2. (a)

1. Save the original user-mode instruction bits in the CPU register

2. In order to execute kernel-mode code, the CPU registers need to be updated with the new location of the kernel-mode instructions.

3. Jump to kernel mode to run kernel tasks.

4. After the system call ends, the CPU registers need to restore the original saved user state, and then switch to the user space to continue running the process.

(b)

When dealing with multi-threaded concurrent tasks, the processor will allocate CPU time slices to each thread, and the threads will occupy the processor and perform tasks within their assigned time slices. When forced to suspend running, there will be another thread to occupy the processor. This process in which one thread gives up the right to use the processor and the other thread obtains the right to use the processor is called thread context switching. One thread gives up the right to use the CPU

processor, which is "cut out"; another thread obtains the right to use the CPU processor, which is "cut in". During the process of cutting in and out, the operating system will save and restore related progress information , and this progress information is called "context", that is, the CPU registers and program counters mentioned.

(c)

ABCD ACBD ACDB

CABD CADB CDAB