（多个线程访问同一个资源时就需要同步机制，解决线程安全问题）

（[(1条消息) 单核cpu与多线程体会\_Windsearcher的博客-CSDN博客](https://blog.csdn.net/qq_40511966/article/details/104149168)）

[(1条消息) 单核CPU仍然存在线程安全问题\_疯狂的蜗牛-CSDN博客](https://blog.csdn.net/weixin_37968613/article/details/105510239)

（单核CPU与多线程是否存在线程安全问题？仍然存在。为什么？多核CUP使得多个线程能够存在并行，但是会存在缓存一致性问题；即使没有使用缓存其实也会存在数据不一致性。再看看单核CPU，多个线程需要不断的轮换，每次轮换会保存该线程的信息，你无法保证什么时候线程会进行切换，那就无法保证存在线程信息中的共享数据信息在内存中没有发生变化，所以也会存在数据安全问题。）

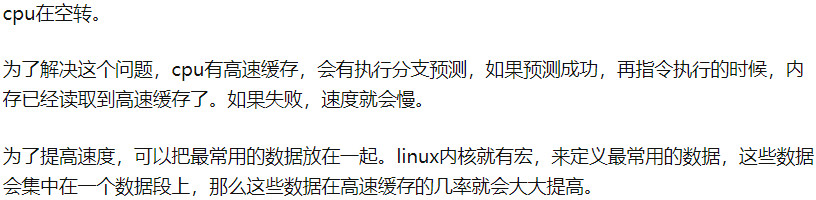
（多核CPU访问内存的机制是怎么样的？）

（寄存器与内存之间会不会存在数据一致性问题？）

（CPU与内存之间速度差异究竟比较的是什么东西？存储设备是cpu暂时缓存运算数据的元件cpu处理数据可以简单认为是读取，运算，回写这一切都需要数据暂存在存储设备我们平时认为的cpu比内存更快是指cpu从内存读取或回写到内存时候用的时间远长于cpu运算所需时间这里的cpu运算也是需要存储设备支持的，就是cpu内部的寄存器在我们看来cpu处理速度高于内存，直观表现就是程序的性能会极大程度上收到内存访问次数的影响，内存访问次数越多，性能越差，所以才引入缓存从而减少cpu访问内存的次数。这里cpu比内存更快实际上是指cpu运算部件和寄存器以及缓存交互的速度比和内存交互的速度更快

）

（CPU访问内存时，CPU在做什么？[CPU在访问内存期间，它在干什么事？ - 知乎 (zhihu.com)](https://www.zhihu.com/question/395249648)

）

（多个ＣＰＵ会不会出现，一个CPU刚拿到数据，然后另一个CPU就立即去更新内存刚才那个数据，这会不会出现问题？

[论CPU是如何保证内存访问冲突？一致性？\_IT专家网 (ctocio.com.cn)](http://storage.ctocio.com.cn/storage/2020/0215/8265.html)

基于mips架构的loongson3A-4核cpu

当一个cpu具有多个物理核心时，他们的虚拟地址时不一样的，但是最后都能访问到内存条，宏观上的路径时怎么走的？

假设一下每个cpu的虚拟地址：

核1:0000\_0000-- ffff--ffff (0-4G)

核2:10000\_0000--1ffff--ffff (4-8G)

核3:20000\_0000--2ffff--ffff (8-12G)

核4:30000\_0000--3ffff--ffff (12-16G)

假设每个cpu为32位，那寻址空间就是4G，这样叠加就是每个cpu的寻址范围了。

那么这样不同的寻址空间，最后是怎么落到同一个内存条的呢？

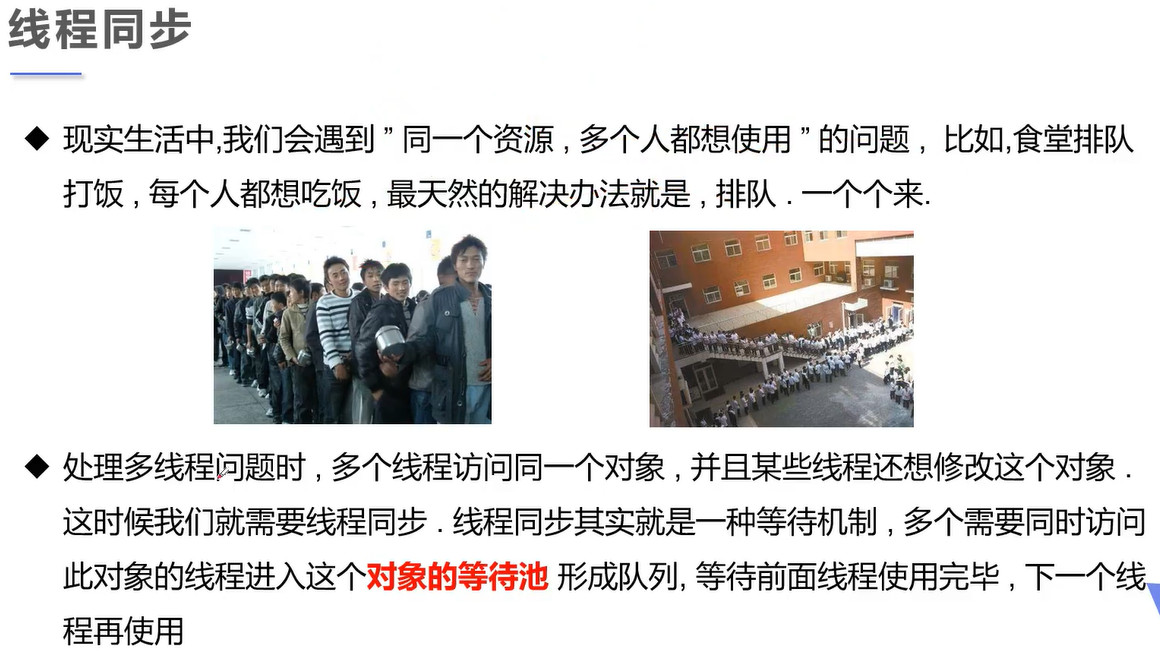
mips架构的cpu每个核心都有自己的地址窗口映射，也就相当于灵活设置不同的核心中不同地址访问的设备是那个，换句话说就是同一个内存条，只要那个核心的虚拟地址通过地址窗口之后，对应到这个内存条上，那么就访问到了。

）

（CPU中的寄存器与内存的关系？[CPU里的寄存器和内存的关系！\_百度知道 (baidu.com)](https://zhidao.baidu.com/question/124766790.html)）

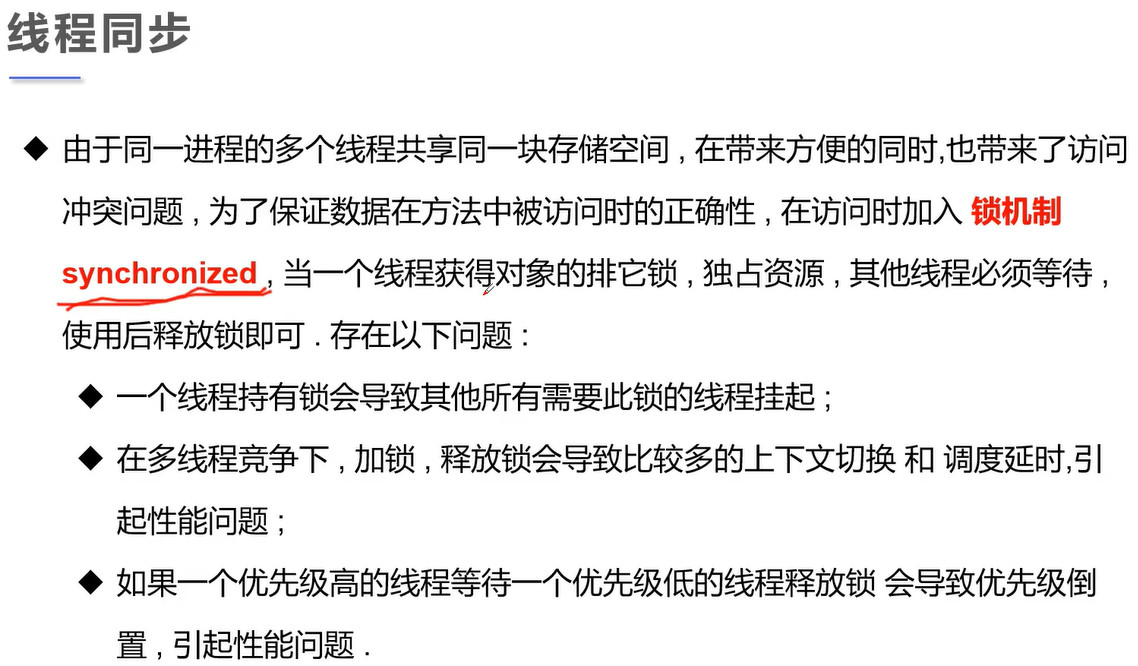
（多个CPU与多核CPU？[多核 CPU 和多个 CPU 有何区别？ - 知乎 (zhihu.com)](https://zhuanlan.zhihu.com/p/85819786)）

（线程切换时，线程状态怎么保存？看具体线程的实现，一般用一种数据结构进行保存。[线程切换时，线程上下文保存在哪？ - SegmentFault 思否](https://segmentfault.com/q/1010000003003454/a-1020000003003489)）



（同步是指多个线程访问某个临界资源时，或者在某个条件下时，需要保证数据的一致性以及逻辑的完整性，于是需要多个线程在相应的时刻进行逻辑上的串行执行的方式，）

（队列加锁，往往是实现线程同步的两个工具）



**以一个Runnable对象创建多个线程相当于将该对象作为资源被多个线程共享**。

（多线程问题往往需要保证内存、缓存、线程数据的一致性，Java使用内存屏障、锁来保证。）

（要细究，就要回到JVM中，每个线程有独立的方法栈，每个方法栈中有局部变量表。线程之间独立运行，所以需要将内存或者缓存中的数据复制一份到各自的线程中，而复制数据这个过程本是原子性的。但是在读取数据的前后往往伴随着改写操作，所以，而多线程又是抢占式执行，所以读取数据的原子性往往不够保证整个业务逻辑中内存中的数据一致性）

