Chpt.01 计算机体系结构概论

计算机层级结构

M5：应用语言机器--------应用语言
M4：高级语言机器--------高级语言
M3：汇编语言机器--------汇编语言
M2：操作系统机器-------作业控制语言
M1：传统机器----------机器指令系统
M0：微程序机器————————微指令系统

编译与解释

编译 compile：
全部N+1级指令 -> N级指令
编译为整体行为，可以优化，效率高，与平台有关。
解释 translation：
一条N+1级指令 -> 一串N级指令
解释为局部行为，不优化，效率低，与平台无关。

系统结构 组成 实现

关系：
具有相同系统结构的计算机可以采用不同的组成，一种计算机组成可以采用多种不同的计算机实现；
采用不同的系统结构会使可以采用的组成技术产生差异，计算机组成也会影响系统结构；
相互关系：
计算机组成的设计，其上决定于计算机系统结构，其下又受限于所用的实现技术，它的发展促进了实现技术的发展，也促进了结构的发展；
计算机实现，特别是器件技术的发展是计算机系统结构和组成的基础，促进了组成与结构的发展；
随着技术的发展，三者关系融合于一体，难以分开，在相互促进中发展。

透明性：本来存在的事物或属性在某个角度上看不到。

性能价格比：

设：D 为研制设计费用，M 为重复生产费用
　　h 为硬件，s 为软件。
则：Dh ~= 100Ds Mh~=100Ms
设 C：该功能在软件实现时许重新设计的次数；
　R：存储介质上出现的次数；
　V：生产的台数；
则硬件的费用： Dh/V+Mh
软件的费用： C*Ds/V+R*Ms

CPU 性能

时钟频率 f
每条指令所花的时钟周期数 CPI = CPU 时钟周期数 / IC
指令条数 IC
则 Tcpu = IC*CPI/f = IC/MIPS*10^6
MIPS Million Instruction Per Second
　= IC/(Tcpu*10^6) = f/CPI*10^-6
MFLOPS Million Floating Point Per Second
　= Ifn/Tcpu*10^-6

软件兼容

向上（下）兼容：指按某一档机器编制的软件，不加修改就能运行于比它高（低）档的机器上。
向前（后）兼容：在按某一时期投入市场的该型号机器上编制的软件，不加修改就能运行宇在它之前（后）投入市场的机器上。
兼容机：把不同公司厂家生产的具有同一系统结构的计算机。

并行性开发途径

时间重叠：是在并行性概念中引入时间因素，让多个处理过程在时间上相互错开，轮流重叠地使用同一套硬件设备的各个部分，以加快硬件周转而赢得速度。 流水线计算机
资源重复：是在并行性概念中引入空间因素，通过重复设置硬件资源来提高可靠性或性能。 阵列处理机
资源共享：是利用软件的方法让多个用户按一定时间顺序轮流地使用同一套资源，以提高其利用率，这样也可以提高整个系统的性能。多处理机系统

Chpt.02 数据表示、寻址方式与指令系统

数据表示：指的是能由机器硬件直接识别和引用的数据类型。由硬件实现的数据类型
数据结构：面向计算机系统软件、面向应用领域所需处理的数据类型。由软件实现的数据类型。

数据类型定义：具有一组值的集合，且定义了作用于该集合的操作集。

带标识符的数据表示的优缺点：

优点:
简化指令系统和程序设计

简化编译程序
便于一致性校验
能由硬件自动完成数据类型的变换
支持数据库系统的实现与数据类型无关的要求
为软件调试和应用软件开发提供支持；
缺点：
使程序所占用的主存空间增加（如下图）
降低指令的执行速度；
必须用专门的指令完成标识符的初始化

数据描述符与带标识符的区别：
标识符是和每一个数据相连的，合存在一个存储单元中，描述单个数据的类型特征。
描述符是和数据分开存放的，专门用来描述所要访问的数据是整块数据还是单块数据，访问该数据块或数据元素需要的地址以及其他特征信息等。

寻址方式种类：
寄存器寻址 立即寻址 直接寻址 间接寻址 相对寻址
变址寻址 寄存器间接寻址 自增自减寻址 比例寻址

逻辑地址：程序员编写程序时使用的地址。
物理地址：程序在主存中的实际地址。

静态再定位：用软件方法把目标程序的逻辑地址变换成物理地址，而在程序的执行过程中，物理地址不再改变。

动态再定位：在执行每条指令时才形成访存物理地址的方法。通过基址寻址。

指令设计的步骤：
根据应用，初拟出指令的分类和具体的指令；
试编出用该指令系统设计的各种高级语言的编译程序；
对各种算法用哪些大量测试程序进行模拟测试，看指令系统的操作码和寻址方式效能是否都比较高；
将程序中高频出现的指令串复合改成一条强功能新指令，即改用硬件方式实现；而将频度很低的指令的操作改成基本的指令组成的指令串来完成，即用软件方式实现；

指令设计原则：
如何支持编译系统能高效、简易地将源程序翻译成目标代码。
规整性 对称性 独立性和全能性 正交性 可组合性 可扩充性

指令格式的优化：
如何用最短的位数来表示指令的操作信息和地址信息，使程序中指令的平均字长最短。

操作码三种编码方法：
固定长度：规整性好，解码简单，空间大。
Huffman 编码：空间小，规整性不好，解码复杂。
扩展编码：折衷方案。

信息源熵 H 和 信息冗余量：
H = -∑pi*log2pi / 信息冗余量 = 1-H / 操作码的实际平均长度

Huffman 主要缺点：
操作码长度很不规整，硬件译码困难
与地址码共同组成固定长的指令比较困难

Chpt03. 存储、中断、总线和 IO 系统

存储器要求：大容量、高速度和低价格

存储器容量 Sm
W：存储体的字长（位、字节）
l： 每个存储体的字数
m：并行工作的存储体个数
Sm = W*l*M

存储器的存取速度
访问时间 TA 是存储器从接到访存读申请，到信息被读到数据总线上所需的时间
存储周期 TM 是连续启动一个存储体所需要的间隔时间
频宽（带宽）BM 存储器可提供的数据传送速率，一般用每秒钟传送的信息位数（或字节数）来衡量
单体的频宽 BM = W/TM
m 个分体的最大频宽 BM = W*m/TM
实际频宽 < 最大频宽

中断的分级
机器校验中断（64 位）一般列为第一级
管理程序调用（8 位）一般列为第二级
程序性中断（指令和数据格式错、程序执行中出现异常，16 位）一般列为第一级
外部中断（16 位）第三极
输入、输出中断（16 位）第四级
重新启动中断 最低级

Chpt04. 存储体系

存储体系定义：

两个或两个以上速度、容量和价格各不相同的存储器用硬件、软件、或软件与硬件相结合的方法连接起来成为一个存储系统。这个系统对应用程序员透明。

访问效率

H 为命中率

$$设：r = T_{A_2} / T_{A_1}$$

$$则：e = \frac{T_{A_1}}{T_A} = \frac{T_{A_1}}{HT_{A_1} + (1-H)T_{A_2}} = \frac{1}{H + (1-H)r}$$

一些定义：

映像方式：低层存储器的块按什么规则装入高层存储器。

映像机构：映像方式的实现。如何识别和查找高层存储器的信息块。

替换策略：访问失效后，如何淘汰信息块，而换新块。

写策略：写操作时采用何种策略以保持相邻两级存储器中数据的一致性，发生写操作失效时是否将被写的块从低层存储器取入高层存储器。

段式管理：

每个程序段都从 0 地址开始编址，长度可长可短，可以在程序执行过程中动态改变程序段的长度。

段式管理优缺点：

优点

程序的模块化性能好。便于程序和数据的共享。程序的动态链接和调度比较容易。便于实现信息保护

缺点

地址变换所花费的时间比较长，做两次加法运算。主存储器的利用率往往比较低。对辅存（磁盘存储器）的管理比较困难

页式管理特点：

页表项简单，查找速度快；

页面大小固定不利于系统的效率，

有些系统可调整其大小。

页式管理在存储空间较大时，由于页表过大，效率降低。

存储空间的保护困难。

页式管理优缺点：

优点

主存储器的利用率比较高。页表相对比较简单。地址变换的速度比较快。对磁盘的管理比较容易

缺点

程序的模块化性能不好。页表很长，需要占用很大的存储空间。

段页式管理：把实存机械地等分成固定大小的页，程序按模块分段，每个段又分成与主存页面大小相同的页。

段页式管理地址变换方法：

先查段表，得到该程序段的页表起始地址和页表长度

再查页表找到要访问的主存实页号

最后把实页号 p 与页内偏移 d 拼接得到主存的实地址。

页式虚拟存储器的构成

$$虚地址：N_s = u + N_v' + N_r$$

$$实地址：n_p = n_v + n_r$$

$$虚存空间：2^u * 2^{N_v'} 个页$$

$$实存空间：2^{n_v} 个页$$

$$其中：N_v = u + N_v'$$

地址映象和变换

地址映象：是将每个虚存单元按某种规则（算法）装入（定位于）实存，即建立多用户虚地址 Ns 与实存地址 np 之间的对应关系。

地址变换：是程序按照这种映象关系装入实存后，在执行时，多用户虚地址 Ns 如何变换成对应的实地址 np。

页面争用（实页冲突）：发生两个以上的虚页想要进入主存中同一个页面位置的现象。

地址变换的原则

减少实页冲突。硬件少、成本低。实现方便、变换速度快。

由于虚存空间远远大于实存空间，因此页式虚拟存储器常采用全相联映像。

替换算法：

随机算法（Random，RAND）：用软的或硬的随机数产生器来形成主存重要被替换页的页号。

简单，易于实现。没有利用历史信息。命中率低，很少使用

先进先出算法（First-In First-Out，FIFO）：选择最早装入主存的页作为被替换的页。

配置计数器字段

虽然利用历史信息，但不一定反映出程序的局部性

近期最少使用算法（Least Recently Used，LRU）：选择近期最

少访问的页作为被替换的页。

配有计数器字段。

比较正确反映程序的局部性。

优化替换算法（Optimal Replacement Algorithm， OPT）:是在时刻 t 找出主存中每个页将要用到时刻 ti，然后选择其中 ti-t 最大的那一页作为替换页。

理想化算法

颠簸现象：连续不断出现页面失效。

堆栈型替换算法

对任意一个程序的页地址流作两次主存页面数分配，分别分配 m 个主存页面和 n 个主存页面，并且有 m≤n。如果在任何时刻 t，主存页面数集合 Bt 都满足关系：Bt(m)⊆Bt(n)，则这类算法称为堆栈型替换算法。

堆栈型替换算法特点：

随着分配给程序的主存页面数增加，主存的命中率也提高，至少不下降。

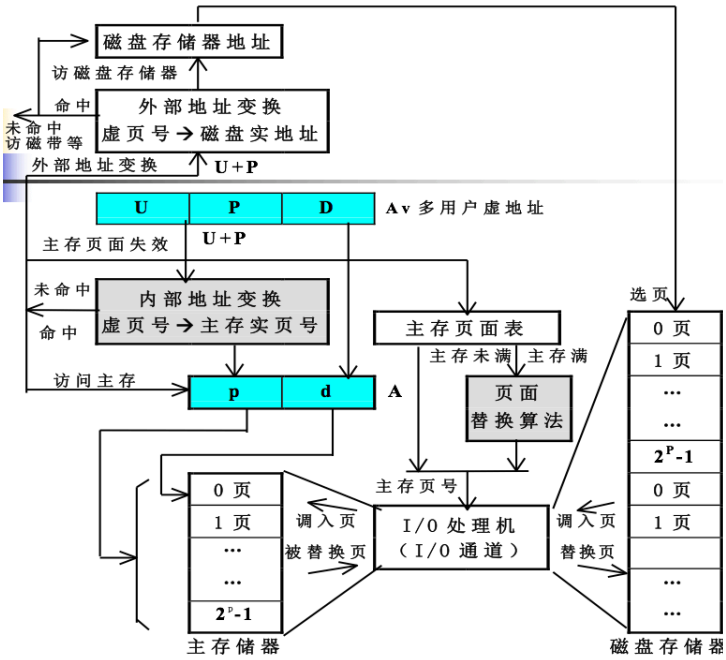
提出使系统性能可以更优的动态算法。

由操作系统来动态调节分配给每道程序的实页数。

LRU、OPT 都是堆栈型算法

FIFO 是非堆栈型算法

图页式虚拟存储器工作原理



S_1：某道程序的主存容量

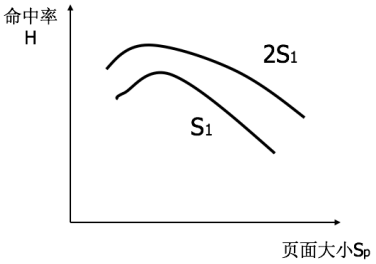
S_p：页面大小

S_1一定时：

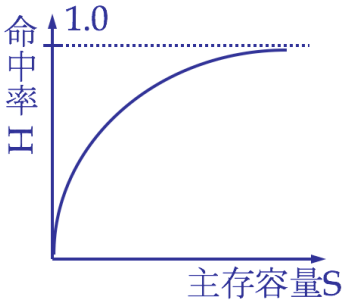
S_p由小增大，H逐渐增大

增达到最大值，下降。

增大S_1可增大S_p。



主存命中率H随着分配给该程序的主存容量S的增加而单调上升。在S比较小的时候，H提高得非常快。随着S的逐渐增，H提高的速度逐渐降低。当S增加到某一个值之后，H几乎不再提高。



地址映象方法：

全相连映象

直接映象

组相连映象：组之间直接映象，组内全相连

段相连映象：组间全相联，组内直接映象。

全相连映象：主存中的任意一块都可以映象到 Cache 中的任意一块。

特点：冲突概率低。空间利用率高。地址变换复杂

直接映象：主存中一块只能映象到 Cache 的一个特定的块中。

特点：硬件简单。冲突概率高。出现大量空闲块。很少使用。

优点：

硬件实现很简单，不需要相联访问存储器

访问速度也比较快，实际上不做地址变换

缺点：块的冲突率较高

组相连映象：

优点：

块的冲突概率比较低。块的利用率大幅度提高。块失效率明显降低

缺点：实现难度和造价要比直接映象方式高

cache 透明：

造成 Cache 与主存的不一致的原因：

由于 CPU 写 Cache，没有立即写主存

由于 IO 处理机或 IO 设备写主存

写

写回法（抵触修改法，WB）：

是在 CPU 执行写操作时，信息只写入 Cache，仅当需要被替换时，才将以被写入过的 Cache 块先送回主存，然后再调入新块。

写直达法（直达法，WT）：

利用 Cache—主存存储层次在处理机和主存之间的直接通路，每当处理机写入 Cache 的同时，也通过此通路直接写入主存。

可靠性，写直达法优于写回法

与主存的通信量,写回法少于写直达法

控制的复杂性：写直达法比写回法简单

硬件实现的代价：写回法要比写直达法好

单处理机（节省成本）：写回法

共享主存的多处理机（保证信息交换可靠）：写直达法

读

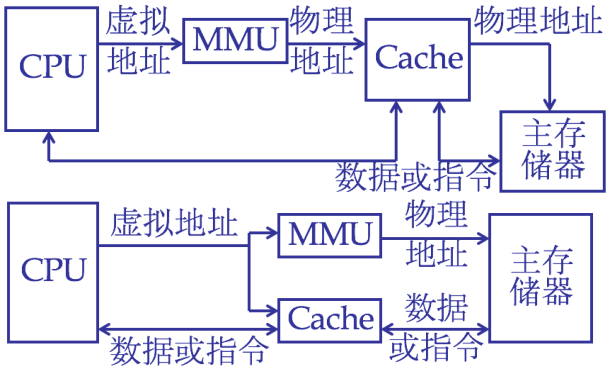
按需取进法：

出现 Cache 块失效时，才将要访问的字所在的块（行）取进。

预取法

恒预取：只要访问到主存第 i 块的某个字，不论 Cache 是否命中，恒发预取命令。 / 不命中时预取：近当访问第 i 块不命中时，才预取命令。

存储系统的两种组织方式



Chpt.05 流水

顺序解释：各条机器指令之间顺序串行地执行，执行完一条指令后采取出下条指令来执行，而且每条指令内部的各个微操作也是顺序串行地执行。

优点：控制简单。

重叠解释：在解释第 k 条指令的操作完成之前，就可开始解释第 k+1 条指令

不能加快一条指令的实现,但能加快相邻两条以至一段程序的解释 .

空间并行性

设置多个独立的操作部件。多操作部件处理机。超标量处理机

时间并行性

采用流水线技术，不增加或只增加少量硬件就能使运算速度提高几倍。

流水线处理机。超流水线处理机

流水线分类

单功能：只能完成一种固定功能的流水线

多功能：流水线的各段通过不同连接实现不同功能

静态：同一段时间内，多功能流水线中的各个功能段只能按照一种固定的方式连接，实现一种固定的功能。

动态：在同一段时间内，多功能流水线中的各段可以按照不同的方式连接，同时执行多种功能。

标量流水 / 向量流水

线性(Linear Pipelining)：每个流水段都流过一次，且仅流过一次

非线性(Nonlinear Pipelining)：在流水线的某些流水段之间有反馈回路或前馈回路

吞吐率 TP Though-put Rate

流水线单位时间里能流出的任务数或结果数。

$$TP_{\max}=\frac{1}{\max\{\Delta t_1,\Delta t_2,\Delta t_3,\Delta t_4\}}$$

$$T=m*\Delta t_0+(n-1)*\Delta t_0=n*\Delta t_0+(m-1)*\Delta t_0$$

$$TP=\frac{n}{m\Delta t_0+(n-1)\Delta t_0}=\frac{1}{\Delta t_0(1+\frac{m-1}{n})}=\frac{TP_{\max}}{1+\frac{m-1}{n}}$$

加速比 Sp Speed Ratio

$$T_{非流水}=n*m*\Delta t_0$$

$$S_p=\frac{T_{非流水}}{T}=\frac{n*m*\Delta t_0}{m\Delta t_0+(n-1)\Delta t_0}=\frac{m}{1+\frac{m-1}{n}}$$

效率 η

如为线性流水线，且各段经过时间相等，则在T时间内，流水线的效率都相等，为η_0，即：

$$\eta_1=\eta_2=\ldots=\eta_m=\frac{n*\Delta t_0}{T}=\frac{n*\Delta t_0}{\mathrm{m}*\Delta t_0+(\mathrm{n}-1)*\Delta t_0}$$

$$=\frac{n}{m+n-1}=\eta_0$$

$$\eta=\frac{\eta_1+\eta_2+\ldots+\eta_m}{m}=\frac{m*\eta_0}{m}=\frac{m*n*\Delta t_0}{m*T}$$

1.分母$m*T$是时空图中m段与T所围成的总面积，而分子是时空图中n个任务是实际占用的面积，效率是面积之比；

2.$\eta = \frac{n*\Delta t_0}{T} = \frac{n}{n+(m-1)} = TP*\Delta t_0$

流水线的效率正比与吞吐率。

3.$\eta = \frac{n}{n+(m-1)} = \frac{1}{1+\frac{m-1}{n}} = \frac{S_p}{m}$

流水线的效率是实际加速比与最大加速比之比。

解决瓶颈子过程的方法：细分（串联）、瓶颈段并联

指令级高度并行的超级计算机

超标量处理机。。超长指令字处理机 。超流水线处理机