as required size, then the block will be removed from unused_block list and add it to used_block list. If the block's size is larger than required size, just add the required size to its current size and updated the size. Also move from unused_block to used_block.

4 Virtual memory release

Traverse the used_block to find the block that we need to free. Then call free_page() to free the correspond pages to the virtual address. Also the block will be moved from used_block to unused_block.

5 is_legitimate

Check if the fault_address is within any allocated virtual memory

Implement paging using recursive lookup

Adding

1. VMPool** registered_pools; All the registered virtual memory pools
2. numberOfPools: count the number of registered virtual memory pool

Using recursive lookup trick

1. In PateTable constructor, get frame from process_mem_pool instead of kernel_mem_pool to store page directory and page table. Map the first 4MB of physical address to logical address still the same.
2. set the last entry of page directory point to itself
3. The page fault handler will first check if the fault address is within any allocated virtual memory. If not, it will return false. Then will check if the page directory entry is empty. If it is empty, first we will have to allocate a frame to it. And then we check if the page table entry is empty. If it is empty, we first get the address of the page table and put a frame into it.

Free Page:

It will calculate the correspond frame number and then call release_frame to free the corresponding frame in FramePool

Register VMPool

Just add the virtual memory pool to registered virtual pool array and increase the counter.