OSC 源创会 Opensource Innovation Meetup | IT大咖说 知识共享平台

2017源创会年终盛典

与电子标准院共建开源标准

12月23日　北京万豪酒店

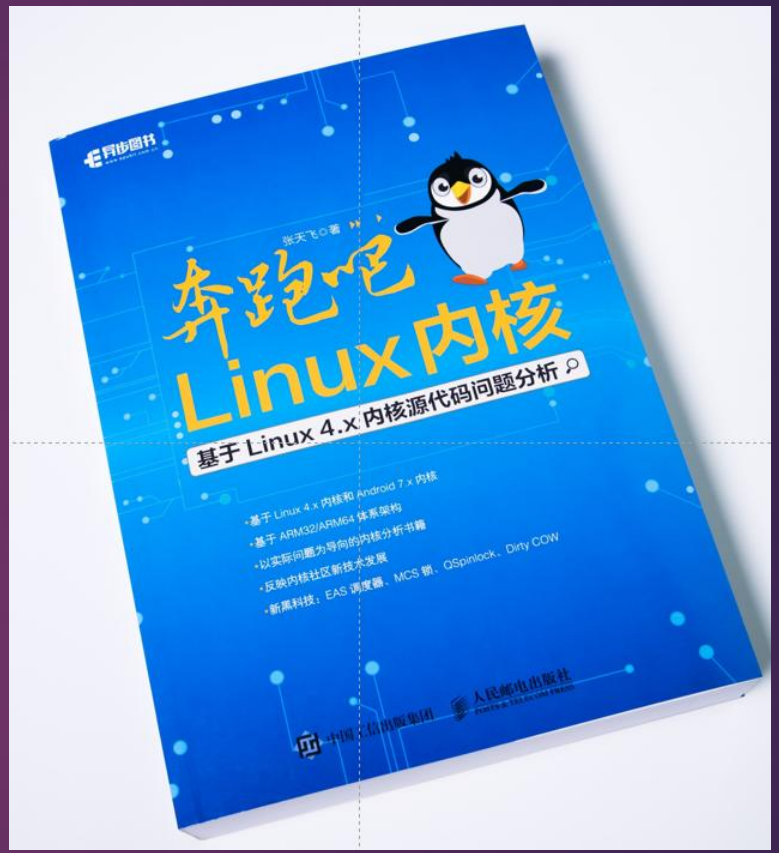
主办方 开源中国 oschina.net　ESI

源创会 Innovation Meetup

个人简介

- 笨叔叔：

沪上紫竹某小公司的小FAE。工作之余写点Linux小段子。2017年不小心出版了《奔跑吧Linux内核》一书。



Linux 4.x内核最新特性简介

笨叔叔

2017.12

Agenda

- Linux内核开发流程简介
- 内存管理最新特性简介
  - 内存管理总览
  - HMM特性
  - 页面回收基于node节点
  - OOM Killer改进
  - swap功能改进
- 进程管理最新特性简介
  - EAS调度器
- 内核学习方法建议

Linux内核开发模式

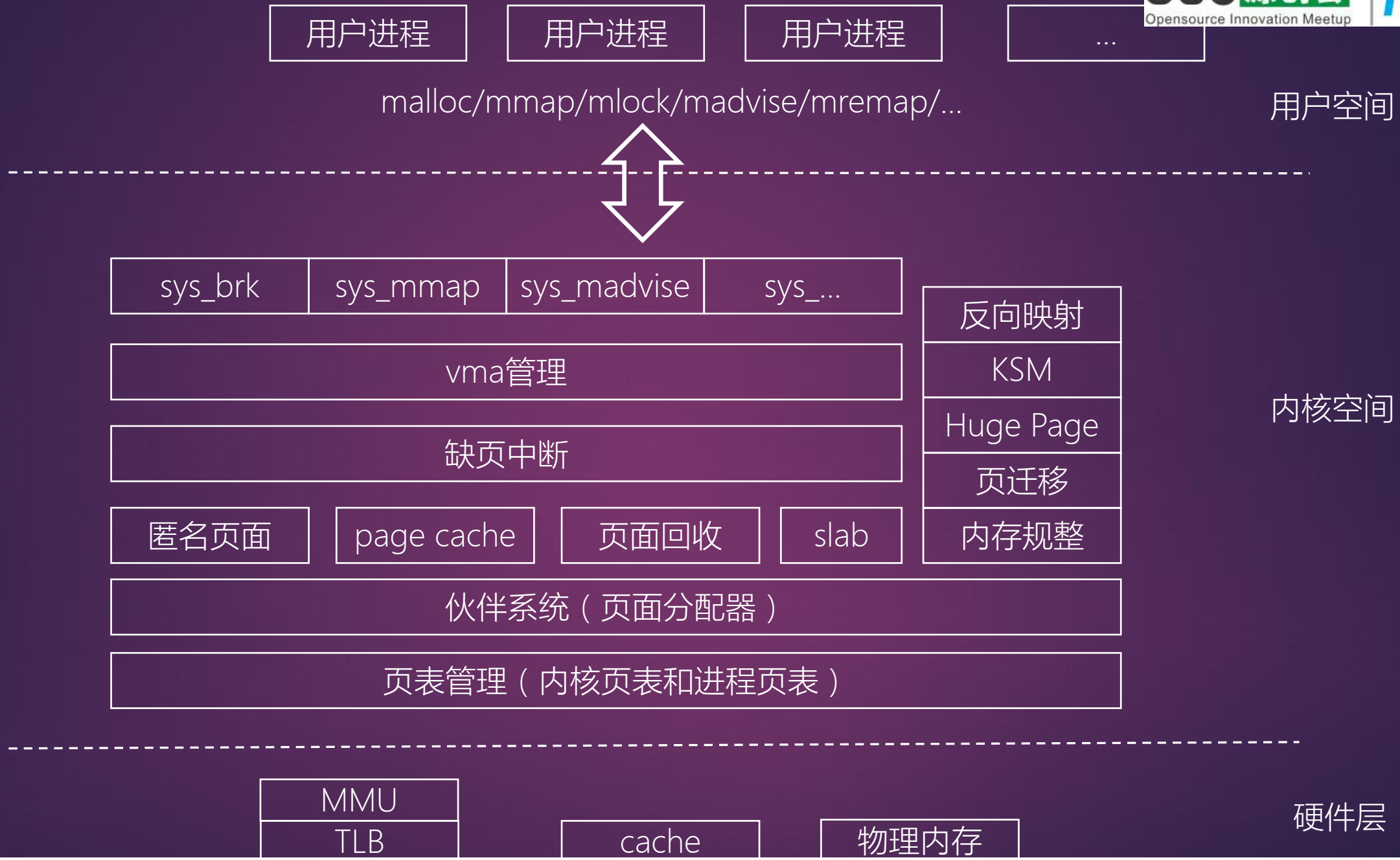
- 现在最新的版本是Linux 4.14 （2017.12）
- 由Linux内核社区开发主导
- 总的maintainer是：Linus Torvalds
- 每个子系统都有maintainer
- 每2~3个月发布一个新版本
- 创新的驱动力来源于各个公司：红帽，Suse，Intel，IMB，谷歌。。。

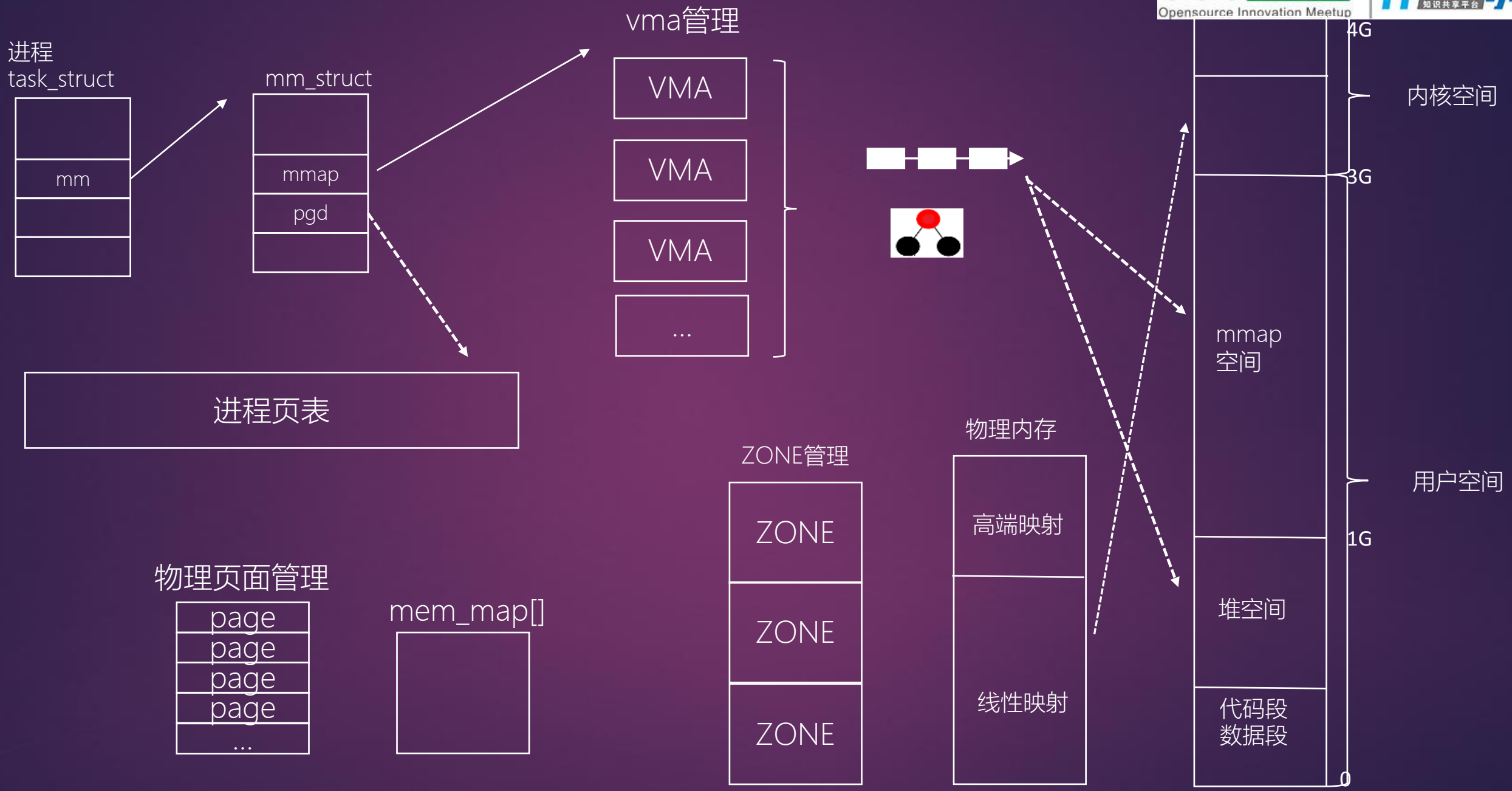
1. 平均60~70发布一个新版本
2. 最新的Linux 4.13内核代码已经超过2千万行代码

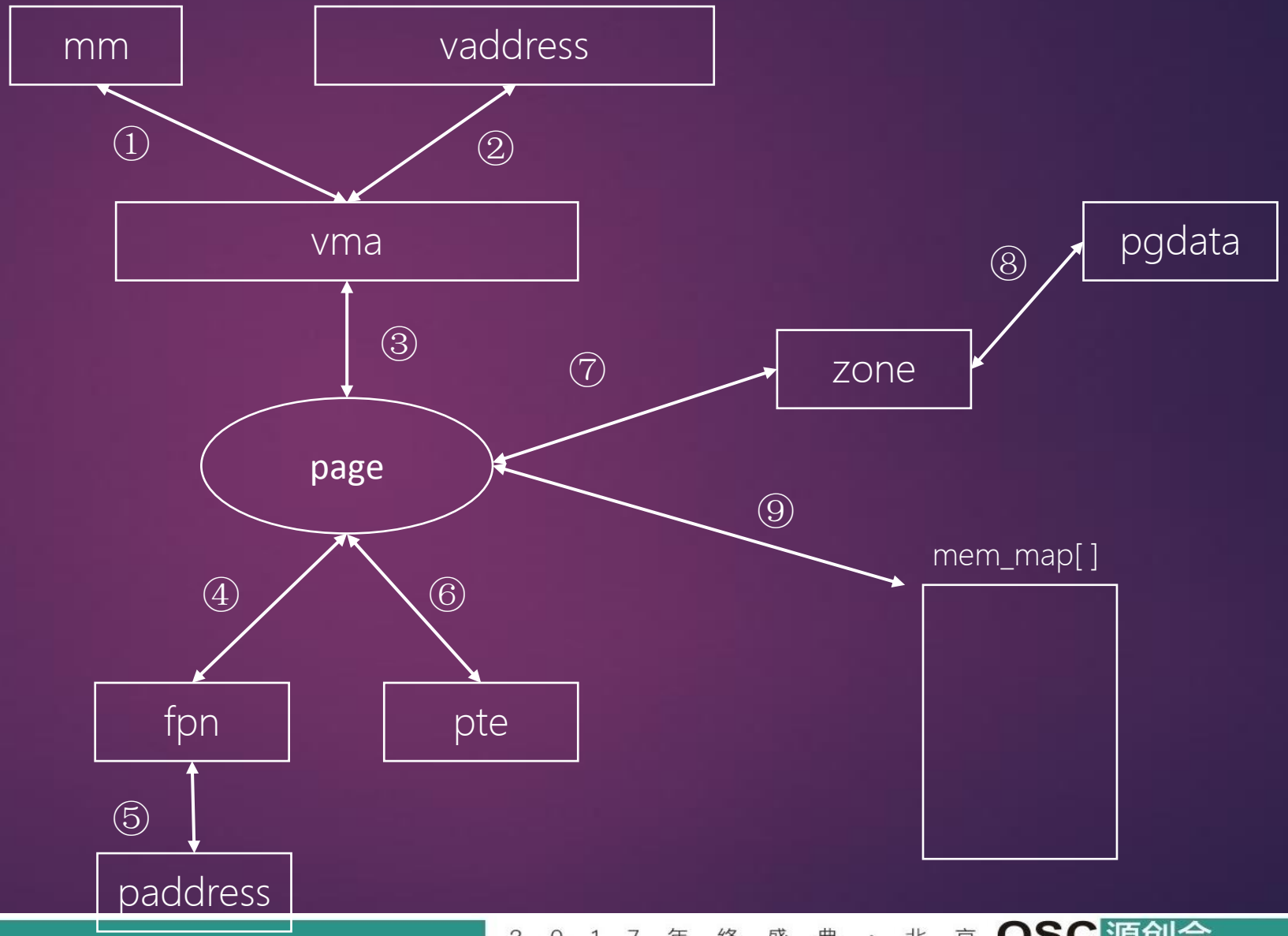
Kernel Version	Release Date	Days of Development
4.8	2016-10-02	70
4.9	2016-12-11	70
4.10	2017-02-19	70
4.11	2017-04-30	70
4.12	2017-07-02	63
4.13	2017-09-03	63

Version	Files	Lines
4.8	55,472	22,070,760
4.9	56,201	22,348,062
4.10	57,167	22,839,361
4.11	57,959	23,137,101
4.12	59,801	24,170,555
4.13	60,538	24,766,703

内存管理总览




进程
task_struct
mm
mm_struct
mmap
pgd
vma管理
VMA
VMA
VMA
...
进程页表
物理页面管理
page
page
page
page
...
mem_map[]
ZONE管理
ZONE
ZONE
ZONE
物理内存
高端映射
线性映射
4G
内核空间
3G
mmap
空间
用户空间
1G
堆空间
代码段
数据段
0


mm
vaddress
①
②
vma
③
⑧
pgdata
⑦
zone
page
⑨
mem_map[ ]
④
⑥
fpn
pte
⑤
paddress

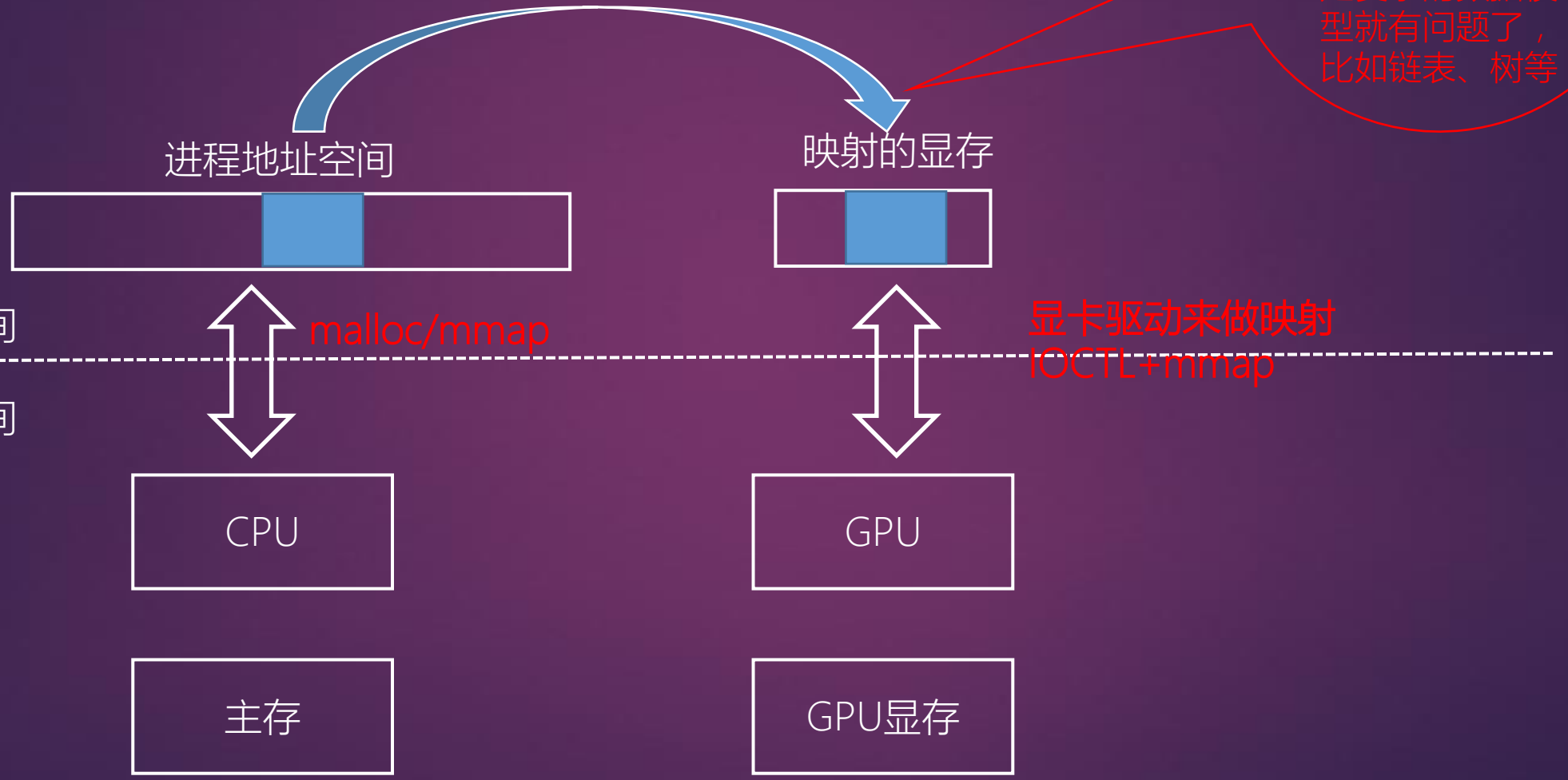
异构内存管理

传统访问方式

被分离的进程地址空间（split address space）

- CPU通过malloc/mmap分配的常规的内存
- GPU通过显卡驱动管理和映射的进程地址空间
- 这两个地址空间是分离的，访问需要相互来回拷贝

相互拷贝：
简单的数据类型比如数组、image等没有问题，但是复杂的数据模型就有问题了，比如链表、树等



传统访问方式小例子

```
mul_mat_on_gpu(float *r, float *a, float *b, unsigned m)
{
    gpu_buffer_t gpu_r, gpu_a, gpu_b;

    gpu_r = gpu_alloc(m*m*sizeof(float));
    gpu_a = gpu_alloc(m*m*sizeof(float));
    gpu_b = gpu_alloc(m*m*sizeof(float));

    gpu_copy_to(gpu_a, a, m*m*sizeof(float));
    gpu_copy_to(gpu_b, b, m*m*sizeof(float));

    gpu_mul_mat(gpu_r, gpu_a, gpu_b, m);

    gpu_copy_from(gpu_r, r, m*m*sizeof(float));
}
```

CPU分配的buffer，属于常规内存，比如通过malloc分配的。这些数据作为input

分配GPU buffer，通过GPU驱动来分配的

把CPU分配的buffer拷贝到GPU buffer里

在GPU中进行计算

把GPU的结果拷贝回CPU的buffer

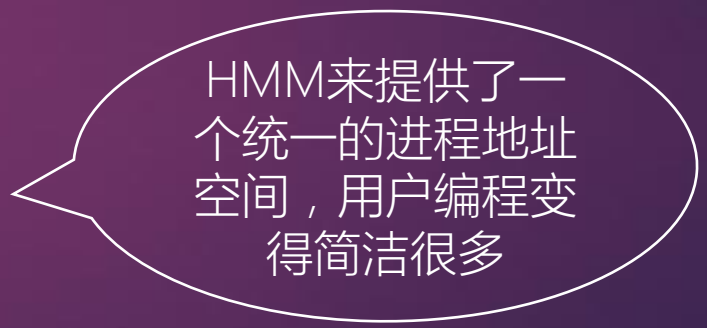
HMM（heterogeneous Memory Management）

- 建立一个统一的进程地址空间（share address space）
  - HMM mirror镜像功能：CPU和GPU都可以同时访问这个统一的进程地址空间（虚拟地址），都能同时访问到相同的物理内存。

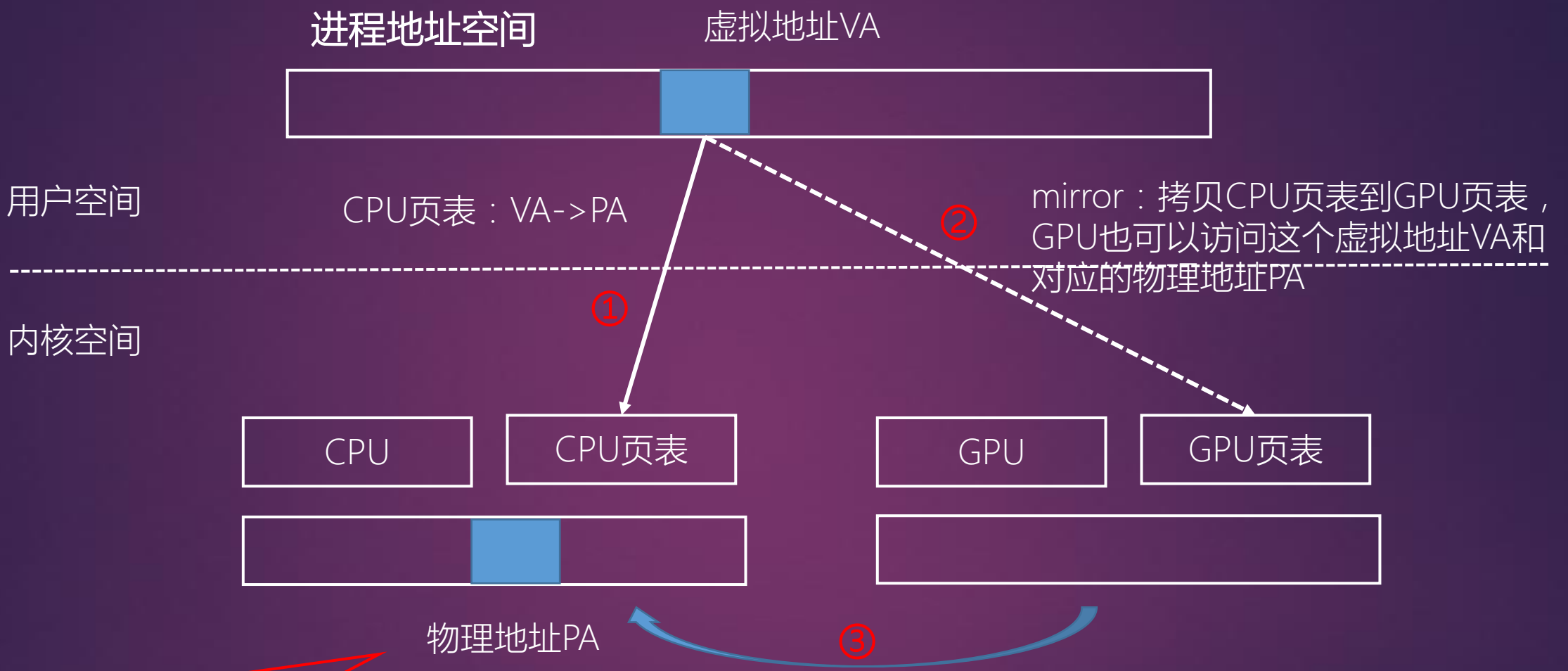
- 页面迁移：在一个进程里可以透明地使用device的内存。
- 已经合并到Linux 4.14内核中
- 支持HMM的开源NV的显卡驱动nouveau正在开发中

支持HMM的应用程序例子：

```
mul_mat_on_gpu(float *r, float *a, float *b, unsigned m)
{
   gpu_mul_mat(r, a, b, m);
}
```



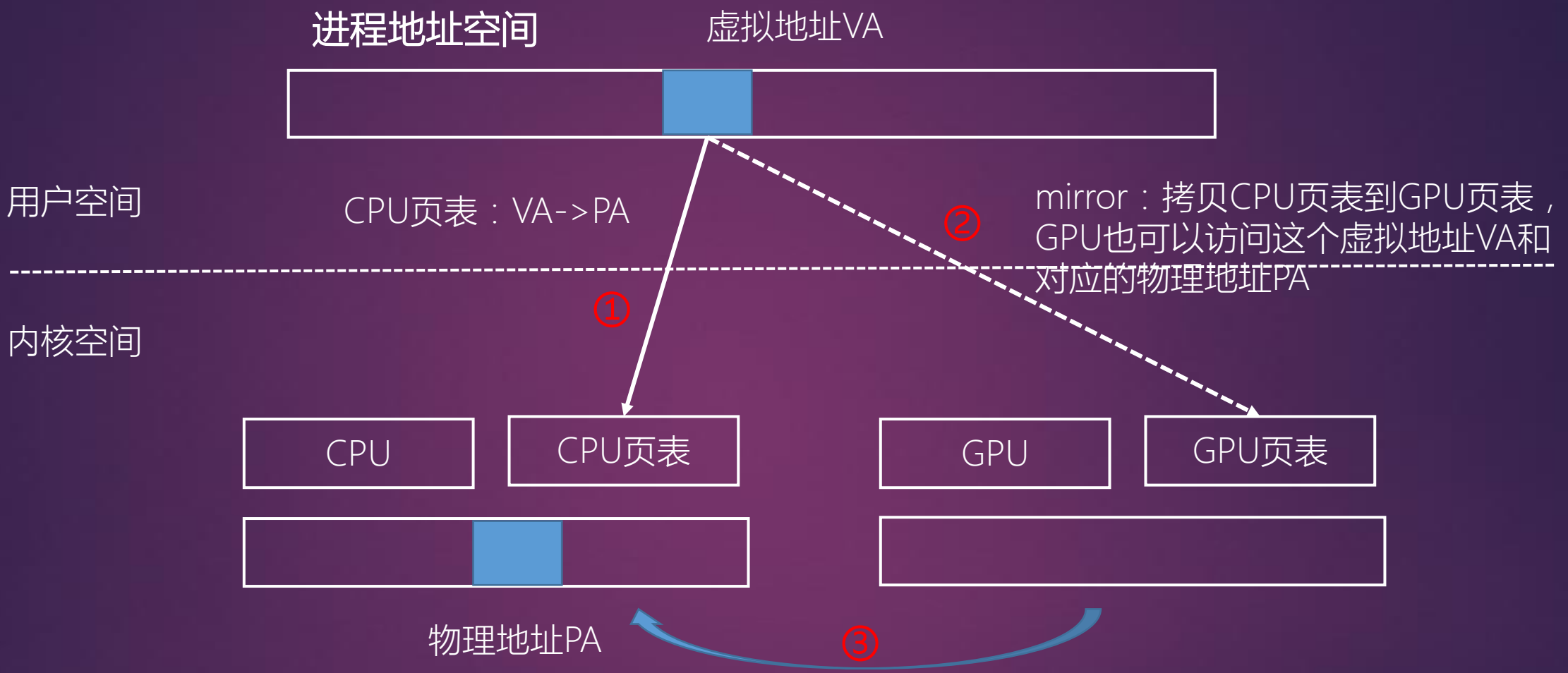
HMM的镜像功能



HMM镜像功能保证：CPU和GPU访问同一个虚拟地址时都能指向相同的物理地址

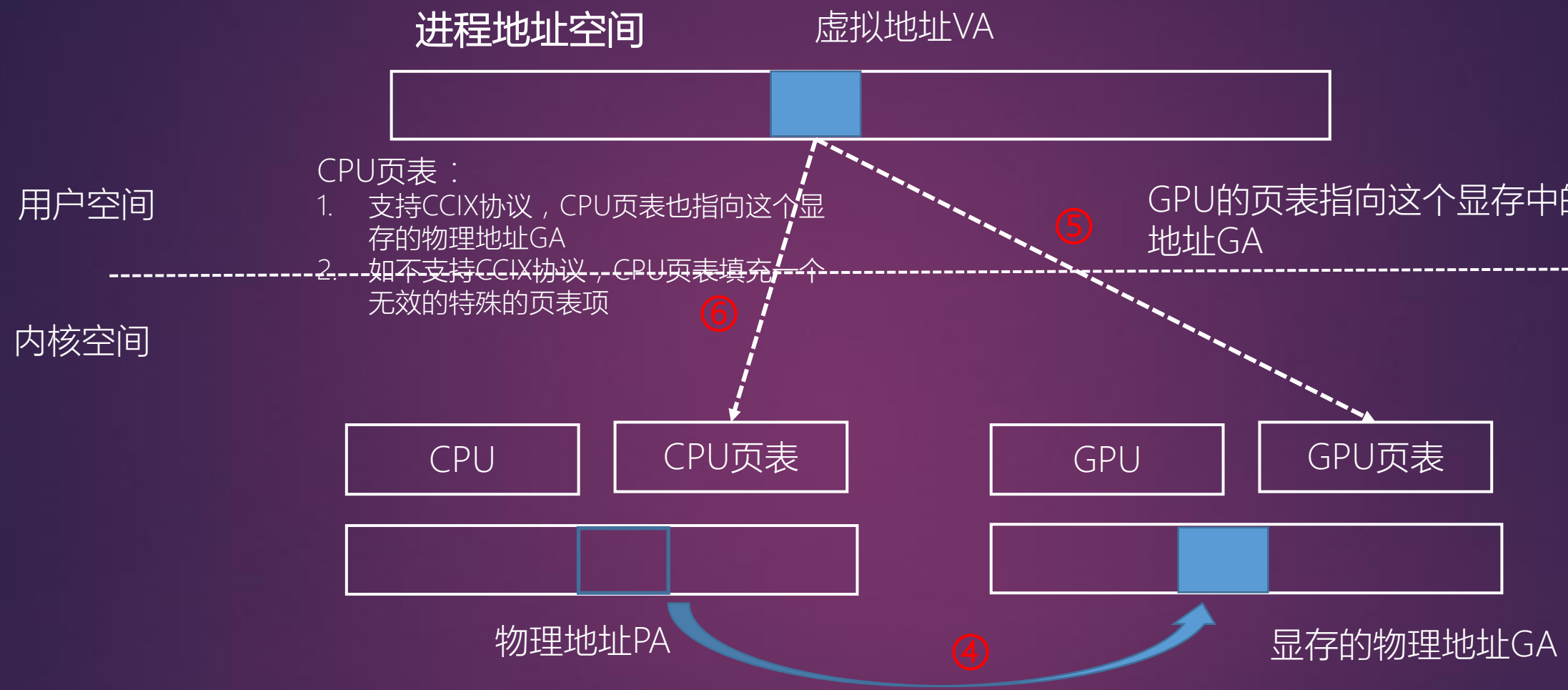
- 那么GPU可以访问这个PA

HMM的迁移功能



- 那么GPU可以访问这个PA

HMM的迁移功能



- 显卡驱动可以迁移这个PA的内容到GPU显存GA里，这样可以利用显存的高带宽
- 如果支持CCIX协议，那么CPU和GPU的页表都能同时指向这个GA
- 如果不支持CCIX协议，那么CPU的页表项要填充一个无效的特殊的页表项，当CPU访问这个虚拟地址时候，会触发一个缺页中断，然后把GA的内容迁移回主存

基于node节点的页面回收机制

- 原来的页面回收机制都是基于ZONE（内存管理区域）
  - 在32位系统里，内核地址空间有限，所以高端内存分在了ZONE_HIGHMEM
- 同一个node里不同的ZONE存在不同的页面老化速度（page age speed）
  - 不同的zone的页面扫描的速度和覆盖率不一致
  - zone里的LRU链表扫描覆盖率也不一致
- 页面回收内核线程kswapd和页面分配器之间复杂的逻辑关系，社区里增加了很多诡异的patch
- Linux 4.8里已经支持基于node节点的页面回收机制

OOM Killer改进

- OOM检测机制
  - Linux 4.7合并的新机制。
  - 在直接回收机制分配内存失败时通常会直接调用OOM，有点鲁莽
  - 新机制，多了一下检测和尝试

- OOM收割机
  - OOM Killer有时候不能回收进程的内存
  - 新机制：在杀死进程之前，先收割进程的匿名页面

swap功能改进

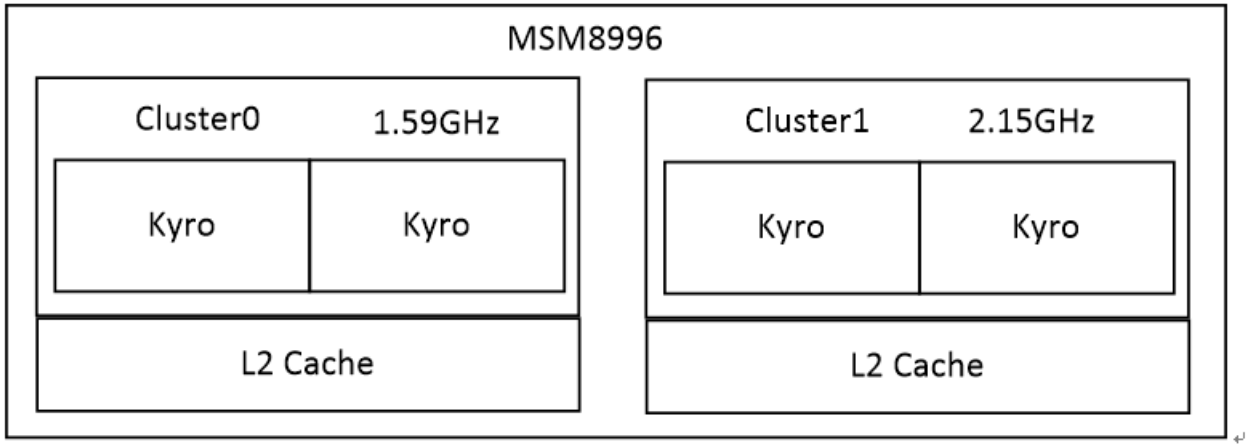
- 提高页面回收效率：
  - 优化LRU算法和页面回收机制
  - 优化SWAP性能

- 随着SSD的普及， swap的性能可以随着优化
  - 解决思路：细化锁，减少锁的争用
  - 细化：swap_info_struct->spinlock
  - 细化：保护address_space的一个锁

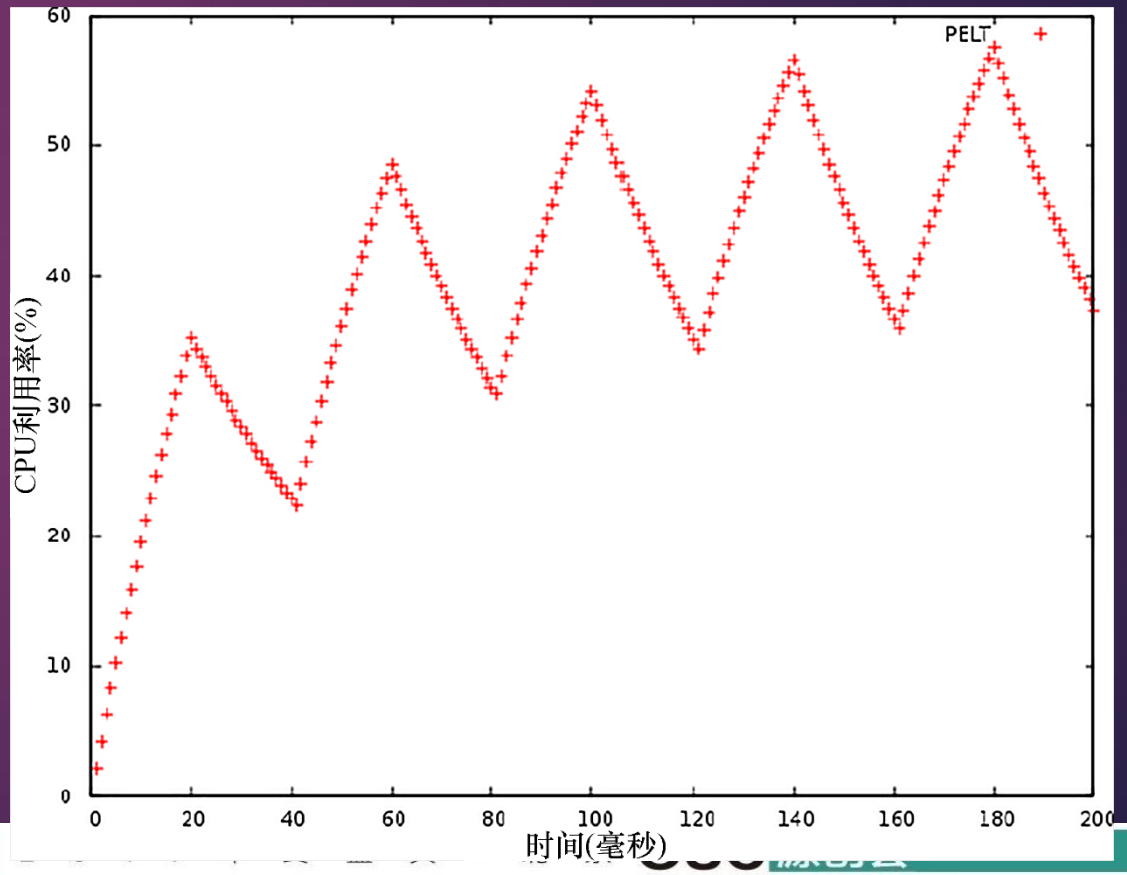
进程管理改进

现有调度器的缺点

- 现有的SMP调度器不支持大小核架构
- 调度器和CPUfreq以及CPUidle等动态调频调压的驱动模块脱节
- 没有考虑功耗场景
- PELT（Pre-entity Load Tracking）负载计算方法会导致负载反应比较迟钝，不太适应手机场景



高通骁龙820处理器架构



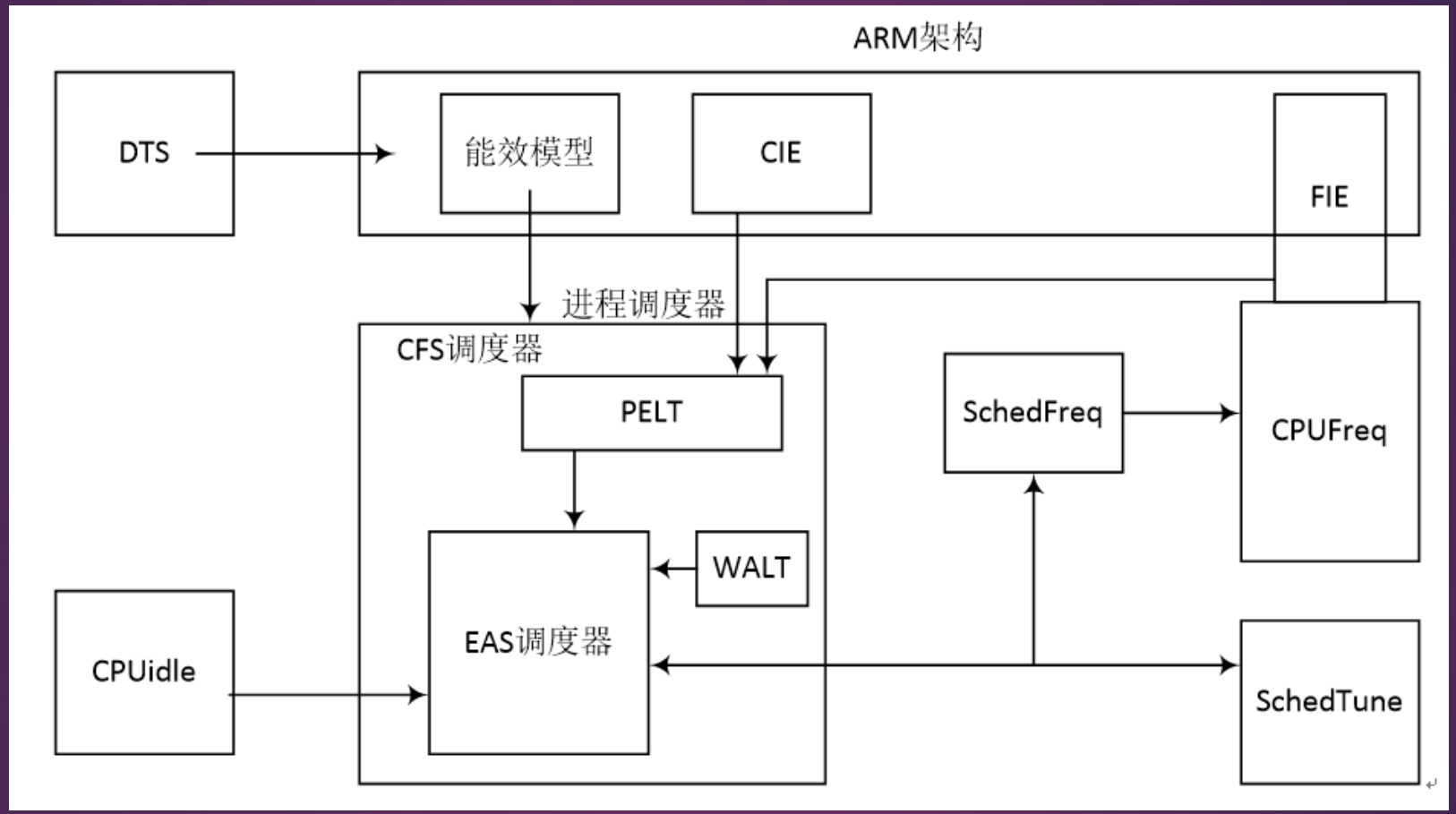
EAS调度器

- EAS（Energy Awareness Scheduling）绿色节能调度器

- 设计目标：
  - 保证系统性能的前提下尽可能减低功耗
  - 统一调度器、CPUfreq和CPUidle

- 核心实现
  - WALT（Window Assisted Load Tracking）算法
  - 可量化的能效模型
  - 可量化的计算能力
  - 复用Linux内核的CPU调度域模型
  - 新增的CPU frequency government

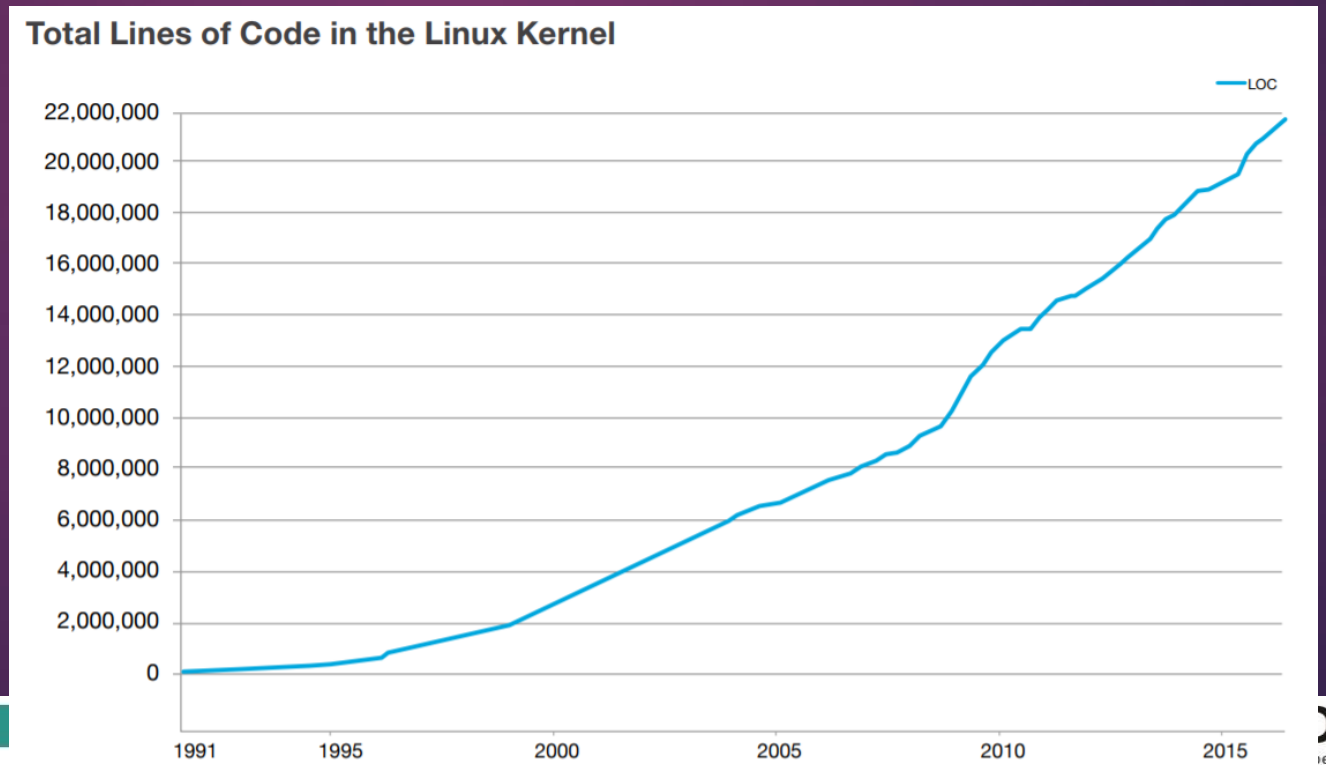
EAS调度器架构图


ARM架构
DTS
能效模型
CIE
FIE
进程调度器
CFS调度器
PELT
SchedFreq
CPUFreq
WALT
CPUidle
EAS调度器
SchedTune

Linux内核学习方法介绍

内核学习常见的疑惑

- 内核代码太庞大，Linux 4.x内核代码超过了2000万行
- 内核更新速度太快
- 内核核心设计经常变化和优化
- 缺乏最新的内核设计资料和文档
- 市面上大部分Linux内核书籍停留在Linux2.4和Linux2.6内核



- 笨式吊打法 – 带着问题去阅读内核代码
  - 深挖实际项目上遇到的问题
  - 阅读代码产生的疑问
  - 面试或者微信论坛上讨论的问题
  - 《奔跑吧Linux内核》中奔跑卷以及每一章的思考题

动手实践

- 可以从一个简单的驱动开始，比如简单的字符设备驱动
- “Qemu+gdb+eclipse+O0” 调试跟踪内核
- 选择感兴趣的内核模块进行调试或者修改

归纳总结

- 数据结构关系图
- 函数调用关系图

兴趣　兴趣　兴趣

Linux内核学习资料推荐

〈奔跑吧Linux内核〉