问答题：

1、What are the two main functions of an operating system？

答：①、自顶向下：为应用开发人员提供简洁易用的资源抽象

②、自底向上：管理种类繁多的硬件资源

2、On early computers, every byte of data read or written was handled by the CPU (i.e., there was no DMA). What implications does this have for multiprogramming?

答：多道程序设计的目的是在某个程序等待 I/O 完成时，能够让 CPU 进行其他任务。 如果没有 DMA，则 CPU 会被 I/O 操作完全占用，此时多道程序设计并没有特殊的意义。但是如果由于其他原因导致 I/O 很慢，则 CPU 可以执行其他任务。

3、What is the difference between kernel and user mode? Explain how having two distinct modes aids in designing an operating system.

答：区别： CPU 在内核模式下可以执行其指令集中的每条指令，并且使用硬件的各种功能。但是用户模式下只能执行部分指令和使用硬件的部分功能。

两种模式的用处：拥有两种模式使得设计人员可以在用户模式下运行用户程序，避免访问到关键指令

4、What is a trap instruction? Explain its use in operating systems.

答：陷阱指令是处理陷阱的指令。在操作系统中，陷阱指令可以把 CPU 的执行模式从用户模式切换到内核模式，使得用户程序可以调用操作系统内核中的函数

5、What type of multiplexing (time, space, or both) can be used for sharing the following resources: CPU, memory, disk, network card, printer, keyboard, and display?

答： Time: CPU, network card, printer, keyboard

Space: memory, disk

Both: display

6、To a programmer, a system call looks like any other call to a library procedure. Is it important that a programmer know which library procedures result in system calls? Under what circumstances and why?

答：我觉得是有必要的，因为一个程序如果不需要进行系统调用，那么它可以更快地完成，因为这样能够避免系统调用中切换用户环境和内核环境时发生的开销。

7、Explain how separation of policy and mechanism aids in building microkernel-based operating systems.

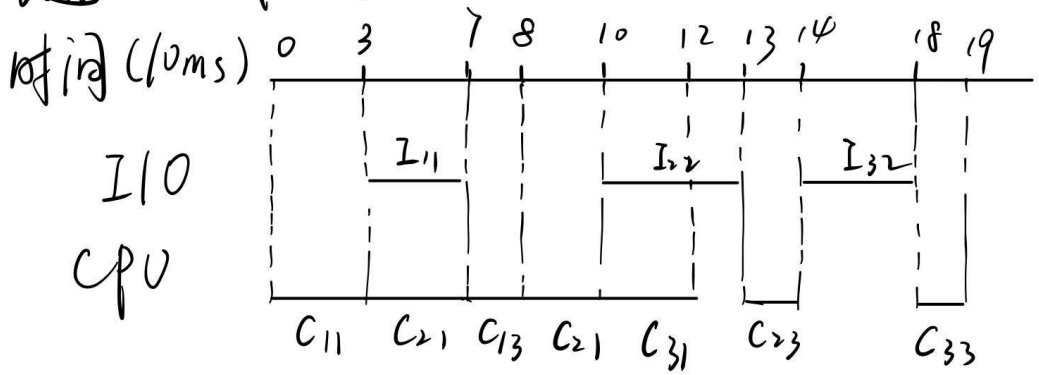
答：策略和机制的分离允许操作系统的设计者在内核中实现少量的基本原语。这些原语由于不依赖任何特定的策略，因此可以被适当简化，进而在用户层面可以用于实现更复杂的机制和策略。

应用题:

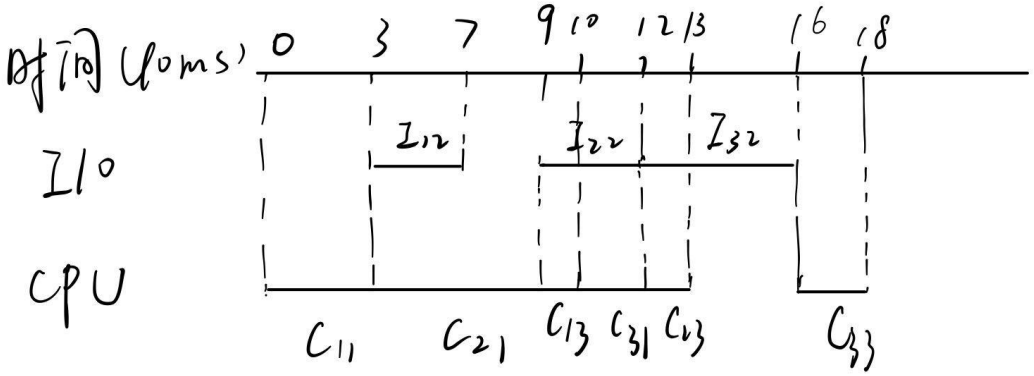
书后应用题 3：

3. 忽略各调度时：单道：260ms

多道（抢占式）：190ms，节省70ms

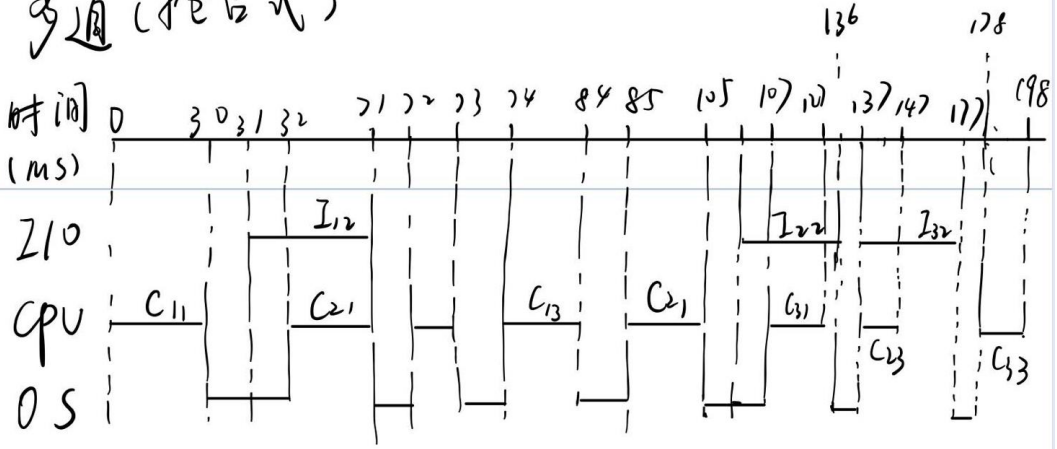


多道（非抢占式）：180ms. 节省80ms

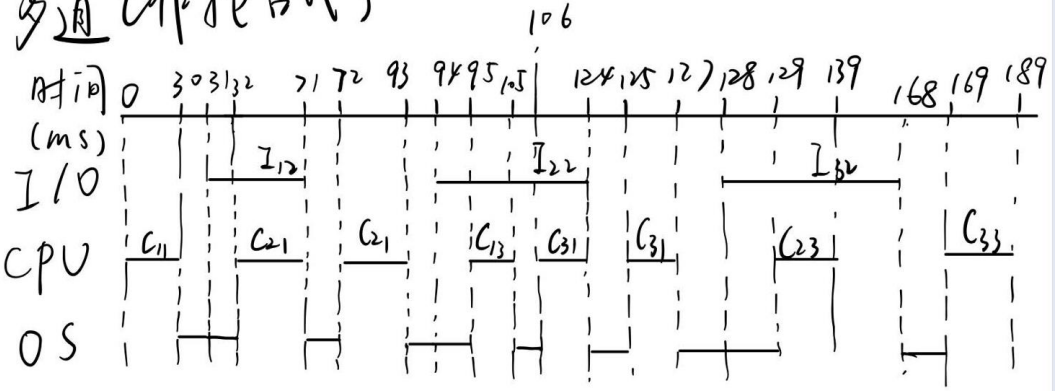


有调度时间时

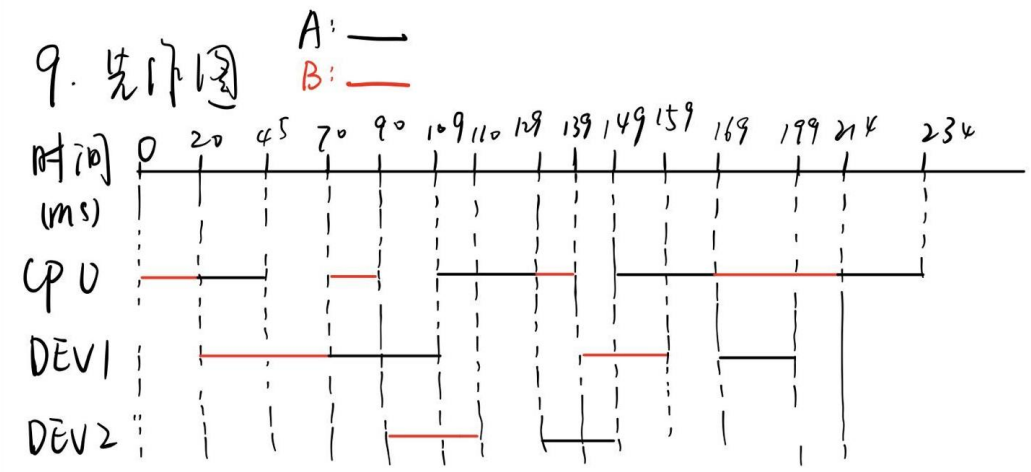
多道（抢占式）



多道（非抢占式）



书后应用题 9：



⑴. B先结束

⑵. 234ms

⑶. $\frac{180}{234} = 76.9\%$

⑷ A等了 20+15 =35ms

⑸ B等了 19+10 =29ms