## CPU多核缓存架构

|  |
| --- |
|  |
|  |

## 缓存一致性协议：

|  |
| --- |
| 其实缓存一致性协议不止一种，但是目前大多数采用的是MESI协议。 |
| **如果两个cpu同时修改？**  一个指令周期内，会进行裁决(也就是裁决下到底由哪个cpu改)。 |
| **缓存行：**  cpu存储的最小单元。有的是32字节，有的是64字节，也有的是128字节。 |
| **缓存一致性协议失效问题：**   1. 如果变量的存储长度大于一个缓存行，缓存一致性协议会失效(不能一个数据横缓多个存行)，这时候是加总线索； 2. cpu本身并不支持缓存一致性协议(比如早期奔腾系列的cpu)。 |

## 线程

|  |
| --- |
|  |
| **为啥cpu级别分为ring0和ring3？**  安全性问题，最高级别的操作只允许内核空间的线程进行。否则会出问题。 |

### ULT(用户级线程) && KLT(内核级线程)

|  |
| --- |
|  |
| **UTL用户及线程**  优点：可以避免过度创建线程，还能避免大量上下文切换  缺点：一个线程阻塞，全部线程阻塞 |
| **KLT内核级线程**  每个进程中可以创建多个线程，每个线程都可以看作是一个微进程 |
| **Java中用的是ULT还是KLT?**  java1.2之前是ULT,  java1.2之后用的是KLT . |

### JAVA线程与内核线程的关系

|  |
| --- |
|  |

### Java线程的生命状态

|  |
| --- |
|  |