운영체제 Project #1

Virtual Memory Manager

Moon Gi Seok

참고

- Textbook의 10장 programming

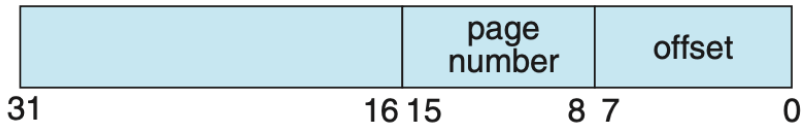
Programming Projects

Designing a Virtual Memory Manager

This project consists of writing a program that translates logical to physical addresses for a virtual address space of size $2^{16} = 65{,}536$ bytes. Your program will read from a file containing logical addresses and, using a TLB and a page table, will translate each logical address to its corresponding physical address and output the value of the byte stored at the translated physical address. Your learning goal is to use simulation to understand the steps involved in translating logical to physical addresses. This will include resolving page faults using demand paging, managing a TLB, and implementing a page-replacement algorithm.

Specific

Your program will read a file containing several 32-bit integer numbers that represent logical addresses. However, you need only be concerned with 16-bit addresses, so you must mask the rightmost 16 bits of each logical address. These 16 bits are divided into (1) an 8-bit page number and (2) an 8-bit page offset. Hence, the addresses are structured as shown as:

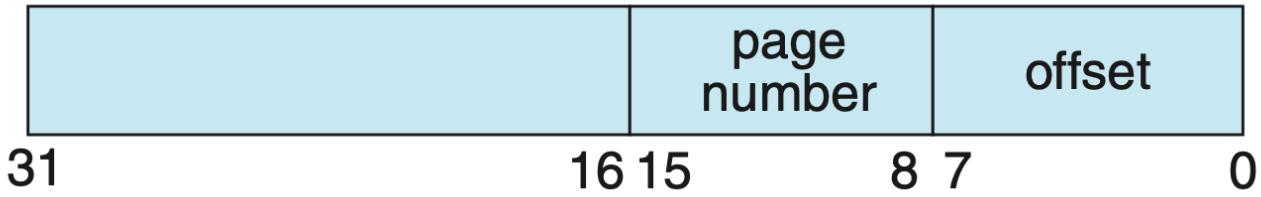


프로젝트 개요

- **목적: 가상 메모리 관리자를 사용하여 논리 주소를 물리 주소로 변환**
- **핵심 구성 요소:**
  - TLB (변환 조회 버퍼)
  - 페이지 테이블
  - 물리 메모리
- **학습 목표:**
  - 주소 변환 이해
  - 수요 페이징 구현
  - TLB 관리
  - 페이지 교체 알고리즘 구현

주소 구조

- **논리 주소는 32비트 정수**
  - 이중 오른쪽 16비트만 다음목적으로 사용
    - ✓ 8비트 페이지 번호
    - ✓ 8비트 오프셋



- **Other Specifications:**
  - 2^8 entries in the page table
  - Page size of 2^8 bytes
  - 16 entries in the TLB
  - Frame size of 2^8 bytes (page size와 동일)
  - 256 frames => Physical memory = 65,53 6bytes (256 frames×256 byte(frame size))

주소 변환

- **1. 논리 주소에서 페이지 번호와 오프셋을 추출**
- **2. TLB에서 페이지 번호를 확인합니다**
- **3. TLB 히트 시: TLB에 저장된 프레임 번호를 사용**
- **4. TLB 미스 시:페이지 테이블을 확인합니다 (demand paging 방식)**
  - 페이지 테이블 미스 (페이지 폴트) 시:
    - BACKING STORE에서 페이지를 로드
      - Backing store는 BACKING_STORE.bin 파일로 표현
      - BACKING STORE에서 256바이트 페이지를 읽은 후, 물리 메모리의 사용 가능한 프레임에 저장함
      - 페이지 테이블과 TLB를 업데이트

TLB and Page Table?

dongguk UNIVERSITY

- **TLB?**
  - 페이지 번호와 프레임 번호의 튜플 리스트.
  - 최대 크기: 16개 항목.
  - 교체 전략: FIFO 또는 LRU.

- **Page table의 엔트리의 값이 (프레임 번호)가 유효하지 않으면 -1저장**

프로젝트 파일 구성

- **virtual_memory_manager.py 실행 파일 (main)**
- **address.txt - 논리주소가 저장된 텍스트**
- **BACKING_STORE.bin**
  - 프로그램들을 위한 페이지들이 저장된 하드 드라이브를 상징
- **global_constants.py**
  - 프로젝트에서 사용되는 전역 상수들이 저장됨 (페이지 크기, 페이지 엔트리 수, 프레임 크기 등)
- **physical_memory.py**
  - 물리 메모리를 관리하는 클래스가 정의됨
- **tlb_fifo.py**
  - Fifo 기반 replacement를 수행하는 TLB 클래스 구현
- **tlb_lru.py**
  - lru 기반 replacement를 수행하는 TLB 클래스 구현

구현 대상

- 파일에서 Todo 표기된 부분 구현 (많지 않음)
- FIFO 기반 TLB 사용시 실행결과 파일 일부 - result_fifo.txt

```
Virtual address: 8940 Physical address: 46572 Value: 0x0
Virtual address: 9929 Physical address: 44745 Value: 0x0
Virtual address: 45563 Physical address: 46075 Value: 0x7e
Virtual address: 12107 Physical address: 2635 Value: 0xd2
Number of Translated Addresses = 1000
Page Faults = 244
Page Fault Rate = 0.244
TLB Hits = 55
TLB Hit Rate = 0.055
```

제출물

- **파이선 파일**
  - virtual_memory_manager.py
  - tlb_lru.py
  - 실행 결과
    - result_fifo.txt
    - result_lru.txt (LRU 기반 TLB를 사용했을 시의 결과)