计算机的概念

定义：存储程序与程序控制

功能：传输、存储和处理信息

指令编码：

指令也是数据，在计算机中 CPU 的指令形式受硬件条件制约。

指令编码格式：操作码——数据码

操作码：指明所要完成何种操作变换

数据码：参与操作变换的数据

指令集：某种 CPU 所有指令的集合，反映这种 CPU 的功能。

RISC：精简指令集

CSIC：复杂指令集

三极管的放大、开关作用

小规模集成电路：逻辑门电路

中规模集成电路：组合、时序电路

中规模集成电路：CPU 逻辑器件

组合逻辑电路——全加器、译码/编码器、三态门等；

全加器：完成一位二进制数的运算；

译码/编码器：编码信号转换；

三态门：控制线路连接状态

时序逻辑电路——触发器：“保持”二进制一个位，能快速触发“翻转”改变状态

计算机 CPU 实现器件——加法器、寄存器、计数器等；

加法器：多位二进制数的运算（结合寄存器）

寄存器：触发器阵列，保存多位二进制数；

计数器：具有自动进位功能的寄存器；

译码器：将指令编码转换为控制信号。

微型计算机硬件基本结构由运算器、控制器、存储器、输入设备和输出设备组成。

其中运算器和控制器合称为中央处理器 CPU。

微型计算机的结构采用总线结构来实现相互之间的信息传送。单一总线结构使微型机结构简单，易于扩充，奠定了产品标准化、模块化基础。

CPU 和存储器通过总线相互连接，输入/输出设备通过 I/O 接口，挂接在总线上。

总线是 CPU、内存储器和 I/O 接口（以上三者简称主机，I/O 设备简称外设）之间相互交换信息的公共通路，其工作方式采用分时方式。

总线由数据总线、地址总线和控制总线组成。

数据总线是从微处理器向内存储器、I/O 接口传送数据的通路，同时它也是从内存储器、I/O 接口向微处理器传送数据的通路，因为它可以在两个方向往返传送数据，称为**双向**总线。

地址总线是微处理器向内存储器和 I/O 接口传送地址信息的通路，它是单向总线，只能从微处理器向外传送

控制总线是微处理向内存储器和 I/O 接口传送的命令信号以及外界向微处理器传送状态信号等信息的通路

CPU 引脚组成 三总线 电源（VCC GND）时钟信号（CLK）

CPU 功能：

指令控制：程序的顺序控制，称为指令控制。

操作控制：管理并产生由内存取出的每条指令的操作信号，把各种操作信号送往相应的部件，从而控制这些部件按指令的要求进行动作。

时间控制：对各种操作实施时间上的定时，称为时间控制。

数据加工：所谓数据加工，就是对数据进行算术运算和逻辑运算处理。

CPU 组成：

传统 CPU 的组成：运算器、控制器。（在诺曼机的定义中）

现代的 CPU 的基本部分有：运算器、控制器和 Cache（高速缓冲存储器）

控制器的组成：程序计数器（PC）、指令寄存器（IR）、指令译码器、时序产生器和操作控制器

控制器的主要功能：

从内存中取出一条指令，并指出下一条指令在内存中的位置。

对指令进行译码或测试，并产生相应的操作控制信号，以便启动规定的动作。

指挥并控制 CPU、内存和输入 / 输出设备之间数据流动的方向。

运算器的组成：算术逻辑单元（ALU）、累加寄存器（AC）、数据缓冲寄存器和状态条件寄存器（PSW）。

运算器的主要功能：

执行所有的算术运算。

执行所有的逻辑运算，并进行逻辑测试。

指令：计算机的程序是由一系列的机器指令组成的。指令就是要计算机执行某种操作的命令

一台计算机中所有机器指令的集合，称为这台计算机的指令系统。

系列计算机，是指基本指令系统相同、基本体系结构相同的一系列计算机。其必要条件是同一系列的各机种有共同的指令集．而且新推出的机种指令系统一定包含所有旧机种的全部指令，即实现一个“向上兼容”

一个完善的指令系统应满足如下四方面的要求：

（1）**完备性**是指用汇编语言编写各种程序时，指令系统直接提供的指令足够使用，而不必用软件来实现。

（2）**有效性**是指利用该指令系统所编写的程序能够高效率地运行。

（3）**规整性**包括指令系统的对称性、匀齐性、指令格式和数据格式的一致性。

（4）**兼容性**至少要能做到“向上兼容”，即低档机上运行的软件可以在高档机上运行。

操作码

指令的操作码表示该指令应进行什么性质的操作。组成操作码字段的位数一般取决于计算机指令系统的规模。

地址码

根据一条指令中有几个操作数地址，可将该指令称为几操作数指令或几地址指令。目前二地址和一地址指令格式用的得最多。

零地址指令的指令子中只有操作码，而没有地址码

从操作数的物理位置（所谓地址）来说，又可归结为三种类型：

1、存储器－存储器（SS）慢

2、寄存器－寄存器（RR）快

3、寄存器－存储器（RS）较快

指令字长度：一个指令字中包含二进制代码的位数，称为指令字长度。

指令和数据的寻址方式：当采用地址指定方式时，形成操作数或指令地址的方式，称为寻址方式。寻址方式分为两类，即指令寻址方式和数据寻址方式

指令寻址的基本方式有两种，一种是**顺序寻址方式**，另一种是**跳跃寻址方式**。

指令周期：是取出并执行一条指令的时间。由于各种指令的操作功能不同，有的简单，有的复杂，因此各种指令的指令周期是不尽相同的。

机器周期：指令周期常常用若干个 CPU 周期数来表示，CPU 周期也称为机器周期。通常用内存中读取一个指令字的最短时间来规定 CPU 周期。

时钟周期：一个 CPU 周期时间又包含有若干个时钟周期。由传入 CPU CLK 引脚的方波周期信号确定。

一个指令周期由若干（数目不固定）机器周期组成；

一个机器周期由若干（数目固定）时钟周期组成。

PC——指令自动执行的心脏

地址暂存寄存器 AR——指明存取指令或数据在存储器中的地址

存储器——为 CPU 传送（提供）指令或数据

数据暂存寄存器 DR——存放由 CPU 得到的指令或数据

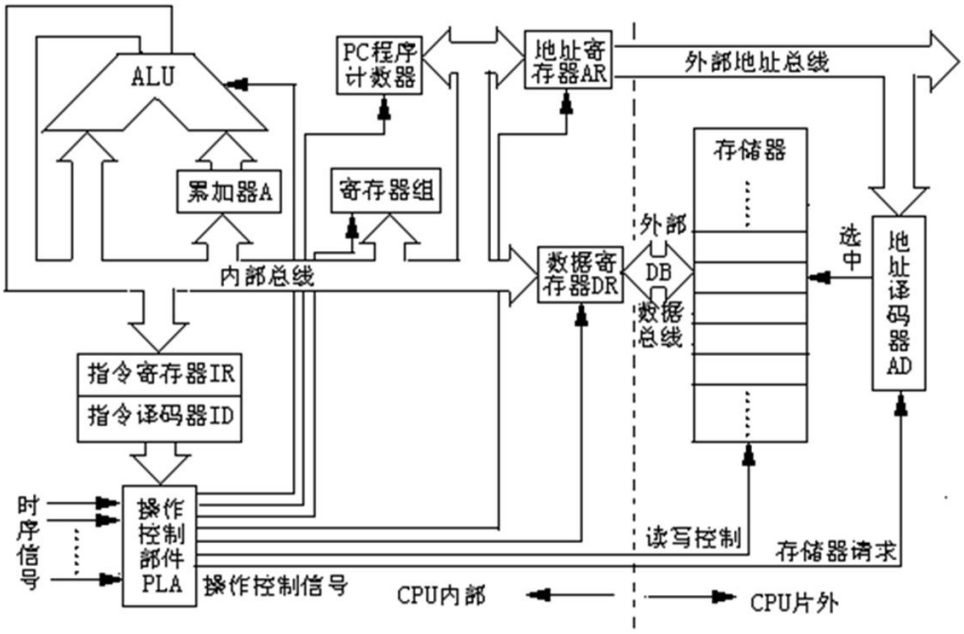
指令寄存器 IR——锁存指令，在该指令的执行周期内，其值保持不变

ALU——完成算术逻辑运算

地址总线——将 AR 指明的地址信息的数据与 CPU 进行交换（包括存储器与外设接口）

数据总线——与 CPU 进行交换（包括存储器与外设接口）数据

CPU逻辑结构图



超标量技术是一种在单个处理器内核中实现指令级并行的技术。它允许 CPU 在一个时钟周期内执行多条独立的指令，从而进一步提高 CPU 的吞吐率。

SIMD（单指令流多数据流）技术是一种实现数据级并行的技术。它允许一条指令同时对多个数据进行相同的操作，这在处理向量和矩阵运算时尤为高效。

按存储方式分

随机存储器：任何存储单元的内容都能被随机存取，且存取时间和存储单元的物理位置无关。

顺序存储器：只能按某种顺序来存取，存取时间和存储单元的物理位置有关

按存储器的读写功能分

随机只读存储器(ROM)：存储的内容是固定不变的，只能读出而不能写入的半导体存储器。

随机读写存储器(RAM)：既能读出又能写入的半导体存储器。

按信息的可保存性分

非永久记忆的存储器：断电后信息即消失的存储器。

永久记忆性存储器：断电后仍能保存信息的存储器

静态随机存储器是有自保持功能的 RAM。

动态随机存储器是需要刷新存储器

物理存储器是指实际存在的具体存储器芯片。

存储地址空间是指对存储器编码（编码地址）的范围。所谓编码就是对每一个物理存储单元（一个字节）分配一个号码，通常叫作“编址”。

地址空间的大小和物理存储器的大小并不一定相等

时间局部性（temporal locality）：最近的访问项（指令或数据）很可能在不久的将来再次被访问。即对最近使用区域的集中访问。

空间局部性（spatial locality）：一个进程访问的各项的地址彼此很近，例如，表操作或数组操作含对地址空间中某一区域的集中访问。

高速缓冲存储器缓存（cache）：是 CPU 内存的一部分容量较小但速度较快的存储器，用来解决 CPU 与内存读写不匹配的问题

以空间换取时间，但空间牺牲不大。

缓存替换算法：

- 最近最少使用（LRU）
- 先进先出（FIFO）
- 用得最少（LFU）

虚拟存储器：当内存不够用时，用硬盘中的一部分空间来充当内存。

虚拟存贮器特性：

- 具有请求调入功能和置换功能，能从逻辑上对内存容量进行扩充的一种存储系统。
- 实质：以时间换空间，但时间牺牲不大。
- 虚拟存储的实现需要依靠硬件的支持，对于不同的 CPU 来说是不同的。但是几乎所有的硬件都采用一个叫 MMU（Memory Management Unit）的部件来进行页映射

虚拟存储器的实现方式：

需要动态重定位

请求分页系统

请求分段系统

虚存特征：

离散性：部分装入（若连续则不可能提供虚存），无法支持大作业小内存运行

多次性：局部装入，多次装入。

对换性：

虚拟性.

I/O 接口的功能与类型：

① 对输入/输出数据进行缓冲、隔离和锁存。

② 信号转换。

③ I/O 端口提供寻址功能。

④ 为 CPU 和 I/O 设备之间提供联络。

总之，I/O 接口的功能就是完成数据、地址和控制三总线的转换和连接

数据缓冲寄存器。它分为输入缓存器和输出缓存器两种。前者的作用是将外设送来的数据暂时存放，以便处理器将它取走；后者的作用是用来暂时存放处理器送往外设的数据。

控制寄存器。控制寄存器用于存放处理器发来的控制命令和其他信息，以确定接口电路的工作方式和功能。控制寄存器是只写寄存器，其内容只能由处理器写入，而不能读出。

状态寄存器。状态寄存器用于保存外设现行各种状态信息。它的内容可以被处理器读出，从而使处理器了解外设状况及数据传送过程中正在发生或最近已经发生的事情，供处理器做出正确的判断，使它能安全可靠地与接口完成交换数据的各种操作。

其他辅助电路：

数据总线和地址总线缓冲器。

端口地址译码器。

内部控制逻辑。

对外联络控制逻辑。

I/O 接口的编址方式：

I/O 端口与存储器单元统一编址，如 M6800 的做法；

I/O 端口独立编址，如 8086/8088 的做法。

I/O 端口独立编址：所谓 I/O 端口独立编址，也称为 I/O 隔离编址或 I/O 指令寻址方式，即 I/O 端口地址区域和存储器地址区域，分别各自独立编址。访问 I/O 端口使用专门的 I/O 指令(**IN、OUT**)，而访问内存则使用 MOV、ADD 等指令

外部设备的工作方式：

无条件传送——传送前，CPU 不需要了解接口的状态，直接传送数据。

查询传送——传送前，CPU 先查询接口的状态，接口就绪后传送。

中断传送——传送请求由外设提出，CPU 视情况响应后，调用预先安排好的子程序来完成数据传送。

直接存储器存取——传送请求由外设向 DMA 控制器提出，后者向 CPU 申请总线，DMAC 利用系统总线完成外设和存储器间的数据传送。该传送完全由硬件实现，具有非常高的传送速率。

采用 I/O 处理机进行数据的传送和处理——CPU 委托专门的 I/O 处理机管理外设，完成传送和相应的数据处理。

中断：CPU 执行程序时，由于发生了某种随机的事件(**外部**或**内部**)，引起 CPU 暂时中断正在运行的程序，转去执行一段特殊的服务程序(称为中断服务程序或中断处理程序)，以处理该事件，该事件处理完后又返回被中断的程序继续执行，这一过程称为**中断**。

中断过程五个步骤：

① 中断请求

② 中断判优(有时还要进行中断源识别)

③ 中断响应

④ 中断服务

⑤ 中断返回

DMA：不经 CPU 的干预，而是在专用硬件电路的控制下直接传送。这种方法称为直接存储器存取，缩写为 DMA。

算法：算法是一组有穷的规则，它们规定了解决某一特定类型问题的一系列运算，是对解题方案的准确与完整的描述

算法的特点：

① **有穷性**：一个算法对任何合法的输入值必须总是在执行有穷步之后结束，并且每步都必须在有穷时间内完成。（利用这一点，可以区别算法与程序。）

② **确定性**：算法中每一条指令必须有确切的含义，不会产生二义性。

③ **可行性**：算法中描述的操作都必须可以通过已经实现的基本运算有限次执行来实现，而且算法必须在有限时间内完成。

④ **输入**：一个算法必须有零个或多个输入量。

⑤ **输出**：一个算法应有一个或多个输出量，输出量是算法计算的结果。

算法的设计要求：

① **有穷性**：一个算法对任何合法的输入值必须总是在执行有穷步之后结束，并且每步都必须在有穷时间内完成。（利用这一点，可以区别算法与程序。）

② **确定性**：算法中每一条指令必须有确切的含义，不会产生二义性。

③ **可行性**：算法中描述的操作都必须可以通过已经实现的基本运算有限次执行来实现，而且算法必须在有限时间内完成。

④ **输入**：一个算法必须有零个或多个输入量。

⑤ **输出**：一个算法应有一个或多个输出量，输出量是算法计算的结果。

时间效率。考虑算法所耗费的运行时间。
算法的执行时间用时间复杂度表示；

$T(n) = O(f (n))$

空间效率。考虑运行算法需要耗费的存储空间，**算法执行过程中所占的临时空间与算法有着密切关系。**这由空间复杂度来衡量

$S（n）= O（f（n））$

算法的表示：

自然语言

流程图

伪代码

计算机程序设计语言

分治法：字面上的解释是“分而治之”，就是把一个复杂的问题分成两个或更多的相同或相似的子问题，再把子问题分成更小的子问题……直到最后子问题可以简单的直接求解，原问题的解即子问题的解的合并

动态规划：其基本思想是：将原问题分解为相似的子问题，在求解的过程中通过子问题的解求出原问题的解。

贪心法：最优子结构的意思是局部最优解能决定全局最优解。简单地说，问题能够分解成子问题来解决，子问题的最优解能递推到最终问题的最优解。

递归：指的是直接或间接调用自身。

迭代：迭是屡次和反复的意思代是替换的意思合起来迭代就是反复替换的意思。

数据结构：数据结构是指相互间存在一种或多种特定关系的数据元素的集合

逻辑结构是对给定数据结构的一种描述方式，是从实际问题抽象出来的数据模型

数据结构分为四类：

集合结构：数据元素之间除了同属于一个集合之外，没有其他关系的数据结构。**集合结构中的任何一个元素都有 0 个前驱，0 个后继**

线性结构：数据元素之间存在“一对一”线性关系的数据结构。**线性结构中的任何一个元素都有 0 或 1 个前驱，0 或 1 个后继。**

树型结构：数据元素之间存在**“一对多”**逻辑关系的数据结构。**树型结构中的任何一个元素都有 0 或 1 个前驱，0 或多个后继**

图型结构：数据元素之间存在**“多对多”**逻辑关系的数据结构。**图型结构中中的任何一个元素都有 0 或多个前驱，0 或多个后继**

访问受限的线性结构：

栈——对它的操作集中在线性结构的一端（压栈 PUSH、出栈 POP）

后进先出，先进后出

队列——对它的操作集中在线性结构的两端（队首出队，队尾入队）

先进先出，后进后出

存储结构又称物理结构，就是数据结构在计算机中的存储方式。它包括数据元素在计算机中的存储方式，还包括元素之间关系在内存中的表示

存储空间的不同分配方式将数据结构分为两类：

顺序存储：所有元素存放在一片连续的存储空间中，逻辑上相邻的元素在内存中的物理位置也是相邻的。

链式存储：所有元素存放在可以不连续的存储单元中，以结点的形式存在，每个结点除了保存数据元素信息外，还通过指针来保存元素之间的关系。逻辑相邻物理不一定相邻。

逻辑结构操作：

1、结构的生成；

2、结构的销毁；

3、在结构中查找满足规定条件的数据元素；

4、在结构中插入新的数据元素；

5、删除结构中已经存在的数据元素；

6、遍历。

操作系统：操作系统是控制和管理计算机硬件和软件资源、合理地组织计算机工作流程以及方便用户有效地使用计算机的程序集合

操作系统特点:

硬件相关、应用无关

核心常驻内存

中断驱动

权威性

庞大、复杂

重要性（无处不在、无时不有）

并发、共享、虚拟、异步（这是操作系统的四个基本特征）

并发性——宏观并行，微观串行

在多道程序环境下，并发性是指两个或多个事件在同一时间间隔内发生，即宏观上有多道程序同时执行，而微观上，在单处理机系统中每一个时刻仅能执行一道程序。

共享性

共享是指系统中的资源可供多个并发执行的进程使用。

虚拟性 ——物理一个，逻辑多个

是指通过某种技术把一个物理实体变成若干个逻辑上的对应物。

异步性

也称不确定性，是指在多道程序环境下，允许多个进程并发执行，由于资源的限制，进程的执行不是“一气呵成”的，是“走走停停”的

操作系统的目标

1．不断提高资源利用率的需要

2．方便用户操作

3．适应硬件的不断更新换代 ——接口标准

4．计算机体系结构的不断发展

操作系统历史：

1. 手工操作阶段（40 年代）
2. 单道批处理阶段（50 年代）
3. 多道批处理（60 年代初）
4. 分时系统（60 年代中）
5. 实时操作系统（60 年代中）
6. 网络操作系统（80 年代中）

进程的特征

动态性：进程具有动态的地址空间（数量和内容），地址空间上包括：

代码（指令执行和 CPU 状态的改变）

数据（变量的生成和赋值）

系统控制信息（进程控制块的生成和删除）

独立性：各进程的地址空间相互独立，除非采用进程间通信手段；

并发性、异步性："虚拟"

结构化：代码段、数据段和核心段（在地址空间中）；程序文件中通常也划分了代码段和数据段，而核心段通常就是 OS 核心（由各个进程共享，包括各进程的 PCB）

进程控制块(PCB, process control block)：**进程控制块**是由 OS 维护的用来记录进程相关信息的一块内存

进程控制块的内容：

进程描述信息：

进程标识符(process ID)，唯一，通常是一个整数；

进程名，通常基于可执行文件名（不唯一）；

用户标识符(user ID)；进程组关系(process group)

进程控制信息：

当前状态;

优先级(priority);

代码执行入口地址;

程序的外存地址;

运行统计信息（执行时间、页面调度）;

进程间同步和通信；阻塞原因

资源占用信息:

虚拟地址空间的现状、打开文件列表

CPU 现场保护结构:

寄存器值（通用、程序计数器 PC、状态 PSW，地址包括栈指针）

进程与程序的区别:

进程是动态的，程序是静态的：程序是有序代码的集合；进程是程序的执行。通常进程不可在计算机之间迁移；而程序通常对应着文件、静态和可以复制。

进程是暂时的，程序是永久的：进程是一个状态变化的过程，程序可长久保存。

进程与程序的组成不同：进程的组成包括程序、数据和进程控制块（即进程状态信息）。

进程与程序的对应关系：**通过多次执行**，一个程序可对应多个进程；**通过调用关系**，一个进程可包括多个程序

操作系统功能:

处理机管理——进程管理——充分利用

存储管理——方便多进程共享

设备管理——与处理机并行

文件管理——组织、存储、保护

作业管理——为用户提供一个使用系统的良好环境

系统调用：系统调用就是用户程序对操作系统的调用

系统调用包括：

外存文件与目录的读写

各种 I/O 设备的使用

在一个程序中启动另一个程序

查询和统计系统资源使用情况

程序的执行有两种方式：

顺序执行：顺序执行是单道批处理系统的执行方式，也用于简单的单片机系统；

并发执行：现在的操作系统多为并发执行，具有许多新的特征。引入并发执行的目的是为了提高资源利用率。

顺序执行的特征

顺序性：按照程序结构所指定的次序（可能有分支或循环）

封闭性：独占全部资源，计算机的状态只由于该程序的控制逻辑所决定

可再现性：初始条件相同则结果相同。如：可通过空指令控制时间关系

并发执行的特征

间断(异步)性："走走停停"，一个程序可能走到中途停下来，失去原有的时序关系；

失去封闭性：共享资源，受其他程序的控制逻辑的影响。如：一个程序写到存储器中的数据可能被另一个程序修改，失去原有的不变特征。

失去可再现性：失去封闭性 ->失去可再现性；外界环境在程序的两次执行期间发生变化，失去原有的可重复特征。

进程分类：

用户进程：运行在目态（用户态），用户态时不可直接访问受保护的 OS 代码。

系统进程：运行在管态（核心态），核心态时执行 OS 代码，可以访问全部进程空间。

处理机调度器是操作系统中的一段代码，它完成如下功能：

把处理机从一个进程切换到另一个进程;

防止某进程独占处理机。

时间片即 CPU 分配给各个程序的时间,每个进程被分配一个时间段，称作它的时间片，即该进程允许运行的时间，使各个程序从表面上看是同时进行的。

进程的状态转换：两状态进程模型 五状态进程模型 挂起进程模型

两状态进程模型：

运行状态(Running)：占用处理机资源;

暂停状态(Not-Running)：等待进程调度分配处理机资源;

五状态进程模型：

运行状态(Running)：占用处理机资源；处于此状态的进程的数目小于等于 CPU 的数目

就绪状态(Ready)：进程已获得除处理机外的所需资源，等待分配处理机资源；只要分配 CPU 就可执行。

阻塞状态(Blocked)：由于进程等待某种条件（如 I/O 操作或进程同步），在条件满足之前无法继续执行。该事件发生前即使把处理机分配给该进程，也无法运行。如：等待 I/O 操作的完成

创建状态(New)：进程刚创建，但还不能运行(一种可能的原因是 OS 对并发进程数的限制)；如：分配和建立 PCB 表项（可能有数目限制）、建立资源表格（如打开文件表）并分配资源，加载程序并建立地址空间表。

结束状态(Exit)：进程已结束运行，回收除 PCB 之外的其他资源，并让其他进程从 PCB 中收集有关信息（如记帐，将退出码 exit code 传递给父进程）。

挂起进程模型（七状态）：

运行状态

创建状态

就绪状态(Ready)：进程在内存且可立即进入运行状态；

阻塞状态(Blocked)：进程在内存并等待某事件的出现；

阻塞挂起状态（Blocked, suspend）：进程在外存并等待某事件的出现；

就绪挂起状态（Ready, suspend）：进程在外存，但只要进入内存，即可运行；

终止状态

两状态模型转换：

进程创建(Enter)：系统创建进程，形成 PCB，分配所需资源，排入暂停进程表(可为一个队列)；

调度运行(Dispatch)：从暂停进程表中选择一个进程(要求已完成 I/O 操作)，进入运行状态；

暂停运行(Pause)：用完时间片或启动 I/O 操作后，放弃处理机，进入暂停进程表；

进程结束(Exit)：进程运行中止；

五状态模型转换：

创建新进程：创建一个新进程，以运行一个程序。可能的原因为：用户登录、OS 创建以提供某项服务、批处理作业。

收容(Admit，也称为提交)：收容一个新进程，进入就绪状态。由于性能、内存、进程总数等原因，系统会限制并发进程总数

调度运行(Dispatch)：从就绪进程表中选择一个进程，进入运行状态；

释放(Release)：由于进程完成或失败而中止进程运行，进入结束状态；

运行到结束：分为正常退出 Exit 和异常退出 abort（执行超时或内存不够，非法指令或地址，I/O 失败，被其他进程所终止）

就绪或阻塞到结束：可能的原因有：父进程可在任何时间中止子进程；

超时（Timeout）：由于用完时间片或高优先进程就绪等导致进程暂停运行；

事件等待（Event Wait）：进程要求的事件未出现而进入阻塞；可能的原因包括：申请系统服务或资源、通信、I/O 操作等；

事件出现（Event Occurs）：进程等待的事件出现；如：操作完成、申请成功等；

七状态模型转换:

挂起（Suspend）：把一个进程从内存转到外存；可能有以下几种情况：

阻塞到阻塞挂起：没有进程处于就绪状态或就绪进程要求更多内存资源时，会进行这种转换，以提交新进程或运行就绪进程；

就绪到就绪挂起：当有高优先级阻塞（系统认为会很快就绪的）进程和低优先级就绪进程时，系统会选择挂起低优先级就绪进程；

运行到就绪挂起：对抢先式分时系统，当有高优先级阻塞挂起进程因事件出现而进入就绪挂起时，系统可能会把运行进程转到就绪挂起状态；

激活（Activate）：把一个进程从外存转到内存；可能有以下几种情况：

就绪挂起到就绪：没有就绪进程或挂起就绪进程优先级高于就绪进程时，会进行这种转换；

阻塞挂起到阻塞：当一个进程释放足够内存时，系统会把一个高优先级阻塞挂起（系统认为会很快出现所等待的事件）进程；

事件出现（Event Occurs）：进程等待的事件出现；如：操作完成、申请成功等；可能的情况有：

阻塞到就绪：针对内存进程的事件出现；

阻塞挂起到就绪挂起：针对外存进程的事件出现；

收容(Admit)：收容一个新进程，进入就绪状态或就绪挂起状态。进入就绪挂起的原因是系统希望保持一个大的就绪进程表（挂起和非挂起）；

五状态模型：两状态模型无法区分暂停进程表中的可运行和阻塞，五状态模型就是对暂停状态的细化。

挂起进程模型（七状态）：

由于进程优先级的引入，一些低优先级进程可能等待较长时间，从而被对换至外存。这样做的目的是：

提高处理机效率：就绪进程表为空时，要提交新进程，以提高处理机效率；

为运行进程提供足够内存：资源紧张时，暂停某些进程，如：CPU 繁忙，内存紧张

用于调试：在调试时，挂起被调试进程（从而对其地址空间进行读写）

进程调度的任务主要有：

保存处理机的现场信息。在进行调度时首先需要保存当前进程的处理机的现场信息，如程序计数器、多个通用寄存器中的内容等。

按某种算法选取进程。调度程序按某种算法从就绪队列中选取一个进程，将其状态改为运行状态，并准备把处理机分配给它。

把处理器分配给进程。由分派程序把处理器分配给该进程，此时需要将选中进程的进程控制块内有关处理机现场的信息装入处理器相应的各个寄存器中，把处理器的控制权交予该进程，让它从上次的断点处恢复运行

存储器分系统区和用户区：OS 工作在系统区，应用程序工作在用户区

为防止多道程序之间的相互干扰，用户区得进行分区：段式 页式 段页式

线程——程序中一个单一的顺序控制流程。在单个程序中同时运行多个线程完成不同的工作，称为多线程。

线程特点：

只拥有必不可少的资源，如：线程状态、寄存器上下文和栈。

同样具有就绪、阻塞和执行三种基本状态。

引入线程的目的是将进程间的多个程序执行流并发，转换为进程内的多个程序执行流并发（多线程），简化线程间的通信，以小的开销来提高进程内的并发程度。

OS 的三类模型：

进程模型：进程既是资源分配单位（存储器、文件），也是 CPU 调度（分派）单位。

进/线程模型：线程是 CPU 调度单位，而进程只作为其他资源分配单位。

协程，又称微线程，纤程，协程是一种用户态的轻量级线程。

线程的优点：

减小并发执行的时间和空间开销（线程的创建、退出和调度），因此容许在系统中建立更多的线程来提高并发程度。

线程的创建时间比进程短；

线程的终止时间比进程短；

同进程内的线程切换时间比进程短；

由于同进程内线程间共享内存和文件资源，可直接进行不通过内核的通信；

进程和线程的比较：

地址空间和其他资源（如打开文件）：进程间相互独立，同一进程的各线程间共享——某进程内的线程在其他进程不可见；

通信：进程间通信 IPC，线程间可以直接读写进程数据段（如全局变量）来进行通信——需要进程同步和互斥手段的辅助，以保证数据的一致性；

调度：线程上下文切换比进程上下文切换要快得多。

计算机网络：将地理位置不同的具有独立功能的多台计算机及其外部设备，通过通信线路连接起来，在网络操作系统，网络管理软件及网络通信协议的管理和协调下，实现资源共享和信息传递的计算机系统。

计算机的两大功能：数据通信和资源共享

连通性——计算机网络使上网用户之间都可以交换信息，好像这些用户的计算机都可以彼此直接连通一样。

共享——即资源共享。可以是信息共享、软件共享，也可以是硬件共享。

网络协议(network protocol)，简称为**协议**，是为进行网络中的数据交换而建立的规

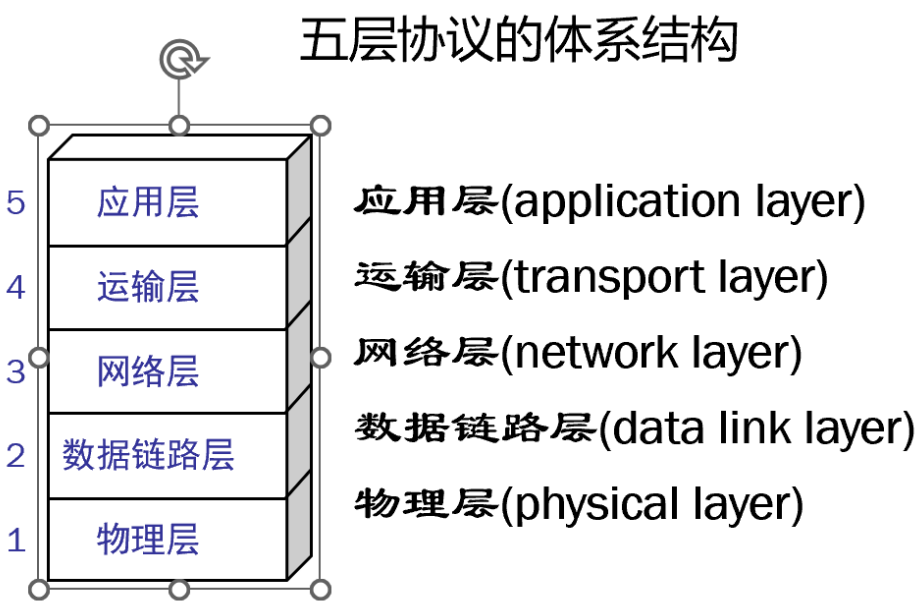
则、标准或约定

网络协议三要素:

语法:数据与控制信息的结构或格式 。

语义:需要发出何种控制信息，完成何种动作以及做出何种响应。

同步（时序）:事件实现顺序的详细说明。



数据(data)——运送消息的实体。

信号(signal)——数据的电气的或电磁的表现。

模拟的”(analogous)——代表消息的参数的取值是连续的。

“数字的”(digital)——代表消息的参数的取值是离散的。

码元(code)——在使用时间域（或简称为时域）的波形表示数字信号时，代表不同离散数值的基本波形。

单向通信（单工通信）——只能有一个方向的通信而没有反方向的交互。

双向交替通信（半双工通信）——通信的双方都可以发送信息，但不能双方同时发送(当然也就不能同时接收)。

双向同时通信（全双工通信）——通信的双方可以同时发送和接收信息。

电缆分类：

非屏蔽双绞线 UTP

屏蔽双绞线 STP

同轴电缆

光纤

OSI/RM 体系结构从下到上为：物理层 数据链路层 网络层 传输层 会话层 表示层 应该层

物理层 传输原始比特流

数据链路层 传输数据帧

网络层 传输数据包

传输层 传输数据段

会话层 表示层 应用层 数据报文

数据链路层使用的信道主要有以下两种类型：

点对点信道。这种信道使用一对一的点对点通信方式。

广播信道。这种信道使用一对多的广播通信方式，因此过程比较复杂。广播信道上连接的主机很多，因此必须使用专用的共享信道协议来协调这些主机的数据发

数据链路层的三个基本功能：

(1) 封装成帧

(2) 透明传输

(3) 差错控制

数据链路层有

两个子层：逻辑链路控制 LLC (Logical Link Control)子层

媒体接入控制 MAC (Medium Access Control)子层。

网络接口板又称为通信适配器(adapter)或网络接口卡 NIC (Network Interface Card)，或“网卡”

网络适配器的重要功能：

进行串行/并行转换。

对数据进行缓存。

在计算机的操作系统安装设备驱动程序。

实现以太网协议。

MAC 地址又称为物理地址 共有 48 位

CSMA/CD：

多点接入

载波监听

碰撞检测

网络层向上只提供简单灵活的、无连接的、尽最大努力交付的数据报服务。

网络层主要协议：

IP 协议：网际协议

ARP 地址解析协议

RARP 反向地址解析协议

ICMP 网际报文协议

RIP 距离矢量路由协议 是一种内部网关协议

BGP 外部网关协议

传输控制协议 TCP 面向连接的 可靠的 三次握手 四次挥手

用户数据包协议 UDP 面向非连接 不可靠

TCP 数据传输利用滑动窗口实现流量控制

URL 统一资源定位符

HTTP 超文本传输协议

文件是按名存取的数据的集合。

文件系统

文件系统：文件与管理文件的程序集合。

它是操作系统的一个子系统。常见的文件系统有：FAT16、FAT32、NTFS、EXT2、 …

文件包括两部分：

文件体：文件本身的信息；

文件说明：文件存储和管理信息；如：文件名、文件内部标识、文件存储地址、访问权限、访问时间等。

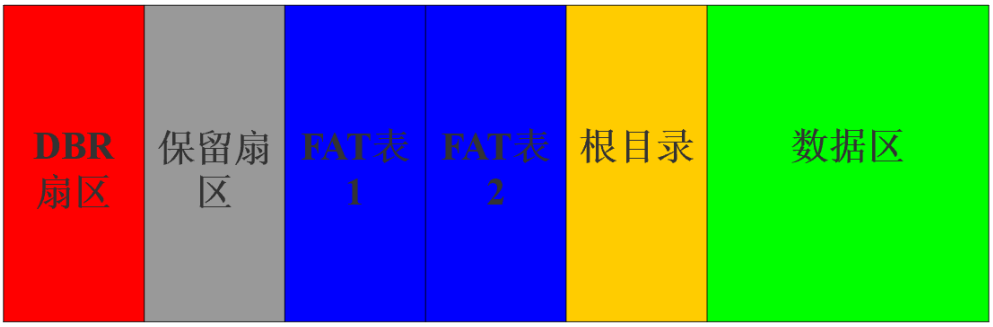
文件分类：普通文件 目录文件 特殊文件

文件访问方式 顺序访问 随机访问

由于基本磁盘块单位扇区（512 字节）粒度太小，一般将若干扇区组成一个较大的块单位“簇”（依据磁盘系统的不同大小从 4K 字节到几十 K 字节不等），以方便管理。

举例：FAT16文件物理组织

硬盘上的数据按照其不同的特点和作用大致可分为5部分：MBR区、DBR区、FAT区、DIR区和DATA区。如下图：



第一代语言 机器语言

第二代语言 汇编语言

第三代语言 高级语言（面向过程）

第四代语言（非过程化语言）

第五代语言（智能化语言）

高级语言的翻译程序有两种 解释和编译

泛型编程 数据类型参数化

编译逻辑过程

1. 词法分析：从左至右读源程序的字符流、识别(拼)单词

2. 语法分析：层次分析.依据源程序的语法规则把源程序的单词序列组成语法短语(表示成语法树).
3. 语义分析：上下文相关性 类型匹配 类型转换

4. 中间代码生成 源程序的内部(中间)表示
5. 代码优化
6. 目标代码生成

数据处理就是将数据信息转换的过程。数据处理的内容主要包括：数据的收集、整理、存储、加工、分类、维护、排序、检索和传输等一系列活动的总和。

数据库系统的组成

计算机硬件

数据库管理系统

数据库

应用程序

数据库用户

模式：模式又称概念模式或逻辑模式，对应于概念级。它是由数据库设计者综合所有用户的数据，按照统一的观点构造的全局逻辑结构。

外模式：外模式又称子模式，对应于用户级。它是某个或某几个用户所看到的数据库的数据视图，是与某一应用有关的数据的逻辑表示。

内模式“：内模式又称存储模式，对应于物理级。它是数据库中全体数据的内部表示或底层描述，是数据库最低一级的逻辑描述，它描述了数据在存储介质上的存储方式和物理结构，对应着实际存储在外存储介质上的数据库。(文件的逻辑结构——文件内容——呈特定数据结构)

数据库系统的二级映射：数据库系统的三级模式是数据在三个级别（层次）上的抽象，**使用户能够逻辑地、抽象地处理数据而不必关心数据在计算机中的物理表示和存**

储。（此种功能由 DBMS 实现）

数据库管理系统的功能：

数据库定义（描述）功能

数据库操纵功能

数据库运行管理功能

数据组织、存储和管理

数据库的建立和维护

通信功能

数据库管理系统的组成：

数据定义语言及其编译处理程序

数据操作语言及其编译程序

数据库运行控制程序

实用程序

数据库系统的特点：

数据共享

减少数据冗余

具有较高的数据独立性

物理数据独立

逻辑数据独立

增强了数据安全性和完整性保护

所谓关系数据库就是采用关系模型作为数据的组织方式，换句话说就是支持关系模型的数据库系统。

关系模型由三个部分构成：关系数据结构、关系数据操作和完整性约束。

层次模型（Hierarchical Model）

用树形结构表示实体和实体间联系的数据模型称为层次模型。

网状模型（Network Model）

用网状结构表示实体和实体之间关系的数据模型称为网状模型。

关系模型（Relational Model）

用二维表来表示实体和实体间联系的数据模型称为关系模型。

关系：一个关系就是一张二维表，通常将一个没有重复行、重复列的二维表看成一个关系，每个关系都有一个关系名。

元组：二维表的每一行在关系中称为元组。

属性：二维表的每一列在关系中称为属性，每个属性都有一个属性名，属性值则是各个元组在该属性上的取值。

域：属性的取值范围称为域。

完整性约束条件是关系数据模型的一个重要组成部分，是为了保证数据库中的数据一致性

完整性约束分为三类：

实体完整性：每个数据表都必须有主键。

参照完整性：不允许引用不存在的实体。

用户定义完整性：用户要求的逻辑。

软件的定义

软件=程序+数据+文档。

程序：程序是按事先设计好的功能和性能要求执行的指令序列。

数据：数据是指程序能正常处理信息的数据和数据结构。

文档：文档是与程序运行和维护有关的图文资料。

软件开发流程:

需求分析

软件设计

编码

测试