操作系统

Operating system

孔维强

大连理工大学

内容纲要

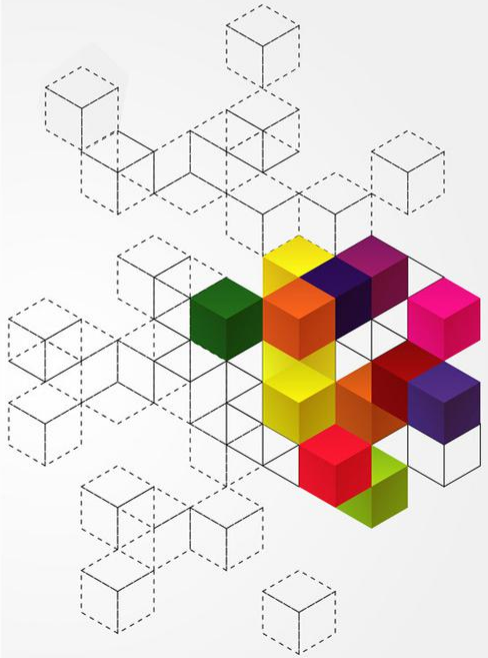
8.1 主存管理基本概念

一、 内存管理的重要性

二、 内存隔离保护

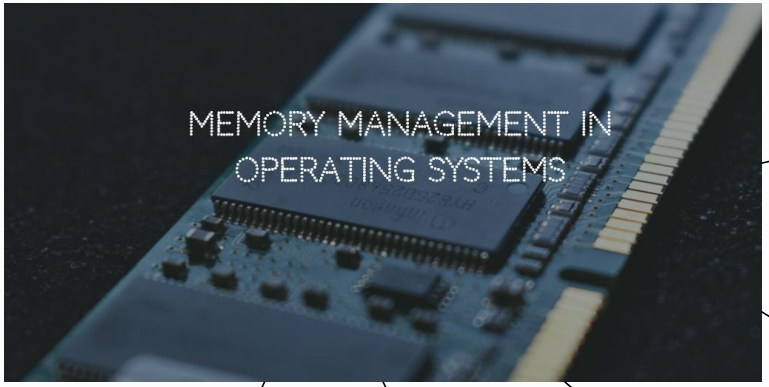
三、 地址绑定

四、 交换的概念

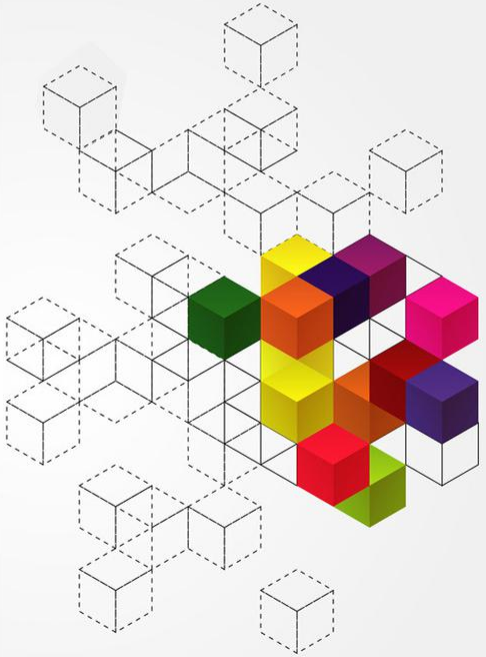


一、内存管理的重要性

内存是计算机系统中的核心资源，需要精心管理

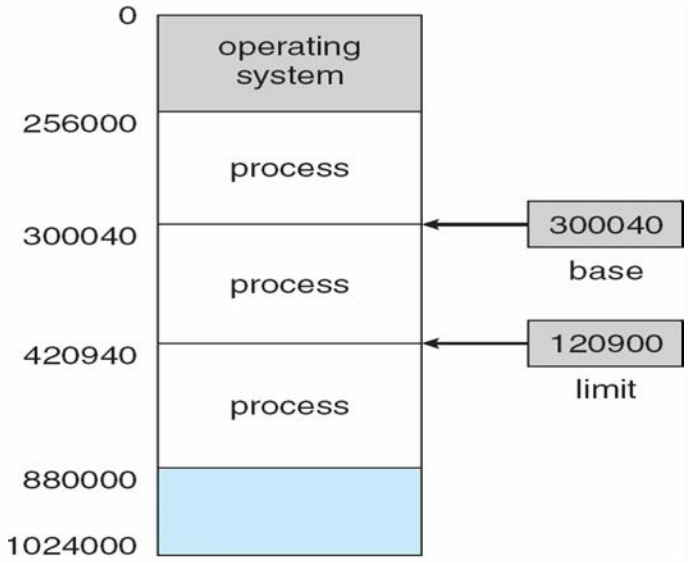


- 内存分配
- 内存保护
- 地址绑定
- 分页/分段
- 内存扩充



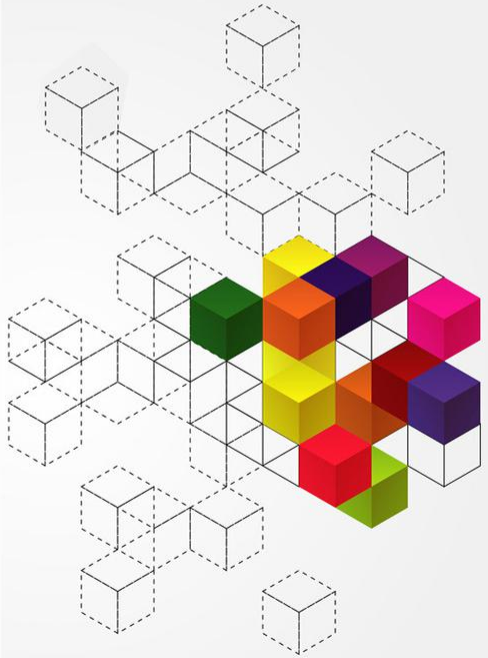
二、内存隔离保护

MM功能1：内存保护



为每个进程分配独立的一段内存空间

基本手段1：
为每个进程分配连续的内存，用两个寄存器分别存放地址上界和地址下界

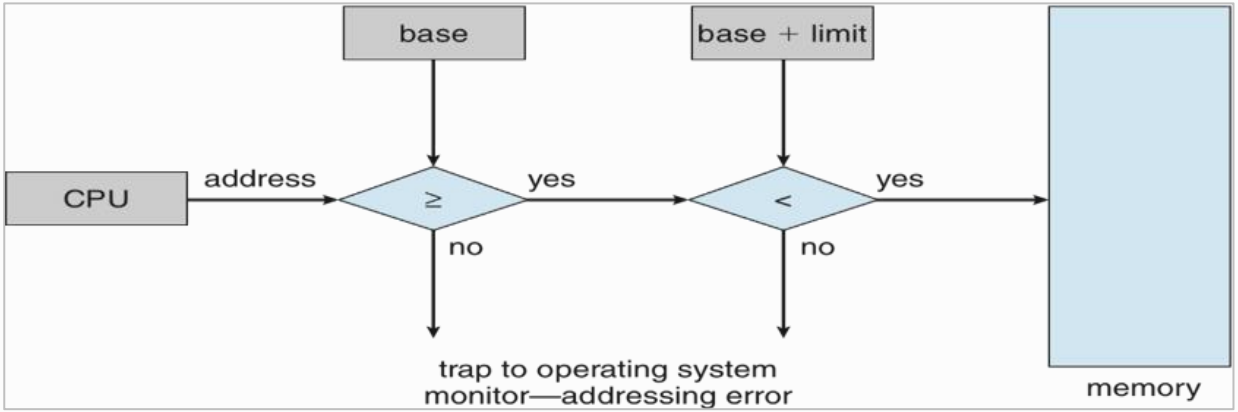


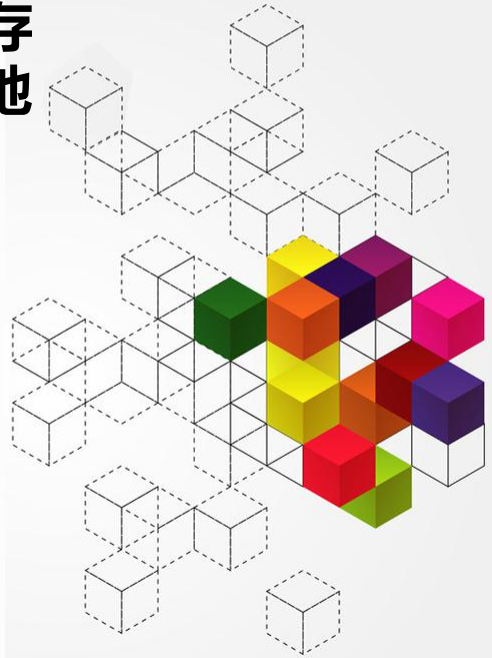
二、内存隔离保护

MM功能1：内存保护（硬件支持）

基本手段1：
为每个进程分配连续的内存，用两个寄存器分别存放地址上界和地址下界

内存保护硬件逻辑

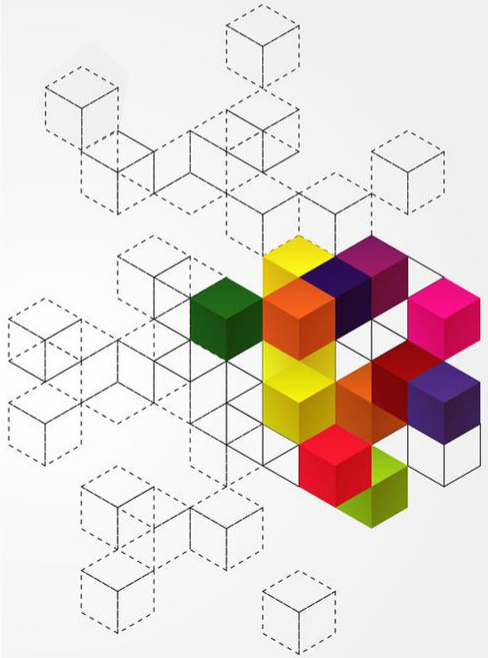




三、地址绑定

用户程序需经过多个步骤才能执行：

- 编译器（compiler）：读取、分析、转换代码为对象文件或错误消息（机器语言、多个对象）
- 链接器（linker）：将一个或多个对象文件（以及可能的库代码）合并为可执行文件、库、或错误消息（1个可执行文件one-executable）
- 加载器（loader）：读取可执行文件至内存、实施地址转换、尝试运行程序，最终运行程序或输出错误消息



三、地址绑定

符号地址（symbolic address 例：程序中的指令、变量）

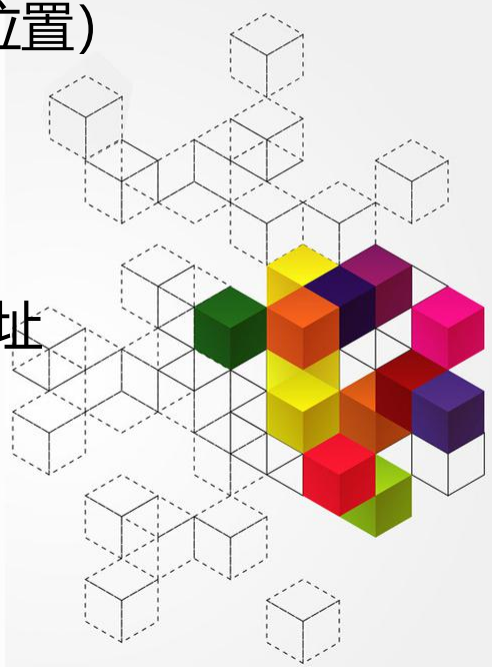
可重定位地址（relocatable address，相对某个指针的位置）

绝对地址（absolute address，内存中的位置）

程序中的指令、数据地址绑定（binding）到内存中的地址

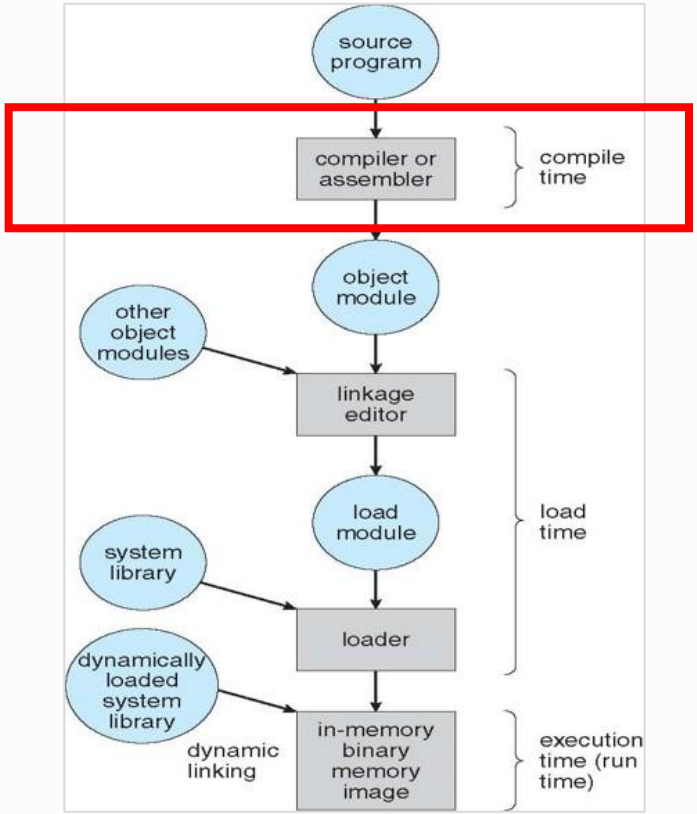
可发生在三个阶段：

Compile time, Load time, execute time

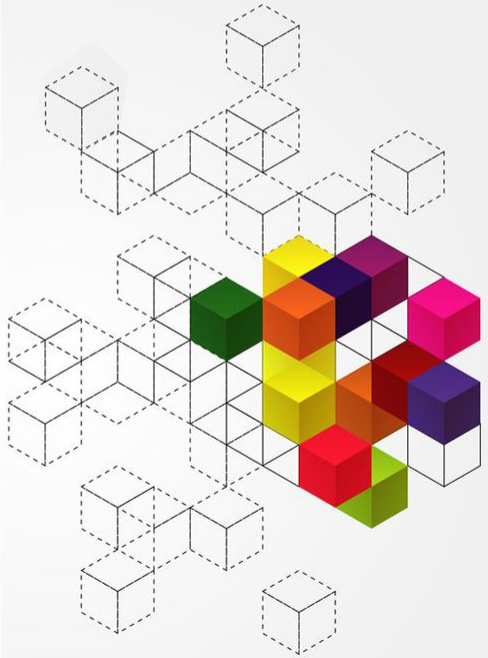


三、地址绑定

MM功能2：地址绑定

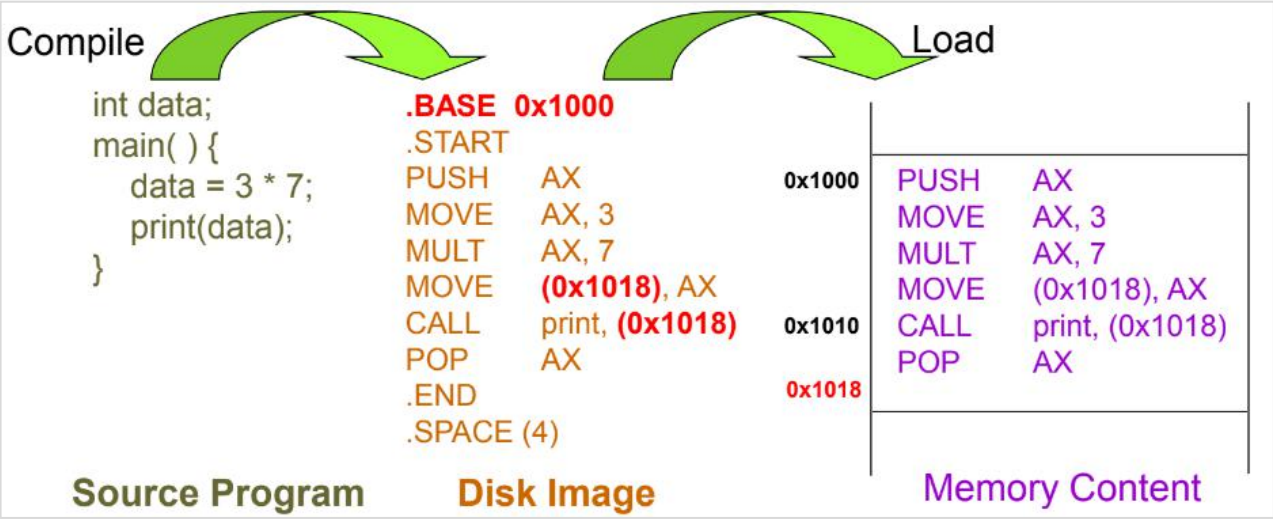


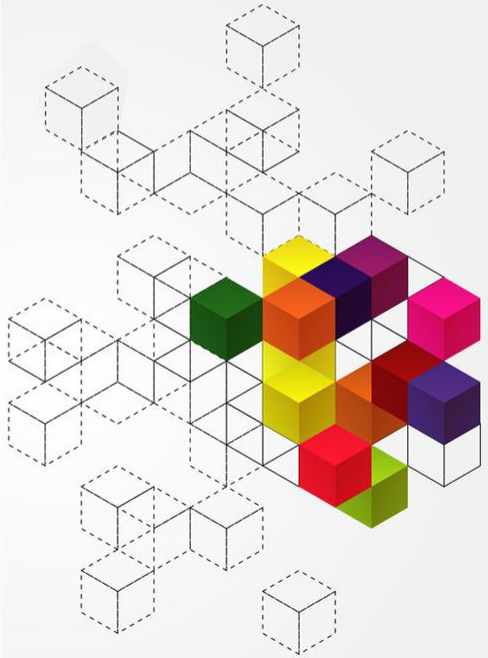
- 编译器直接将源代码中的符号地址转换为物理地址
- 如果程序装载位置发生变化，则需要重新编译



三、地址绑定

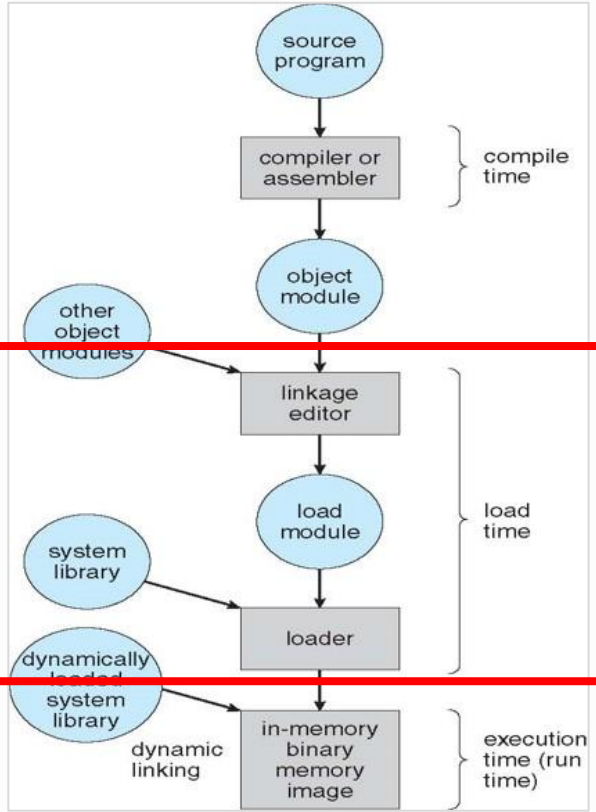
MM功能2：地址绑定（编译时绑定示例）



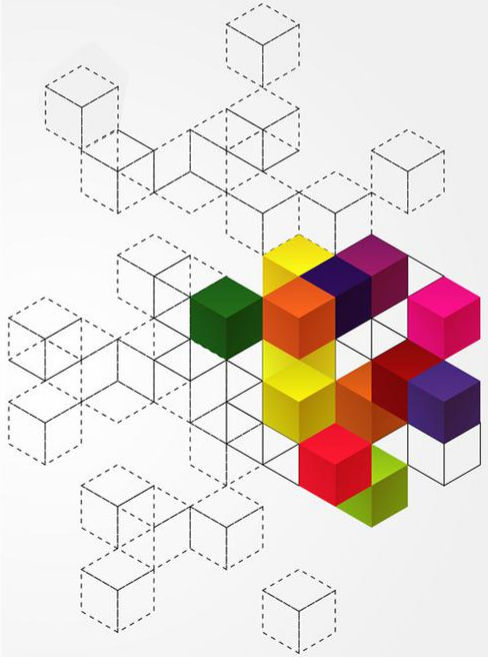


三、地址绑定

MM功能2：地址绑定

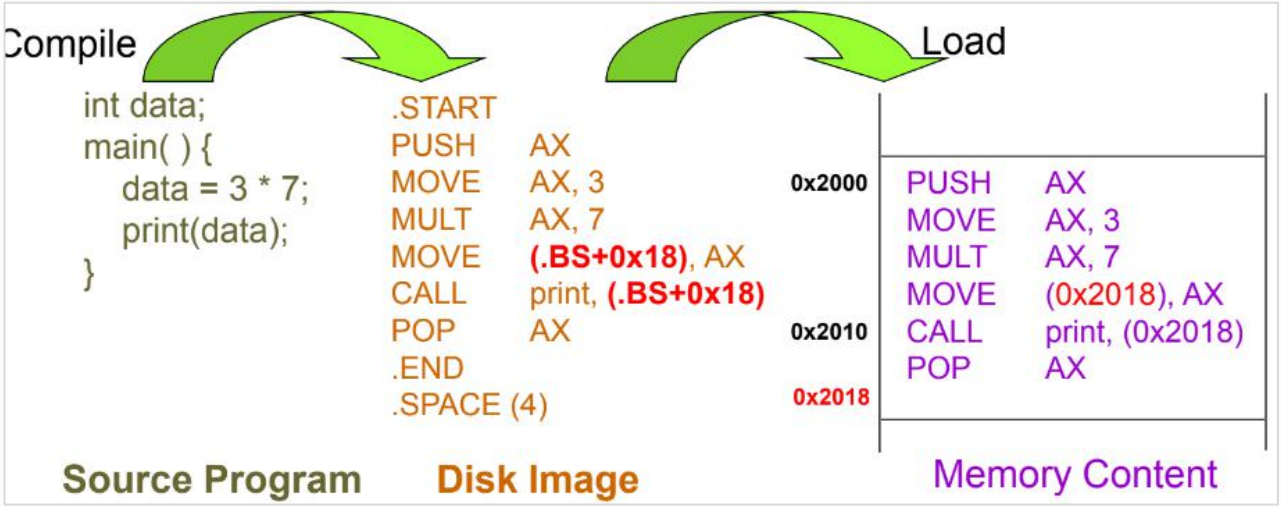


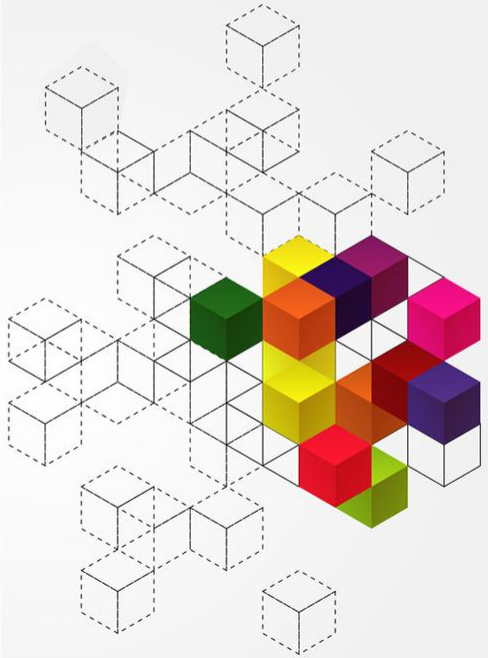
- 编译器将源代码中的符号地址翻译为可重定位地址
- 可执行程序中的可重定位地址在加载时被转换为物理内存地址



三、地址绑定

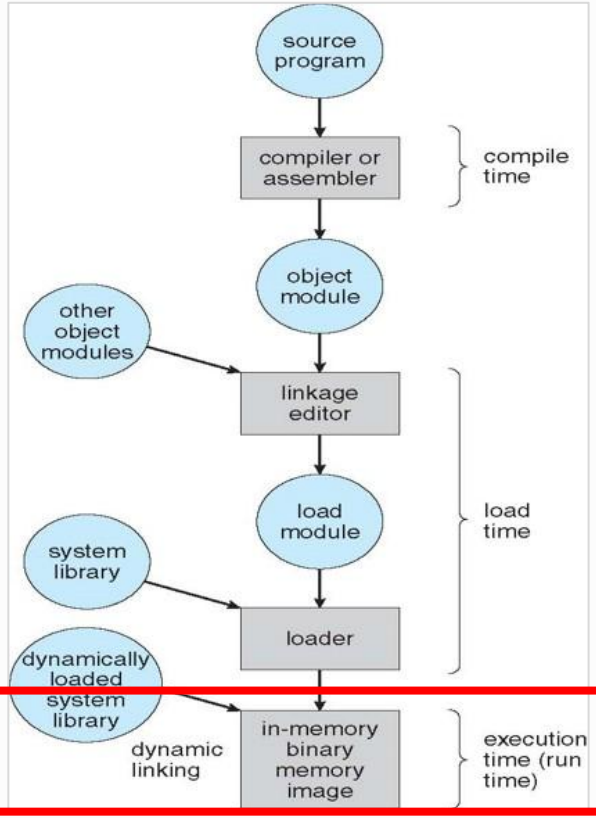
MM功能2：地址绑定（加载时绑定示例）



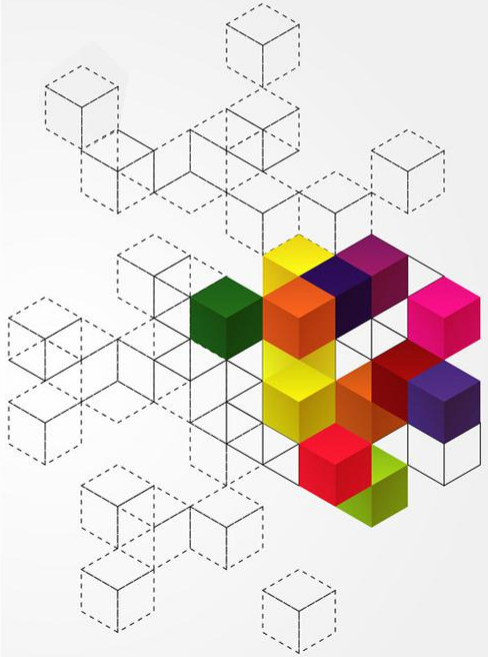


三、地址绑定

MM功能v2：地址绑定

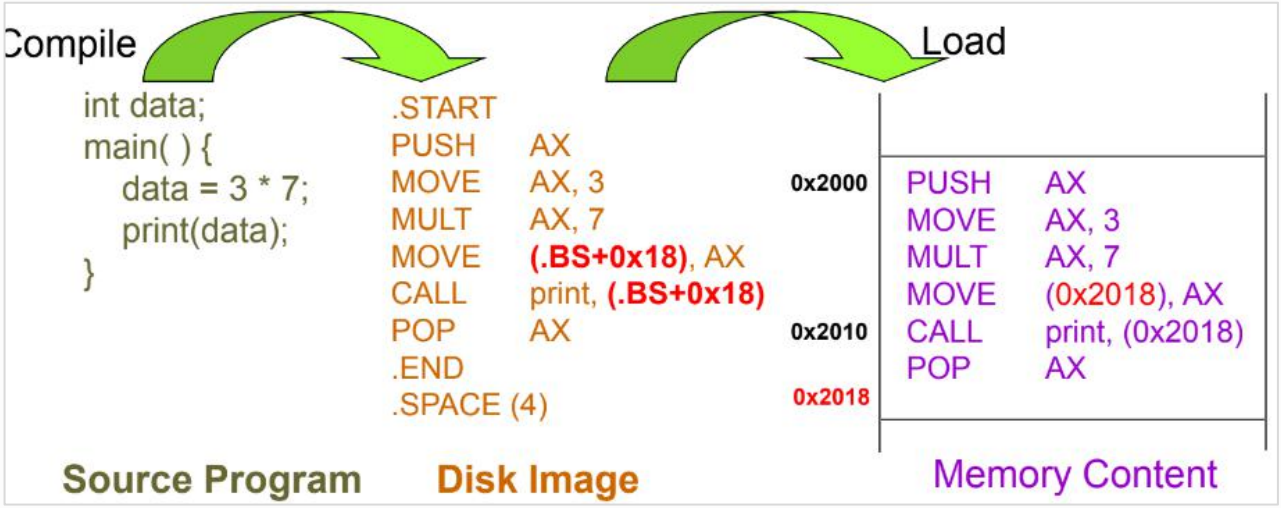


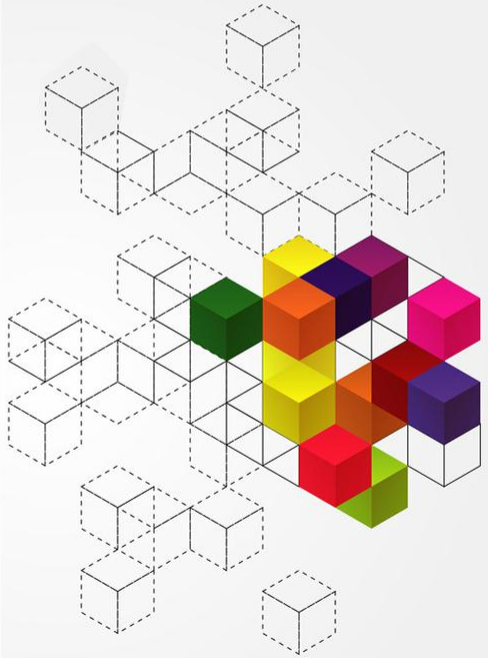
- 编译器将源代码中的符号地址翻译为逻辑地址
- 可执行程序中的可重定位地址在运行时被转换为物理内存地址
- CPU执行的逻辑地址到内存地址的绑定，是MM的最核心的功能



三、地址绑定

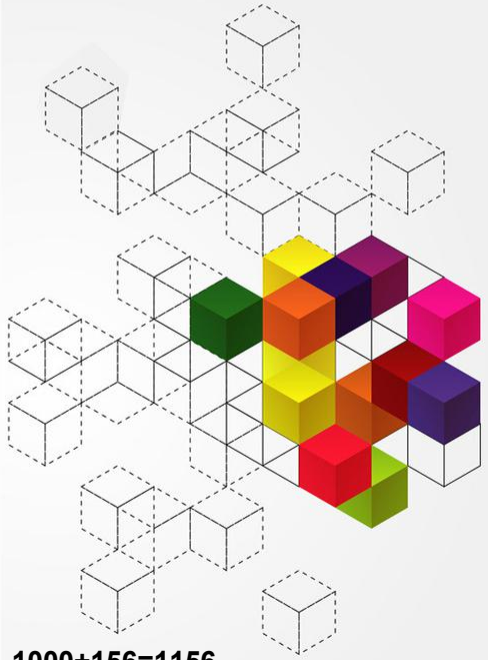
MM功能2：地址绑定（运行时绑定示例）





三、地址绑定（示例）

编译时绑定	1.编译后 (目标文件中)	2.加载到内存后
…	…	…
Integer count	1156:(存放count的值)	1156 (存放count的值)
…	…	
Count=3	move 1156 3	move 1156 3
…		…
加载时绑定		
…	…	
Integer count	156:(存放count的值)	1156 (存放count的值)
…	…	
Count=3	move 156 3	move 1156 3
…		…
运行时绑定		
…	…	
Integer count	156:(存放count的值)	1156 (存放count的值)
…	…	
Count=3	move 156 3	move 156 3

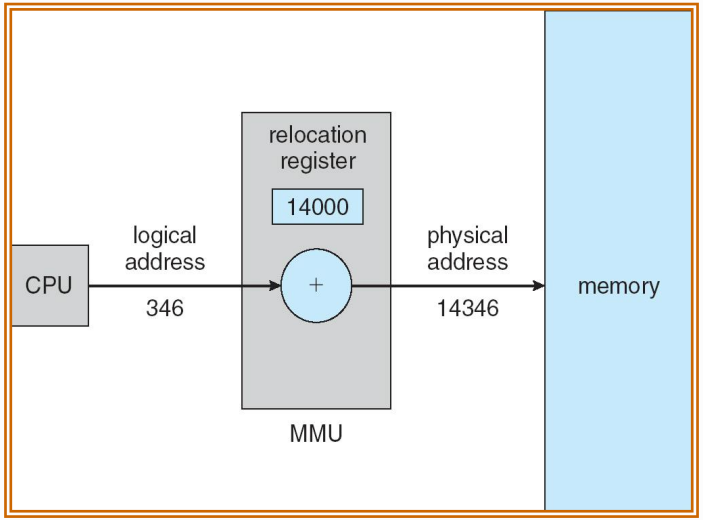


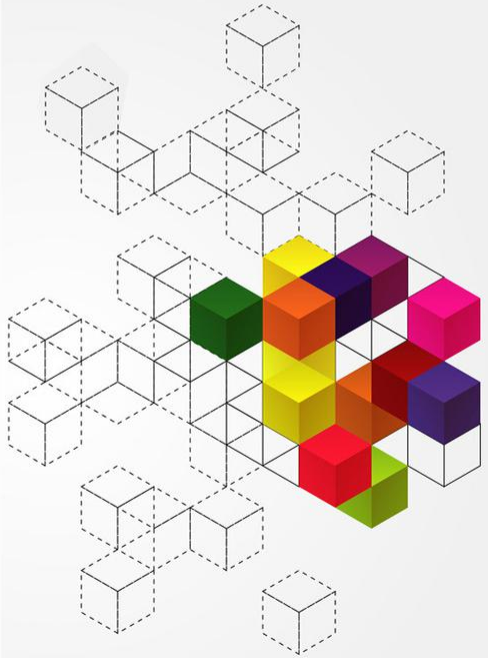
-〉1000+156=1156

（执行该指令时转换）

三、地址绑定（示例）

- 逻辑地址到物理地址的运行时绑定由MMU完成
- MMU = Memory Management Unit

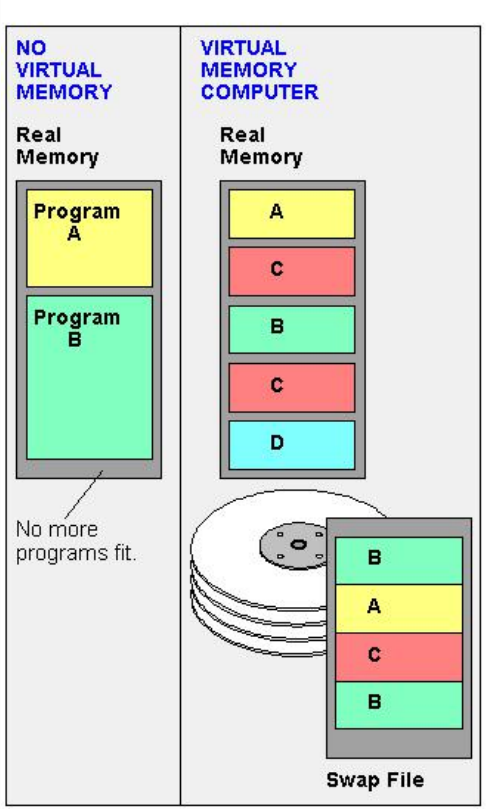




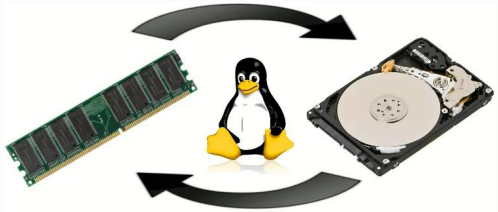
- 用户程序处理逻辑地址（不了解真实的物理地址）
  - 运行时绑定发生在访问内存中的位置时（逻辑地址->物理地址）

四、交换概念

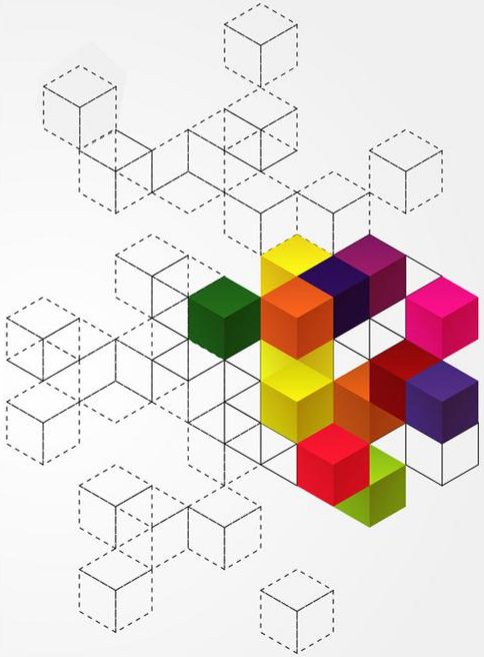
MM功能3：交换



Memory is extended to storage

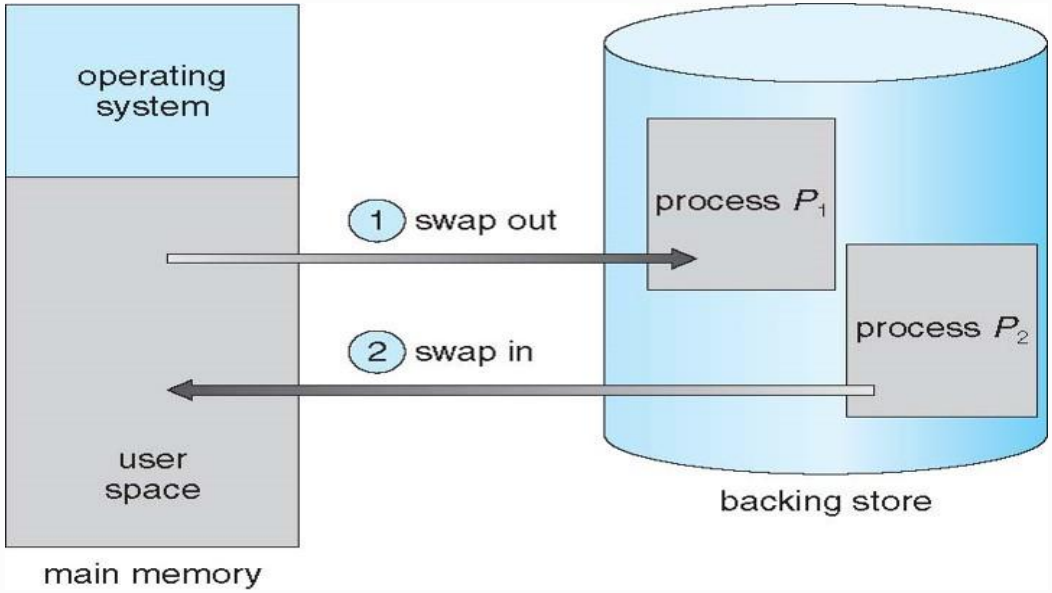


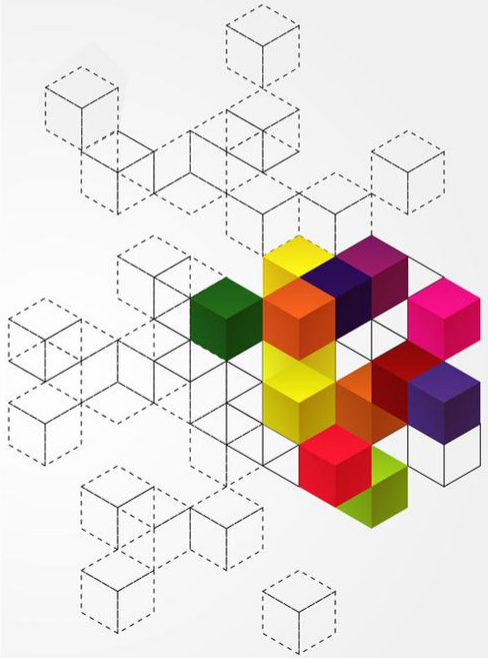
- 进程可从内存中临时交换到backing store，然后在后续再拿回内存中
  - **这样使得多个进程的总物理内存空间大于真实的物理内存**
- **Backing store** – 快速磁盘



四、交换概念

MM功能3：交换



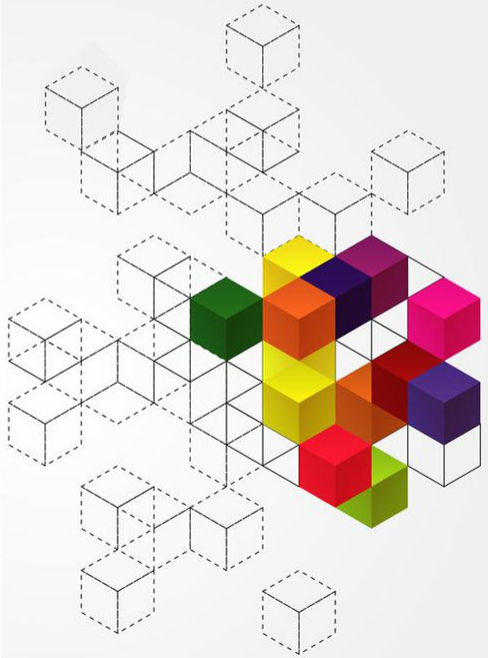


当内存资源紧张时，可以将进程内存映像转移到外存交换空间（扩充形式之一）

四、交换概念

MM功能3：交换

- 将已经交换至backing store 的进程拿回内存时，是否需要拿至交换前相同的物理地址？
- 依赖地址绑定方法（如编译时绑定则必须）
- 交换技术应用在许多系统中（如UNIX，Linux，Windows）
  - 交换一般被禁用
  - 当内存使用超过某个阈值时启用
  - 当内存使用低于某个阈值时再次禁用



本讲小结

- 内存管理重要性
- 内存隔离保护
- 地址绑定
- 交换概念