

1과목 데이터베이스

001

- 자료(Data) : 현실 세계로부터 단순한 관찰이나 측정을 통해 수집된, 가공(처리)되지 않은 값이나 사실
- 정보(Information) : 자료를 처리해서 의사결정에 필요한 형태로 가공한 결과물로서 사용자가 목적하는 값

002

데이터베이스(Database)의 정의

- 통합된 데이터(Integrated Data)
- 저장된 데이터(Stored Data)
- 운영 데이터(Operational Data)
- 공유 데이터(Shared Data)

003

데이터베이스의 특성

- 실시간 접근성(Real Time Accessibility)
- 내용에 의한 참조(Content Reference)
- 동시 공유(Concurrent Sharing)
- 계속적 변화(Continuous Evolution)

004

DBMS의 필수 기능

- 정의 기능(Definition Facility)
- 조작 기능(Manipulation Facility)
- 제어 기능(Control Facility)

005

- 외부 스키마 : 사용자 접근 측면 기술(전체 중 일부 기술)
- 개념 스키마 : 논리적인 측면 기술(DB전체 기술)
- 내부 스키마 : 물리적 저장 기술

006

- 단말 사용자 : 질의어나 응용 프로그램을 사용하여 데이터베이스 접근
- 응용 프로그래머 : 응용 프로그램을 작성하여 데이터에 접근, 사용자 인터페이스 제공
- DBA : 데이터베이스의 생성과 모든 관리 운영에 대한 책임과 권한을 가짐

007

데이터 모델의 구성 요소

- 데이터 구조(Structure)
- 연산(Operations)
- 제약 조건(Constraints)

008

E-R 다이어그램 구성 요소

기호	기호 이름	의미
	사각형	개체
	타원	속성
	밑줄 타원	기본키 속성
	마름모	관계
	관계	1:1, 1:n, n:m 등의 관계 유형
	선, 링크	개체와 속성의 연결, 개체 간의 연결

개념적 설계 모델 : 개체-관계(E-R)모델

009

논리적 설계 모델

- 관계데이터 모델 : Table로 표현
- 계층 데이터 모델 : Tree로 표현
- 망 데이터 모델 : Graph로 표현

010

데이터베이스의 설계 순서

요구조건 분석 → 개념적 설계 → 논리적 설계 → 물리적 설계 → 데이터베이스 구현

011

관계 데이터 모델의 용어

릴레이션(Relation)	<ul style="list-style-type: none"> • 2차원 구조의 테이블 • 릴레이션 스키마와 릴레이션 인스턴스로 구성됨
속성(Attribute)	<ul style="list-style-type: none"> • 개체가 가지는 성질을 나타내며, 테이블의 각 열(Column)을 의미함 • 릴레이션 내에서 유일한 이름을 가지며, 속성들 간에는 순서가 없음 • 속성값은 논리적으로 더 이상 분해할 수 없는 원자값이어야 함
도메인(Domain)	하나의 속성이 취할 수 있는 범위
튜플(Tuple)	<ul style="list-style-type: none"> • 테이블의 각 행(Row)을 의미함 • 중복된 튜플이 존재하지 않으며, 튜플 간의 순서는 없음
차수(Degree)	한 릴레이션에서 속성의 수
카디널리티(Cardinality)	한 릴레이션에서 튜플의 수

012

무결성(Integrity)

- 개체 무결성 : 한 릴레이션의 기본키를 구성하는 어떠한 속성값도 널 값이나 중복값을 가질 수 없음
- 참조 무결성 : 릴레이션은 참조할 수 없는 외래키 값을 가질 수 없음
- 영역 무결성 : 속성값들은 정해진 범위 내에 있어야 함

013

널 값(Null Value)

- 아직 알려지지 않은 값 또는 모르는 값
- '해당 없음'의 이유로 정보 부재(Missing)를 나타내기 위해 사용하는 특수 데이터

014

관계 대수

구 분		기호	설 명
일반 집합	합집합(Union)	\cup	두 릴레이션의 튜플의 합집합을 구함
	교집합(Intersection)	\cap	두 릴레이션에서 중복되는 튜플을 선택함
	차집합(Difference)	$-$	두 릴레이션에서 중복되는 튜플을 제거함
연산자	교차곱(Cartesian Product)	\times	두 릴레이션에 정의된 튜플을 곱으로 계산함
순수 (특별)	선택(Select)	σ	선택 조건에 맞는 튜플의 수평적 부분 집합
	프로젝트(Project)	Π	선택 조건에 맞는 튜플의 수직적 부분 집합
관계	조인(Join)	\bowtie	두 개 이상의 릴레이션에서 튜플이나 애트리뷰트를 조합한 새로운 릴레이션을 생성함
연산자	디비전(Division)	\div	두 릴레이션 A, B에서 B 릴레이션의 모든 조건을 만족하는 튜플들을 릴레이션 A에서 분리해 내어 프로젝션 함

015

관계 해석

- 질의어 형태로 원하는 릴레이션을 정의하는 방법을 제공하며, 수학의 Predicate Calculus에 기반을 둠
- 원하는 정보가 무엇이라는 것만 정의하는 비절차적 언어의 특성을 지님

016

SQL

- 데이터 정의어(DDL) : CREATE, ALTER, DROP
- 데이터 조작어(DML) : SELECT-FROM-WHERE-, INSERT-INTO-, DELETE-FROM-, UPDATE-SET
- 데이터 제어어(DCL) : GRANT, REVOKE

017

뷰(View)의 특징

- 사용자에게 접근이 허용된 자료만을 제한적으로 보여주기 위해 하나 이상의 기본 테이블로부터 유도된 가상 테이블
- 뷰에 대한 검색은 기본 테이블과 동일하며, 뷰를 이용한 또 다른 뷰의 생성이 가능함
- 뷰는 한 번 정의되면 변경할 수 없으므로, 정의를 변경하려면 삭제하고 다시 만들어야 함

018

시스템 카탈로그

- 시스템이 필요로 하는 모든 객체에 대한 정보를 가지고 있는 시스템 데이터베이스로 사용자 데이터베이스와 구별됨
- 데이터 사전(Data Dictionary)이라고도 하며, 카탈로그에 저장된 데이터를 메타 데이터(Meta Data)라고 함
- 카탈로그 자체도 시스템 테이블로 구성되어 있어 일반 사용자도 SQL을 이용하여 검색해 볼 수 있음
- 카탈로그의 정보를 INSERT, DELETE, UPDATE 문으로 카탈로그를 직접 갱신하는 것은 불가능함
- 사용자가 SQL문을 실행시켜 기본 테이블, 뷰, 인덱스 등에 변화를 주면 DBMS가 자동으로 카탈로그를 갱신함

019

- 삽입 이상 : 데이터 삽입 시 불필요한 데이터까지 함께 삽입되는 현상

- 삭제 이상 : 한 튜플을 삭제함으로써 일어나는 연쇄 삭제 현상
- 갱신 이상 : 튜플의 속성값을 갱신할 때 일부 튜플의 정보만 갱신되어 정보에 모순이 생기는 현상

020
정규화 과정



021
트랜잭션의 특성(ACID)

- 원자성(Atomicity) : 트랜잭션은 완벽하게 수행되거나 전혀 수행되지 않아야 함
- 일관성(Consistency) : 트랜잭션 수행 전과 트랜잭션 수행 완료 후가 같아야 함
- 격리성(Isolation, 독립성) : 트랜잭션이 실행될 때마다 다른 트랜잭션의 간섭을 받지 않아야 함
- 영속성(Durability) : 트랜잭션이 일단 그 실행을 성공적으로 완료하면 그 결과는 영속적이어야 함

022
트랜잭션의 연산 : Commit(완료), Rollback(복귀)

023

로킹 단위	로크(Lock)의 수	관리(병행제어 기법)	병행성 수준
작을수록	많아짐	복잡해짐	높아짐
클수록	적어짐	쉬워짐	낮아짐

024
분산 데이터베이스의 장·단점

장점	<ul style="list-style-type: none"> · 분산 데이터의 효과적 처리와 데이터의 공유성 향상 · 신뢰성, 융통성, 효율성, 가용성, 확장성 증가 · 지역 특성에 맞는 H/W, S/W를 구축
단점	<ul style="list-style-type: none"> · 구현이 복잡하며, 소프트웨어 개발 비용과 처리 비용의 증가 · 잠재적 오류가 증가함 · 보안을 위한 추가 기술이 필요

025

- 선형구조 : 리스트(List), 스택(Stack), 큐(Queue), 데크(Deque)
- 비선형구조 : 트리(Tree), 그래프(Graph)

026

이진 트리의 운행법

- 전위(Preorder) 운행법 : Visit → Left → Right
- 중위(Inorder) 운행법 : Left → Visit → Right
- 후위(Postorder) 운행법 : Left → Right → Visit

027

산술식 표기법

- 중위식(Infix) 표기법 : 피연산자 - 연산자 - 피연산자
- 전위식(Prefix) 표기법 : 연산자 - 피연산자 - 피연산자
- 후위식(Postfix) 표기법 : 피연산자 - 피연산자 - 연산자

028

해싱(Hashing)

- 해시 테이블(Hash Table)이라는 기억공간을 할당하고, 해시 함수(Hash Function)를 이용하여 레코드 키에 대한 해시 테이블 내의 홈 주소(Home Address)를 계산한 후, 주어진 레코드를 해당 기억장소에 저장하거나 검색 작업을 수행하는 방식
- 키 값으로부터 레코드가 저장되어 있는 주소를 직접 계산하여, 산출된 주소로 바로 접근하는 방법이며, 키-주소 변환 방법이라고도 함
- 검색 방법 중 속도가 가장 빠르지만, 기억공간이 많이 요구됨
- 종류 : 제산 방법(Division Method), 중간 제곱 방법(Mid-Square Method), 중첩 방법(Folding Method), 기수 변환 방법(Radix Conversion Method), 무작위 방법(Random Method), 계수 분석 방법(Digit Analysis Method)

029

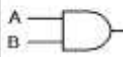
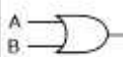
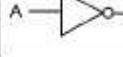





파일 조직 기법

- 순차 파일(Sequential File) : 입력되는 데이터의 논리적인 순서에 따라 물리적으로 연속된 위치에 기록하는 방식으로, 처리 속도가 빠르고, 저장 매체의 공간 효율이 좋지만 검색 시 불편함
- 색인 순차 파일(Indexed Sequential File) : 키 값에 따라 순차적으로 정렬된 데이터를 저장하는 데이터 영역과 이 지역에 대한 포인터를 가진 색인 영역으로 구성된 파일로서, 기본 영역, 색인 영역(트랙/실린더/마스터), 오버플로우 영역으로 구성됨
- VSAM 파일(Virtual Storage Access Method File) : 동적 인덱스 방법을 이용한 색인 순차 파일로서, 기본 데이터 영역과 오버플로우 영역을 구분하지 않음
- 직접 파일(Direct File) : 특정 레코드에 접근하기 위해서 디스크의 물리적 주소로 변환할 수 있는 함수인 해싱 함수를 사용하며, 시간이 빠르고, 랜덤 처리에 적합하나 메모리 효율이 떨어짐

2과목 전자계산기

030

주요 논리회로

이름	기호	논리식	의미	진리표															
AND		$Y = A \cdot B$ $= AB$	입력 값이 모두 1일 때만 1을 출력	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	0	1	0	0	1	1	1
A	B	Y																	
0	0	0																	
0	1	0																	
1	0	0																	
1	1	1																	
OR		$Y = A + B$	입력 값 중 한 개라도 1이면 1을 출력	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	1
A	B	Y																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	1																	
NOT		$Y = \overline{A}$ $= A'$	입력 값의 반대 값을 출력	<table><tr><th>A</th><th>Y</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	Y	0	1	1	0									
A	Y																		
0	1																		
1	0																		
BUFFER		$Y = A$	입력 값을 그대로 출력	<table><tr><th>A</th><th>Y</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	A	Y	0	0	1	1									
A	Y																		
0	0																		
1	1																		
NAND		$Y = \overline{A \cdot B}$ $= \overline{AB}$	AND의 부정 (NOT + AND)	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	1	0	1	1	1	0	1	1	1	0
A	B	Y																	
0	0	1																	
0	1	1																	
1	0	1																	
1	1	0																	
NOR		$Y = \overline{A + B}$	OR의 부정 (NOT + OR)	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	1	0	1	0	1	0	0	1	1	0
A	B	Y																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	0																	
XOR (Exclusive-OR)		$Y = A \oplus B$ $= \overline{A}B + A\overline{B}$	입력 값이 모두 같으면 0, 한 개라도 다르면 1을 출력	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	0
A	B	Y																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	0																	
XNOR (Exclusive-NOR)		$Y = A \odot B$	XOR의 부정 (NOT + XOR)	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	1	0	1	0	1	0	0	1	1	1
A	B	Y																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	1																	

031

반가산기(Half-Adder)

- 2개의 비트 X, Y를 더한 합 S와 자리올림 C를 구하는 회로
- 1개의 XOR 회로와 1개의 AND 회로로 구성

$$C = X \cdot Y \quad S = X \oplus Y = \overline{X} \cdot Y + X \cdot \overline{Y}$$

전가산기(Full-Adder)

- 3개의 입력과 2개의 출력
- 2개의 반가산기와 1개의 OR회로로 구성

032

플립플롭

- 1bit 기억소자로, 레지스터에 사용됨
- RS, JK, T, D플립플롭이 있음

033

- SR 플립플롭 : S=1, R=1일때 표현불가
- JK 플립플롭 : SR의 단점을 보완, J=1,K=1일 때 상태 반전 표현
- T 플립플롭 : JK플립플롭을 하나로 묶어 표현

034

- Bit : 정보 표현의 최소단위
- Nibble : 4Bit 구성문자
- Field : 정보의 논리적 표현의 최소단위
- n Bit로 표현 가능한 표현 가지수 : 2^N

035

음수의 표현과 표현범위

종 류	표현 범위
부호와 절대치	$-(2^{n-1}-1) \sim 2^{n-1}-1$
부호와 1의 보수	$-(2^{n-1}-1) \sim 2^{n-1}-1$
부호와 2의 보수	$-2^{n-1} \sim 2^{n-1}-1$

036

고정 소수점(Fixed Point) 표현

- 10진 연산 : 10진수 1자리를 2진수 4자리로 표현하는 방식
- 2진 연산 : 정수값을 2진수로 변환하여 표현하는 방식

037

부동 소수점(Floating Point) 표현

- 고정 소수점 표현보다 매우 작은 수와 매우 큰 수의 표현에 적합하며, 표현의 정밀도를 높일 수 있음
- 표현 방법

0	1	2	7	8	n-1
부호 (Sign)	지수부 (Exponent)	가수부 (Fraction)			

038

문자코드

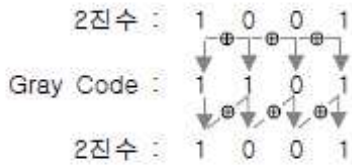
- BCD 코드 : 1개의 문자를 2개의 Zone 비트와 4개의 Digit 비트로 표현함
- ASCII 코드 : 1개의 문자를 3개의 Zone 비트와 4개의 Digit 비트로 표현하며, 데이터 통신 및 마이크로 컴퓨터에서 주로 사용함
- EBCDIC 코드 : 1개의 문자를 4개의 Zone 비트와 4개의 Digit 비트로 표현하며, 대형 컴퓨터에서 주로 사용함

039

숫자코드

- BCD 코드 : 10진수 1자리의 수를 2진수 4Bit로 표현하며, 8421 코드라고도 함
- 3 초과 코드 : BCD 코드에 3(0011)을 더한 것과 같음
- 그레이 코드
 - BCD 코드의 인접하는 비트를 XOR 연산하여 만든 코드

-변환 방법



040

오류 검출/교정 코드

- 패리티 검사 코드 : 코드의 오류를 검사하기 위해서 Data Bit 외에 1Bit의 패리티 체크 비트를 추가하며, 1 Bit의 오류만 검출 가능
- 해밍 코드 : 오류 검출(2Bit) 및 교정(1Bit)이 가능한 코드로, 1, 2, 4, 8, 16, ... 2^n 번째는 오류 검출을 위한 패리티 비트임

041

중앙처리장치의 구성

- 제어장치(Control Unit) : 주기억장치에 기억된 명령을 꺼내서 해독하고, 시스템 전체에 지시 신호를 내는 장치
- 연산장치(ALU : Arithmetic and Logic Unit) : 제어장치의 명령에 따라 실제로 연산을 수행하는 장치
- 레지스터(Register) : 컴퓨터의 중앙처리장치에서 사용되는 고속 메모리로, 처리에 필요한 내용을 일시적으로 기억하는 역할을 수행하며, 메모리 중에서 속도가 가장 빠름

042

레지스터의 종류

프로그램 카운터 (PC : Program Counter)	다음에 실행할 명령어의 주소를 기억하는 레지스터
명령 레지스터 (IR : Instruction Register)	현재 실행중인 명령을 기억하는 레지스터
메모리 주소 레지스터 (MAR : Memory Address Register)	기억장치에서 사용하는 데이터의 주소를 기억하는 레지스터
메모리 버퍼 레지스터 (MBR : Memory Buffer Register)	기억장치에서 사용하는 데이터를 기억하는 레지스터
베이스 레지스터 (Base Register)	명령이 시작되는 최초의 번지를 기억하는 레지스터
프로그램 상태 레지스터 (PSWR : Program Status Word Register)	시스템 내부의 순간순간의 상태를 기록하고 있는 정보인 PSW(Program Status Word)를 저장하고 있는 레지스터
인덱스 레지스터 (Index Register)	주소 변경, 서브루틴 연결, 반복 연산 수행 등의 역할을 하는 레지스터
메이저 스테이트 레지스터 (Major State Register)	CPU의 메이저 상태를 저장하고 있는 레지스터

043

명령어의 구성

연산자부 (Operation Code)	<ul style="list-style-type: none"> · 실행할 명령이 들어 있음 · 연산자 부분이 나타낼 수 있는 것 : 명령어의 형식, 연산자, 자료의 종류 · 연산자부의 비트 수가 n Bit \rightarrow 명령어 개수는 최대 2^n개
자료부 (Operand)	<ul style="list-style-type: none"> · 자료부 = 주소부 = 어드레스 필드(Address Field) · 실제 데이터에 대한 정보를 표시함 · 자료부가 n Bit \rightarrow 메모리 용량은 최대 2^n

044

연산자의 4대 기능

- 함수 연산 기능
- 자료 전달 기능
- 제어 기능
- 입/출력 기능

045

0-주소 명령어

- Operand부가 없이 OP-code부만으로 구성됨(자료의 주소를 지정할 필요가 없음)
- 주소의 사용 없이 스택에 연산자와 피연산자를 넣었다 꺼내어 연산한 후 결과를 다시 스택에 넣으면서 연산하기 때문에 원래의 자료가 남지 않음

1-주소 명령어

- Operand부가 1개로 구성되며, 연산의 결과는 Operand 1에 기록됨
- 하나의 Operand가 누산기 속에 포함되고, 연산 결과가 항상 누산기에 저장됨

2-주소 명령어

- Operand부가 2개로 구성되며, 연산의 결과는 Operand 1에 기록됨
- 여러 개의 범용 레지스터를 가진 컴퓨터에 사용됨
- 연산 이전의 데이터 일부를 기억하지 못함(Operand 1에 있던 원래의 자료가 파괴됨)

3-주소 명령어

- Operand부가 3개로 구성되며, 연산의 결과는 Operand 3에 기록됨
- 연산 후에 입력 자료가 변하지 않고 보존됨
- 프로그램의 길이는 짧아지지만, 명령어 한 개의 길이가 길어짐

046

접근 방식에 의한 주소지정 방식

묵시적 주소지정 방식 (Implied Addressing Mode)	<ul style="list-style-type: none"> · Operand가 명령어에 묵시적으로 정의되어있는 경우 · 스택을 이용하는 0-주소 명령어에 사용됨
즉시 주소지정 방식 (Immediate Addressing Mode)	<ul style="list-style-type: none"> · 명령어의 Operand 부분에 데이터를 기억하는 방식 · 별도의 메모리에 접근하지 않고 CPU에서 곧바로 자료를 이용하므로 실행속도가 가장 빠름 · 메모리 참조횟수 : 0
직접 주소지정 방식 (Direct Addressing Mode)	<ul style="list-style-type: none"> · 명령어의 Operand 부분에 유효 주소 데이터가 있는 방식 · 메모리 참조횟수 : 1
간접 주소지정 방식 (Indirect Addressing Mode)	<ul style="list-style-type: none"> · 명령어의 Operand 부분에 실제 데이터의 위치를 찾을 수 있는 번지를 가지고 있는 방식 · 메모리 참조횟수 : 2회 이상

047

계산에 의한 주소지정 방식

절대 주소지정 방식 (Absolute Addressing Mode)	기억장치 고유의 번지로서 0, 1, 2, 3과 같이 16진수로 약속하여 순서대로 정해놓은 번지인 절대 주소로 주소를 지정하는 방식
상대 주소지정 방식 (Relative Addressing Mode)	유효주소 : 명령어의 주소 부분 + PC
베이스 레지스터 주소지정 방식 (Base Register Addressing Mode)	· 유효주소 : 명령어의 주소 부분 + Base Register · 베이스 레지스터(Base Register) : 명령이 시작되는 최초의 번 지를 기억하고 있는 레지스터
인덱스 레지스터 주소지정 방식 (Indexed Addressing Mode)	· 유효주소 : 명령어의 주소 부분 + Index Register · 인덱스 레지스터(Index Register) : 주소 변경, 서브루틴 연결, 반복 연산 수행 등의 역할을 하는 레지스터

048

마이크로 오퍼레이션(Micro Operation)

- 명령을 수행하기 위해 클록 펄스(Clock Pulse)에 기준을 두고 CPU 내의 레지스터와 플래그의 상태
변환을 일으키는 동작
- 레지스터에 저장된 데이터에 의해서 이루어짐
- 마이크로 오퍼레이션을 순서적으로 일어나게 하려면 제어 신호(Control Signal)가 필요함
- 메이저 상태(Major State) : 현재 CPU가 하고 있는 작업의 상태
- PSW(Program Status Word) : 시스템의 순간 순간의 상태 기억

049

마이크로 사이클 타임(Micro Cycle Time)

동기 고정식 (Synchronous Fixed)	· 모든 마이크로 오퍼레이션 중에서 수행시간이 가장 긴 마이크로 오퍼레이션 의 수행시간을 마이크로 사이클 타임으로 정함 · 모든 마이크로 오퍼레이션의 수행시간이 유사한 경우 유리한 방식
동기 가변식 (Synchronous Variable)	· 각 마이크로 오퍼레이션에 따라서 수행시간을 다르게 정할 수 있음 · 각 마이크로 오퍼레이션의 사이클 타임이 현저한 차이를 나타낼 경우 유리한 방식
비동기식 (Asynchronous)	· 모든 마이크로 오퍼레이션에 대하여 서로 다른 마이크로 사이클 타임을 정 의하는 방식

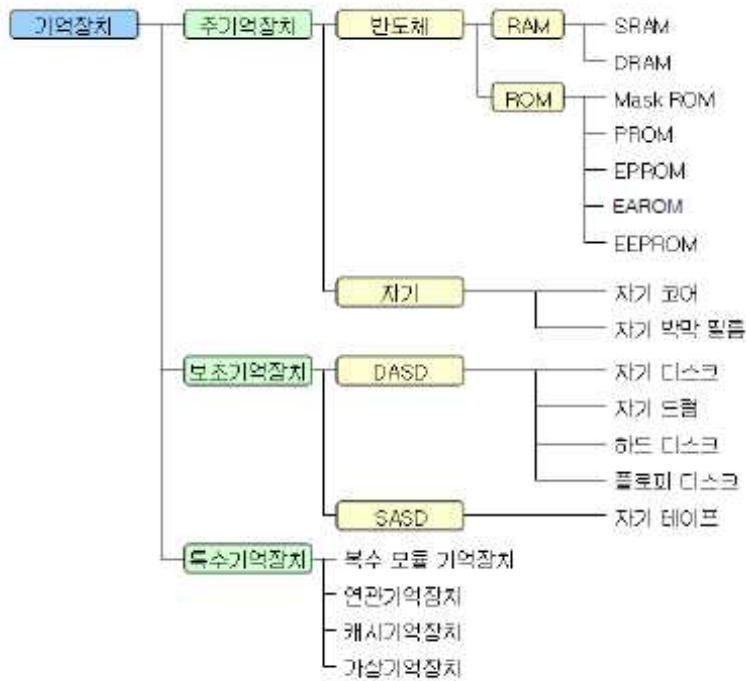
050

메이저 스테이트 변환 과정

- 인출 단계(Fetch Cycle)
- 간접 단계(Indirect Cycle)
- 실행 단계(Execute Cycle)
- 인터럽트 단계(Interrupt Cycle)

051

기억장치의 분류 및 계층구조



052
ROM의 종류

Mask ROM	반도체 공장에서 프로그램 되어 생산되며, 한 번 수록된 데이터는 수정할 수 없는 ROM
PROM (Programmable ROM)	사용자가 한 번만 내용을 기입할 수 있지만, 지울 수 없는 ROM
EPROM (Erasable PROM)	이미 기억된 내용을 자외선을 이용하여 여러 번 지우고 다시 사용할 수 있는 ROM
EAROM (Erasable Alterable ROM)	전기적 특성을 이용하여 기록된 정보의 일부를 바꿀 수 있는 ROM
EEPROM (Electronic EPROM)	이미 기억된 내용을 전기적인 방법을 이용하여 여러 번 지우고 다시 사용할 수 있는 ROM

053

구 분	DRAM(Dynamic RAM)	SRAM(Static RAM)
특 징	주기적으로 재충전(Refresh)이 필요함	전원이 공급되는 동안에는 기억 내용이 유지됨
구성 소자	컨덴서	플립플롭
소비 전력	낮음	높음
접근 속도	느림	빠름
용 도	일반 주기억장치	캐시 메모리

054

RAM/ROM의 용량 계산법

- $2^{\text{워드 수}} \times \text{워드의 크기}$
- 워드의 수 = 입력 번지(주소)선의 수 = MAR = PC
- 워드의 크기 = 출력 데이터선의 수 = Data Bus의 비트 수 = MBR = DR = IR

055

보조기억장치

자기 디스크(Magnetic Disk)

- 자성 물질을 입힌 금속 원판을 여러 장 겹쳐서 만든 기억장치
- DASD(Direct Access Storage Device) 장치로서 순차적 또는 직접 접근이 가능함
- 용량이 크고, 접근속도가 빠름
- 데이터 접근 시간(Access Time) = Seek Time + Rotational Delay Time + Transmission Time

자기 테이프(Magnetic Tape)

- 주소의 개념을 사용하지 않고, 처음부터 차례대로 처리하는 순차처리(SASD)만 할 수 있음
- 대량의 자료를 장시간 보관하는 데 가장 유리한 장치

056

연관 메모리(Associative Memory)

- 기억장치에서 자료를 찾을 때 주소에 의해 접근하지 않고, 기억된 내용의 일부를 이용하여 접근할 수 있는 기억장치
- CAM(Content Addressable Memory ; 내용 지정 메모리)이라고도 함
- 주소에 의해서만 접근이 가능한 기억장치보다 정보 검색이 신속함
- 외부의 인자와 내용을 비교하기 위한 병렬 판독 논리회로가 필요하므로 하드웨어 비용이 증가함

057

복수 모듈 기억장치

- 개별적으로 데이터를 저장할 수 있는 기억장치 모듈을 여러 개 가진 기억장치
- 기억장소의 접근을 보다 빠르게 하여 주기억장치와 CPU의 속도 차의 문제점을 개선함
- 메모리 인터리빙(Memory Interleaving) : 기억장소의 연속된 위치를 서로 다른 뱅크로 구성하여 하나의 주소를 통하여 여러 개의 위치에 해당하는 기억장소를 접근할 수 있도록 하는 방법

058

캐시 메모리(Cache Memory)

- CPU의 처리 속도와 주기억장치의 접근 속도 차이를 줄이기 위해 사용하는 고속 Buffer Memory
- 주기억장치에서 미리 사용할 데이터를 CPU에 근접하는 속도를 가진 캐시 메모리에 기억시켜서 수행 속도를 향상시킴

059

가상 메모리(Virtual Memory)

- 기억용량이 작은 주기억장치를 마치 큰 용량을 가진 것처럼 사용할 수 있도록 하는 기법
- 사용자는 프로그램의 크기에 제한 받지 않고 프로그램의 실행이 가능함
- 보조기억장치로 DASD 장치(자기디스크)를 많이 사용함
- 가상 메모리의 관리 기법 : 페이징(Paging) 기법, 세그먼테이션(Segmentation) 기법

060

입/출력의 제어방식

Programmed I/O

- 필요한 I/O 완료 여부를 체크하기 위해 CPU가 상태 Flag를 계속 조사하는 방식
- CPU에 의한 I/O라고도 함
- I/O 작업 시 CPU는 계속 I/O 완료 여부를 체크해야 하므로 다른 작업을 할 수 없음

Interrupt I/O

- CPU가 계속 Flag를 검사하지 않고 데이터가 준비되면 인터페이스가 CPU에 입·출력을 요구하고, 입·출력 전송이 완료되면 CPU는 수행 중이던 프로그램으로 되돌아가서 수행을 재개하는 방식
- CPU가 계속 Flag를 검사하지 않아도 되기 때문에 Programmed I/O보다 효율적임

DMA(Direct Memory Access)에 의한 I/O

- 데이터 입·출력 전송이 CPU를 통하지 않고 직접 주기억장치와 주변장치 사이에서 수행되는 방식
- CPU를 거치지 않고 메모리와 입·출력장치가 직접 통신하기 때문에 CPU에 부하가 증가되지 않으며, CPU의 상태 보존이 필요 없음
- 대량의 데이터를 고속으로 전송할 때 유리한 방식

Channel에 의한 I/O

- 채널(Channel) : 신호를 보낼 수 있는 전송로로써, 입·출력 장치와 주기억장치를 연결하는 중개 역할을 담당하는 부분
- CPU의 도움 없이 입·출력 동작을 수행하도록 지시하고, 작업이 끝나면 CPU에게 인터럽트로 알려줌
- 채널의 종류 : Selector Channel, Multiplexer Channel, Block Multiplexer Channel

061

- 버퍼링(Buffering) : 주기억장치의 일부 공간을 버퍼 공간으로 할당하여 처리할 데이터를 일시 기억하여 처리하는 방식
- 스푼링(Spooling : Simultaneous Peripheral Operation On-Line) : 입·출력 장치의 효율을 높이기 위해 입·출력의 내용을 디스크에 모아두었다가 처리하는 방식
- 버퍼링과 스푼링의 차이점

구 분	버퍼링	스푼링
구현 방식	하드웨어	소프트웨어
운영 방식	단일작업 단일사용자	다중작업 다중사용자
저장 위치	주기억장치	보조기억장치(디스크)

062

외부 인터럽트(External Interrupt) 종류

- 전원 이상 인터럽트
- 기계 검사 인터럽트

- 외부 신호 인터럽트
- 인·출력 인터럽트

063

프로그램 실행 중에 트랩(Trap)이 발생하는 조건

- 0에 의한 나눗셈
- 불법적인 명령
- Overflow 또는 Underflow가 발생한 경우
- 보호 영역 내의 메모리 어드레스를 Access하는 경우

064

인터럽트 수행 순서

- ① 인터럽트 요청 신호 발생
- ② 현재 수행 중인 명령을 완료하고, 상태를 기억시킴
- ③ 어느 장치가 인터럽트를 요청했는지 찾음
- ④ 인터럽트 취급 루틴 수행
- ⑤ 보존한 프로그램 상태 복귀

065

인터럽트 우선순위

전원 이상 > 기계 이상 > 외부 신호 > 인·출력 > 명령어 잘못 > 프로그램 검사 > SVC

066

인터럽트 우선순위 판별 방법

- 폴링(Polling) : 소프트웨어적인 방법
- 데이지 체인(Daisy Chain) : 하드웨어적인 방법

3과목 운영체제

067

시스템 소프트웨어의 분류

- 제어 프로그램 : 감시 프로그램, 작업 제어 프로그램, 자료 관리 프로그램
- 처리 프로그램 : 언어 번역 프로그램, 서비스 프로그램, 문제 프로그램

068

언어 번역 과정

원시 프로그램 → (컴파일러/어셈블러) → 목적 프로그램 → (링커) → 로드 모듈 → (로더) → 실행

069

매크로(Macro)

- 반복되는 코드를 한번만 작성하여 특정 이름으로 정의한 후 정의된 이름이 사용될 때 마다 작성된 코드를 삽입해서 실행하는 방법
- 매크로 정의 내에 또 다른 매크로 정의를 할 수 있음

부프로그램(Sub-program)

- 반복적으로 사용되는 코드를 별도의 프로그램으로 작성하여 필요할 때 호출하여 사용할 수 있도록 제작한 프로그램
- 부프로그램과 매크로의 공통점 : 여러 번 중복되는 부분을 별도로 작성하여 사용함

매크로 프로세서(Macro Processor)의 기능

- 매크로 정의 인식
- 매크로 정의 저장
- 매크로 호출 인식
- 매크로 호출 확장

070

운영체제의 목적 및 역할

- 사용자와 컴퓨터 시스템 간의 인터페이스 기능 제공
- 자원의 효율적인 운영 및 자원 스케줄링
- 데이터 공유 및 주변장치 관리
- 처리 능력 및 신뢰성 향상
- 응답시간과 반환시간 단축
- 시스템 오류 처리

071

운영체제의 운용 기법

- 일괄 처리 시스템 : 일정량의 데이터를 모아서 한꺼번에 처리함
- 다중 프로그래밍 시스템 : 하나의 CPU와 주기억장치를 이용하여 여러 개의 프로그램을 동시에 처리함
- 다중 처리 시스템 : 여러 개의 CPU와 하나의 주기억장치를 이용하여 여러 개의 프로그램을 동시에 처리함
- 시분할 처리 시스템 : 여러 명의 사용자가 사용하는 시스템에서 컴퓨터가 사용자들의 프로그램을 번갈아가며 처리함
- 실시간 처리 시스템 : 데이터 발생 또는 데이터에 대한 처리 요구가 있는 즉시 처리하여 응답함
- 다중 모드 시스템 : 일괄 처리 시스템, 시분할 시스템, 다중 처리 시스템, 실시간 처리 시스템을 한 시스

템에서 모두 제공함

- 분산 처리 시스템 : 여러 개의 컴퓨터를 통신 회선으로 연결하여 하나의 작업을 처리함

072

운영체제의 발달 순서 : 일괄 처리 → 다중 프로그래밍/다중 처리/시분할 처리/실시간 처리 → 다중 모드
→ 분산 처리

073

프로세스(Process)의 정의

- 현재 실행 중인 프로그램
- PCB를 가진 프로그램
- 프로세서(CPU)가 할당되는 실체
- 프로시저가 활동 중인 실체
- 비동기적 행위를 일으키는 주체

074

PCB에 저장되어 있는 정보

- 프로세스의 현재 상태
- 프로세스 고유 식별자(PID)
- 스케줄링 및 프로세스의 우선순위
- CPU 레지스터 정보
- 각종 자원의 포인터

075

스케줄링의 목적(평가 기준)

- 처리율 및 CPU 이용률 증가
- 오버헤드 최소화
- 응답시간, 반환시간, 대기시간 최소화

076



077

비선점 기법

- FIFO 스케줄링 : 준비상태 큐에 먼저 들어온 작업에게 CPU를 먼저 할당하는 기법
- SJF 스케줄링 : 준비상태 큐에서 기다리고 있는 프로세스들 중에서 작업이 끝나기까지의 실행시간 추정치가 가장 작은 작업을 먼저 실행하는 기법
- HRN 스케줄링 : 우선순위=(대기시간+서비스시간)/서비스시간
- 기한부(Deadline) 스케줄링 : 제한된 시간 내에 반드시 작업이 완료되도록 하는 기법

078

선점 기법

- RR 스케줄링 : 주어진 시간 할당량 안에 작업을 마치지 않으면 준비상태 큐의 가장 뒤로 배치됨
- SRT 스케줄링 : 현재 실행 중인 프로세스의 남은 시간과 준비상태 큐에 새로 도착한 프로세스의 실행 시간을 비교하여 가장 짧은 실행 시간을 요구하는 프로세스에게 CPU를 할당함
- 다단계 큐 스케줄링 : 프로세스를 특정 그룹으로 분류할 수 있을 경우 그룹에 따라 각기 다른 준비상태 큐를 사용하는 기법
- 다단계 피드백 큐 스케줄링 : 여러 개의 준비상태 큐를 두어 낮은 단계로 내려갈수록 프로세스의 시간 할당량을 크게 하는 방식

079

Context Switching(문맥 교환) : 다중 프로그래밍 시스템에서 운영체제에 의하여 CPU가 할당되는 프로세스를 변경하기 위하여 현재 CPU를 사용하여 실행되고 있는 프로세스의 상태 정보를 저장하고, 앞으로 실행될 프로세스의 상태 정보를 설정한 다음에 CPU를 할당하여 실행이 되도록 하는 작업

080

병행 프로세스

- 일계 구역 : 다중 프로그래밍 운영체제에서 공유 자원에 대하여 한 순간에는 반드시 하나의 프로세스에 의해서만 자원 또는 데이터가 사용되도록 지정된 영역
- 상호 배제 : 공유 자원을 어느 시점에서 단지 한 개의 프로세스만이 사용할 수 있도록 하며, 다른 프로세스가 공유자원에 대하여 접근하지 못하게 제어하는 기법
- 동기화 : 세마포어, 모니터

081

교착상태 발생 조건

- 상호 배제(Mutual Exclusion)
- 점유와 대기(Hold and Wait)
- 비선점(Non-Preemption)
- 환형(순환) 대기(Circular Wait)

082

교착상태 해결 방법

- 예방 기법(Prevention)
- 회피 기법(Avoidance)
- 회복 기법(Recovery)

083

기억장치 관리 전략

- 반입 전략 : 보조기억장치의 프로그램/데이터를 주기억장치로 언제 가져올 것인가를 결정
- 배치 전략 : 보조기억장치의 프로그램/데이터를 주기억장치의 어디로 위치시킬 것인가를 결정
- 교체 전략 : 주기억장치에 새로 배치될 프로그램을 위해 어떤 프로그램/데이터를 제거할지 결정

084

- 내부 단편화 : 분할된 영역이 할당될 프로그램의 크기보다 커서 프로그램이 할당된 후 사용되지 않고 남아 있는 빈 공간
- 외부 단편화 : 분할된 영역이 할당될 프로그램의 크기보다 작아서 프로그램이 할당되지 못해 사용되

지 않고 남아 있는 빈 공간

085

가상기억장치의 구현 방식

- 페이징(Paging) 기법 : 가상기억장치에 보관되어 있는 프로그램과 주기억장치의 영역을 동일한 크기로 나눈 후, 나뉜 프로그램(Page)을 동일하게 나뉜 주기억장치의 영역(Page Frame)에 적재시켜 실행하는 기법
- 세그멘테이션(Segmentation) 기법 : 가상기억장치에 보관되어 있는 프로그램을 다양한 크기의 논리적인 단위(세그먼트, Segment)로 나눈 후 주기억장치에 적재시켜 실행시키는 기법

086

구역성(Locality)

- 시간 구역성 : 한 번 참조된 기억장소가 가까운 미래에도 계속 참조될 가능성이 높음을 의미함(루프, 스택, 서브루틴, 서브프로그램 등)
- 공간 구역성 : 하나의 기억장소가 가까운 장래에도 계속 참조될 가능성이 높음을 의미함(배열 순례, 순차적 코드의 실행 등)

087

워킹 셋(Working Set)

- 프로세스가 자주 참조하는 페이지의 집합
- 자주 참조되는 워킹 셋을 주기억장치에 상주시킴으로써 스래싱 현상을 최소화할 수 있음

088

페이지 부재(Page Fault) : 가상 페이지 주소를 사용하여 데이터를 접근하는 프로그램이 실행될 때, 프로그램에서 접근하려고 하는 페이지가 주기억장치에 있지 않은 경우 발생하는 현상

089

스래싱(Thrashing)

- 하나의 프로세스가 작업 수행 과정에서 수행하는 기억장치 접근에서 지나치게 페이지 부재가 발생하여 프로세스 수행에 소요되는 시간보다 페이지 이동에 소요되는 시간이 더 커지는 현상
- 다중 프로그래밍 정도가 높아지면 어느 시점까지는 CPU의 이용률이 높고 스래싱의 발생 빈도도 낮아지지만, 어느 시점을 넘어서면 CPU의 이용률이 낮아지고 스래싱의 발생 빈도도 높아짐

090

페이지 교체 알고리즘

- OPT(OPTimal replacement) : 앞으로 가장 오랫동안 사용되지 않을 페이지를 교체하는 기법(실현 가능성 희박)
- FIFO(First In First Out) : 가장 오랫동안 주기억장치에 상주했던 페이지를 교체하는 기법
- LRU(Least Recently Used) : 현 시점에서 가장 오랫동안 사용되지 않았던 페이지를 먼저 교체하는 기법
- LFU(Least Frequently Used) : 각 페이지들이 얼마나 자주 사용되었는가에 중점을 두어 참조된 횟수가 가장 적은 페이지를 교체하는 기법
- NUR(Not Used Recently) : 최근에 참조되지 않은 페이지를 교체하는 기법으로, 최근의 사용 여부를 나타내기 위해서 두 개의 하드웨어 비트(참조 비트, 변형 비트)가 필요함
- SCR(Second Chance Replacement) : 가장 오랫동안 주기억장치에 상주했던 페이지 중에서 자주 참

조되는 페이지의 교체를 예방하기 위한 기법

091

디스크 스케줄링

- FCFS(First Come First Service) : 디스크 대기 큐에 먼저 들어온 순서대로 서비스 하는 기법
- SSTF(Shortest Seek Time First) : 현재 헤드의 위치에서 가장 가까운 트랙에 대한 요청을 먼저 서비스하는 기법
- SCAN : 헤드가 현재 진행 중인 방향으로 가장 짧은 탐색 거리에 있는 트랙에 대한 요청을 먼저 서비스하고, 안쪽 실린더 도착 시 다시 바깥쪽 실린더 쪽으로 헤드가 이동하면서 서비스하는 기법
- C-SCAN(Circular SCAN) : 헤드가 항상 바깥쪽 실린더에서 안쪽으로 움직이면서 가장 짧은 탐색시간을 가지는 요청을 서비스하는 기법
- N-Step SCAN : 어떤 방향의 진행이 시작될 당시에 대기 중이던 요청들만 서비스하고, 진행도중 도착한 요청들은 한데 모아져서 다음의 반대 방향 진행 때 최적으로 서비스 할 수 있도록 배열되는 기법
- Look : SCAN 기법을 사용하되, 진행 방향의 마지막 요청을 서비스한 후, 그 방향의 끝으로 이동하는 것이 아니라 곧바로 역방향으로 진행하는 기법

092

파일 시스템의 기능

- 파일의 생성, 변경, 제거
- 파일에 대한 여러 가지 접근 제어 방법 제공
- 예비(Backup)이나 복구(Recovery) 등의 정보 손실이나 파괴 방지

093

파일 디스크립터

- 파일이 액세스되는 동안 시스템(운영체제)이 관리 목적으로 알아야 할 정보를 모아 놓은 자료구조
- 파일 디스크립터의 정보 : 파일 이름, 파일 ID 번호, 파일 크기, 파일 구조, 보조기억장치 내의 파일 주소, 보조기억장치의 유형, 접근 제어 정보 등

094

파일 구조

순차 파일(Sequential File)

- 파일을 구성하는 레코드를 논리적인 처리 순서에 따라 연속된 물리적 저장 공간에 기록
- 기억장치의 효율이 높고, 다음 레코드의 접근이 빠름
- 삽입/삭제/검색 시에 효율이 낮음
- 기억 매체로 자기 테이프를 주로 사용함

직접 파일(Direct File)

- 직접 접근 기억장치의 물리적 주소를 통해 직접 레코드에 접근
- 접근 시간이 빠르며, 판독이나 기록의 순서에는 제약이 없음
- 기억 매체로 자기 디스크를 주로 사용함

색인 순차 파일(Indexed Sequential File)

- 순차 파일과 직접 파일을 결합한 형태
- 각 레코드는 레코드 키값에 따라 논리적으로 배열되며, 시스템은 각 레코드의 실제 주소가 저장된 색인을 관리함
- 디스크 기억장치에 많이 이용됨

- 구성 : 기본 영역, 색인 영역(트랙/실린더/마스터), 오버플로우 영역

095

디렉토리 구조

- 1단계 디렉토리 : 모든 파일이 하나의 디렉토리 내에 위치하여 관리되는 구조
- 2단계 디렉토리 : 중앙에 마스터 파일 디렉토리가 있고, 그 아래에 각 사용자에게 할당하는 디렉토리가 있는 구조
- 트리 구조 : 하나의 루트 디렉토리에 여러 개의 서브 디렉토리로 구성된 구조
- 비순환 그래프 구조 : 사이클이 허용되지 않는 구조
- 일반 그래프 구조 : 트리 구조에 링크를 첨가시켜 사이클을 허용하는 구조

096

자원 보호

- 접근 제어(Access Control) : 접근 제어 행렬, 접근 제어 리스트
- 권한(자격) 리스트(Capability List)
- 전역 테이블(Global Table)

097

파일 보호 기법

- 파일 명명(File Naming)
- 비밀번호>Password)
- 암호화(Cryptography)
- 접근 제어(Access Control)

098

보안

- 보안 요건 : 기밀성, 무결성, 가용성, 인증, 부인방지
- 보안 유지 기법 : 외부 보안(시설 보안, 운용 보안), 내부 보안, 사용자 인터페이스 보안

099

다중 처리기의 운영체제

주/종 처리기

- 하나의 프로세서를 Master(주프로세서)로 지정하고, 나머지는 Slave(종프로세서)로 지정하는 구조
- 주프로세서는 운영체제를 수행하며, 입·출력과 연산을 담당함
- 종프로세서는 연산만 담당하고, 입·출력 발생 시 주프로세서에게 요청함

분리 수행 처리기

- 주/종 처리기의 문제점을 보완하여 각 프로세서가 독자적인 운영체제를 수행하도록 구성한 구조

대칭적 처리기

- 분리 수행 처리기의 비대칭성을 보완하여 여러 프로세서들이 완전한 기능을 갖춘 하나의 운영체제를 공유하여 수행하는 구조

100

분산 처리 시스템

- 독립적인 처리 능력을 가진 컴퓨터 시스템을 통신망으로 연결한 시스템으로 약결합 시스템
- 설계 목적 : 자원 공유, 연산 속도 향상, 신뢰도 향상

• 특징

장점	<ul style="list-style-type: none"> • 자원 공유 및 통신이 용이함 • 빠른 반응 시간, 작업 처리량이 항상, 신뢰성 항상 • 점진적인 확장이 용이함
단점	<ul style="list-style-type: none"> • 집중형 시스템에 비해 소프트웨어 개발이 어려움 • 보안이 어렵고, 오류의 잠재성이 큼

101

분산 시스템의 위상(Topology)에 따른 분류

- 스타(Star)형 : 모든 사이트가 하나의 중앙 사이트에만 직접 연결된 구조
- 링(Ring)형 : 각 사이트는 인접하는 다른 두 사이트와만 직접 연결된 구조
- 다중 접근 버스 연결 구조(Multi Access Bus Connection) : 모든 사이트들이 공유 버스에 연결된 구조
- 망(Network)형 : 각 사이트는 시스템 내의 모든 사이트들과 직접 연결되어 있는 구조
- 트리(Tree)형 : 각 사이트들이 트리 형태로 구성되며, 루트 사이트와 서브 사이트가 존재함

102

UNIX의 특징

- 시분할 시스템을 위해 설계된 대화식 운영체제
- 소스가 공개된 개방형 시스템
- 단순하고, 강력한 명령어를 제공함
- 대부분 C언어로 작성되어 이식성과 확장성이 높음
- 멀티유저 및 멀티태스킹을 지원함
- 트리 구조의 파일 시스템

103

UNIX 시스템의 구성

커널(Kernel)	<ul style="list-style-type: none"> • UNIX 시스템의 가장 핵심적인 부분 • 하드웨어를 보호하고, 하드웨어와 프로그램 간의 인터페이스 역할을 담당함 • 프로세스 관리, 기억장치 관리, 파일 관리, 입·출력 관리, 프로세스 간의 통신 등을 수행함
셸(Shell)	<ul style="list-style-type: none"> • 명령어 해석기로 사용자의 명령어를 인식하여 필요한 프로그램을 호출하고 그 명령을 수행함 • 사용자와 시스템 간의 인터페이스를 담당함
유틸리티(Utility)	<ul style="list-style-type: none"> • 사용자의 편의를 위한 프로그램으로 DOS의 외부 명령어와 유사함

104

UNIX 파일 시스템 구조

부트 블록 (Boot Block)	부트스트랩에 필요한 파일들이 존재하며, 루트 영역 외에는 해당되지 않음
슈퍼 블록 (Super Block)	파일 시스템을 관리하는데 필수적인 정보가 저장됨
i-node 블록 (Index Node Block)	파일 소유자의 사용자 번호, 파일 크기, 파일 유형, 파일 생성 시간, 최종 수정 시간, 파일 링크 수, 데이터가 저장된 블록의 주소 등이 기록됨
데이터 블록 (Data Block)	실제 데이터가 저장됨

UNIX 명령어

• 프로세스 관련 명령어

fork	새로운 프로세스 생성
exec	새로운 프로세스 수행
exit	프로세스 수행 종료
wait	자식 프로세스의 하나가 종료될 때까지 부모 프로세스를 일시로 중지시킴
getpid	자신의 프로세스 아이디 구함
getppid	부모 프로세스 아이디 구함
pipe	두 프로세스를 연결하여 프로세스 간 통신을 가능하게 함

• 시스템 관련 명령어

create/open/close/ cp/mv/rm	파일의 생성/준비/종료/복사/이동 및 이름변경/삭제
cat	파일의 내용 표시
chmod	파일에 대한 액세스 권한을 설정하여 파일의 사용 허가를 지정
chown	소유자 변경
mount	기존 파일 시스템에 새로운 파일 시스템을 서브디렉토리에 연결할 때 사용
mkfs	파일 시스템 생성
fsck	파일 시스템 검사
mkdir/chdir/rmdir	디렉토리 생성/위치변경/삭제
ls	현재 디렉토리 내의 파일 목록 확인

4과목 시스템 분석 및 설계

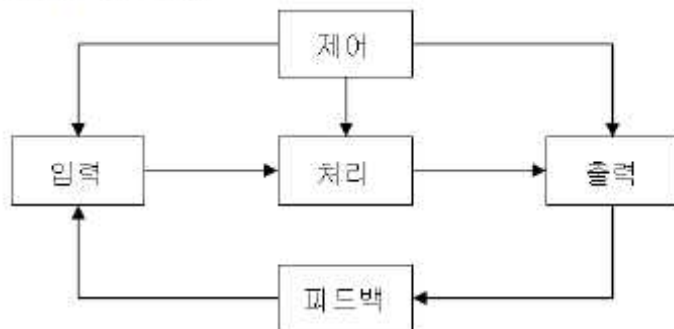
106

시스템(System)

- 어떤 목적 또는 목표를 위하여 여러 기능 요소가 상호 관련하여 결합된 절차나 방법의 유기적 집합체
- 특성 : 목적성, 자동성, 제어성, 종합성

107

시스템 기본 요소



108

시스템 개발 생명 주기

조사 → 분석 → 설계 → 구현 → 시험 → 운영 → 유지보수

109

유지보수의 종류

- 수정 유지보수(Corrective Maintenance)
- 적응 유지보수(Adaptive Maintenance)
- 기능(완전) 유지보수(Perfective Maintenance)
- 예방 유지보수(Preventive Maintenance)

110

시스템 분석가의 기본 조건

- 기업 목적의 정확한 이해
- 컴퓨터 장치와 소프트웨어에 대한 지식
- 시간 배정과 계획 등의 빠른 파악
- 창조적, 인간 중심적

111

코드의 기능

- 코드의 3대 기능 : 식별, 분류, 배열 기능
- 기타 기능 : 표준화, 간소화, 연상, 암호화, 오류 검출 기능

112

코드 설계 순서

① 코드화 대상의 결정

- ② 코드화 목적의 명확화
- ③ 번호 부여 대상 수 확인
- ④ 사용 범위 결정
- ⑤ 사용 기간의 결정
- ⑥ 코드화 대상의 특성 분석
- ⑦ 번호 부여 방식의 결정

113

코드의 종류

- 순차 코드(Sequence Code) : 코드화 대상 항목을 어떤 일정한 배열로 일련번호를 배당하는 코드
- 블록 코드(Block Code) : 코드화 대상 항목에 미리 공통의 특성에 따라서 임의의 크기를 블록으로 구분하여 각 블록 안에서 일련번호를 배정하는 코드
- 그룹 분류식 코드(Group Classification Code) : 코드화 대상 항목을 소정의 기준에 따라 대분류/중분류/소분류로 구분하고 순서대로 번호를 부여하는 코드
- 10진 코드(Decimal Code) : 좌측부는 그룹분류에 따르고 우측부는 10진수의 원칙에 따라 세분화하는 코드
- 표의 숫자 코드(Significant Digit Code) : 코드화 대상 항목의 길이, 넓이, 부피, 무게 등을 나타내는 문자나 숫자, 기호를 그대로 코드로 사용하는 코드
- 연상 코드(Mnemonic Code) : 코드화 대상의 품목 명칭 일부를 약호 형태로 코드 속에 넣어 대상 항목을 쉽게 파악할 수 있는 코드

114

코드의 오류

- 필사 오류(Transcription Error) : 입력 시 한 자리를 잘못 기록하는 오류
- 전위 오류(Transposition Error) : 입력 시 좌우 자리를 바꾸어 발생하는 오류
- 이중 오류(Double Transposition Error) : 전위 오류가 두 개 이상 발생하는 오류
- 생략 오류(Missing Error) : 입력 시 한 자리를 빼고 기록하는 오류
- 추가 오류(Addition Error) : 입력 시 한 자리를 추가해서 기록하는 오류
- 임의 오류(Random Error) : 두 가지 이상의 오류가 결합해서 발생하는 오류

115

입력 설계 순서

- ① 입력 정보의 발생
- ② 입력 정보의 수집
- ③ 입력 정보의 매체화
- ④ 입력 정보의 투입
- ⑤ 입력 정보의 내용

116

입력 매체 시스템

- 진중 매체화 시스템
- 분산 매체화 시스템
- 턴 어라운드 시스템
- 직접 입력 시스템

117

원시 전표 설계 시 고려사항

- 기입량이 적고, 기입이 쉽도록 해야 함
- 일정 순서대로 기입할 수 있어야 함
- 고정 항목은 미리 인쇄하거나 선택 가능해야 함
- 기입상 혼란을 일으킬 수 있는 경우는 전표상에 기입 요령을 명시해야 함

118

출력 설계 순서

- ① 출력 정보의 내용
- ② 출력 정보의 매체화
- ③ 출력 정보의 분배
- ④ 출력 정보의 이용

119

출력 매체 시스템

- 인쇄 출력 시스템
- 디스플레이 출력 시스템
- 파일 출력 시스템
- 음성 출력 시스템
- COM(Computer Output Microfilm) 시스템
- 턴 어라운드 시스템

120

파일의 내용에 따른 분류

- 원시 파일(Source File)
- 마스터 파일(Master File)
- 트랜잭션 파일(Transaction File)
- 요약 파일(Summary File)
- 히스토리 파일(History File)
- 백업 파일(Backup File)
- 트레일러 파일(Trailer File)

121

파일 설계 순서

- ① 파일의 성격 검토
- ② 파일의 항목 검토
- ③ 파일의 특성 조사
- ④ 파일 매체의 검토
- ⑤ 편성법 검토

122

프로세스 표준 처리 패턴

- 매체 변환(Conversion)
- 병합(Merge)
- 갱신(Update)
- 분배(Distribution)

- 추출(Extract)
- 조합(Collate)
- 대조(Matching)
- 생성(Generate)

123

오류 검사 시스템

컴퓨터 입력 단계에서의 검사 방법

- 체크 디지트 검사(Check Digit Check)
- 균형 검사(Balance Check)
- 범위 검사(Limit Check)
- 일괄 합계 검사(Batch Total Check)
- 데이터 수 검사(Data Count Check)
- 논리 검사(Logical Check)

계산 처리 단계에서의 검사 방법

- 중복 레코드 검사(Double Record Check)
- 불일치 레코드 검사(Unmatched Record Check)
- 한계 초과 검사(Overflow Check)

124

프로그램 설계서

- 작성 효과 : 비용 절감, 수정/유지보수 용이, 시스템 변경 시 적용 용이, 인사이동시 결함 방지
- 구성 : 시스템명, 코드명, 설계 방침, 입·출력 설계표, 프로그래밍 지시서 등
- 프로그래밍 지시서 포함 사항 : 설계서 작성자명, 프로그램의 작성 기간 및 비용, 입·출력 일람, 처리 개요

125

시스템 평가

- 시스템 평가 목적 : 시스템 운영 관리의 타당성 파악, 시스템의 성능과 유용도 판단, 처리 비용과 효율 면에서 개선점 파악
- 시스템 평가 항목 : 신뢰성 평가, 기능 평가, 성능 평가

126

폭포수 모형(Waterfall Model)

- Boehm이 제시한 고전적 생명주기 모형으로, 소프트웨어 개발 과정의 각 단계가 순차적으로 진행됨
- 개발 단계 : 계획 → 요구 분석 → 설계 → 구현 → 시험(검사) → 운용 → 유지보수
- 장점 : 적용 경험과 성공 사례 많음, 단계별 정의 분명, 전체 공조의 이해 용이, 단계별 산출물 명확
- 단점 : 개발 과정 중에 발생하는 새로운 요구나 경험을 설계에 반영하기 어려움, 이전 단계의 오류 수정이 어려움

127

프로토타입 모형(Prototype Model)

- 실제 개발될 시스템의 견본(Prototype)을 미리 만들어 최종 결과물을 예측하는 모형
- 개발 단계 : 요구 수집 → 빠른 설계 → 프로토타입 구축 → 고객 평가 → 프로토타입 조정 → 구현
- 장점 : 요구사항이 충실히 반영됨, 프로토타입은 발주자/개발자에게 공동 참조 모델을 제공함
- 단점 : 프로토타입과 실제 소프트웨어와의 차이로 인한 사용자의 혼란 야기

128

나선형 모형(Spiral Model)

- Boehm이 제시하였으며, 반복적인 작업을 수행하는 점증적 생명 주기 모형
- 개발 단계 : 계획 수립 → 위험 분석 → 공학적 개발 → 고객 평가
- 장점 : 위험 분석 단계에서 기술과 관리의 위험 요소들을 하나씩 제거함으로써 완성도 높은 소프트웨어 개발 가능
- 단점 : 위험 분석 단계에서 발견 못한 위험 요소로 인한 문제 발생, 적용 경험이나 성공 사례가 많지 않음

129



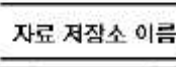
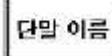
IPT 기법

- IPT 기법의 분류 : 기술적 측면(구조적 설계, 구조적 코딩, 하향식 프로그래밍), 관리적 측면(책임 프로그래머팀, 구조적 검토회, 결렬, 개발 지원 라이브러리)
- IPT 기법의 도입 목적 : 개발자의 생산성 향상, 프로그래밍의 표준화 유도, 개인적인 차이 해소
- HIPO 패키지의 3단계 다이어그램 : 도식 목차(Visual Table of Contents), 총괄 도표(Overview Diagram), 상세 도표(Detail Diagram)

130

구조적 분석 도구

- 자료 흐름도(DFD : Data Flow Diagram)

구성 요소	의 미	표기법
처리 공정 (Process)	자료를 변환시키는 시스템의 한 부분(처리 공정)을 나타냄	
자료 흐름 (Data Flow)	자료의 이동(흐름)을 나타냄	
자료 저장소 (Data Store)	시스템에서의 자료 저장소(파일, 데이터베이스)를 나타냄	
단말 (Terminator)	자료의 출처(생산자)와 도착지(소비자)를 나타냄	

- 자료 사전(DD : Data Dictionary)

기호	의미	설명
=	자료의 정의	~로 구성되어 있다.(is compose of)
+	자료의 연결	그리고(and)
()	자료의 생략	생략 가능한 자료(optional)
[]	자료의 선택	또는(or)
{ }	자료의 반복	{ } _n : 최소 n번 이상 반복 { } ⁿ : 최대 n번 이하 반복 { } _m ⁿ : m번 이상 n번 이하 반복
* *	자료의 설명	주석(Comment)

- 소단위 명세서(Mini-Specification)
- 개체 관계도(ERD : Entity Relationship Diagram)
- 상태 전이도(STD : State Transaction Diagram)

131

구조적 설계 순서

구조 도표 작성 → 구조 도표 평가 → 모듈 설계 → 데이터베이스 설계 → 설계의 패키징

132

결합도(Coupling)

- 한 모듈과 다른 모듈 간의 상호 의존도 또는 두 모듈 사이의 연관 관계
- 독립적인 모듈이 되기 위해서는 각 모듈 간의 결합도가 약해야 하며, 의존하는 모듈이 적어야 함
- 종류

데이터 결합도	스탬프 결합도	제어 결합도	외부 결합도	공통(공유) 결합도	내용 결합도
결합도 약함 ←					→ 결합도 강함

133

응집도(Cohesion)

- 모듈 안의 요소들이 서로 관련되어 있는 정도
- 종류

기능적 응집도	순차적 응집도	교환적 응집도	절차적 응집도	시간적 응집도	논리적 응집도	우연적 응집도
응집도 강함 ←						→ 응집도 약함

134

객체지향(Object Oriented) 기법 구성 요소

- 객체(Object) : 필요한 자료 구조(어트리뷰트)와 이에 수행되는 함수(메소드)들을 가진 모듈
- 어트리뷰트(Attribute) : 객체의 상태를 나타냄
- 메소드(Method) : 객체가 메시지를 받아 실행해야 할 객체의 구체적인 연산
- 클래스(Class) : 하나 이상의 유사한 객체들을 묶어 공통된 특성을 표현한 데이터 추상화를 의미함
- 메시지(Message) : 객체들 간의 상호 작용은 메시지를 통해 이루어짐

135

객체의 특징

- 캡슐화(Encapsulation) : 데이터와 데이터를 조작하는 연산을 하나로 묶는 것
- 정보 은닉(Information Hiding) : 객체가 다른 객체로부터 자신의 자료를 숨기고 자신의 연산만을 통하여 접근을 허용하는 것
- 추상화(Abstraction) : 주어진 문제나 시스템 중에서 중요하고 관계있는 부분만을 분리하여 간결하고 이해하기 쉽게 만드는 것
- 상속성(Inheritance) : 상위 클래스의 속성과 메소드를 하위 클래스가 물려받는 것
- 다형성(Polymorphism) : 많은 상이한 클래스들이 동일한 메소드명을 이용하는 것

136

럼바우(Rumbaugh)의 객체지향 분석 절차

객체 모델링(Object Modeling) → 동적 모델링(Dynamic Modeling) → 기능 모델링(Functional Modeling)

5과목 데이터 통신

137

변조 속도

- 초당 발생한 신호의 변화 횟수 • 속도 단위 : baud(보오)
- 변조 시 상태 변화 수 : 1비트(모노비트), 2비트(디비트), 3비트(트리비트), 4비트(쿼드비트)
- 변조 속도(baud) = 신호 속도(bps) / 변조 시 상태 변화 수

138

신호 속도

- 초당 전송되는 비트 수 • 속도 단위 : bps(bit/sec)
- 데이터 신호 속도(bps) = 변조 속도(baud) × 변조 시 상태 변화 수

139

통신 방식에 따른 종류

- 단방향 통신(Simplex) : 한쪽 방향으로만 전송이 가능한 방식(라디오, TV)
- 반이중 통신 (Half-Duplex) : 데이터를 양쪽방향으로 모두 전송할 수 있으나 동시에 양쪽방향에서 전송할 수 없는 전송 방식(ON-OFF 무전기)
- 전이중 통신(Full-Duplex) : 동시에 양쪽 방향으로 전송이 가능한 전송 방식(전화)

140

비동기식 전송

- Byte와 Byte를 구분하기 위하여 스타트(Start) 비트와 스톱(Stop) 비트를 붙여 전송하는 방식
- 2~3Bit의 오버헤드를 요구하여 전송 효율이 떨어지므로 저속, 단거리 통신 시스템에 이용됨

동기식 전송

- 문자 또는 비트들의 데이터 블록(프레임)을 송·수신 하는 방식으로, 동기를 유지하기 위해 동기 문자를 계속적으로 전송함
- 시작과 끝부분에 플래그(Flag) 신호를 삽입하여 동기화함
- 종류 : 문자 동기 방식, 비트 동기 방식

141

신호 변환 방식

아날로그 데이터 ↓ 아날로그 신호	• 아날로그 데이터를 아날로그 회선으로 전송하기 위해 아날로그 신호로 변조하는 것 • 아날로그 변조라고도 함 • 종류 : 진폭 변조(AM : Amplitude Modulation), 주파수 변조(FM : Frequency Modulation), 위상 변조(PM : Phase Modulation)
아날로그 데이터 ↓ 디지털 신호	• 아날로그 데이터를 디지털 회선으로 전송하기 위해 디지털 신호로 변환하는 것 • 펄스 부호 변조(PCM : Pulse Code Modulation) -아날로그 데이터(음성)를 디지털 신호로 전송하기에 적합한 변조 방법 -PCM 방식의 변조 순서 : 표본화 → 양자화 → 부호화
디지털 데이터 ↓ 아날로그 신호	• 디지털 데이터를 아날로그 회선으로 전송하기 위해 아날로그 신호로 변조하는 것 • 디지털 변조(=브로드밴드 변조)라고도 함 • 종류 : 진폭 편이 변조(ASK), 주파수 편이 변조(FSK), 위상 편이 변조(PSK), 진폭 위상 편이 변조(QAM)
디지털 데이터 ↓ 디지털 신호	• 디지털 데이터를 디지털 회선으로 전송하기 위해 디지털 신호로 변환하는 것 • DSU(Digital Service Unit)를 이용함

142

다중화(Multiplexing)

- 효율적인 전송을 위하여 넓은 대역폭을 가진 하나의 전송링크를 통하여, 여러 신호/데이터를 동시에 실어 보내는 기술
- 선로의 공동 이용이 가능하므로 전송 효율이 높아짐
- 종류 : 주파수 분할 다중화, 시분할 다중화

143

주파수 분할 다중화(FDM : Frequency-Division Multiplexing)

- 통신 회선의 주파수 대역폭을 여러 개의 작은 대역폭으로 분할하여 여러 대의 단말기가 동시에 사용할 수 있도록 하는 기법
- 채널 간의 대역폭이 겹치지 않도록 완충지역으로 보호 대역(Guard Band)이 필요함
- 가드 밴드의 이용으로 채널의 이용률이 낮아짐으로써 시분할 다중화기에 비해 비효율적임
- 케이블 혹은 TV 공중파에 적용됨

144

시분할 다중화(TDM : Time-Division Multiplexing)

- 통신 회선의 대역폭을 일정한 시간 폭(Time Slot)으로 분할하는 기법
- 대역폭의 이용도가 높아 고속 전송에 용이함
- 종류 : 동기식 시분할 다중화, 비동기식 시분할 다중화

동기식 시분할 다중화 (Synchronous TDM)	<ul style="list-style-type: none"> • 전송 매체상의 전송 프레임마다 해당 채널의 타임 슬롯이 고정적으로 할당되는 다중화 방식 • 전송할 데이터가 없는 단말장치에도 타임 슬롯이 고정적으로 할당되므로 타임 슬롯이 낭비될 수 있음
비동기식 시분할 다중화 (Asynchronous TDM)	<ul style="list-style-type: none"> • 실제로 전송할 데이터가 있는 단말장치에만 타임 슬롯을 할당함으로써 전송 효율을 높임 • 통계적 시분할 다중화, 지능 다중화라고도 함

145

전송 제어 절차

데이터 통신 회선의 접속 → 데이터 링크 설정(확립) → 정보 메시지 전송 → 데이터 링크의 종료(해제)
→ 데이터 통신 회선의 절단

146

BSC(Binary Synchronous Control)

- 바이트 단위로 구성된 프레임에 전송 제어 문자를 삽입하여 전송을 제어하는 문자 위주의 프로토콜
- 프레임 구조

SYN	SYN	SOH	Heading	STX	TEXT	ETX	BCC
-----	-----	-----	---------	-----	------	-----	-----

- 전송 제어 문자

SYN(SYNchronous idle)	문자 동기 유지
SOH(Start Of Heading)	헤딩의 시작
STX(Start of TeXt)	헤딩의 종료 및 본문의 시작
ETX(End of TeXt)	본문의 종료
ETB(End of Transmission Block)	전송 블록의 종료
ENQ(ENquiry)	상대편의 데이터 링크 설정 및 응답 요구
EOT(End Of Transmission)	한 개 또는 그 이상의 전송 종료를 표시
DLE(Data Link Escape)	인접하여 뒤따르는 제한된 수의 문자나 의미를 바꿈
ACK(ACKnowledge)	수신측에서 송신측으로 긍정 응답으로 보내는 문자
NAK(Negative Acknowledge)	수신측에서 송신측으로 부정 응답으로 보내는 문자

147

HDLC(High-level Data Link Control)

- 각 프레임에 데이터 흐름을 제어하고 오류를 검출할 수 있는 비트열을 삽입하여 전송하는 비트 위주의 프로토콜
- 전송 효율과 신뢰성이 높음
- 정보 전송 단위 : 프레임(Frame)
- 프레임 구조

플래그	주소부	제어부	정보부	검사부	플래그
-----	-----	-----	-----	-----	-----

- 플래그(Flag) : 프레임의 동기를 제공하기 위해 프레임의 시작과 끝을 표시하며, 항상 '01111110'의 형식을 취함
- 주소부(Address Field) : 프레임을 송신 및 수신하는 스테이션을 구별하기 위해 사용되는 부분
- 제어부(Control Field) : 프레임의 종류를 식별하기 위해 사용되는 부분(정보/감독/비번호 프레임)
- 정보부(Information Field) : 실제 데이터가 들어있는 부분
- 검사부(FCS : Frame Check Sequence Field) : 전송 오류를 검출하는 기능을 수행하는 부분
- 데이터 전송 모드 : 정규(표준) 응답 모드(NRM), 비동기 응답 모드(ARM), 비동기 평형 모드(ABM)

148

에러 발생 원인

- 감쇠(Attenuation) : 전송 신호가 전송 매체를 통과하는 과정에서 거리에 따라 점차 약해지는 현상
- 지연 왜곡(Delay Distortion) : 전송 매체를 통한 신호 전달이 주파수에 따라 그 속도를 달리 함으로써 유발되는 신호 손상
- 잡음(Noise) : 상호변조 잡음, 열 잡음, 누화 잡음, 충격 잡음

149

에러 검출 방식

- 패리티 검사(Parity Check) : 1비트의 검사 비트인 패리티 비트를 추가하여 전송 부호의 에러 검출
- 순환 중복 검사(CRC : Cyclic Redundancy Check) : 특정 다항식에 의한 연산 결과를 데이터에 삽입하여 에러 검출
- 해밍 코드(Hamming Code) : 자기 정정 부호의 하나로 비트 에러를 검출해서 1비트 에러 정정
- 상층 코드 방식 : 순차적 디코딩과 한계값 디코딩을 사용하여 에러 수정

150

자동 반복 요청(ARQ : Automatic Repeat reQuest)

- 통신 경로에서 에러 발생 시 수신측은 에러의 발생을 송신측에 통보하고 송신측은 에러가 발생한 프레임을 재전송하는 방식

- 종류 : 정지-대기 ARQ, 연속 ARQ(Go-Back-N ARQ, 선택적 재전송), 적응적 ARQ

151

전송 에러 제어 방식

- 전진 에러 수정 : 송신측에서 오류 정정을 위한 제어 비트를 추가하여 전송하고, 수신측에서 이 비트를 사용하여 에러를 검출하고 수정하는 방식
- 후진 에러 수정 : 데이터 전송 과정 중 에러가 발생하면 송신측에 재전송을 요구하는 방식

152

흐름 제어

- 통신망 내의 트래픽 제어의 원활한 흐름을 위해 전송하는 프레임(패킷)의 양이나 속도 규제
- 종류 : 정지-대기, 슬라이딩 윈도우

혼잡 제어

- 네트워크 내에서 패킷의 대기 지연(Queuing Delay)이 너무 높아지게 되어 트래픽이 붕괴되지 않도록 네트워크 측면에서 패킷의 흐름 제어

153

회선 제어 방식

- 경쟁 방식(Contention) : 송신 요구를 먼저 한 쪽이 송신권을 가짐
- 폴링(Polling) : 주컴퓨터가 단말기에게 전송할 데이터의 유무를 묻는 방식
- 셀렉션(Selection) : 주컴퓨터가 단말기에게 데이터를 수신할 수 있는지를 묻는 방식

154

회선 교환 방식

- 음성 전화망과 같이 메시지가 전송되기 전에 발생지에서 목적지까지의 물리적 통신 회선 연결이 선행되어야 하는 교환 방식
- 일단 통신경로가 설정되면 데이터의 형태, 부호, 전송제어 절차 등에 의한 제약을 받지 않음
- 전용 전송로와 고정 대역폭(Band Width)을 사용함
- 회선 교환 방식의 종류 : 시분할 교환 방식, 공간 분할 교환 방식

155

메시지 교환 방식

- 하나의 메시지 단위로 축적-전달(Store-and-Forward) 방식에 의해 데이터를 교환하는 방식
- 각 메시지마다 수신 주소를 붙여서 전송을 하며, 전송 경로가 다름
- 응답시간이 느려 대화형 데이터 전송을 위해서는 부적절하나, 수신측이 준비 안 된 경우에도 지연 후 전송이 가능함

156

패킷 교환 방식(Packet Switching)

- 메시지를 일정 길이의 전송 단위(Packet)로 정보를 나누어 전송하는 방식
- 모든 사용자 간에 빠른 응답 시간을 제공하기 위해 사용하며, 음성보다 데이터 전송에 더 적합함
- 전송량 제어와 전송 속도 변화가 가능하며, 융통성이 매우 크며 우선순위가 허용됨
- 전송 실패 패킷의 재전송이 가능함(장애 발생 시 대체 경로 선택이 가능함)
- 대량의 데이터 전송 시 전송 지연이 발생함
- 패킷 교환망의 기능 : 순서 제어, 경로 설정 제어, 트래픽 제어, 에러 제어, 패킷 다중화, 논리 채널

- 종류 : 가상 회선 방식, 데이터그램 방식

157

경로 설정(Routing, 라우팅)

- 데이터 패킷을 출발지에서 목적지까지 이용 가능한 전송로를 찾아본 후에 가장 효율적인 전송로를 선택하는 것
- 경로 배정 요소(Parameter) : 성능 기준, 경로의 결정 시간과 장소, 네트워크 정보 발생지
- 경로 설정 프로토콜 : IGP(RIP, OSPF), EGP, BGP
- 경로 설정 알고리즘 : 범람 경로 제어, 고정 경로 제어, 적응 경로 제어, 임의 경로 제어

158

네트워크 구성 형태

- Mesh형(망형) : 모든 노드 간의 연결이 이루어진 형태로, 많은 단말기로부터 많은 양의 통신을 필요로 하는 경우에 유리함
- Star형(중앙 집중형) : 중앙에 Host Computer가 있고 이를 중심으로 Terminal들이 연결
- Ring형(루프형) : 양쪽 방향으로 접근이 가능하여 통신회선 장애에 대한 융통성이 있음
- Bus형 : 시스템 내의 모든 사이트들이 공유 버스에 연결된 구조
- Tree형(계층형) : 분산 처리 시스템을 구성하는 방식

159

LAN(Local Area Network, 근거리 통신망)

- 한 건물 또는 공장, 학교 구내, 연구소 등의 일정지역 내의 설치된 통신망으로서 각종 기기 사이의 통신을 실행하는 통신망
- 광대역 전송 매체(꼬일선, 동축 케이블, 광섬유 케이블 등)의 사용으로 고속 통신이 가능함
- 확장성과 재배치성이 좋고, 경로 설정이 필요 없음

160

매체 접근 제어(MAC : Media Access Control)

- CSMA/CD(Carrier Sense Multiple Access / Collision Detection)
- 토큰 버스(Token Bus)
- 토큰 링(Token Ring)

161

VAN(Value Added Network, 부가가치 통신망)

- 공중 통신 회선에 교환설비, 컴퓨터 및 단말기 등을 접속시켜 새로운 부가 기능을 제공하는 통신망
- VAN의 계층 구조 : 전송 계층, 네트워크 계층, 통신처리 계층, 정보처리 계층

162

ISDN(Integrated Service Digital Network, 종합 정보 통신망)

- 모든 통신 서비스를 단일 통신망으로 통합한 것
- 컴퓨팅, 교환, 디지털 전송 장치간의 구분이 없어지고, 음성, 데이터 및 이미지 전송에 동일한 디지털 기술이 적용된 통합 시스템

163

인터넷 주소 체계

- 인터넷에 연결된 모든 컴퓨터는 고유의 IP 주소(Internet Protocol Address)를 가짐
- 현재는 IP 주소를 32bit 크기로 8bit씩 4개의 필드로 분리 표기하는 IPv4(Internet Protocol version 4)가 사용됨
- IP 주소 클래스(Class)

A class	<ul style="list-style-type: none"> ▪ 대형 기관 및 기업에서 사용 ▪ $2^{24}(=16,777,216)$ 중 16,777,214 개의 호스트 사용 가능
B class	<ul style="list-style-type: none"> ▪ 중형 기관 및 기업에서 사용 ▪ $2^{16}(=65,536)$ 중 65,534 개의 호스트 사용 가능
C class	<ul style="list-style-type: none"> ▪ 소형 기관 및 기업에서 사용 ▪ $2^8(=256)$ 중 254 개의 호스트 사용 가능
D class	▪ 멀티캐스트용
E class	▪ 실험용

164

도메인 이름(Domain Name)

- 사용자들이 쉽게 인식하고 사용할 수 있도록 숫자로 표기된 주소를 문자 형태로 표시하는 인터넷 주소
- DNS(Domain Name System) : IP 주소와 호스트 이름 간의 변환을 제공하는 분산 데이터베이스

165

인터넷워킹 장비

- 리피터(Repeater) : 신호가 약해지거나 왜곡될 경우 원래의 신호로 재생하여 재송신하는 장비
- 브리지(Bridge) : 두 개의 LAN이 데이터 링크 계층에서 서로 결합되어 있는 경우에 이들을 연결하는 장비
- 라우터(Router) : 서로 다른 형태의 네트워크를 상호 접속하는 3계층 장비
- 게이트웨이(Gateway) : 서로 다른 프로토콜을 사용하는 네트워크 연결

166

프로토콜(Protocol)

- 정보통신을 위해 통신을 원활하게 수행할 수 있도록 해주는 통신 규약
- 프로토콜의 기본 요소 : 구문(Syntax), 의미(Semantic), 타이밍(Timing)
- 프로토콜의 기능 : 동기 제어, 단편화/재결합, 요약화, 흐름 제어, 주소 지정, 경로 제어, 에러 제어, 다중화

167

OSI 참조모델

- 서로 다른 시스템 간의 원활한 통신을 위해 ISO(국제표준화기구)에서 제안한 통신 규약
- OSI 7계층(Layer) 구조 : 물리 계층(Physical Layer) → 데이터 링크 계층(Data Link Layer) → 네트워크 계층(Network Layer) → 전송 계층(Transport Layer) → 세션 계층(Session Layer) → 표현 계층(Presentation Layer) → 응용 계층(Application Layer)

168

TCP/IP(Transmission Control Protocol / Internet Protocol)

- 인터넷에서 사용하고 있는 프로토콜로서 서로 다른 기종의 컴퓨터들 간에 데이터 송·수신이 가능하도록 해주는 표준 프로토콜

- TCP 프로토콜과 IP 프로토콜의 결합적 의미로써 TCP가 IP보다 상위층에 존재함
- 특징 : 접속형 서비스, 전이중 전송 서비스, 신뢰성 서비스 제공
- TCP/IP의 계층 구조 : 링크 계층(Link Layer) → 네트워크 계층(Network Layer) → 전송 계층(Transport Layer) → 응용 계층(Application Layer)

169

X.25 프로토콜

- 패킷망으로 정보를 전송할 때 패킷 터미널을 제안한 표준 규격안
- 흐름 및 오류 제어기능을 제공함
- 연결형 네트워크 프로토콜임
- X.25의 계층 구조 : 물리 계층(Physical Layer) → 링크 계층(Link Layer) → 패킷 계층(Packer Layer)