Assignment 10

CS341: Operating System Lab

Virtual Memory Address Translation Simulator in C

Write a program in C to simulate virtual memory address translation using paging. Each part below builds on the previous, ultimately creating a complete virtual memory translation system.

PART 1: Initialize Virtual Memory Parameters

Problem statement: Write a program in C that calculates the number of pages in the virtual address space and the number of frames in the physical memory. These values are based on the given virtual address space, page size, and physical memory size.

Input:

- Virtual address space in bits (e.g., 32 bits).
- Page size in KB (e.g., 4 KB).
- Physical memory size in MB (e.g., 4 GB).

Output:

- Number of pages in the virtual address space.
- Number of frames in the physical memory.

PART 2: Implement a Page Table Structure

Problem statement: Write a C program to create a page table where each page is assigned to a frame randomly.

Input:

- Number of pages (from Part 1).
- Number of frames (from Part 1).

Output:

- A page table that maps each page number to a frame number.

PART 3: Translate a Virtual Address to Physical Address

Problem Statement:

Write a C program to translate a virtual address to a physical address using the page table.

Input:

- Virtual address in hexadecimal.
- Page table (from Part 2).

Output:

- Page number, offset, frame number, and physical address.

PART 4: Handle Page Faults

Problem Statement:

Modify the program to handle cases where a page number does not exist in the page table, simulating a page fault.

Input:

- Virtual address in hexadecimal.
- Page table, with some entries left unmapped.

Output:

- “Page Fault” if the page is unmapped; otherwise, the translated physical address.