一。

操作系统的两个主要功能是：

1. **控制执行（进程调度，并发进程控制等）**：操作系统负责管理和协调计算机上的各种应用程序。它分配系统资源（如CPU、内存和磁盘空间），确保程序能够按照预期运行，并且不会相互干扰。
2. **资源管理（处理器管理，存储管理，设备管理，文件管理等）**：操作系统管理硬件资源，包括文件系统、网络、外设等。它负责分配和释放内存、处理进程间的通信、处理输入输出请求，以及维护系统的稳定性和安全性。

操作系统是位于底层硬件与用户之间的沟通桥梁。用户通过操作系统的用户界面输入指令，操作系统解释这些指令并驱动硬件设备，最终实现用户的需求。

二。

因为中断是多道程序并发的基础，每次应用程序处理内部事件或外部事件时，都需要通过中断机制产生中断信号来进入内核态工作。在接收到中断信号后，CPU就会从用户态翻转到核心态，而操作系统是运行在核心态的软件，此时操作系统就会上来处理中断，因此操作系统内核是中断驱动的。

以下是几种典型的中断发生场景：

1.时钟中断： 计算机系统中的时钟定时器会定期生成时钟中断。当时钟中断发生时，操作系统内核会更新系统时间，调度任务，以及执行其他与时间相关的操作。

2.硬件设备中断： 当外部硬件设备需要处理时，例如，键盘输入、鼠标移动、网络数据包到达或打印机准备好接收数据时，硬件设备会向处理器发送中断请求。操作系统内核会相应地处理这些中断，将数据传输到适当的缓冲区或通知相应的设备驱动程序来处理数据。

3.异常中断： 异常中断是由于计算机系统中出现了异常情况而触发的，例如，除零错误、内存访问冲突或非法指令执行等。当这些异常发生时，操作系统内核必须能够捕获并适当地处理它们，以确保系统不会崩溃或者导致数据丢失。

4.系统调用： 系统调用是用户程序请求操作系统内核提供服务的一种方式。当用户程序执行系统调用时，会触发一个中断，使操作系统内核能够在用户态和内核态之间切换，并执行用户程序请求的操作，例如文件操作、进程管理等。

三。

1.提供库函数来封装系统调用的好处有几点：

（1）简化接口: 库函数可以提供更高级别的抽象，使应用程序开发人员能够更轻松地使用系统调用。这意味着他们可以通过更简单的接口来完成复杂的操作，而不必直接操作系统调用。

（2）跨平台性: 库函数可以在不同的操作系统上提供相似的接口，从而使应用程序更容易在不同的平台上移植和运行。

（3）安全性: 库函数可以提供错误检查和安全性措施，以防止应用程序开发人员错误地使用系统调用，从而增加系统的稳定性和安全性。

2.用户程序不能直接访问系统内核提供的服务，这是因为直接访问存在安全隐患。在嵌入式系统中，通常需要操作系统来管理资源，支持多任务运行。允许程序直接访问系统资源会引发许多问题，因此资源管理和分配由操作系统负责。用户程序必须通过操作系统发出服务请求，由操作系统处理相关代码。在Linux中，为了保护内核空间，将运行空间分为用户空间和内核空间。用户进程通常不允许访问内核数据

或使用内核函数，只能操作用户数据和调用用户空间函数。当用户进程需要内核服务时，操作系统通过系统调用将其进入内核空间处理完后再返回用户空间。

3.如果用户程序可以直接访问系统内核提供的服务，需要了解以下额外信息：

（1）系统结构和资源分配: 用户程序需要了解系统的结构和资源分配情况，以便直接操作硬件或系统资源。这包括了硬件组成、寄存器地址、内存分配等。

（2）系统调用接口: 如果用户程序需要与内核进行通信，就需要了解系统调用的接口及参数传递方式。这可能涉及到调用约定、参数传递规则等。

（3）安全性考虑: 直接访问系统资源可能会引入安全风险，因此需要对访问权限进行严格控制。用户程序需要了解如何进行安全的资源访问，以防止意外或恶意操作。

（4）错误处理: 直接访问系统资源时可能会出现错误或异常情况，用户程序需要了解如何处理这些错误，以确保系统的稳定性和可靠性。

（5）系统状态和调度: 用户程序可能需要了解系统当前的状态和调度情况，以便合理地进行资源分配和任务调度，以及避免与其他程序的冲突。

（6）版本和兼容性: 系统的版本和配置可能会影响到用户程序的操作方式和支持的功能，因此需要了解系统的版本信息和相关的兼容性问题。

四。

1.宏内核：

定义：宏内核将许多基本服务集中在内核中，例如文件系统、网络协议栈和硬件驱动程序。

优点：性能高，因为各个功能模块可以直接调用。

缺点：耦合度高，一个模块出现问题可能影响其他模块。

2.微内核：

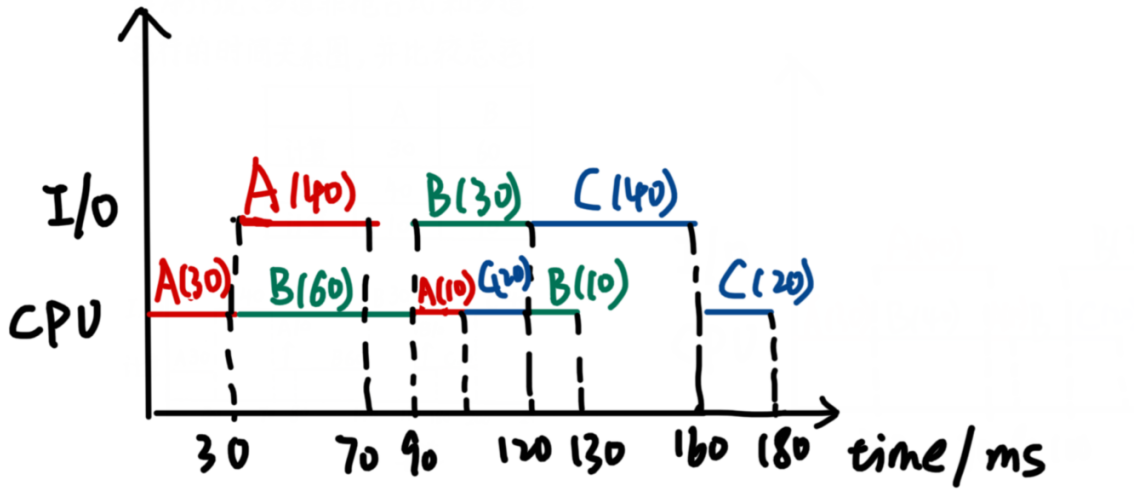
定义：微内核只提供最核心的功能，如任务调度和中断处理。其他功能模块（如进程管理、存储器管理和文件管理）被移出内核，成为特殊的用户进程。

优点：模块之间独立，不会相互影响；稳定性高。

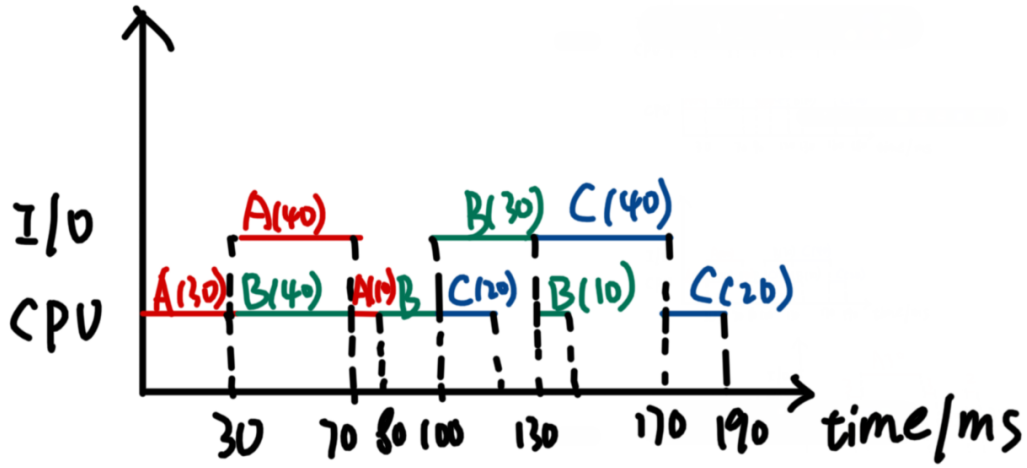
缺点：性能相对宏内核较低。

五。

多道运行的时间关系图（非抢占式）



多道运行的时间关系图（抢占式）



单道运行时间： 30+40+10+60+30+10+20+40+20=260 ms

多道非抢占式运行时间：30+60+30+40+20=180 ms

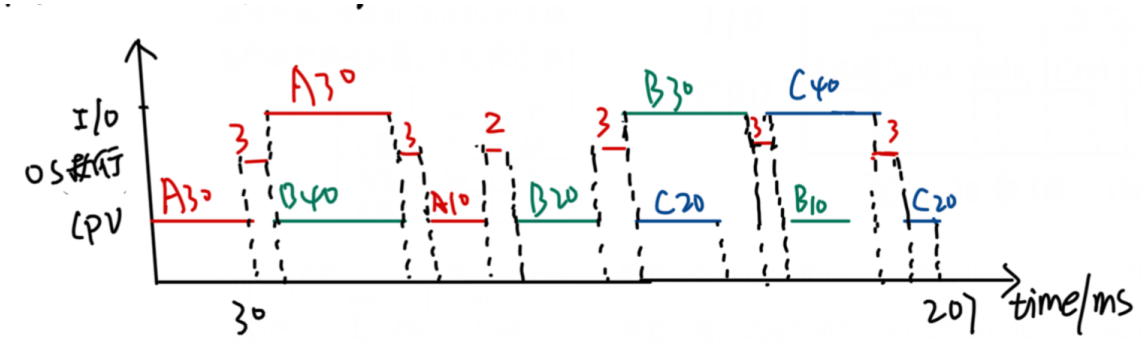
比单道运行节省：260-180=80 ms

多道抢占式运行时间：30+40+30+70+20=190 ms

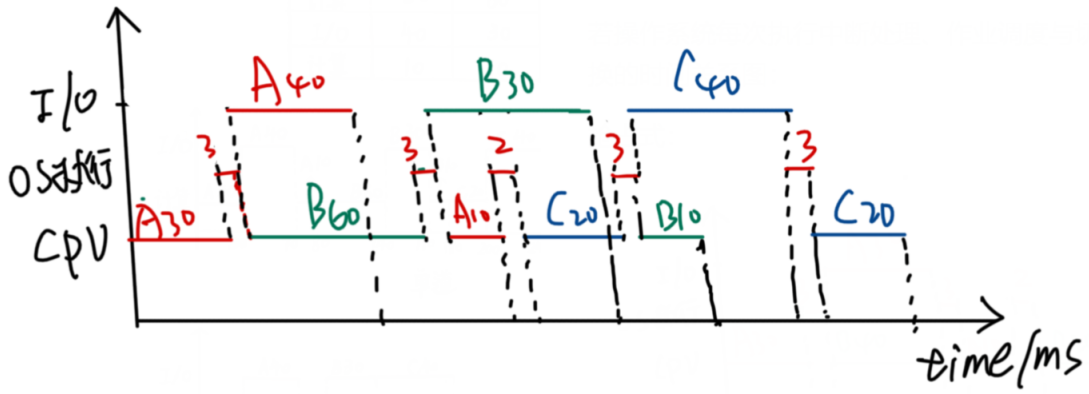
比单道运行节省：260-190=70 ms

若操作系统每次执行中断处理、作业调度与切换等任务，每个任务花费1 ms，各程序状态转换的时间关系图：

抢占式：


I/O
OS
CPU
A30
B40
A10
B20
C20
B10
C20
3
3
2
3
3
3
30
207
time/ms

非抢占式：


I/O
OS
CPU
A40
B30
C40
A30
B60
A10
C20
B10
C20
3
3
2
3
3
3
time/ms