운영체제 과제

컴퓨터소프트웨어공학부

202195039 임서윤

(1-1) 컴퓨터의 네가지 구성요소를 나열하고 자세하게 설명하시오.

* Processor(CPU) : 명령어를 해석하여 실행하는 장치로, 인간으로 치면 두뇌에 해당한다.
* Main Memory : 작업에 필요한 프로그램과 데이터를 저장하는 장소로, 바이트 단위로 분할되어 있으며 분할 공간마다 주소로 구분한다.
* I/O modules( I/O controllers, I/O processors) : 컴퓨터 시스템에서 입력과 출력을 처리하는 하드웨어 구성 요소입니다.
* System interconnection : 컴퓨터 시스템 내에서 다양한 구성 요소들이 상호 연결되어 데이터를 주고받는 방식을 의미합니다. 이러한 연결 구조는 시스템의 성능과 효율성에 매우 중요한 역할을 하며, CPU, 메모리, I/O 장치, 저장 장치 등 컴퓨터의 주요 하드웨어 구성 요소들이 데이터를 교환할 수 있도록 설계되어 있습니다.

(1-2) 강의 노트(os-lecture-01) pp.30~32에서 설명한 가상 처리기가 다음과 같이 두 개의 입출력 명령어를 추가로 제공한다고 가정하자.

0011 = I/O로부터 AC 적재 0111 = I/O에 AC 저장

이 경우, 12-비트 주소는 특정 외부장치를 식별한다. 강의 노트 pp.32의 그 림과 같은 형식으로 다음 프로그램의 수행 과정을 보이시오.

① 장치 5로부터 AC에 적재

② 메모리 주소 960의 내용을 더함

③ AC에 적재된 값을 장치 6에 저장 장치 5로부터 다음에 검색될 값을 3으로 가정하고 주소 960에는 2가 저장되어 있다고 가정한다.

(1-3) CPU(Processor) 장치 구조에 대해 자세히 설명하고, 제어용 레지스터에 대해 자세히 설명하시오.

* 연산장치 : 컴퓨터의 중앙처리장치(CPU) 내에서 중요한 구성 요소로, 산술 및 논리 연산을 수행하는 장치입니다. ALU는 컴퓨터가 데이터를 처리하는 데 필수적인 역할을 하며, 모든 연산을 실제로 수행하는 부분입니다.
* 레지스터 : CPU 내에서 데이터를 임시로 저장하는 고속 메모리입니다. 명령어 실행 중 필요한 데이터를 빠르게 저장하고 접근하는 공간입니다. 레지스터는 일반적으로 매우 작지만 속도가 빠릅니다.
* 제어장치 : CPU 내에서 명령을 해석하고, 그에 따라 다른 구성 요소와 하드웨어가 어떻게 동작할지 제어합니다. 프로그램 카운터(Program Counter, PC)를 사용해 실행할 명령을 결정하고, 그 명령을 해석하여 데이터 이동 및 연산을 제어합니다.

(1-4) 인터럽트를 정의하고 인터럽트 처리 과정에 대해 자세히 설명하시오.

* CPU의 작업과 저장장치의 데이터 이동을 독립적으로 운영함으로써 시스템의 효율을 높힌다. 즉 데이터의 입출력이 이루어지는 동안 CPU가 다른 작업을 할 수 있다.
* 인터럽트 처리 과정

1. CPU가 입출력 관리자에게 입출력 명령을 보낸다.

2. 입출력 관리자는 명령받은 데이터를 메모리에 가져다 놓거나 메모리에 있는 데이터를 저장장치로 옮긴다.

3. 데이터 전송이 완료되면 입출력 관리자는 CPU에 완료 신호를 보낸다.

(1-5) 프로그램 구동 입출력(I/O) 방식과 인터럽트 구동 입출력 방식이 어떻게 작동 하는지 비교하여 설명하시오.

* 프로그램 구동 입출력(Programmed I/O)방식은 CPU가 I/O 장치와 직접 데이터를 주고받는 방법입니다. CPU는 장치의 상태를 계속 감시(polling)하고, 장치가 준비될 때 데이터를 송수신합니다.
* 인터럽트 구동 입출력(Interrupt-driven I/O)방식에서는 I/O 장치가 준비되었을 때 인터럽트신호를 보내 CPU에게 알립니다. 이 방식에서 CPU는 I/O 장치의 상태를 지속적으로 체크할 필요가 없으며, 장치가 준비될 때만CPU가 I/O 처리를 수행합니다.

장치가 데이터 전송을 완료하거나 데이터를 처리할 준비가 되면, CPU에 인터럽트 신호를 보냅니다. 이 신호를 받으면 CPU는 현재 작업을 멈추고 I/O 처리를 위한 인터럽트 서비스 루틴(Interrupt Service Routine, ISR)을 실행합니다.

(1-6) 메모리 계층 구조에 대해 자세히 설명하시오. 유닉스 계열 운영체제는 보조 기억장치(HDD 등) 일부 영역을 swap-area로 할당하는데, 이 swap-area에 대해 자세히 설명하시오.

* 메모리 계층 구조는 컴퓨터 시스템에서 속도와 용량, 가격간의 균형을 맞추기 위해 다층 구조로 설계된 메모리 구조입니다. 메모리는 속도와 가격이 비례하는 특성을 가지고 있기 때문에, 빠르고 용량이 작은 메모리와 느리고 용량이 큰 메모리를 계층적으로 배치하여 시스템의 성능을 최적화합니다.

(1-7) 어떤 컴퓨터가 캐시, 주기억장치 그리고 가상메모리를 위한 디스크를 포함하 고 있다. 만일 참조된 워드가 캐시에 있으면 그것을 접근하는데 60ns가 걸린 다. 주기억장치를 접근하는 시간(캐시 접근 시간 포함)은 300ns가 걸린다. 파 일은 읽기 또는 쓰기 모드로 열릴 수 있다. 쓰기 연산은 주기억장치와 캐시 (write-through cache)를 모두 접근한다. 읽기 연산은 접근하고자 하는 워드 가 캐시에 있을 때와 없을 때에 따라 캐시만 접근할 수도 있고, 캐시와 주기 억장치 모두 접근할 수도 있다. 읽기 연산은 전체 연산의 80%를 차지한다고 알려져 있다. 만약 읽기 연산을 위한 캐시 적중률이 0.9일 때 이 시스템의 평 균 접근 시간은 얼마인가?

• **캐시 히트(Case 1)**: 참조된 데이터가 캐시에 있을 때, 캐시에서만 데이터를 읽으면 됩니다.

• 캐시 접근 시간 = 60ns

• 확률 = 0.9 (캐시 적중률)

• **캐시 미스(Case 2)**: 참조된 데이터가 캐시에 없을 때, 캐시와 함께 주기억장치에 접근해야 합니다.

• 캐시 접근 시간 + 주기억장치 접근 시간 = 300ns

• 확률 = 0.1 (캐시 미스율)

읽기 연산의 평균 접근 시간=(0.9×60ns)+(0.1×300ns)=54ns+30ns=84ns

쓰기 연산의 평균 접근 시간 = 300ns

전체 평균 접근 시간=(0.8×읽기 연산의 평균 접근 시간)+(0.2×쓰기 연산의 평균 접근 시간)

=(0.8×84ns)+(0.2×300ns)= (0.8 \times 84ns) + (0.2 \times 300ns)

=(0.8×84ns)+(0.2×300ns)

=67.2ns+60ns=127.2ns

(2-1) 운영체제의 프로그램 실행 방식에서 멀티프로그래밍 일괄처리 시스템과 시분 할 처리 시스템을 비교하여 설명하시오.

• **멀티프로그래밍 일괄처리 시스템**은 사용자 상호작용 없이 여러 작업을 동시에 처리하며, CPU 자원을 효율적으로 사용하는 방식입니다. 주로 대규모 일괄 작업에서 사용됩니다.

• **시분할 처리 시스템**은 다중 사용자 환경에서 각 사용자에게 CPU 시간을 짧게 할당하여 실시간 상호작용을 가능하게 하며, 사용자들에게 빠른 응답을 제공합니다.

(2-2) 일괄처리 운영체제 시스템에 세 개의 작업 J1, J2, J3가 실행을 위해 들어왔 다. 각 작업은 하나의 입출력 활동과 한 번의 CPU 처리 그리고 또 다른 하 나의 입출력 활동으로 구성되어 있다. 작업 J1은 총 20ms가 걸리고, 그 중 2ms는 CPU 시간이다. 작업 J2는 총 30ms가 걸리고, 그 중 6ms는 CPU 시 간이다. 작업 J3은 총 15ms가 걸리고, 그 중 3ms는 CPU시간이다. 단일 프 로그래밍 실행환경과 멀티프로그래밍 실행환경 각각에서 CPU 이용률은 어떻 게 되는지 계산하시오.

단일프로그래밍 실행환경

• **J1**실행: CPU는 2ms 동안 일을 하고, 나머지 18ms 동안은 입출력을 기다리며 유휴 상태입니다.

• **J2**실행: CPU는 6ms 동안 일을 하고, 나머지 24ms 동안은 입출력을 기다리며 유휴 상태입니다.

• **J3**실행: CPU는 3ms 동안 일을 하고, 나머지 12ms 동안은 입출력을 기다리며 유휴 상태입니다.

• 총 실행 시간 = J1(20ms) + J2(30ms) + J3(15ms) = **65ms**

• CPU가 실제로 작업한 시간 = J1(2ms) + J2(6ms) + J3(3ms) = **11ms**

CPU 이용률=전체 실행 시간CPU가 실제로 일한 시간​=11ms/65ms​≈0.1692=16.92%

멀티프로그래밍 실행환경

CPU가 실제로 작업한 시간은 여전히 **J1(2ms) + J2(6ms) + J3(3ms) = 11ms**입니다.

CPU 이용률=전체 실행 시간CPU가 실제로 일한 시간​=11ms/30ms​≈0.3667=36.67%

(2-3) C 표준 입출력 함수와 입출력 시스템 호출 함수를 비교하여 설명하시오.

• **C 표준 입출력 함수**는 고수준의 입출력 인터페이스로, 사용자에게 친숙하고 코드의 이식성이 높지만, 시스템 레벨에서의 세밀한 제어는 어렵습니다. 버퍼링을 통해 성능을 최적화하는데 적합합니다.

• **입출력 시스템 호출 함수**는 저수준의 인터페이스로, 더 정밀한 제어와 실시간 처리가 가능하지만, 사용이 복잡하며 운영체제에 종속적입니다.

(2-4) 초기 유닉스 운영체제에서 시스템 호출이 실행되는 과정에 대해 간단히 설명 하시오.

• **사용자 프로그램이 시스템 호출 요청**(read(), write()등).

• **트랩 명령어로 사용자 모드에서 커널 모드로 전환**.

• **커널 모드에서 시스템 호출 처리**(파일 열기, 입출력 처리 등).

• **처리가 완료되면 커널 모드에서 사용자 모드로 복귀**.

• **사용자 프로그램은 시스템 호출 결과를 받아 계속 실행**.