

# 计算机组成和体系结构

## 第十一讲

四川大学网络空间安全学院

2020年5月11日

封面来自weblistposting.com

# 版权声明

---

课件中所使用的图片、视频等资源版权归原作者所有。

课件原创内容采用 [创作共用署名—非商业使用—相同方式共享4.0国际版许可证\(Creative Commons BY-NC-SA 4.0 International License\)](#) 授权使用。

Copyright@四川大学网络空间安全学院计算机组成与体系结构课程组，2020



# 上期内容回顾

---

- 输入输出系统
  - 子系统性能与计算机系统整体性能的关系：阿姆达尔定律
- I/O子系统组成和体系结构
  - I/O子系统示例
  - 如何控制I/O设备：五种控制方法、字符和块I/O、总线控制与时序图
- 数据传输模式：并行传输和串行传输
- 持久存储介质
  - 硬盘、固态硬盘、光盘、磁带
- 独立磁盘冗余阵列
- 分布式存储简介

# 本期学习目标

---

- 操作系统
  - 操作系统的发展、分类和标志技术
- 保护环境
  - 虚拟机、子系统和逻辑分区
  - XEN、KVM、AIX和Docker
  - Intel VT-x和VT-d

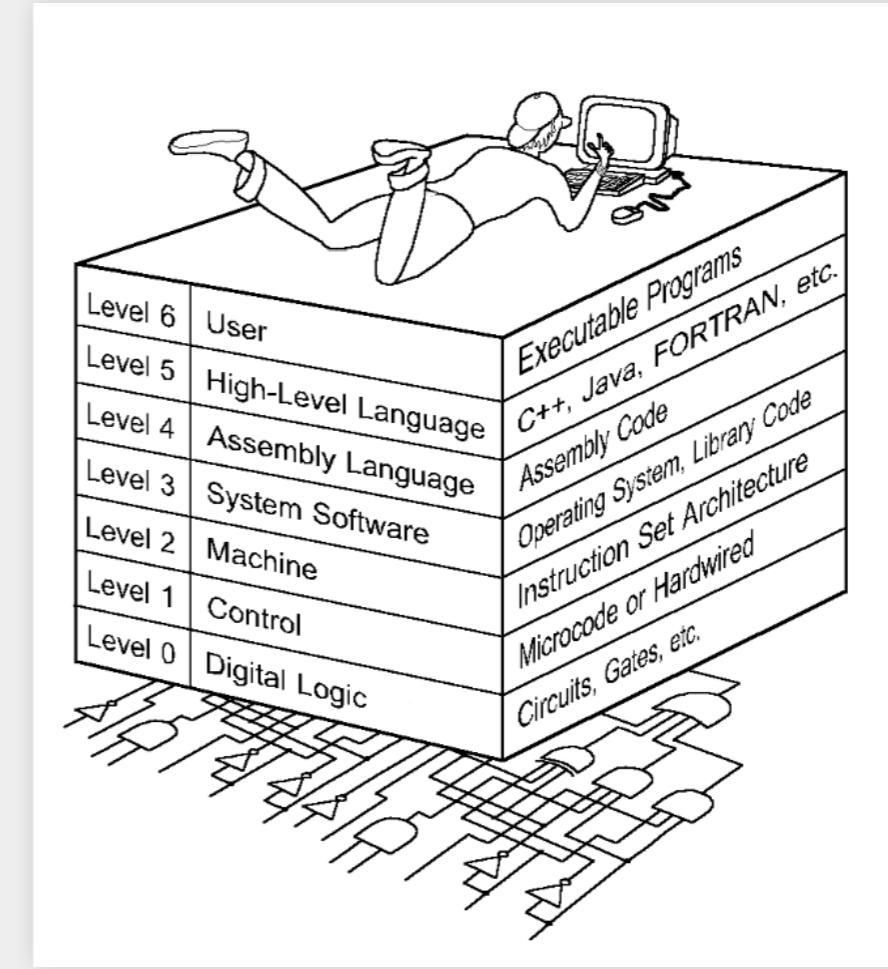
# 中英文缩写对照表

英文缩写	英文全称	中文全称
VM	Virtual Machine	虚拟机
LPAR	Logical Partition	逻辑分区
PV	Paravirtualization	半虚拟化
HVM	Hardware Virtual Machine	硬件虚拟化/全虚拟化
KVM	Kernel-based Virtual Machine	基于内核的虚拟机
VMX	Virtual Machine Extension	虚拟机扩展指令

# 系统软件层

系统软件层位于机器层之上、汇编语言层之下，是连接计算机硬件实现和用户实际服务的重要组成部分。

其中操作系统是系统软件层最具代表性和最重要的组成部分。



# 操作系统的 历史

---

**Operating Systems timeline (<https://prezi.com/f4ajvsukiz-c/>)** By Dan Taune



# 操作系统的标志性技术

---

- 多任务(multitasking)
- 多用户(multi-user)
- GUI
- 抢占式(preemptive)
- **实时(Real-time)**
- **内核(Kernel)**

# 多任务

---

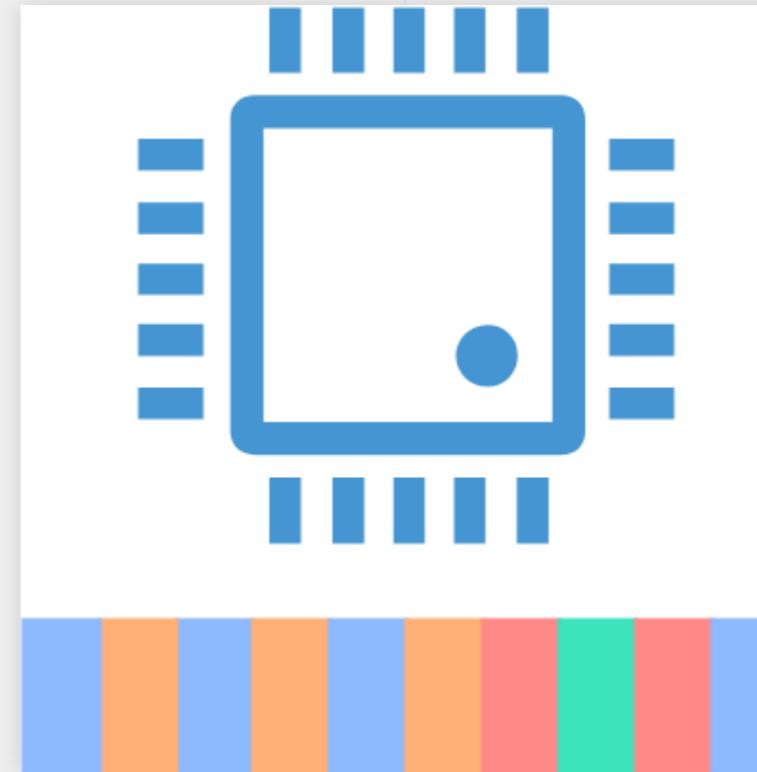
- 早期的操作系统采用的是监视器模式：每次执行一个命令
- 多任务操作系统可以“同时”执行多个程序
- 主要是通过时间分片共享(timesharing)实现的
- 现代计算机操作系统几乎都是多任务操作系统

# 时分系统

---

假设右图中CPU主频为1 GHz，将一秒钟分为十个时间片，分配给四个进程。

等价于四个进程同时使用一个单独的CPU，频率分别为400 MHz（蓝色），300 MHz（黄色）、200 MHz（红色）和100 MHz（绿色）。



# 多用户

---



- 公用计算机上可能有多个用户
- 恶意用户可能危害系统安全、数据安全
- 权限管理、资源隔离成为操作系统的重要功能
- 保护环境

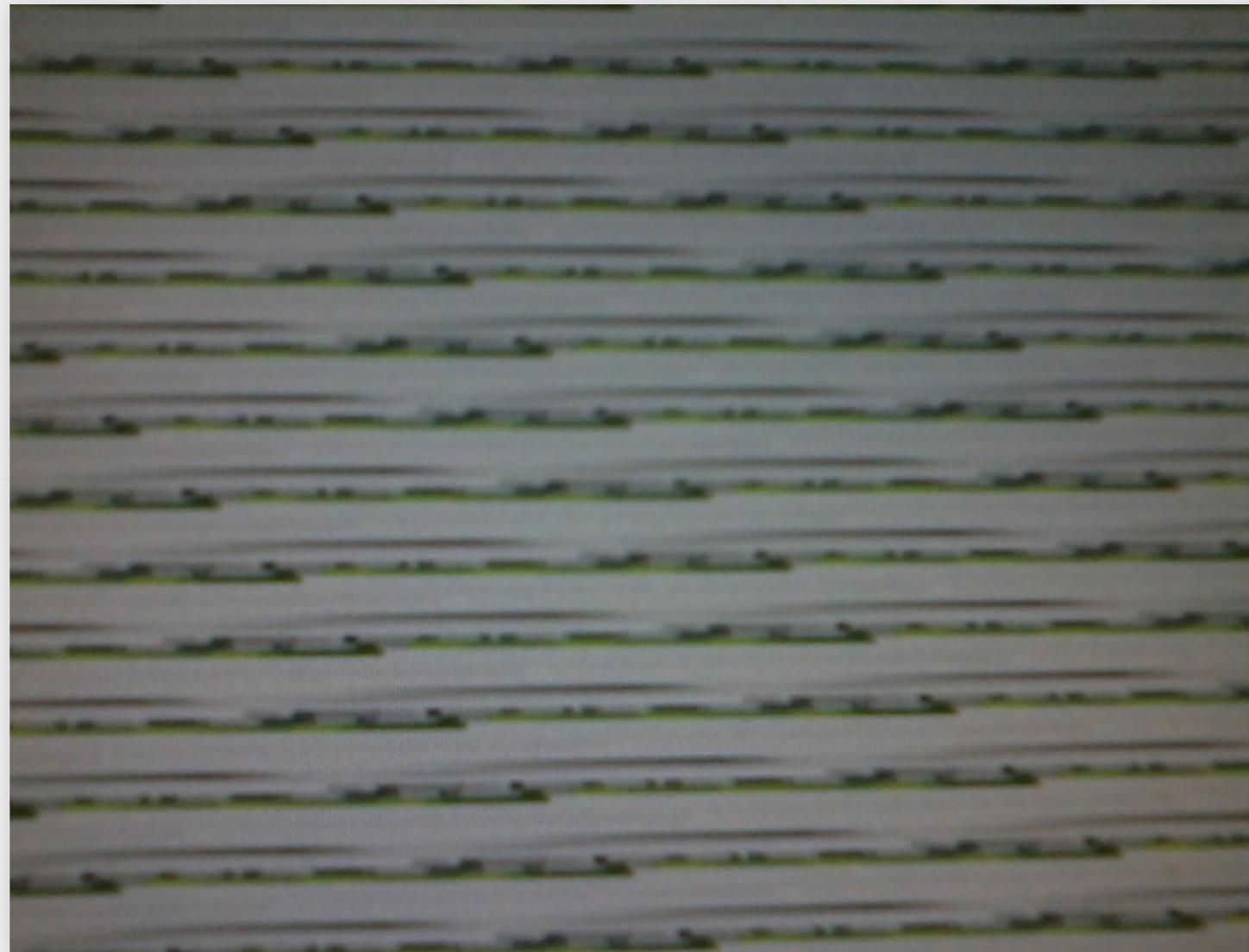
# 图形界面

- 图形界面的作用：人机交互模式的改变
- 现代个人计算机的普及很大程度上得益于图形界面
- 几乎所有现代个人计算机操作系统都带有图形界面



# 图形界面 (2)

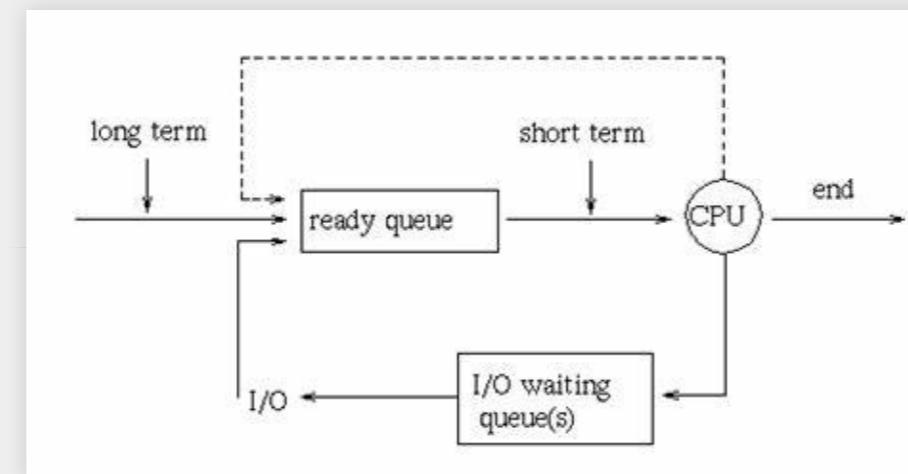
---



成功配置好图形界面是学会使用Linux的重要标志。

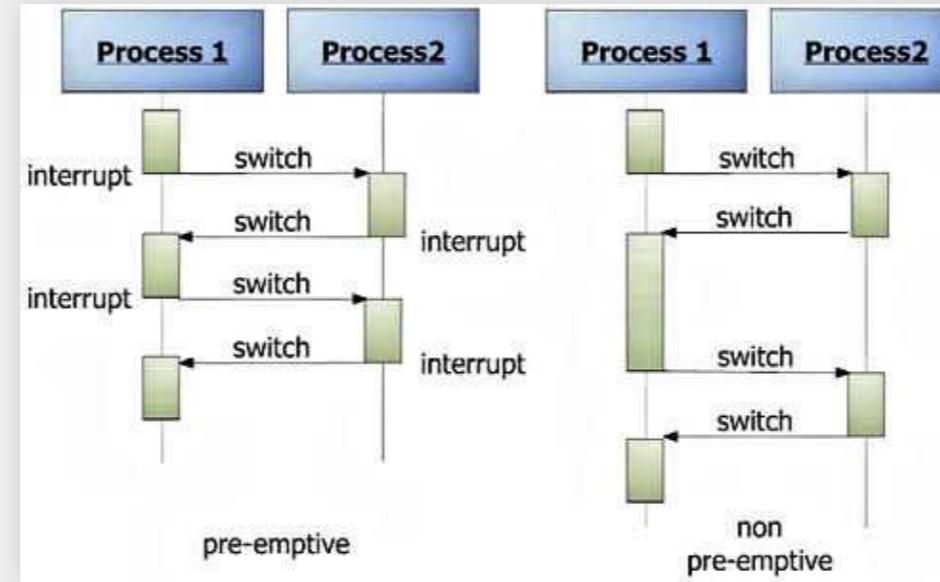
# 任务调度

- 多任务系统对CPU资源的划分依靠任务调度
- 长程调度(long-term scheduling): 决定哪些线程可以放到内存中准备执行
- 短程调度(short-term scheduling): 决定哪些线程可以使用当前的CPU时间片
- 抢占式和非抢占式都属于短程调度



# 抢占式和非抢占式

- 抢占式调度：调度器可以中止正在执行的任务，切换到另一个任务
- 非抢占式调度：调度器总是等当前调度的任务执行完毕才切换到另一个任务



- 注意：进程在调度过程中优先级可能变化

# 任务调度方法

---

- 先到先服务(First-come, first-served): 进程按照到达时间依次进行处理
- 最小任务优先(Shortest job first): 优先调度当前结束时间最短的进程
- 轮询(Round robin): 每个进程分配同等时间片依次进行调度
- 优先级(Priority): 优先调度优先级最高的进程，可以抢占低优先级进程

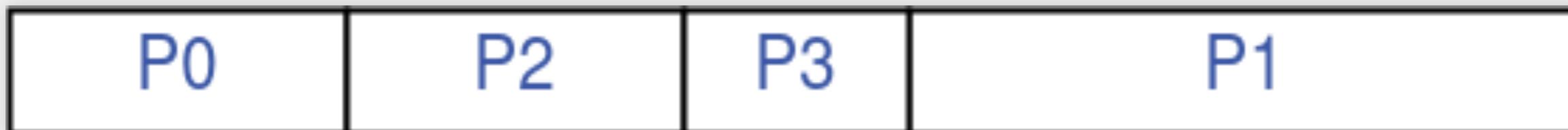
# 先到先服务

进程	到达时间	执行时间	优先级	开始时间	结束时间
P0	0	3	1	0	3
P1	2	6	0	6	12
P2	1	3	2	3	6
P3	4	2	3	12	14



# 最小任务优先

进程	到达时间	执行时间	优先级	开始时间	结束时间
P0	0	3	1	0	3
P1	2	6	0	8	14
P2	1	3	2	3	6
P3	4	2	3	6	8



# 轮询

进程	到达时间	执行时间	优先级	开始时间	结束时间
P0	0	3	1	0	9
P1	2	6	0	2	14
P2	1	3	2	1	10
P3	4	2	3	3	8



# 优先级调度

进程	到达时间	执行时间	优先级	开始时间	结束时间
P0	0	3	1	0	9
P1	2	6	0	2	8
P2	1	3	2	9	12
P3	4	2	3	12	14



# 实时系统

---

- 在一些场景下，某些任务必须在特定的时间内完成
- 例如工控系统、机器人等
- 实时系统经常被用在嵌入式系统中

# 内核

---

- 内核实现了操作系统的功能，一般可以分为两类
- 微内核(microkernel): 内核只实现核心功能
  - 通常安全性更好，但是性能相对较差（某些常用功能运行在用户态）
  - 例子：Windows 2000, GNU/Hurd等
- 宏内核(monolithic): 内核中还包含其它一部分功能，例如IO等
  - 性能相对更高，安全性相对更差（例如某些驱动问题可以导致系统死机）
  - 例子：GNU/Linux、Max OS等

# 保护环境

保护环境，~~人人~~有责  
操作系统

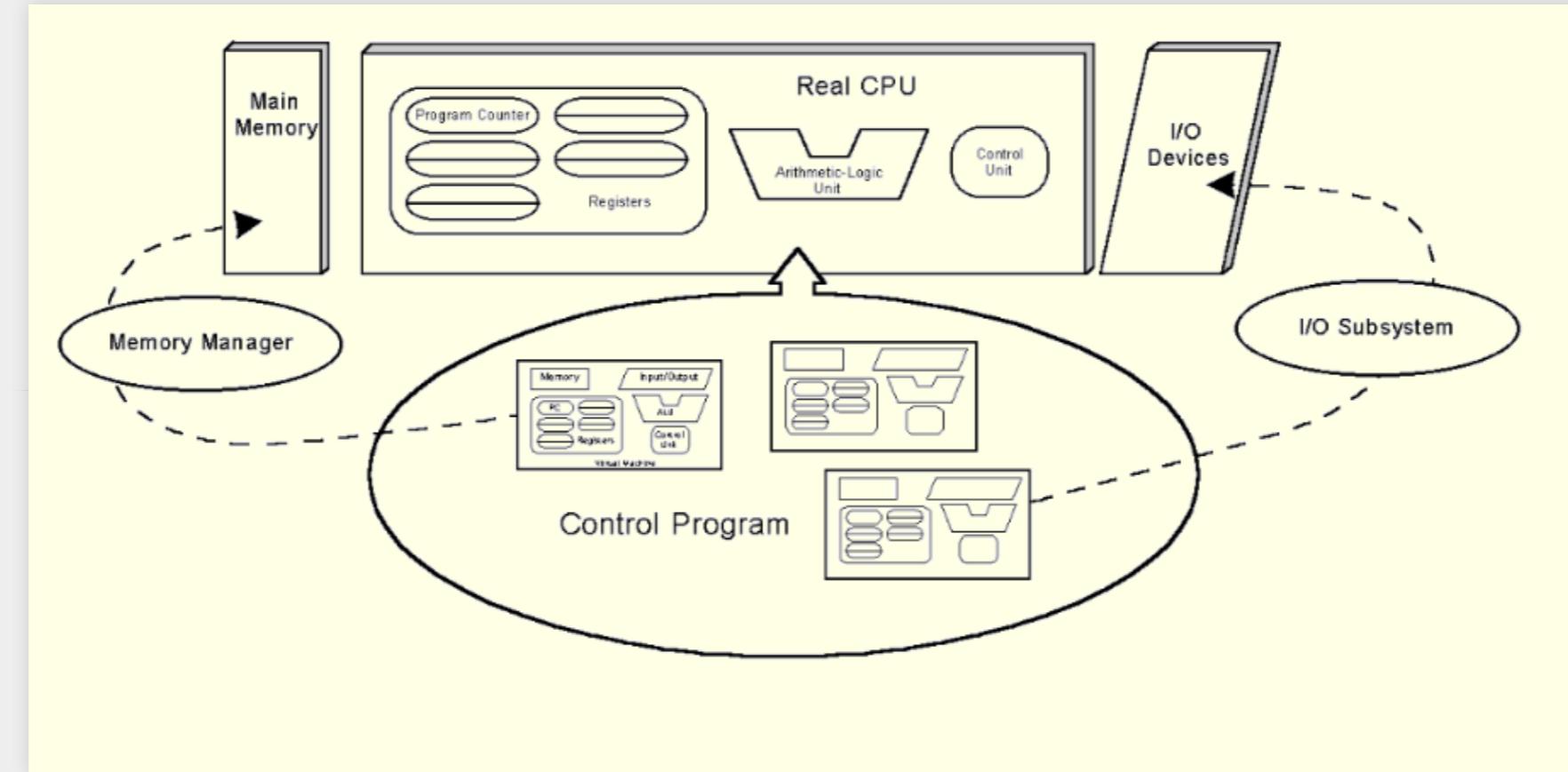
- 受到操作系统**保护**的**环境**(Protected Environments)
- 通常需要将资源、进程、数据等进行隔离
- 实现保护环境的方法
  - 虚拟机(Virtual Machine, VM)
  - 子系统(Subsystem)
  - 逻辑分区(Logical Partitioning, LPAR)

# 虚拟机

---

- 一个虚拟机对于用户进程来说等价于一台物理机
- 虚拟机技术有多种实现方式，从纯软件到硬件支持
- 虚拟设备将相关请求发送给宿主机操作系统，由宿主机操作系统进行实际操作

# 虚拟机示意图

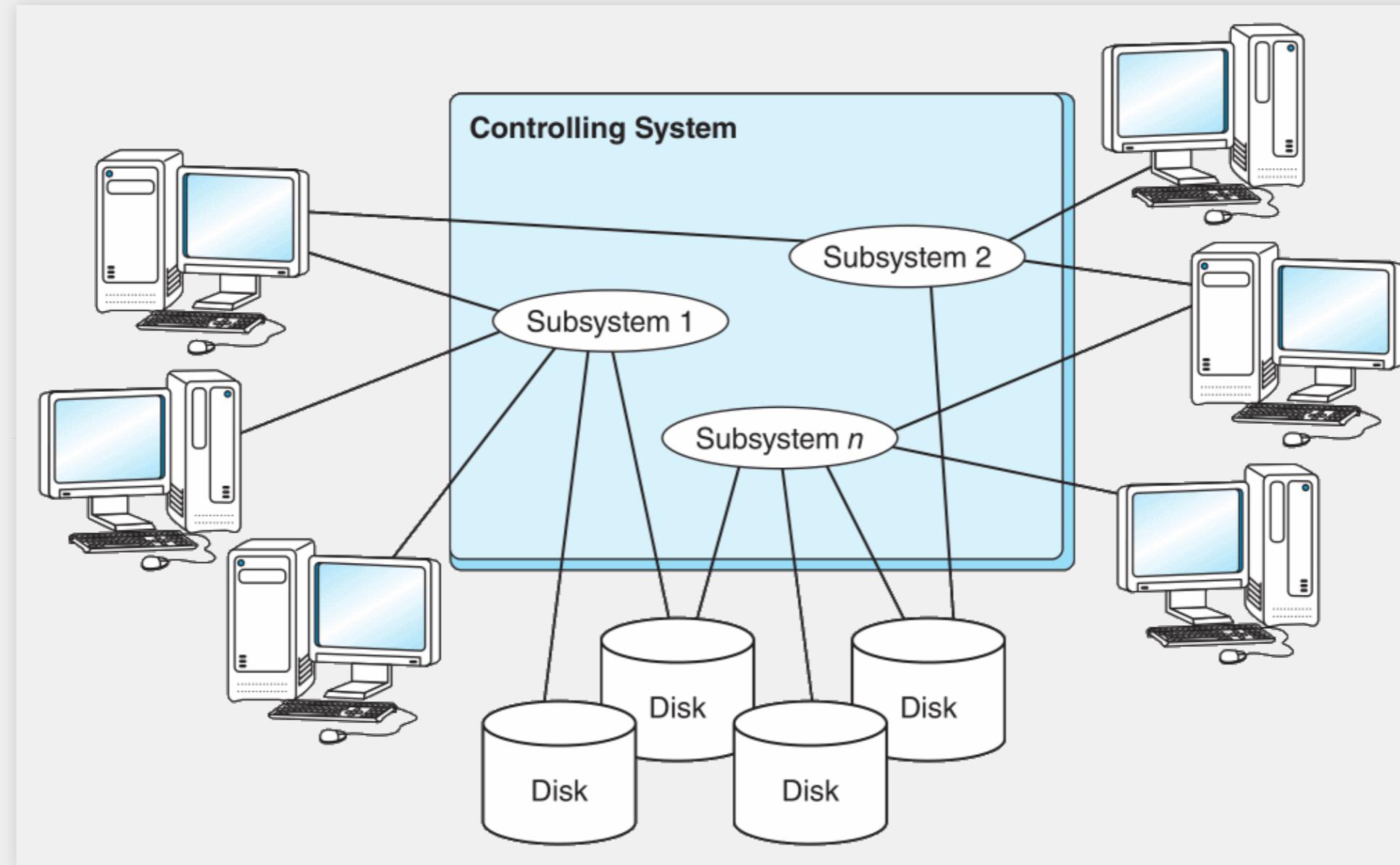


# 子系统

---

- 操作系统将资源划分到若干逻辑上独立的环境中
- 每个子系统可以独立地被控制和管理
- 例子：
  - Windows Subsystem for Linux (WSL)
  - Linux Networking Subsystem

# 子系统示意图

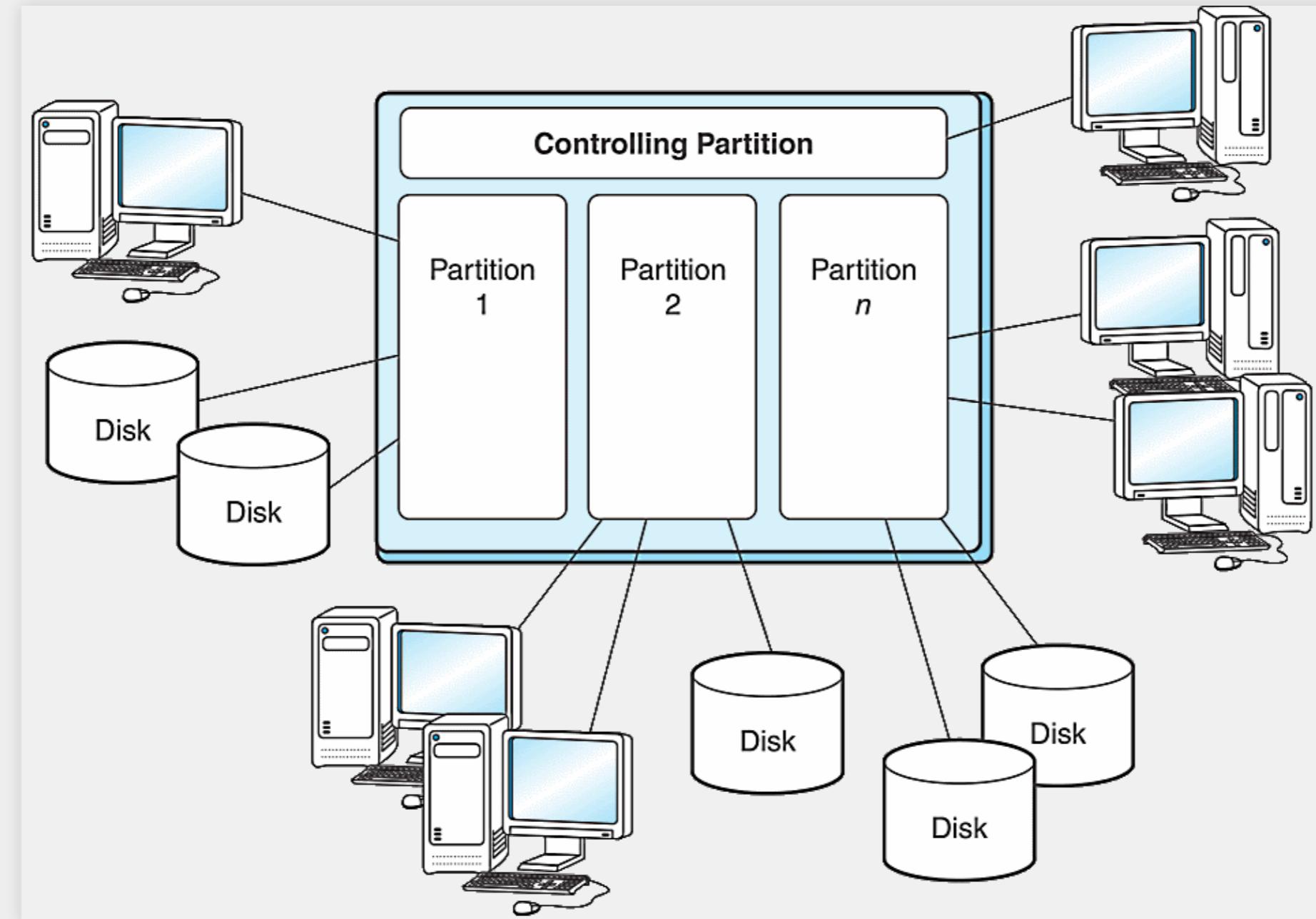


# 逻辑分区

---

- 逻辑分区(logical partitioning, LPAR): 属于不同逻辑分区的进程等同于运行在不同设备上
- 是系统级别的保护环境，注意和硬盘的逻辑分区进行区别

# 逻辑分区示意图

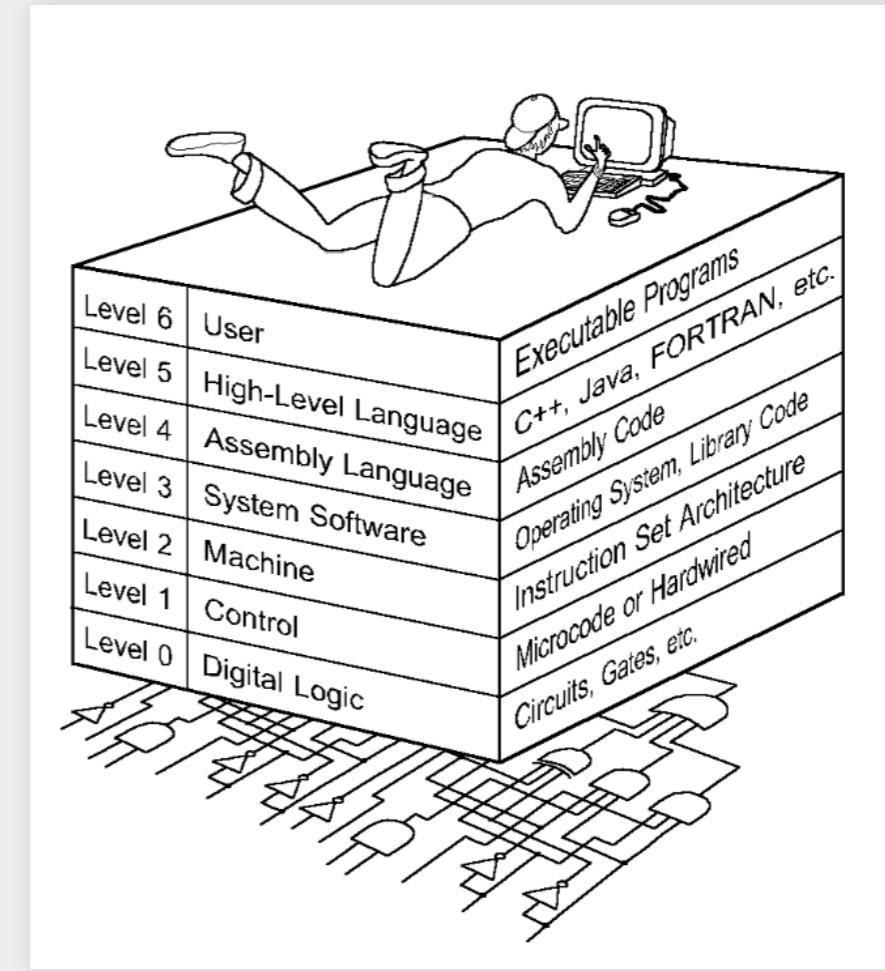


# 虚拟化技术

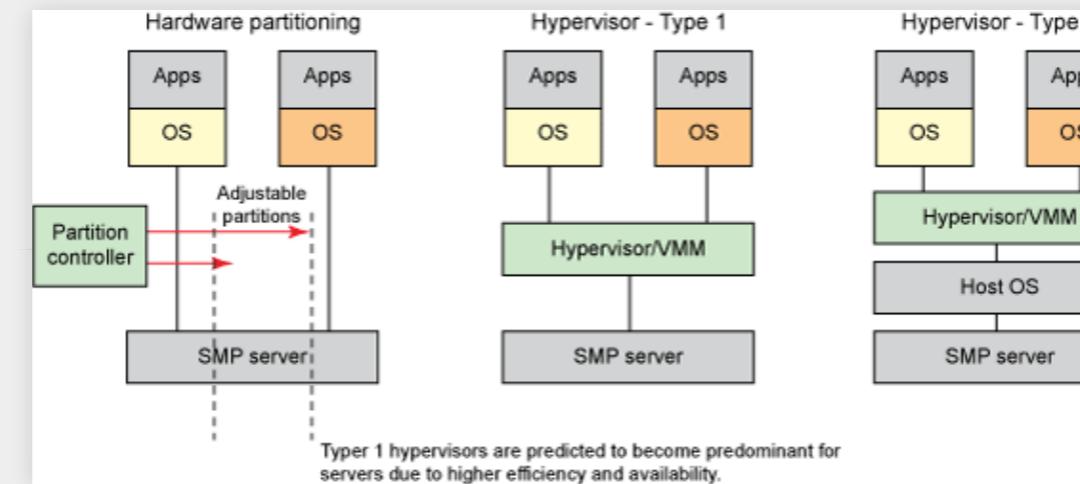
在云计算中提供了不同层次的“虚拟化”。

更好地支持上层的虚拟化是现代CPU设计的一个重要方面。

- 虚拟化技术：XEN、KVM、AIX
- 处理器上的虚拟化支持：VT-x、VT-d

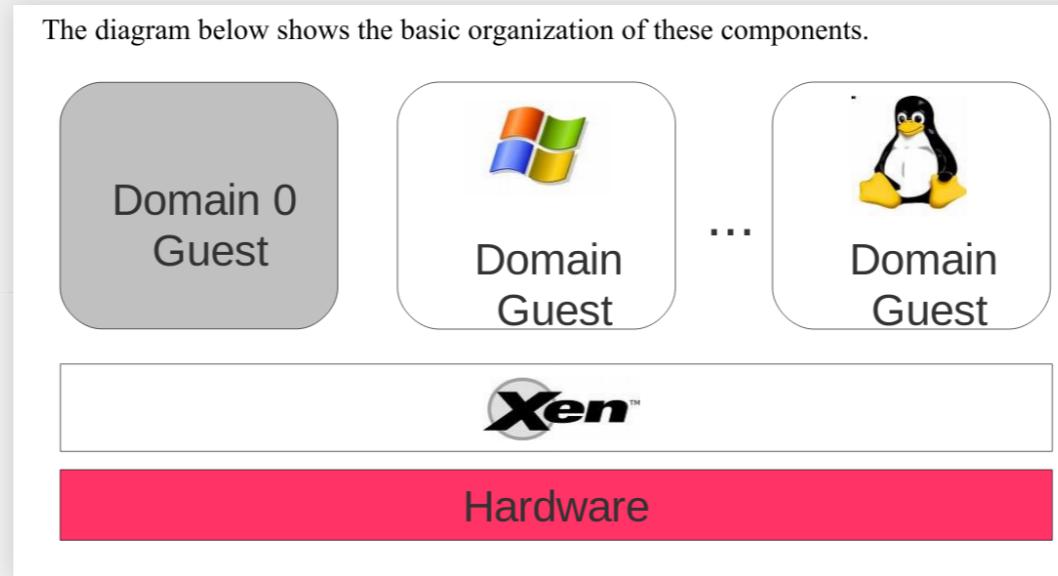


# 虚拟化技术的分类



图片来源：<https://www.ibm.com/developerworks/aix/library/au-aixhypervirtualization/index.html>

# XEN

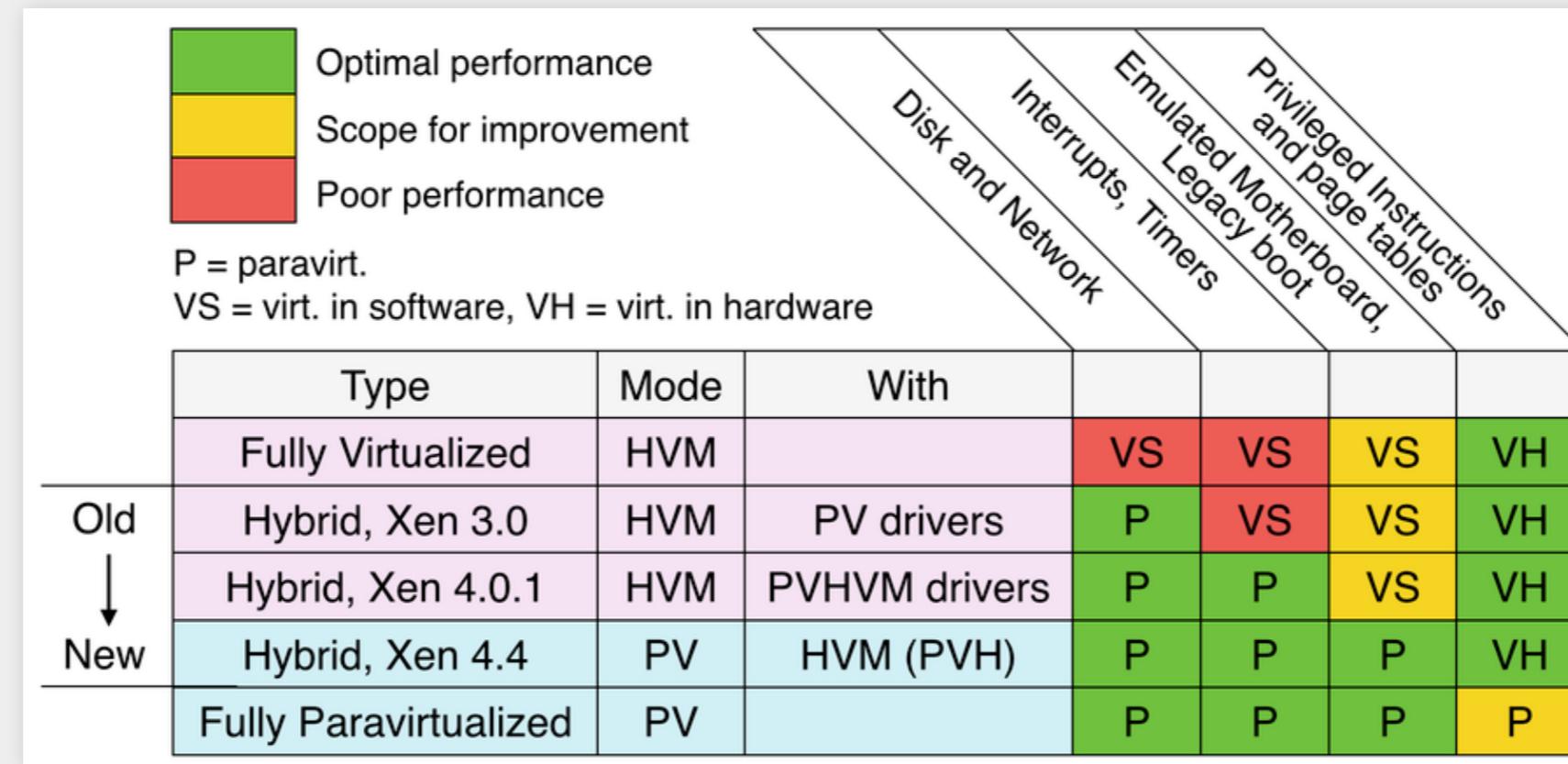


- Domain 0: 具有最高权限的虚拟机，可以访问实际物理资源，起到虚拟机管理的作用
- Domain U: 无权限的虚拟机，采用全虚拟化技术称为Domain U HVM Guest，采用半虚拟化的虚拟机称为Domain U PV Guest

图片来源：<http://www-archive.xenproject.org/files/Marketing/HowDoesXenWork.pdf>

# XEN (2)

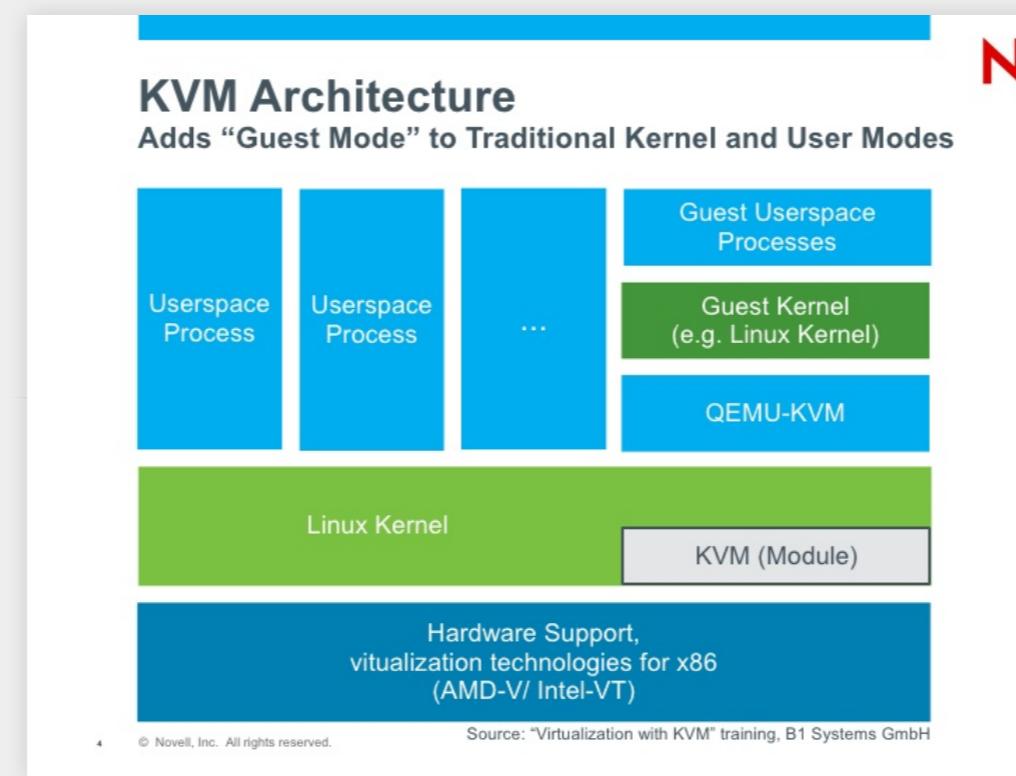
- PV和HVM都是为了提高虚拟机的性能，避免软件模拟
- 硬件对虚拟化更好的支持提高了虚拟机的性能



图片来源: [https://wiki.xenproject.org/wiki/Understanding\\_the\\_Virtualization\\_Spectrum](https://wiki.xenproject.org/wiki/Understanding_the_Virtualization_Spectrum)

# KVM

- KVM全称为Kernel-based Virtual Machine
- 基于Linux，实现在Linux内核中

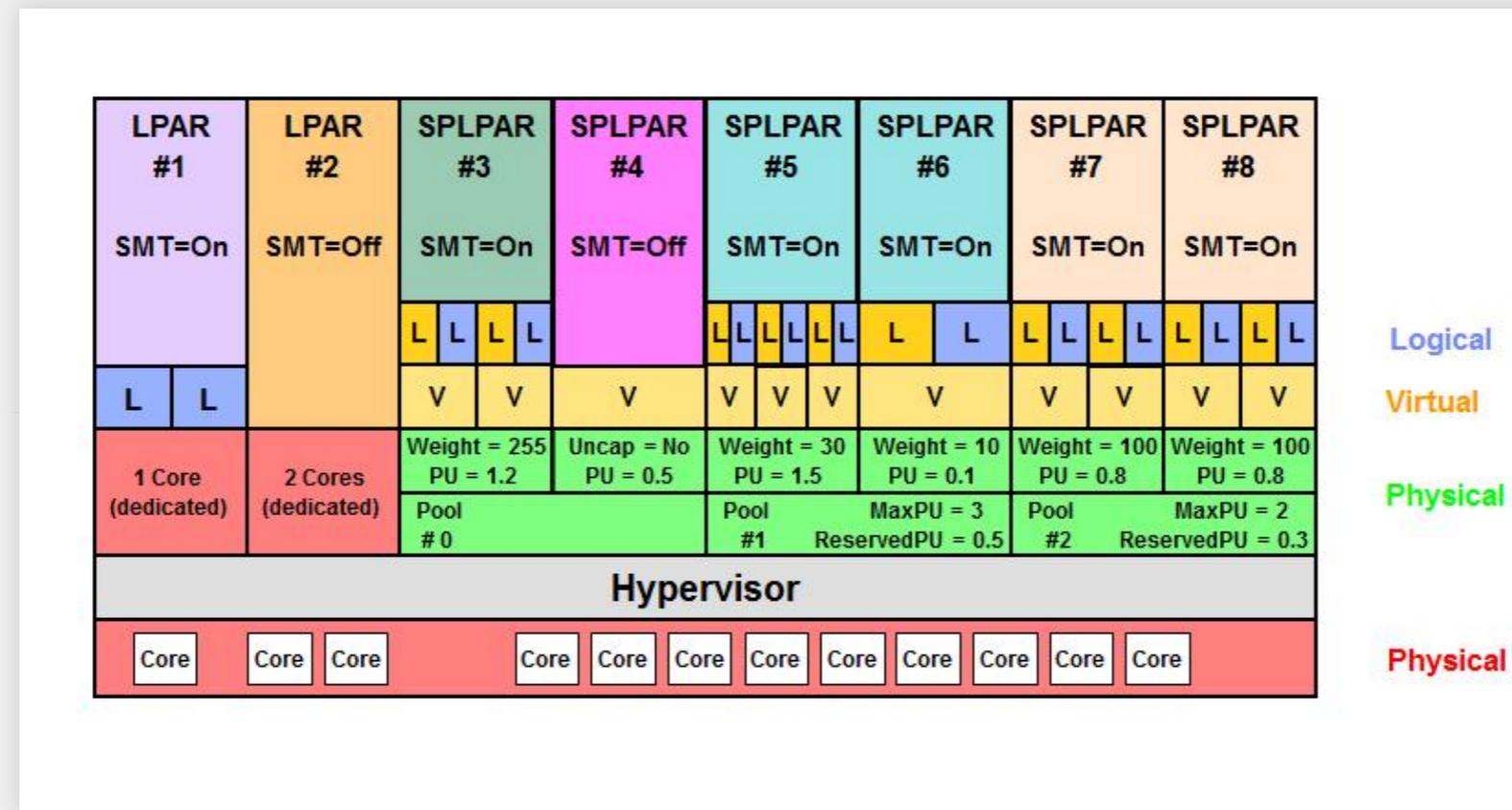


图片来源: Novell. KVM Architecture. [Online]. <https://www.slideshare.net/NOVL/virtualization-with-kvm-kernelbased-virtual-machine>

# AIX

---

- AIX全称Advanced Interactive eXecutive，是IBM开发的一个Unix系统
- AIX运行在IBM的PowerPC体系结构上
- AIX支持多种虚拟化配置

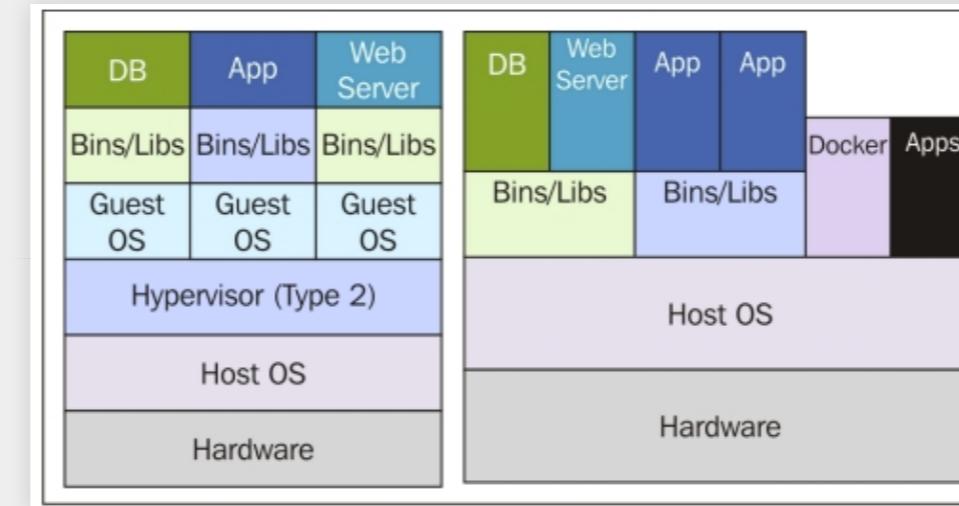


图片来源: db2tutorial.net

# DOCKER

---

- docker是容器虚拟化的代表
- 容器是一种轻量化的虚拟方案，不需要创建运行新的OS，而是对软件应用和运行库进行虚拟化
- 其它容器解决方案：LXC、OpenVZ、AIX WPAR等



图片来源：<https://jaxenter.com/containerization-vs-virtualization-docker-introduction-120562.html>

# INTEL VT-X

---

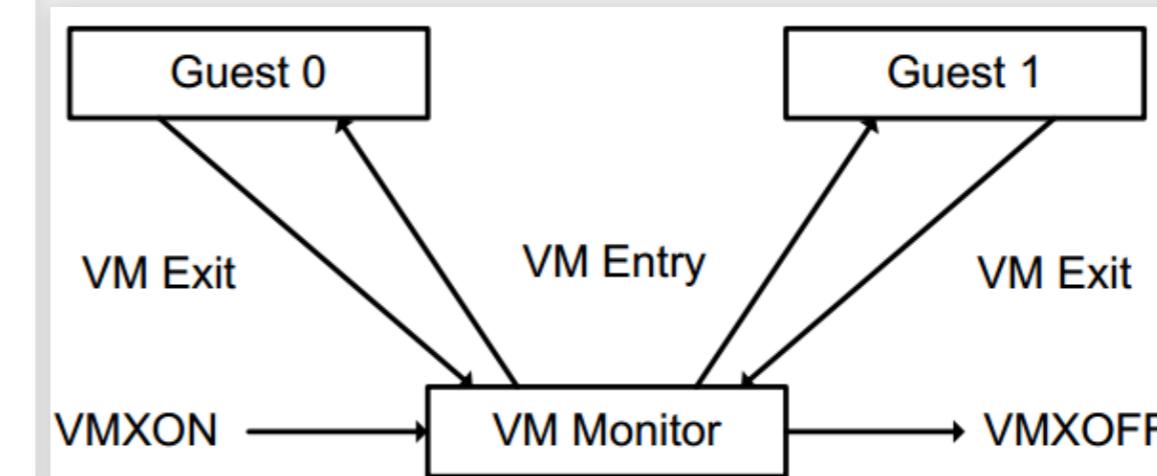
- 硬件可以通过支持虚拟化提高虚拟机的性能
- 例如Intel的VT-x就是对x86指令集的虚拟化支持
- 增加了额外的VMX指令用于进行虚拟机的管理和运行
- 支持扩展页表(Extended Page Table)，每个虚拟机拥有独立的页表
- VMX指令分为root操作和非root操作
- 权限的转换通过VM entry和VM exit指令完成

# INTEL VMX

## VMX Instructions

VMX introduces the following new instructions.

Intel/AMD Mnemonic	Description
INVEPT	Invalidate Translations Derived from EPT
INVPID	Invalidate Translations Based on VPID
VMCALL	Call to VM Monitor
VMCLEAR	Clear Virtual-Machine Control Structure
VMFUNC	Invoke VM function
VMLAUNCH	Launch Virtual Machine
VMRESUME	Resume Virtual Machine
VMPTRLD	Load Pointer to Virtual-Machine Control Structure
VMPTRST	Store Pointer to Virtual-Machine Control Structure
VMREAD	Read Field from Virtual-Machine Control Structure
VMWRITE	Write Field to Virtual-Machine Control Structure
VMXOFF	Leave VMX Operation
VMXON	Enter VMX Operation

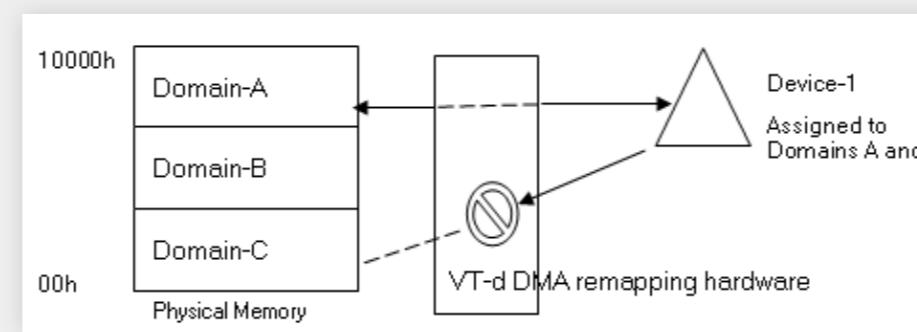


图片来源：<https://rayanfam.com/topics/hypervisor-from-scratch-part-1/>

# INTEL VT-D

---

- VT-d代表Virtualization with Directed I/O
- 实现DMA的硬件重映射(DMA remapping)和IO直接分配
- 在DMA重映射的同时对系统资源进行保护
- 例如下图中物理内存被分为三个保护域，虚拟设备1只能访问Domain A和B，在虚拟设备1进行地址转换时会在硬件层面检查访问权限

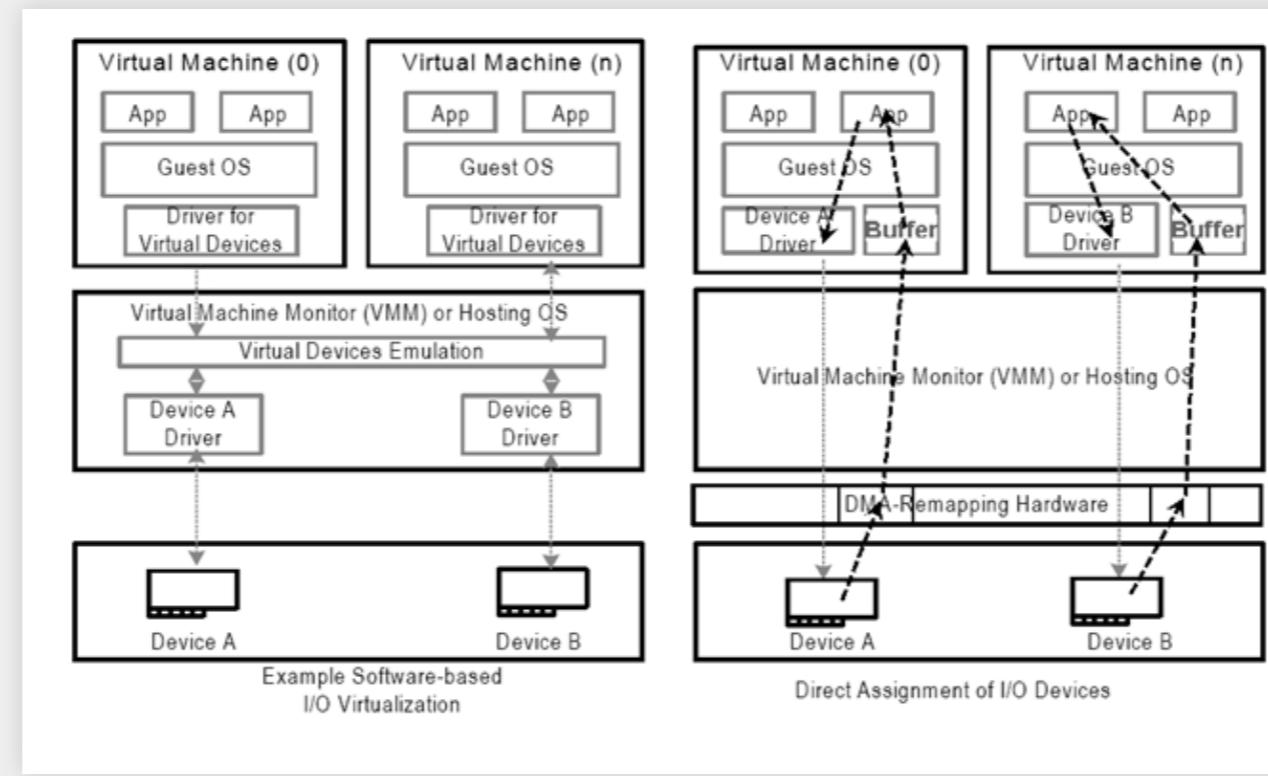


图片来源：<https://software.intel.com/content/www/us/en/develop/articles/intel-virtualization-technology-for-directed-io-vt-d-enhancing-intel-platforms-for-efficient-virtualization-of-io-devices.html>

# INTEL VT-D

---

- 左边是使用软件模拟虚拟设备的数据传输流程
- 右图是使用IO直接分配的数据传输流程，允许虚拟机通过DMA将数据写入保护的内存，达到写入设备的效果



图片来源：<https://software.intel.com/content/www/us/en/develop/articles/intel-virtualization-technology-for-directed-io-vt-d-enhancing-intel-platforms-for-efficient-virtualization-of-io-devices.html>

# 第十一讲结束

# 本期内容总结

---

- 操作系统
  - 操作系统的发展、分类和标志技术
- 保护环境
  - 虚拟机、子系统和逻辑分区
  - XEN、KVM、AIX和Docker
  - Intel VT-x和VT-d



# Q & A