

Operating System - P1

[内核态：操作系统具有对所有硬件的完全访问权] [User态：只使用机器指令的一个子集，禁止影响机器控制或 I/O 操作的指令]

[内核态] [用户态]

硬件 < 操作系统 < 用户接口程序 < 用户程序

软件

Computer 体系结构：指令集、存储组织、I/O 和总线结构

复杂 → 抽象 抽象的定义与实现，随时用抽象解决问题，如操作系统对文件的抽象，Python 中类 - 实例

OS 的任务 在相互竞争的程序之间有序地控制对处理器(CPU)、存储器(Reg, Cache, memory)以及其他 I/O 接口设备的分配 —— 记录程序在使用什么资源，对资源请求进行分配 —— 评估使用代价，调解相互冲突的资源请求

多路复用 时间上复用：如 CPU 轮流执行多个程序

空间上复用：如 分割内存以存放多个程序

历史 第一代(1945~1955)：真空管和穿孔卡片

第二代(1955~1965)：晶体管和批处理系统(batch system)

第三代(1965~1980)：集成电路和多道程序设计(multiprogramming)

L 方案是分割内存使每一部分存放不同作业

L 分时系统(timesharing)，第一个通用的是兼容分时系统(CTSS)

L MULTICS → UNIX → IEEE 提出 POSIX 标准

第四代(1980~)：个人计算机，LSI(大规模集成)的发展

L CP/M(基于磁盘的操作系统)(Control Program for Microcomputer)

L DOS(Disk Operating System)

L MS-DOS → Win 95(独立的) → Win 98(16位) → Win Me

L Win NT → Win 2000 → Win XP → Win Vista → Win 7

L 网络操作系统和分布式操作系统

第五代(1990~)：移动计算机

L 个人数字助理(Personal Digital Assistant)

L 智能手机(smartphone)

Operating

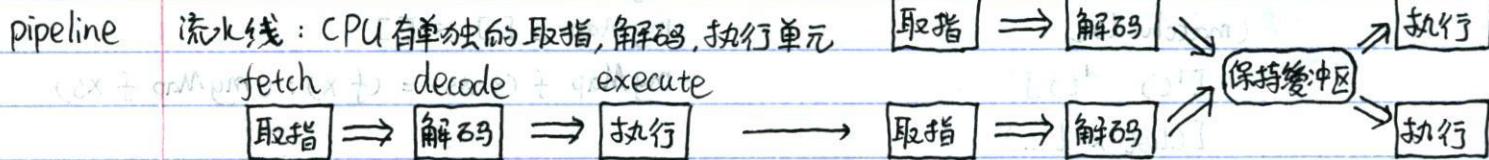
System - P. 2

IS 每个CPU都有一套可执行的专门指令集(Instruction Set),还有寄存器(Register,通用寄存器)

PC 专用寄存器如:程序计数器(Program Counter),保存将要取出的下一条指令的内存地址
堆栈指针,指向内存中当前栈的顶端

PSW 程序状态字(Program Status Word)包含条件码位, CPU优先级, 用户态/内核态

多路复用(time multiplexing)CPU中,每次停止一个程序必须保存所有寄存器值,再次运行时重新装入



超标量(CPU):有多个执行单元,用于不同运算,隐患是程序指令不按顺序执行,硬件保证结果与顺序相同

TRAP 为了从操作系统获得服务,用户程序必须使用系统调用(system call),TRAP指定由用户态切换内核态

多线程(multithreading)或超线程(hyperthreading)允许CPU在进程间以纳秒级的时间尺度切换,但并不是真正的并行处理

CPU 与之相对的是GPU(Graphics Processing Unit),由大量微核构成,擅长大量并行的简单计算

存储器 CPU寄存器(Register) 高速缓存(cache) 主存(Memory) 磁盘(Disk)

典型访问时间 1 ns ~ 10 ns 10 ns ~ 10 ms

典型容量 < 1 KB 4 MB 1~8 GB 1~4 TB

Register 32位(CPU中寄存器为 32×32 位),程序必须自行管理寄存器

Cache 分割为64字节的高速缓存行(cache line),读入字时如果存在于cache中称为高速缓存命中,需两个时钟周期

考虑:何时将新内容放入缓存,放在哪一行,如果需要把旧行内容移走,移走的内容放在何处

现代CPU 多级缓存:每一级比前一级慢但容量更大

L1缓存:总在CPU中,将已解码的指令调入执行,典型容量16 KB

L2缓存:相比L1延时1~2倍时钟周期,存放近来使用过的内容,典型容量4~16 MB



多核芯片中,可能一个L2缓存为所有核共享(Intel),也可能每个核有自己的L2缓存(AMD)

Operating

System - P3

Memory

RAM

ROM

EEPROM

CMOS

Disk

通常称为随机访问存储器 (Random Access Memory, RAM), 过去称磁芯存储器
在高速缓存未满足的访问请求会转往主存, 另外 RAM 是易失性的
只读存储器 (Read Only Memory), 非易失性的随机访问存储器, 速度快且便宜
用于启动计算机的引导加载模块存放在 ROM, 或用于 I/O 卡处理底层设备控制
电可擦除可编程 ROM (Electrically Erasable PROM) 和闪存 (Flash memory) 都是非易失性的
但可以擦除和重写,

互补金属氧化物半导体 (Complementary Metal Oxide Semiconductor), 易失性的, 消耗电能少
用于保持当前时间和日期, 以及保存配置参数, 配置电池失效时会造成计算机的“Alzheimer 症”

磁道 (track): 给定臂位置可读取的一段环形区域, 合并起来成一个柱面 (cylinder)
磁道划分为扇区, 典型容量为 512 字节
随机移到一个柱面用时 5~10ms, 等待扇区 5~10ms, 读写速率 50~160 MB/s
固态硬盘 (Solid State Disk) 实际上是一种闪存存储器

虚拟内存是将程序放在磁盘上, 而将主存作为缓存, 用来保存最频繁使用的部分程序

MMU 存储器管理单元 (Memory Management Unit) 用于快速映射像内存地址

context switch 上下文切换, 指多道程序系统中, 从一个程序切换到另一个程序

SATA 串行高级技术附件 (Serial Advanced Technology Attachment)

I/O 设备 包括设备控制器和设备本身, 因为实际对设备接口隐藏在控制器中, 所以操作系统看到的是控制器接口

设备驱动程序 装入操作系统的三种方式:

device driver UNIX: 内核与 driver 重新链接, 然后重启系统

Windows: 在操作系统文件中设置入口并通知文件需要 driver, 然后重启系统

USB 和 IEEE 1394: 操作系统在运行时接受新设备, 需要动态可装载设备驱动程序

I/O 端口空间 所有设备寄存器的集合

占用地址空间: 将设备寄存器映射到地址空间, 可以像普通存储器读写,

用户被硬件阻挡 (基址和界限寄存器)

专门的 I/O 指令: 设备寄存器放入专门的 I/O 端口空间, 有专门的端口地址.

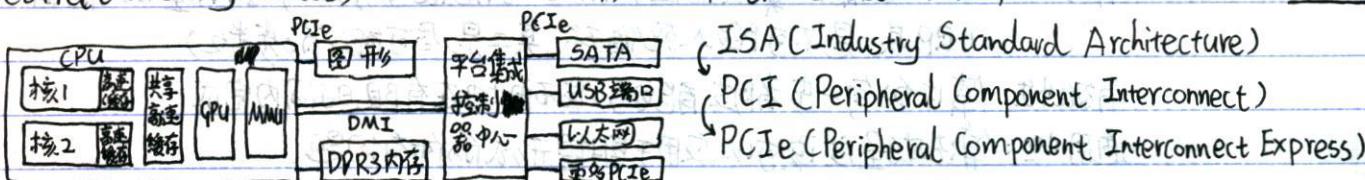
提供内核态中的专门 IN/OUT 指令, 供 driver 读写存储器

Operating System - P4

I/O 实现

- 忙等待：用户程序发出系统调用，内核翻译成对 driver 的过程调用，在一个连续循环中查看是否完成（busy waiting）；随后将控制返回调用者，缺点是在 CPU 轮询设备时一直占用 CPU
- 中断（interrupt）：让设备在操作完成时发出中断，期间阻塞调用者并执行其他工作，driver 检测到完成时发出中断通知，设备编号可作为寻找中断处理程序的地址，即中断向量（vector）
- 直接存储器访问：DMA 芯片控制内存和某些控制器之间的位流，无需 CPU 干预，CPU 设置 DMA 引脚（Direct Memory Access），即启动 DMA 芯片，DMA 在完成时引发一个中断

总线



共享总线架构
shared bus architecture

直到 PCIe 之前，大多数总线都是并行的，表示多个设备使用一些相同的导线传输数据。当多个设备同时需要发送数据，由仲裁器决定哪个设备使用，而 PCIe 使用分离的端到端链路。

并行总线架构 (parallel bus architecture) 表示多条导线发送数据的每一个字，包括传统的 PCI 及之前的串行总线架构 (serial bus architecture) 通过一条称为数据通路的链路传递集合了所有位的一条信息，包括 PCIe。

CPU 通过 DDR3 总线与内存对话，通过 PCIe 总线与外围图形设备对话。

通过 (Direct Media Interface) 总线 经集成中心与所有其他设备对话。

集成中心通过 SATA 总线与硬盘对话，通过 PCIe 传输以太网络帧。

通过 (Universal Serial Bus) 与所有慢速 I/O 设备对话。

SCSI

(Small Computer System Interface) 总线是一种高速总线，用于需要较大带宽的设备，如 ^{高速硬盘} 服务器。

即插即用

每块 I/O 卡有固定的中断请求级别和用于 I/O 寄存器的固定地址，操作系统自动地收集 I/O 设备信息。

plug and play

集中赋予中断级别和 I/O 地址，然后通知每块卡使用的数值，具有相同级别的设备可能引发冲突。

BIOS

基本输入输出系统 (Basic Input Output System)，存放在 RAM 中，非易失性的但可以由操作系统更新。

启动

1. 检查所安装的 RAM 数量，键盘及其他设备是否正常响应。
2. 通过尝试存储在 CMOS 中的设备清单决定启动设备，CD-ROM (USB) → 硬盘。
3. 启动第一个扇区读入内存并执行，包含对保存在启动扇区末尾的分区表检查的程序。
4. 读入第二个启动装载模块至操作系统，并启动。
5. 操作系统向 BIOS 以获得配置信息，检查对应设备驱动程序是否存在，然后初始化表格，启动背景进程。

Operating

System - P5

操作系统类型	
大型机	批处理：不需要交互式用户干预的周期性作业，如定期销售报告 事务处理：大量小的请求，如支票处理和航班预定 分时系统：允许多个远程用户同时在计算机上运行作业，如数据库查询
服务器	通过网络同时为若干用户提供服务，允许用户共享硬件和软件资源，典型如 Solaris, FreeBSD, Linux, Windows
多处理器	按连接和共享方式分为并行计算机，多计算机或多处理器
操作系统的变体	操作系统为配有通信、连接和一致性等专门功能的服务器操作系统的变体
个人计算机	为单个用户提供良好支持
掌上计算机	如平板电脑，智能手机和个人数字助理 (Personal Digital Assistant, PDA)
嵌入式	不可见的软件无法运行，所有软件保存在 ROM 中，应用程序间不存在保护，典型如嵌入式 Linux, QNX 和 VxWorks
传感器节点	配置微处理器，使传感器节点联网，可以彼此通信并且使用无线电通信基站 事件驱动，可以响应外部事件或周期性测量的小型但真实的的操作系统
实时操作系统	时间作为关键参数，保证让特定动作在给定时间完成，分为硬实时（工控、民航、军事）和软实时（数字音频） 操作系统是简单与应用程序链接的库，各个部分必须紧密耦合且彼此没有保护，如 eCos
智能卡	包含一块 CPU 芯片的信用卡，有非常严格的运行能耗和存储空间限制，部分 ROM 里有 Java 虚拟机解释器 (Java Virtual Machine, JVM)
PROCESS	进程本质上是一个正在运行的程序，是容纳运行一个程序所需的所有信息的容器
address space	地址空间是一个存储位置的列表，进程可以在其中读写，存放有可执行程序、程序的数据和堆栈
资源集	寄存器 (程序计数器 PC 和堆栈指针)、打开文件的清单，突出的报警，有关进程清单
process table	进程表是数组或链表结构，用以保证再次启动时的状态与先前锁定时完全相同 所以挂起的进程包括地址空间 (也称磁芯映像, core image) 和对应的进程表项
进程创建与终止	典型如命令解释器 (Command interpreter) 从终端读命令，须先创建一个编译进程，结束后执行一个系统调用终止 若一个进程能够创建一个进程 (子进程)，则可以得到进程树
interprocess communication	进程间通信指合作完成作业的相关进程彼此通信以便同步它们的行为
UID	系统授予权每个进程使用一个给定的 (User IDentification)，子进程拥有与父进程一样的 UID
GID	用户可以是某个组的成员，每个组也有一个 (Group IDentification) 在 UNIX 中有一个特殊 UID 为超级用户 (superuser)，对应 Windows 中的管理员 (administrator)
多道程序	复杂操作系统允许在内存中同时运行多道程序，而为了避免它们相互干扰的保护机制是硬件形式的 由操作系统掌控

Operating System - P6

虚拟内存 为了管理进程的地址空间，每个进程都有一些可使用的地址集合。而为了解决进程大于主存的问题，操作系统创建了一个地址空间的抽象，作为进程可引用的地址的集合，使地址空间与物理内存解耦。

文件系统 大多数操作系统支持目录(directory)的概念，而文件与进程类似，可以组织成树状结构。每个文件都通过从根目录(root directory)开始的路径名(path name)确定。

绝对路径 包含从根目录到该文件的所有目录清单，Windows中以反斜线(\)分隔。

working directory 每个进程有一个工作目录，对于未给出绝对路径的文件在此目录下查找。

读写文件前先检查访问权限，许可则返回一个十进制数字为文件描述符(file descriptor)，若禁止则返回错误码。

mount 系统调用允许把CD-ROM(或其他存储器)上的文件连接到程序希望的根文件系统上。

special file 特殊文件是为了使I/O设备看起来像文件一般，有块特殊文件(block special file)和字符特殊文件(character special file)。
块特殊文件指可随机存放的块组成的设备，如磁盘。
字符特殊文件用于打印机、调制解调器和其他接受或输出字符串的设备。

pipe 管道是一种虚文件，可以连接两个进程，若进程想发现输入输出的是真实文件/管道，需要特殊的系统调用。

I/O子系统 操作系统用来管理其I/O设备，另外相比设备专用的driver，有I/O软件是设备独立的，可以应用于许多或全部I/O设备。

rwx位 管理系统的安全性完全依靠OS，例如文件仅供授权客户访问，如UNIX中的rwx位。
保护代码为9位，分3个字段：rwx rwx rwx，r:读，w:写，x:执行。
例如可写为rwxr-x--x，所有者与所有者同组其他人，而对目录而言，x的含义为查询。

shell UNIX的命令解释器称为shell，“其提示符(prompt)为'\$'”
操作系统的进行系统调用如代码、编辑器、编译器、汇编程序、链接程序、应用程序以及命令解释器。
虽然非重要且有用，但不是OS的组成部分。

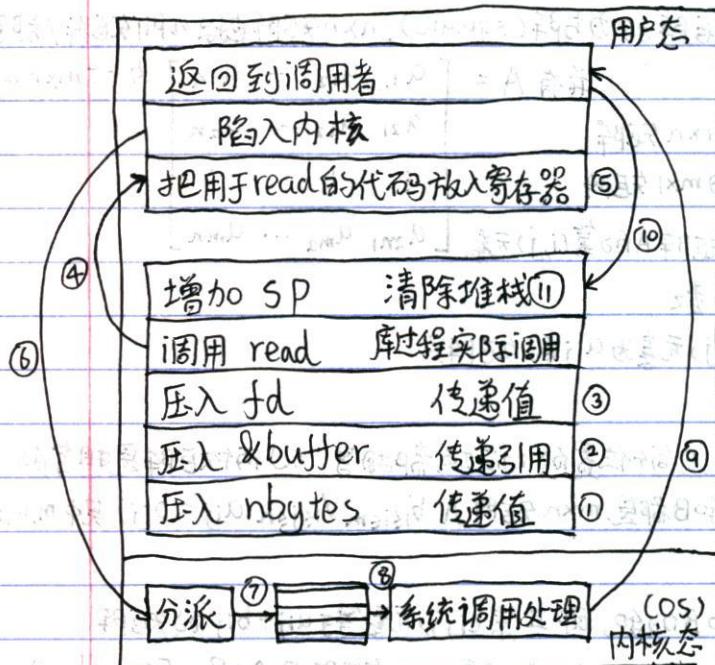
《物种起源》(On the Origin of the Species)出版后，德国生物学家Ernst Haeckel 论述了“个体重复系统发育”，“ontogeny recapitulates phylogeny”，其含义是个体重复着物种演化过程。

技术变化会导致某些思想过时并迅速消失，但又可能使已消失的思想再次复活。

Operating

System - P7

系统调用 UNIX 中的 read 系统调用, read 的三个参数: 指定文件, 指向缓冲区, 要读出的字节数



0xFF...F

①②③: 参数压进堆栈, C/C++ 编译器采用逆序

⑤: 把系统调用的编号放入 OS 所期望的地方

⑥: 执行 TRAP, 进入内核态, 从固定地址开始执行

⑦查找系统调用处理器的指针表.

以系统调用编号为索引, 分派给正确的处理器

⑨返回用户态, 系统调用可能堵塞调用者

(控制可能跟随在 TRAP 后的指令中返回给地址 0x0, 用户空间的库过程)

UNIX

UNIX 与

Windows

进程管理

Windows (Win32)

fork CreateProcess 创建一个新进程

waitpid WaitForSingleObject 等待一个进程退出

execve CreateProcess = fork + execve

exit Exit Process 终止执行

open CreateFile 打开一个已有文件或创建一个文件

close Close Handle 关闭一个文件

read ReadFile 从一个文件读数据

write WriteFile 把数据写入一个文件

lseek Set File Pointer 移动文件指针

stat Get File AttributesEx 取得文件的属性

mkdir Create Directory 创建一个新目录

rmdir Remove Directory 删除一个空目录

link/unlink / Delete File Win32 不支持 link / 销毁一个已有文件

mount/unmount / / Win32 不支持 mount / unmount

chdir Set Current Directory 改变当前工作目录

chmod / / Win32 不支持安全性 (但 Win NT 支持)

kill / / Win32 不支持信号

time Get Local Time 获得本地时间 (从 1970 年 1 月 1 日起)

杂项

Operating

System - P8

fork

创建一个原有进程的精确副本，包括文件描述符、寄存器等内容。

fork之后原有进程和副本(父与子)分开，后续变化不再影响另一个，但共享的程序正文是不可改变的。

PID

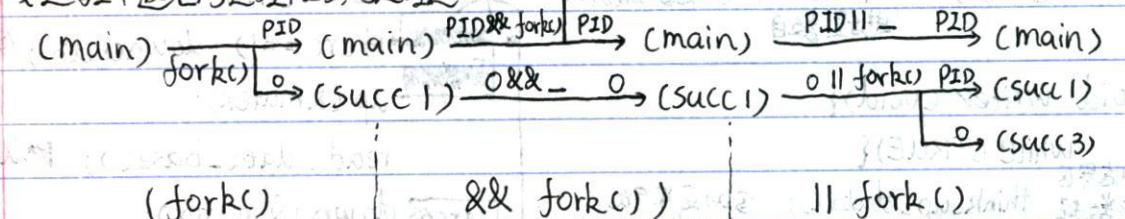
进程标识符(Process IDentifier)由fork返回，父进程中为子进程的PID，子进程中为0。

考虑C语言下的如下语句 fork() && fork() || fork() 会创建多少个子进程

首先注意到运算是左结合的，即 (fork() && fork()) || fork()，且 && 和 || 会忽略第二个操作数。

另外fork()是一次调用两次返回。

父进程中返回子进程PID，子进程返回0



execve

(将要执行的文件名字符串 name, 指向变量数组的指针 argv, 指向环境数组的指针 environp)

execve会引起其整个核心映像被一个文件所替代，考虑 cp file1 file2 将 file1 复制到 file2

CP 的主程序 main(argc, argv, envp), argv 为该命令行内有关参数数目的计数器

argv 是一个指向数组的指针，元素 i 是指向该命令行第 i 个字符串的指针，如 argv[0] 指向“CP”

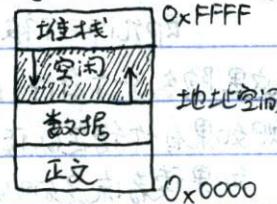
指向环境的指针 envp 用于将诸如终端类型以及根目录等信息传递给程序

存储空间

分为正文段(程序代码)，数据段(变量)，堆栈段。

其中数据段向上增长而堆栈段向下增加。

且数据段的扩展是显示地通过 brk 进行的。



lseek

此调用可以改变指向文件当前位置的指针的值，在顺序读写时，该指针指向下一个读出/写入的字节。有三个参数：文件的描述符，文件位置(offset)，该(offset)是相对于文件开头、结尾或当前位置。修改指针后，lseek返回值为文件中的绝对位置。

link

允许同一个文件以两个或多个名称出现，典型应用是同一开发团队中允许成员共享一个共同文件。

在UNIX中，每个文件都有唯一的i-编号，用以标识文件。

i-编号是对于节点，表格的一个引用，一一对应，说明该文件的拥有者、磁盘块位置。

目录是包含了(i-编号, ASCII名字)对集合的文件。

节点中的一个域记录指向该文件的目录项。

UNIX看到尚且存在的文件没有目录项，就会从磁盘移除。

16	mail	30	bin
81	games	70	note
70	note	38	prog
		指向	同一文件

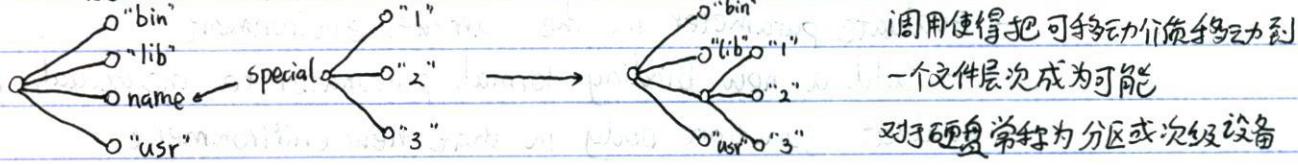
Operating

System - P9

mount

调用允许将两个文件系统合并成一个，调用语句为 `mount(special, name, flag)`

`special` 为安装的块特殊文件的名称，`name` 为被安装的树中位置，`flag` 表示文件系统可读写或只读



工作目录

消除了总是键入“绝对路径名(长的)的需要。UNIX中调用 `chdir` 改变当前的工作目录

保护模式

UNIX中包括针对所有者、组和其他用户的读-写-执行位，`chmod` 使文件对所有者以外的用户只读

kill 调用

供用户和用户进程发送信号用。若进程准备好捕捉特定信号，运行信号处理程序，否则杀掉该进程

UNIX 与 Windows

主要区别在于编程方式；UNIX包括做各种处理的代码以及完成特定服务的系统调用

与 Windows

Windows程序主要是事件驱动程序，主程序等待某些事件发生，然后调用一个进程处理该事件

库调用

UNIX中系统调用(如 `read`)与所使用的库过程(如 `read`)之间几乎是一一对应的

Windows中几乎是不对应的，而是定义了一套过程，称为Win32 API(Application Program Interface)

进程层次

Windows中不存在父进程与子进程的概念，创建者和被创建者是平等的

OS 结构

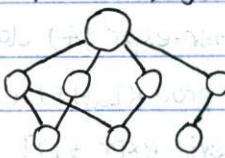
典型的六种设计包括单体系统，层次式系统，微内核，客户端-服务器模式，虚拟机，外核

单体系统

OS在内核态以单一程序的方式运行，整个OS以过程集合的方式编写，并链接成一个大型可执行二进制程序
不受限制的彼此调用的过程可以很高效，但导致系统笨拙，难于理解且易于崩溃

编译单个过程或文件 → 链接成单一文件，每个过程对其他过程都是可见的

主程序：处理服务过程请求



服务过程：执行系统调用

实用过程：辅助服务过程

可装载

UNIX中被叫做共享库(shared library)

的扩展

Windows中被称为动态链接库(Dynamic-Link Library, DLL)

扩展类型均为.dll。