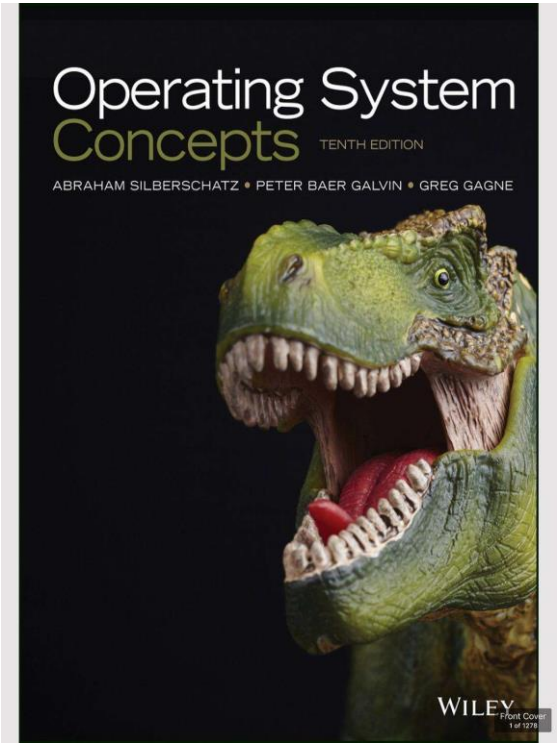
操作系统 D类 SJTU CS2302-3

- QQ群号：

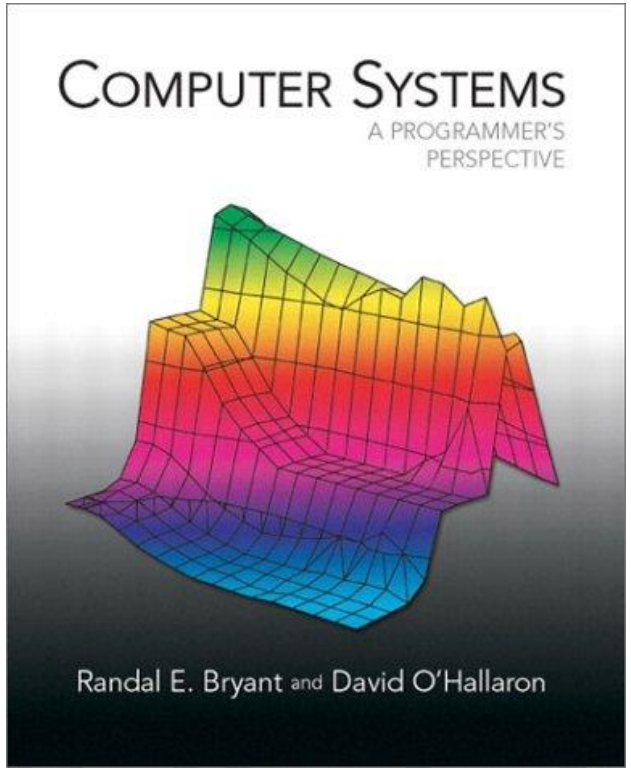
- 课程canvas主页：
- **https://oc.sjtu.edu.cn/courses/40736**
- 教师：邓倩妮
  - **qndeng@sjtu.edu.cn**
  - **deng-qn@sjtu.edu.cn**

教材



Required: Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne, "**Operating System Concepts**", 9th or 10th Edition, John Wiley & Sons, Inc.

参考书



Computer Systems: A Programmer's Perspective, 3/E (CS:APP3e)

Final Grades

- Attendance and Quizzes(Close Book, On Classes) 10%
  - Qizz 会提前两周通知
- Homework 15%
  - Three assignments
- Project 15% (Presentation and Report)
  - 3 projects
  - Some students will be selected to give presentation on classes
  - 《操作系统实验》可以替代成绩
- Final Exam 60% (Close Book)

Learning by Doing

- Individual Labs and Homework assignments
  - Intro
  - User Programs
  - Build your own Shell
  - Threads
  - Scheduling
  - Memory Mapping and Management
  - File Systems

First Assignments

- “Lab 0” (1 point)

- 实验楼
- 实验课程地址：https://www.shiyanlou.com/courses/1637
- 课程邀请码：YHJK0D6Y

Chapter 1: Introduction

- 什么是操作系统？
- Operating-System Basic Operations
  - Virtualization
  - Resource Management
  - Security and Protection …
- 计算机的基本组成结构
- 计算机的指令系统
- 中断、系统调用的概念
- 用户模式、系统模式的概念
- 操作系统引导的过程
- 其他：虚拟机、分布式操作系统、开源操作系统等等

Operating Systems

Windows

redhat

ubuntu

debian

AIX

openSUSE

AMIGA OS

Chrome OS

fedora

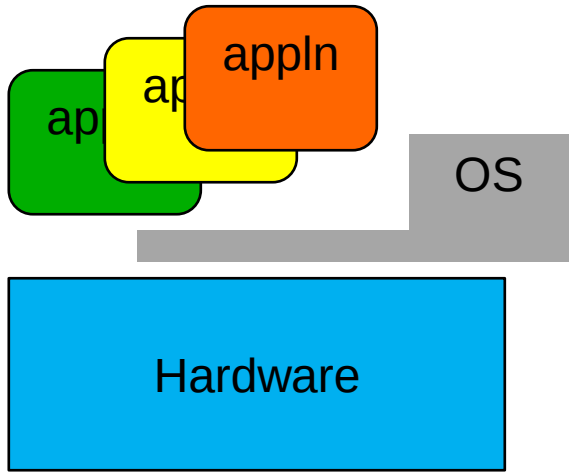
MS DOS

solaris

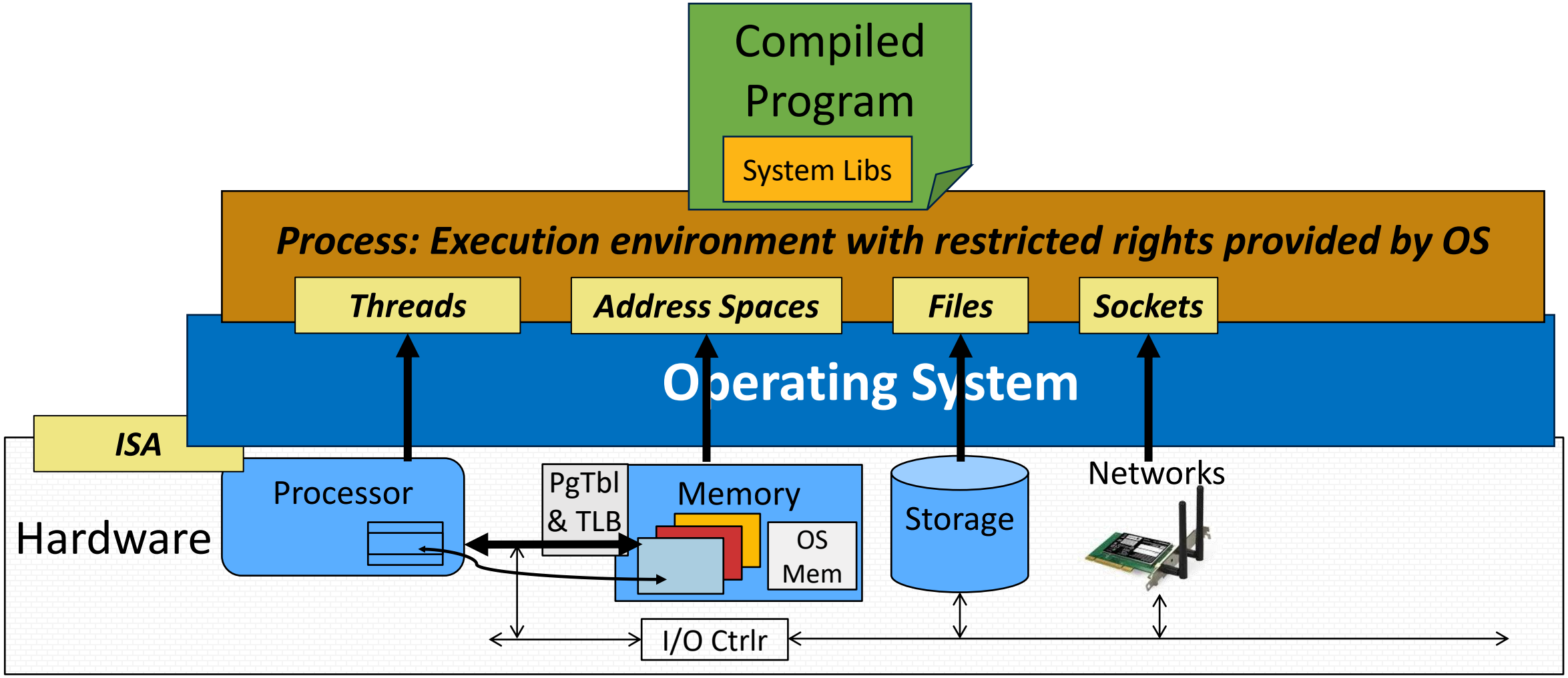
One Definition of an Operating System

- An OS is a program that acts as an intermediary between a user of a computer and the computer hardware

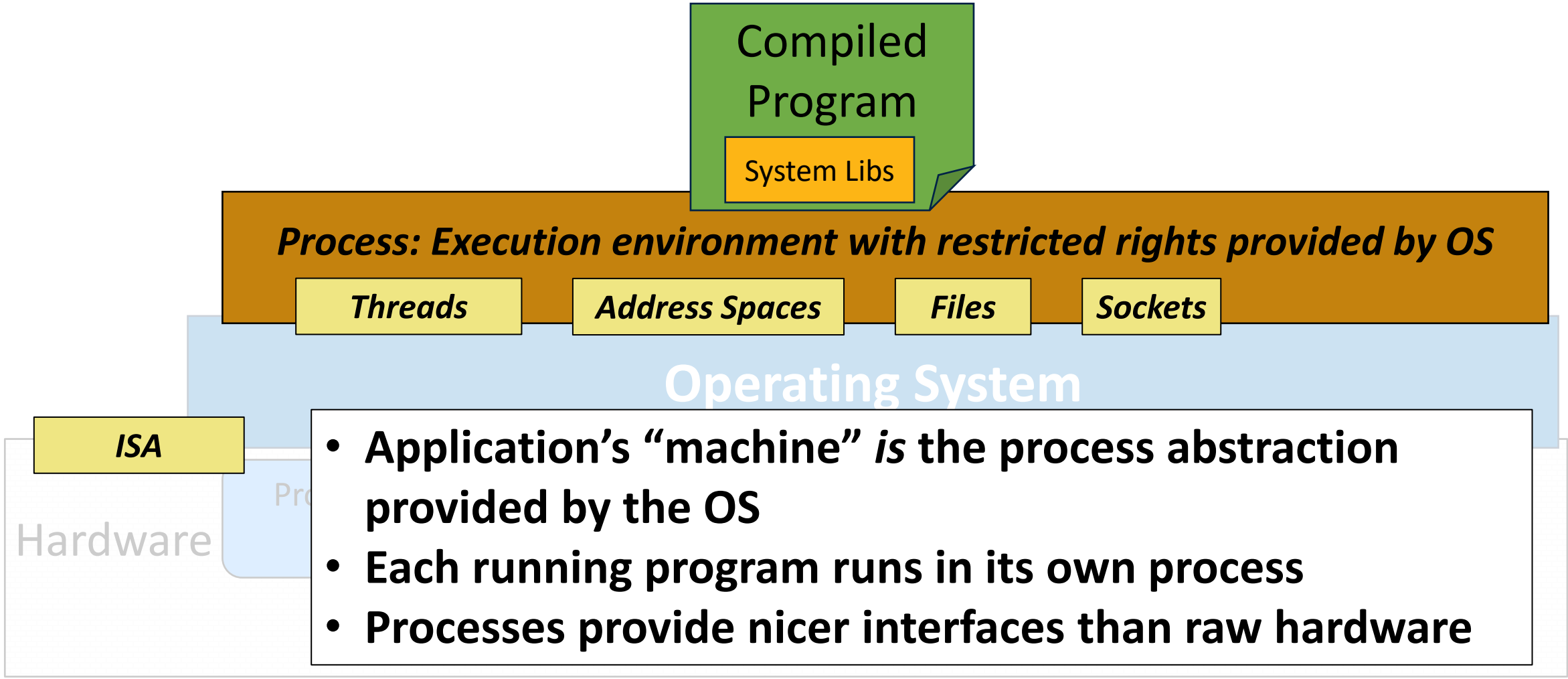
  - 抽象和虚拟化
  - 屏蔽硬件设备的复杂性
  - 分配和保护共享资源
  - 安全与验证



OS Basics: Virtualizing the Machine

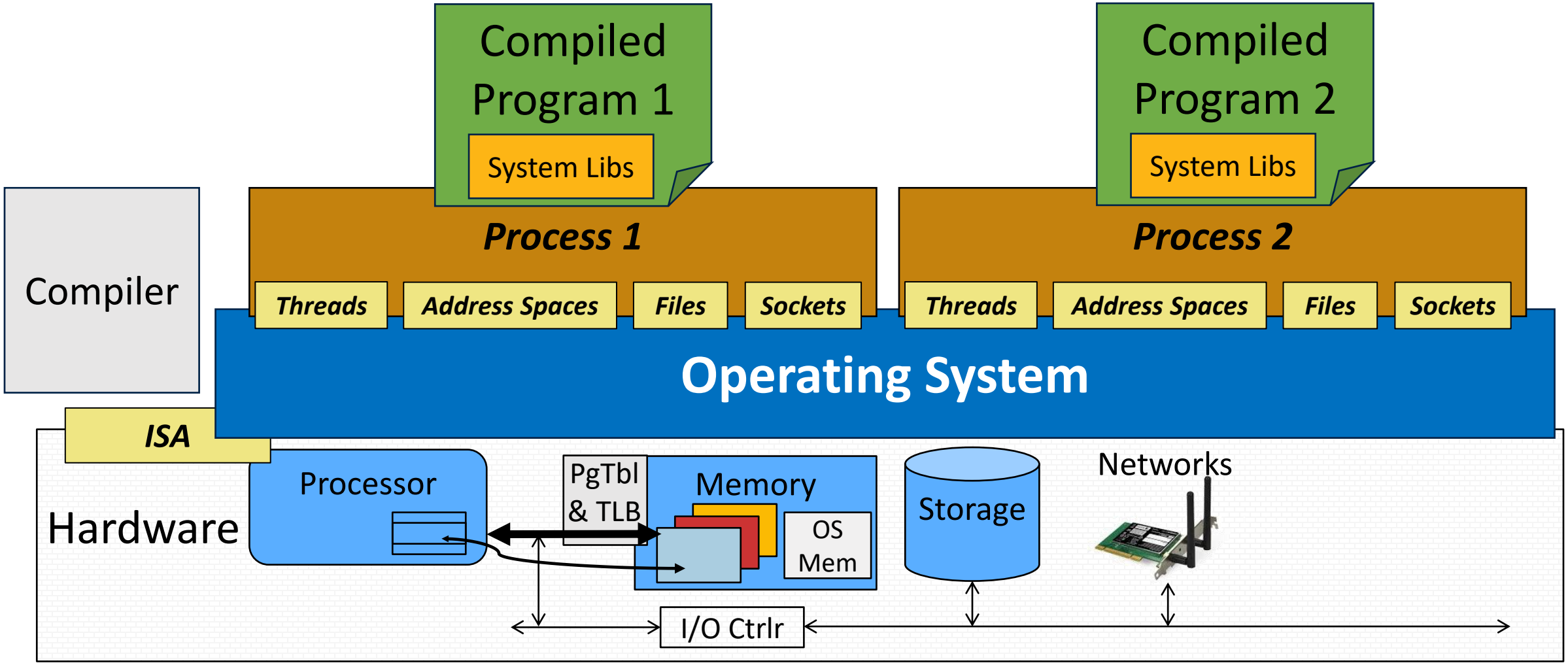


Compiled Program's View of the World



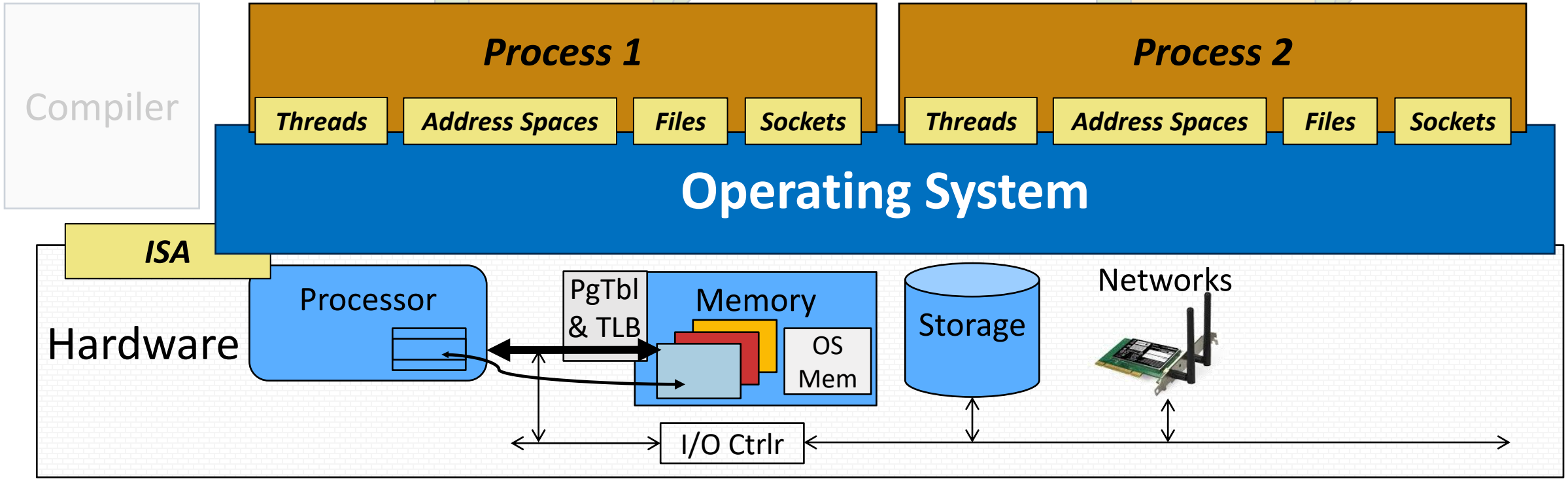
- **Application's "machine" *is* the process abstraction provided by the OS**
- **Each running program runs in its own process**
- **Processes provide nicer interfaces than raw hardware**

Operating System's View of the World

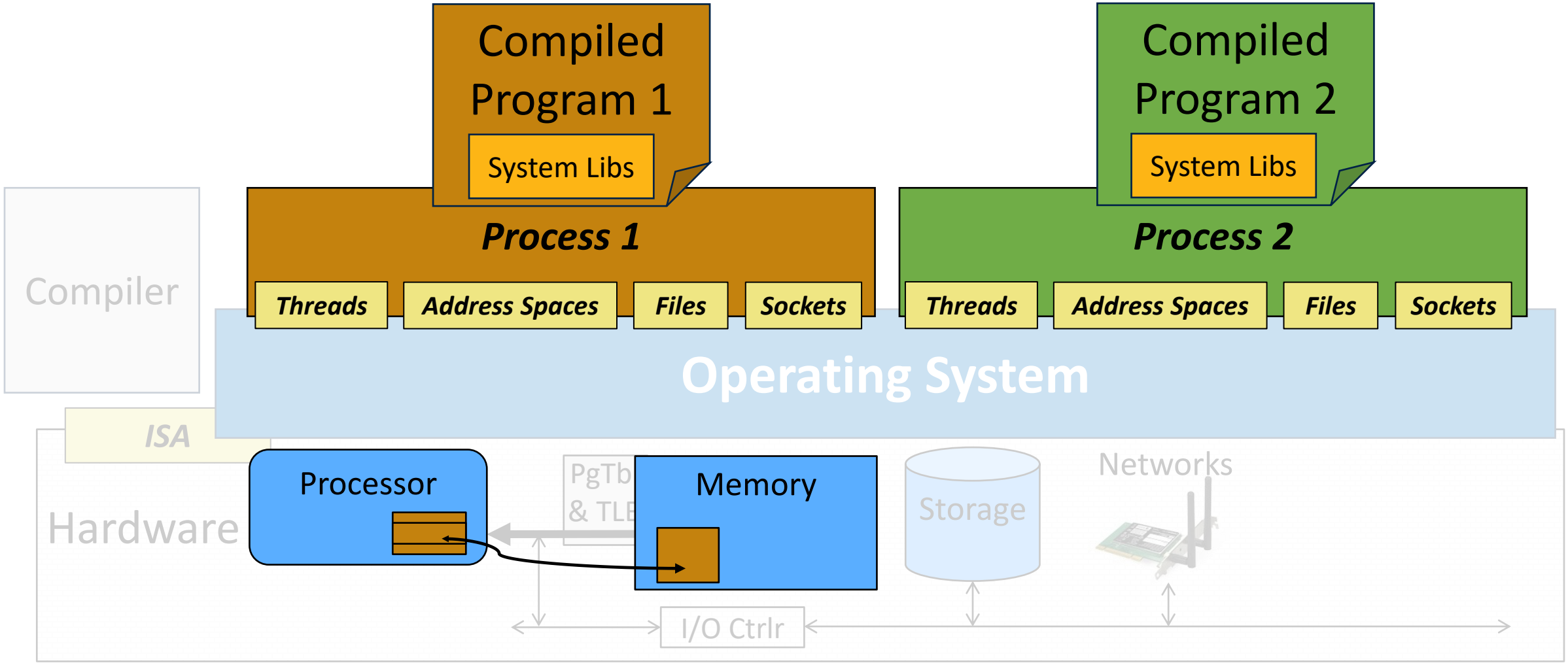


Operating System's View of the World

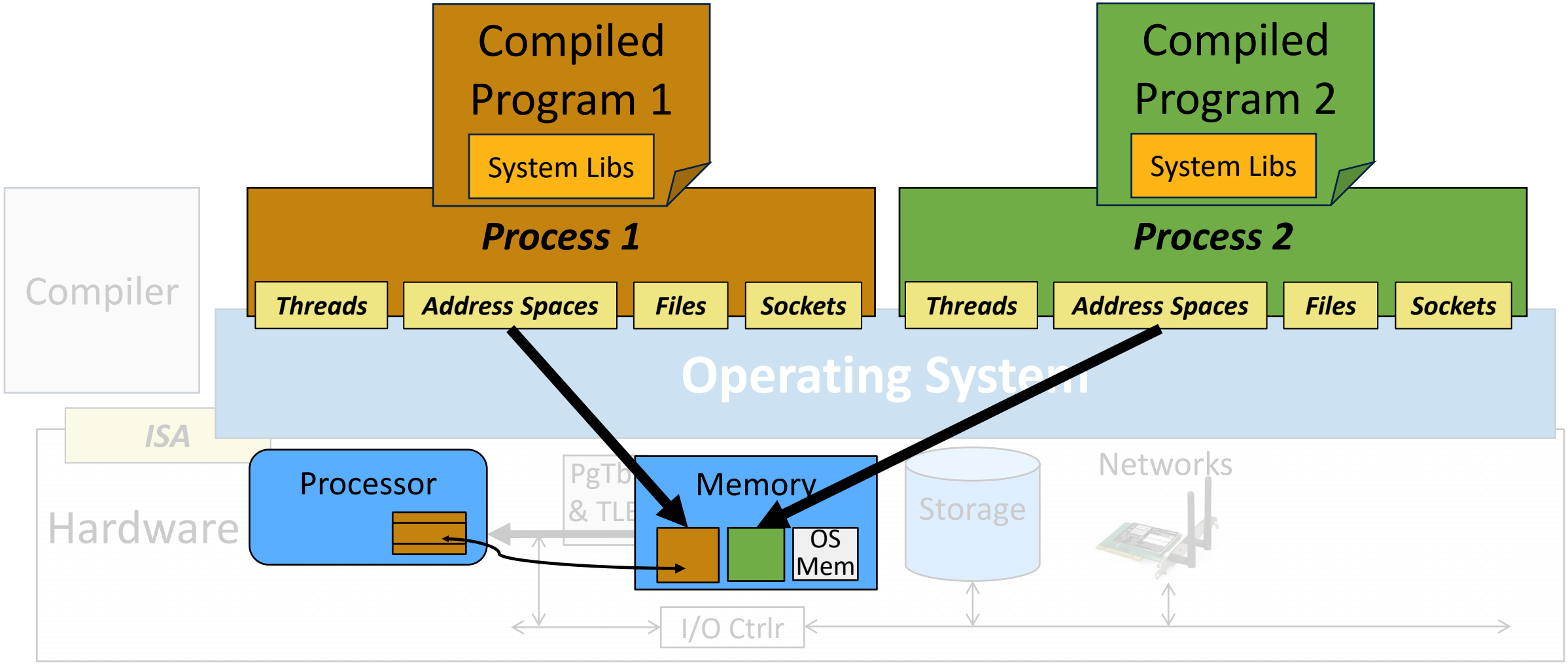
- OS translates from hardware interface to application interface
- OS provides each running program with its own process



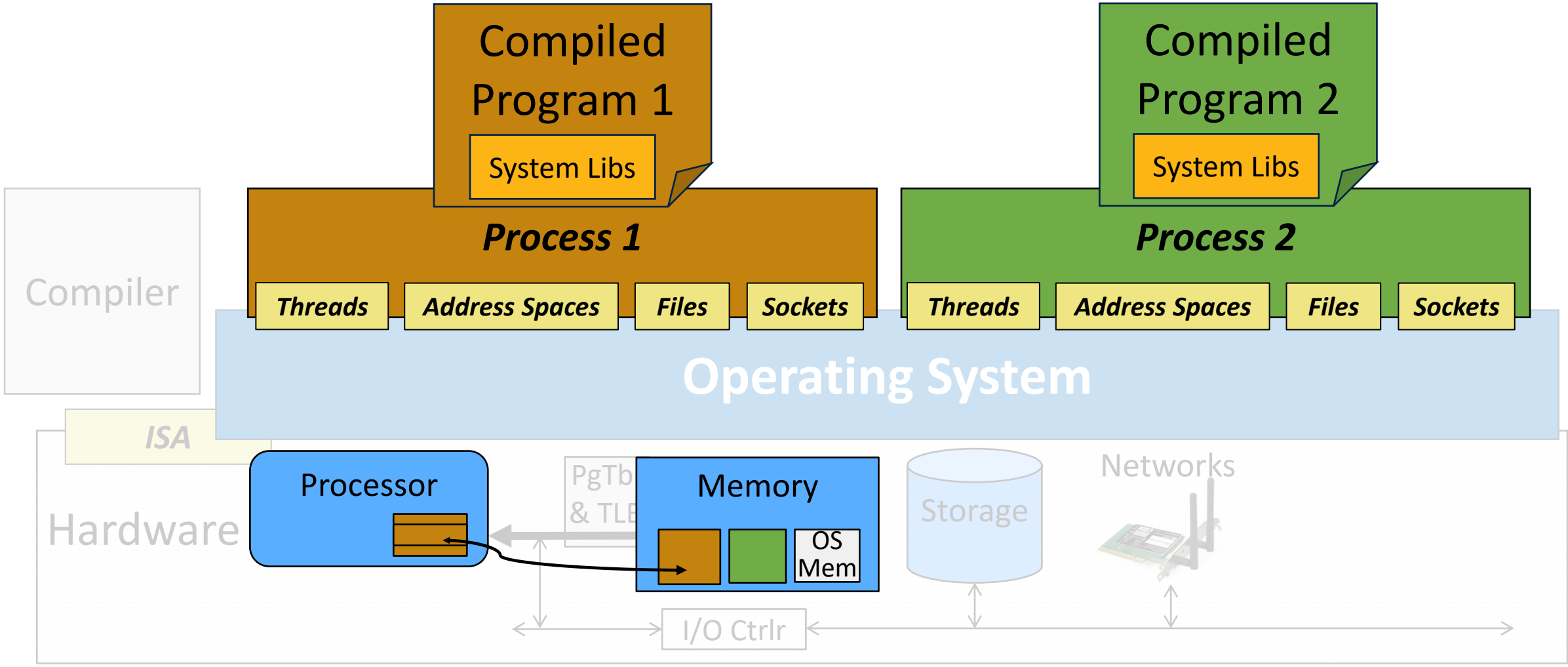
OS Basics: Running a Process



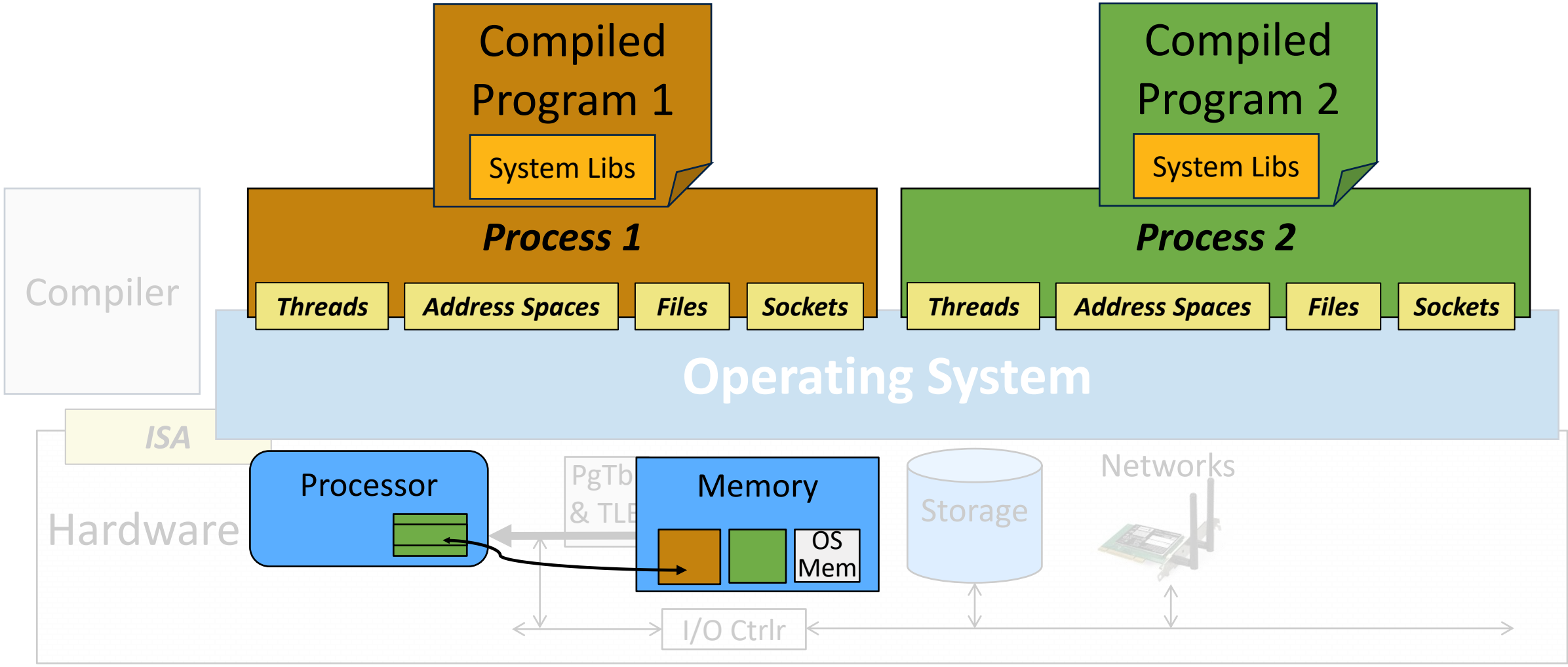
OS Basics: Switching Processes



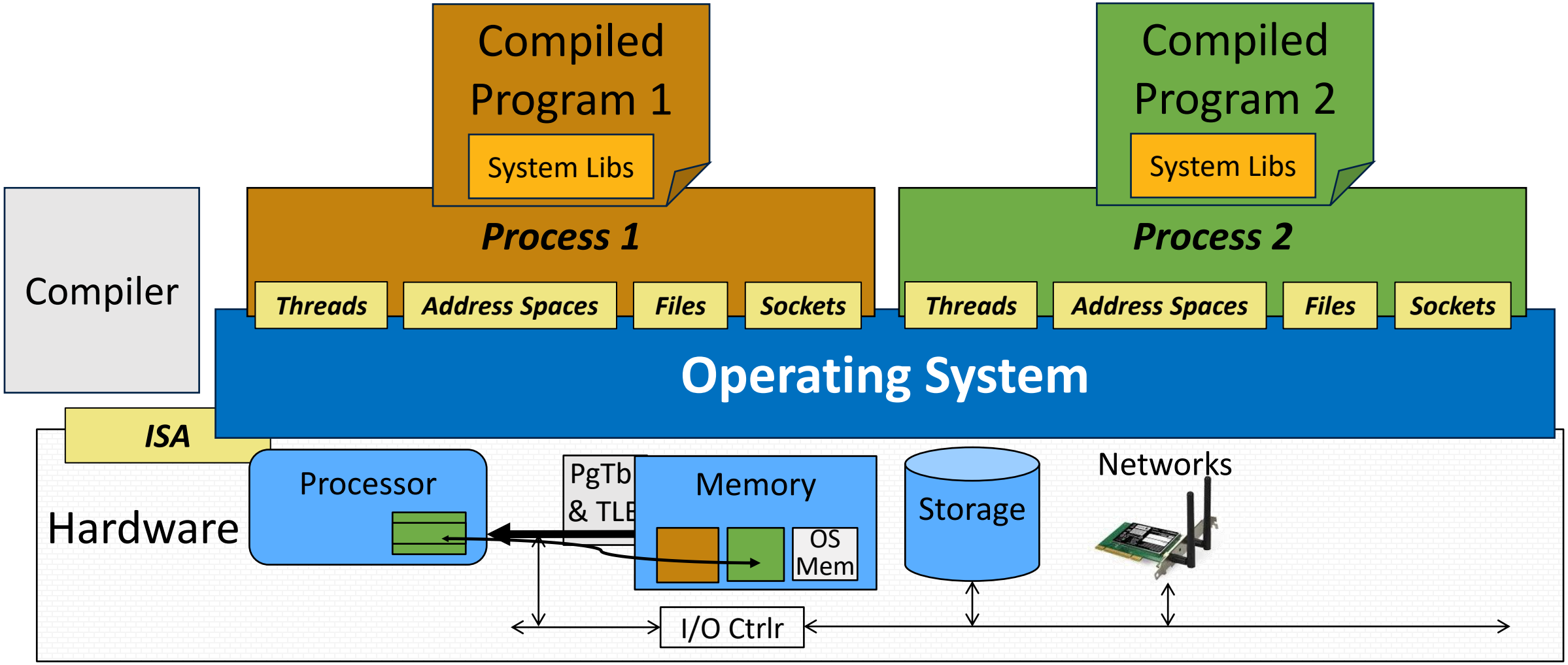
OS Basics: Switching Processes



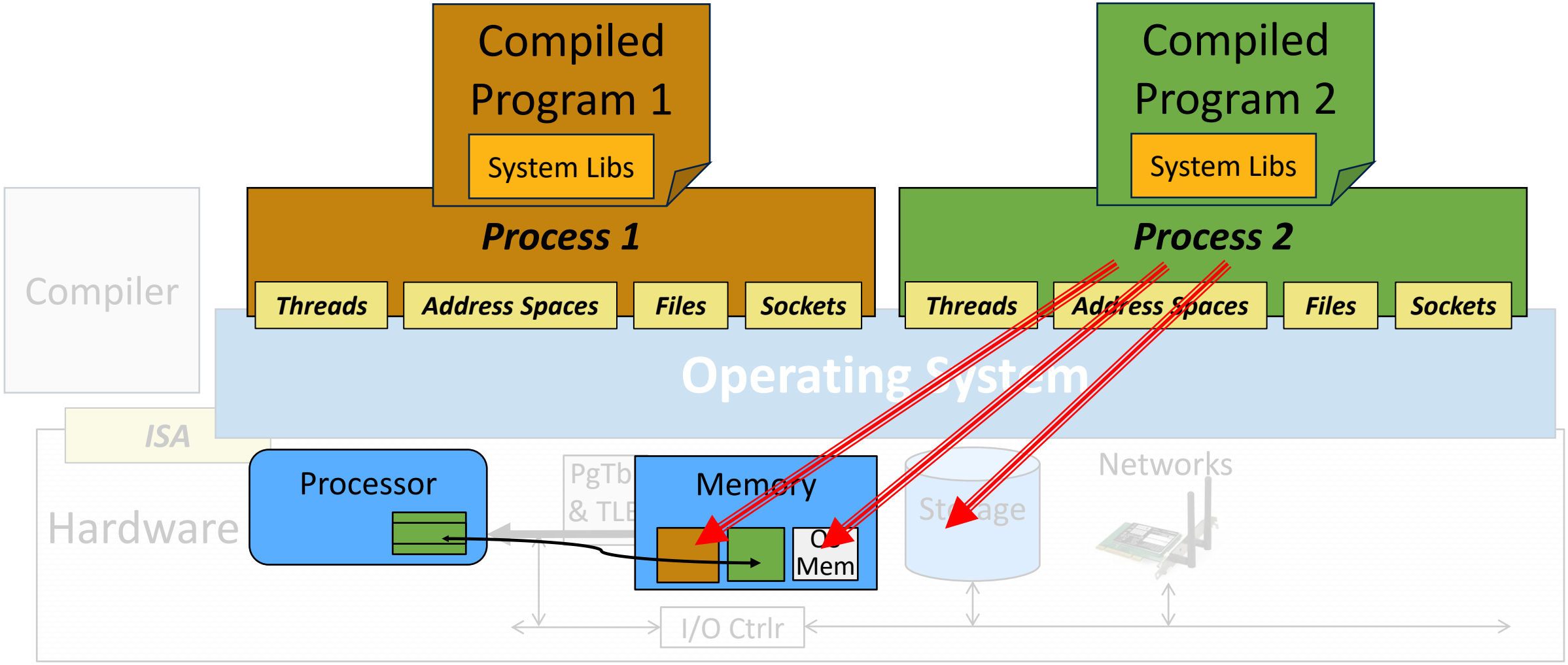
OS Basics: Switching Processes



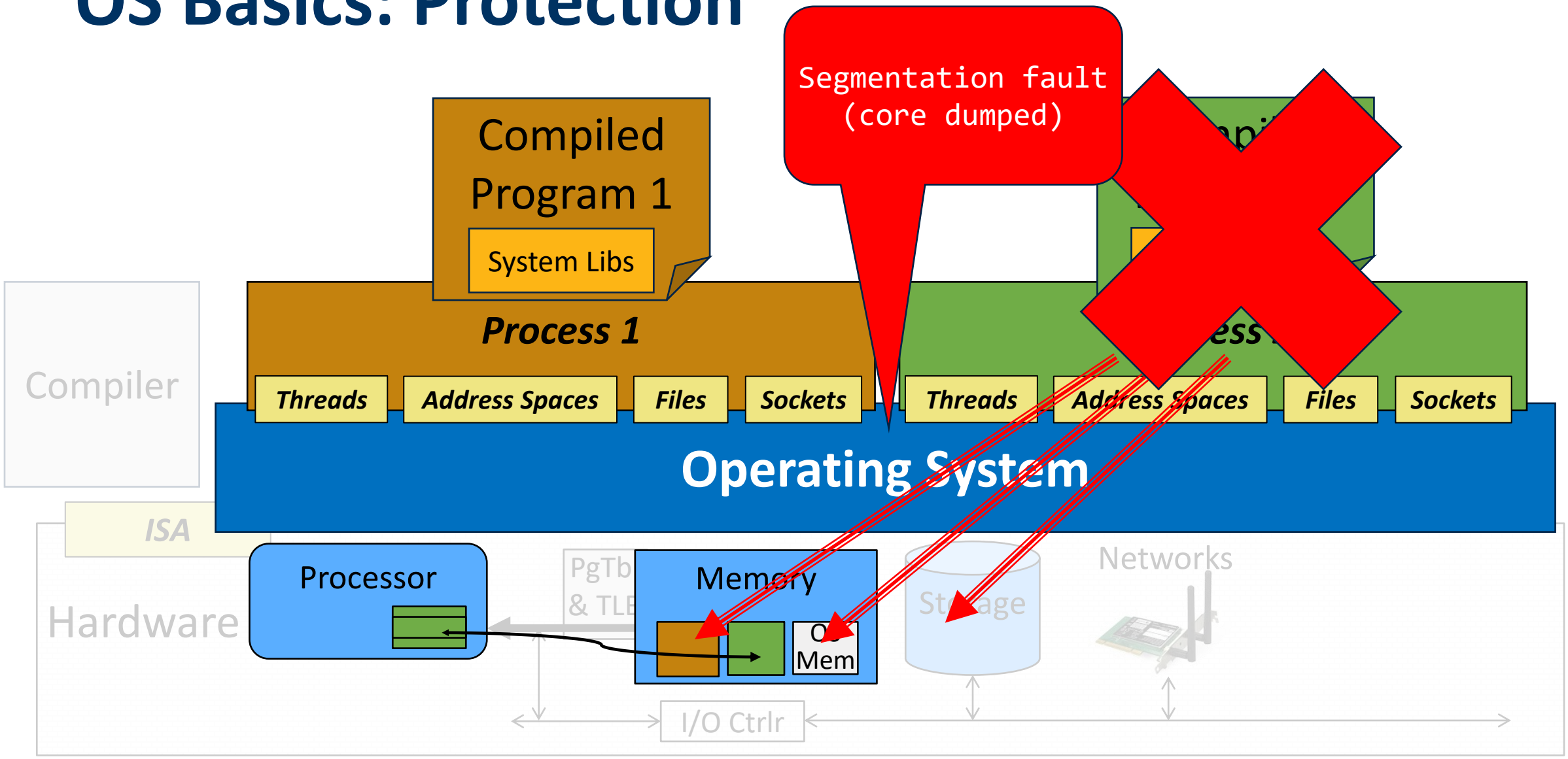
OS Basics: Switching Processes



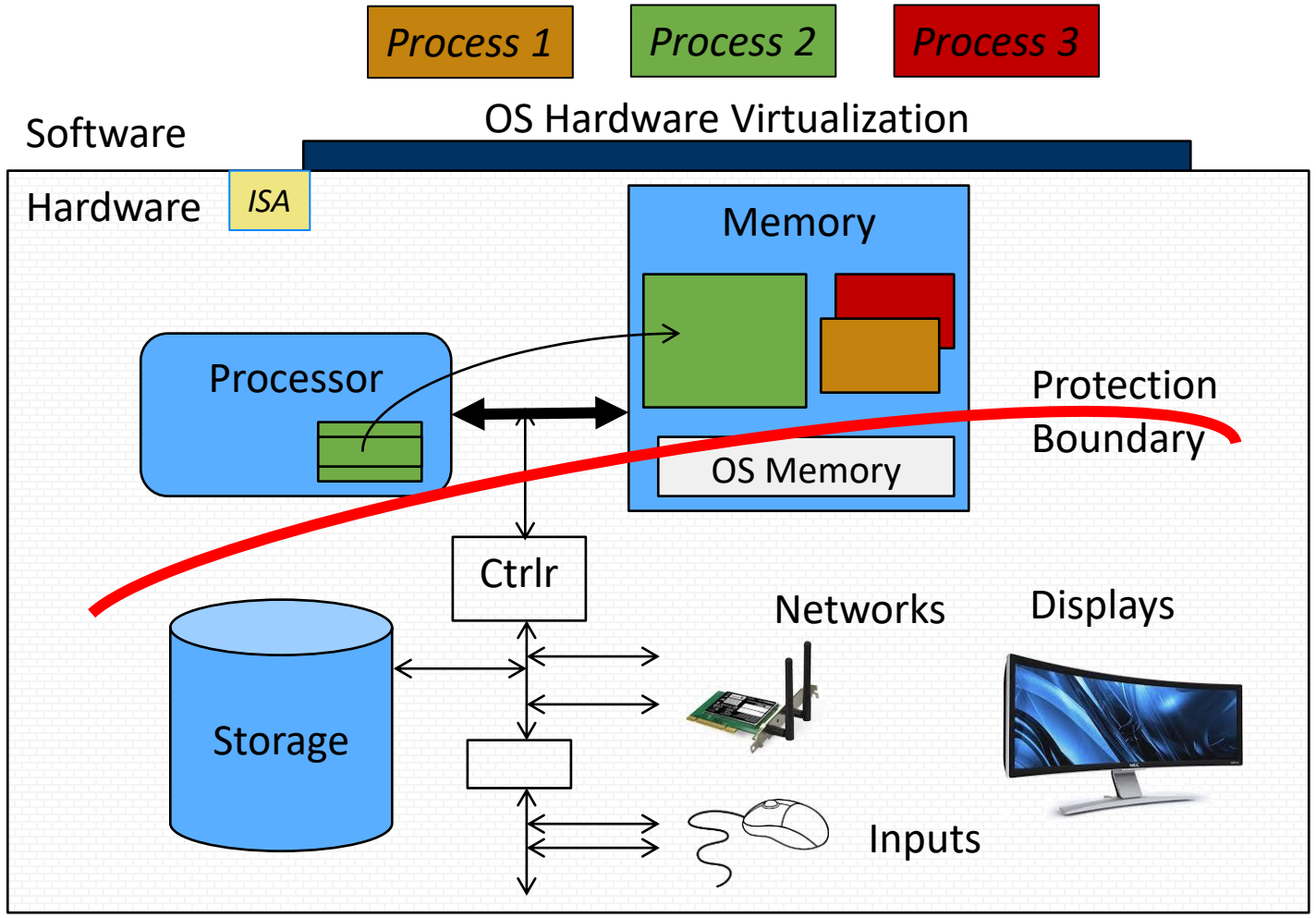
OS Basics: Protection



OS Basics: Protection

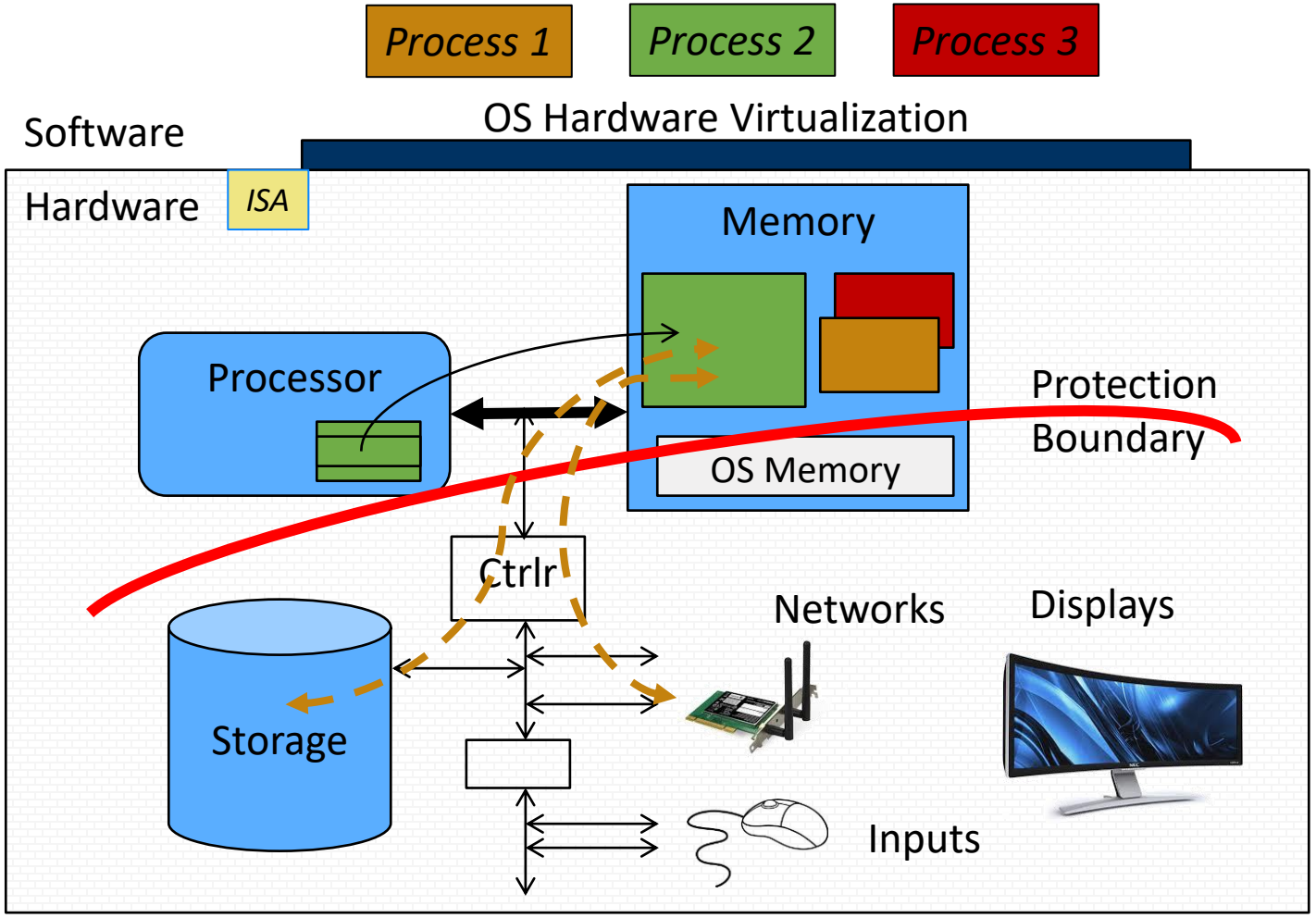


OS Basics: Protection



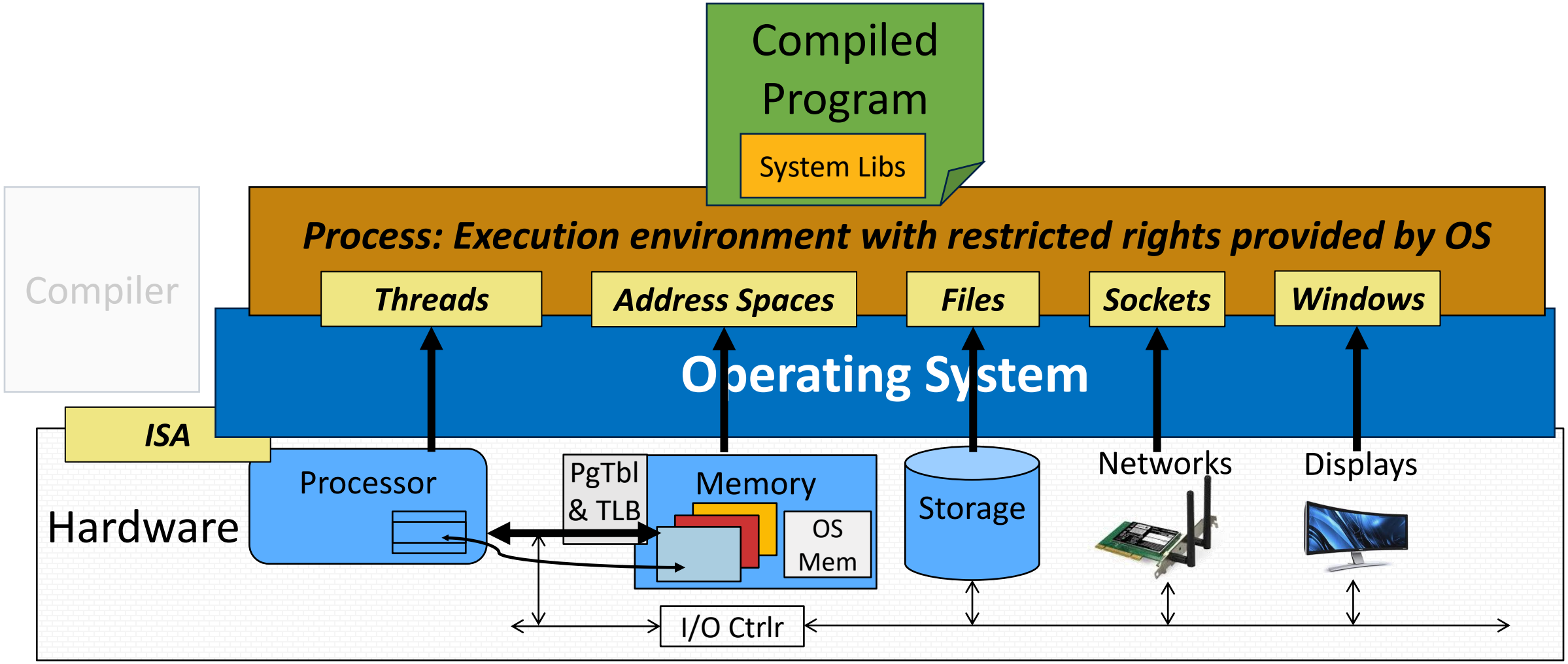
- **OS *isolates* processes from each other**
- **OS *isolates* itself from other processes**
- **… even though they are actually running on the same hardware!**

OS Basics: I/O

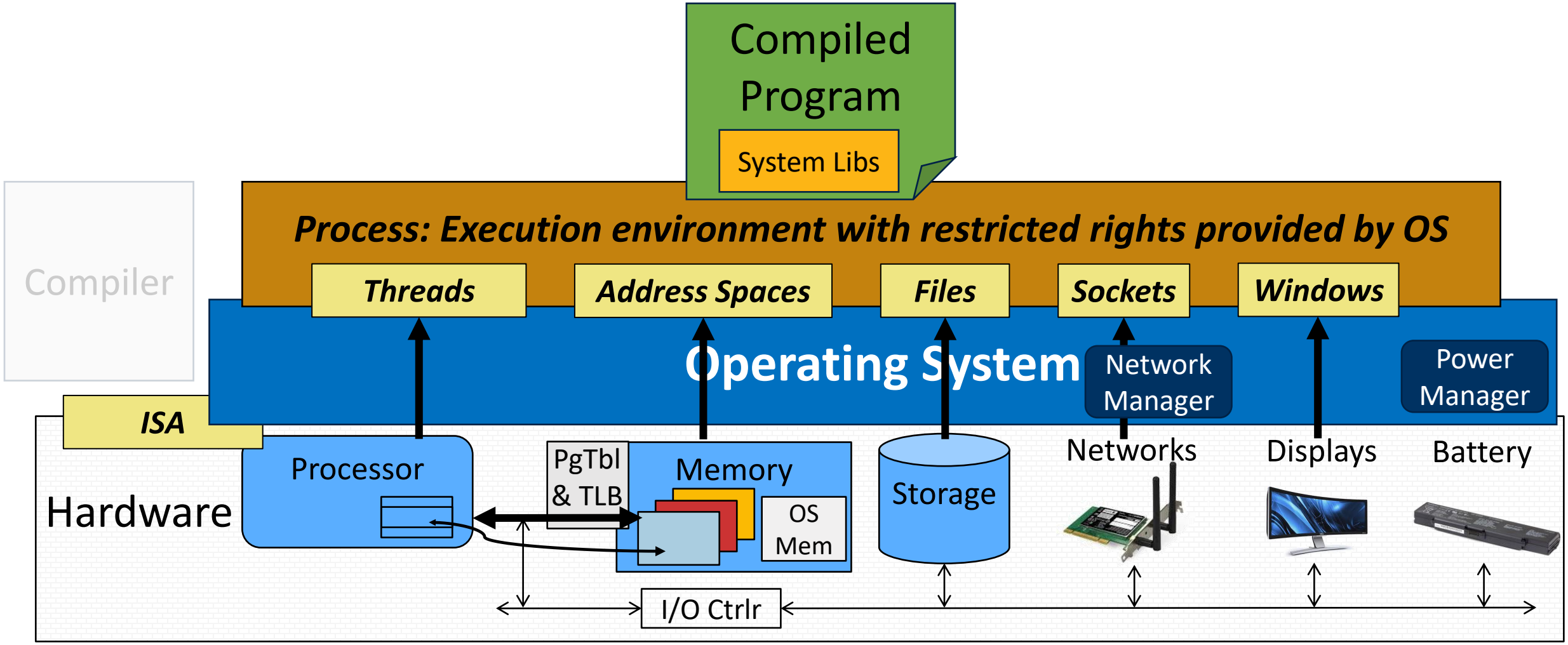


- **OS provides common services in the form of I/O**

OS Basics: Look and Feel



OS Basics: Background Management



Operating Systems…

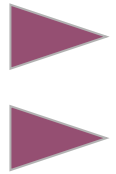
- Provide consistent abstractions to apps, even on diverse hardware
  - File systems, Window Systems, Communications, …
  - Processes, threads
  - VMs, containers
  - Naming systems
- Manage resources shared among multiple applications:
  - Memory, CPU, storage, …
- Achieved by specific algorithms and techniques:
  - Scheduling
  - Concurrency
  - Transactions
  - Security

OS Basics: Hardware Support

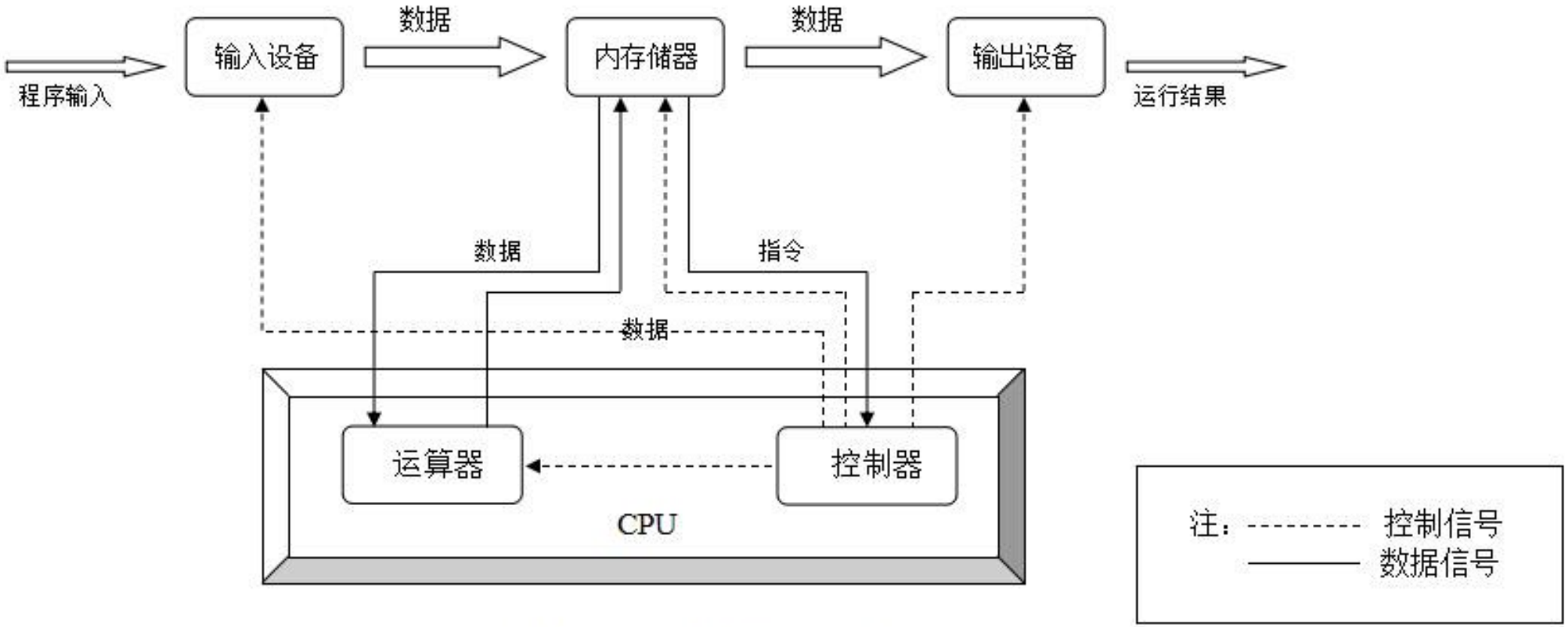
- OS bottom line: support applications!
  - Ideally, OS should have very low performance overhead over the raw hardware
- Rely on support **from the *underlying hardware* to implement OS abstractions** efficiently:
  - **Dual-mode operation, Interrupts, Traps, Precise exceptions**, Memory Management Unit, Translation Lookaside Buffer, etc.
- **Hardware support and OS design continue to co-evolve**…
  - … as hardware performance improves (e.g., faster storage/network), …
  - … and application requirements change.
  - What we study in this class is the result of decades of co-evolution!

Chapter 1: Introduction

- 什么是操作系统？
- Operating-System Basic Operations
  - Virtualization
  - Resource Management
  - Security and Protection …
- 计算机的基本组成结构
- 计算机的指令系统
- 中断的概念
- 用户模式、系统模式的概念
- 操作系统引导的过程
- 其他：分布式操作系统、开源操作系统

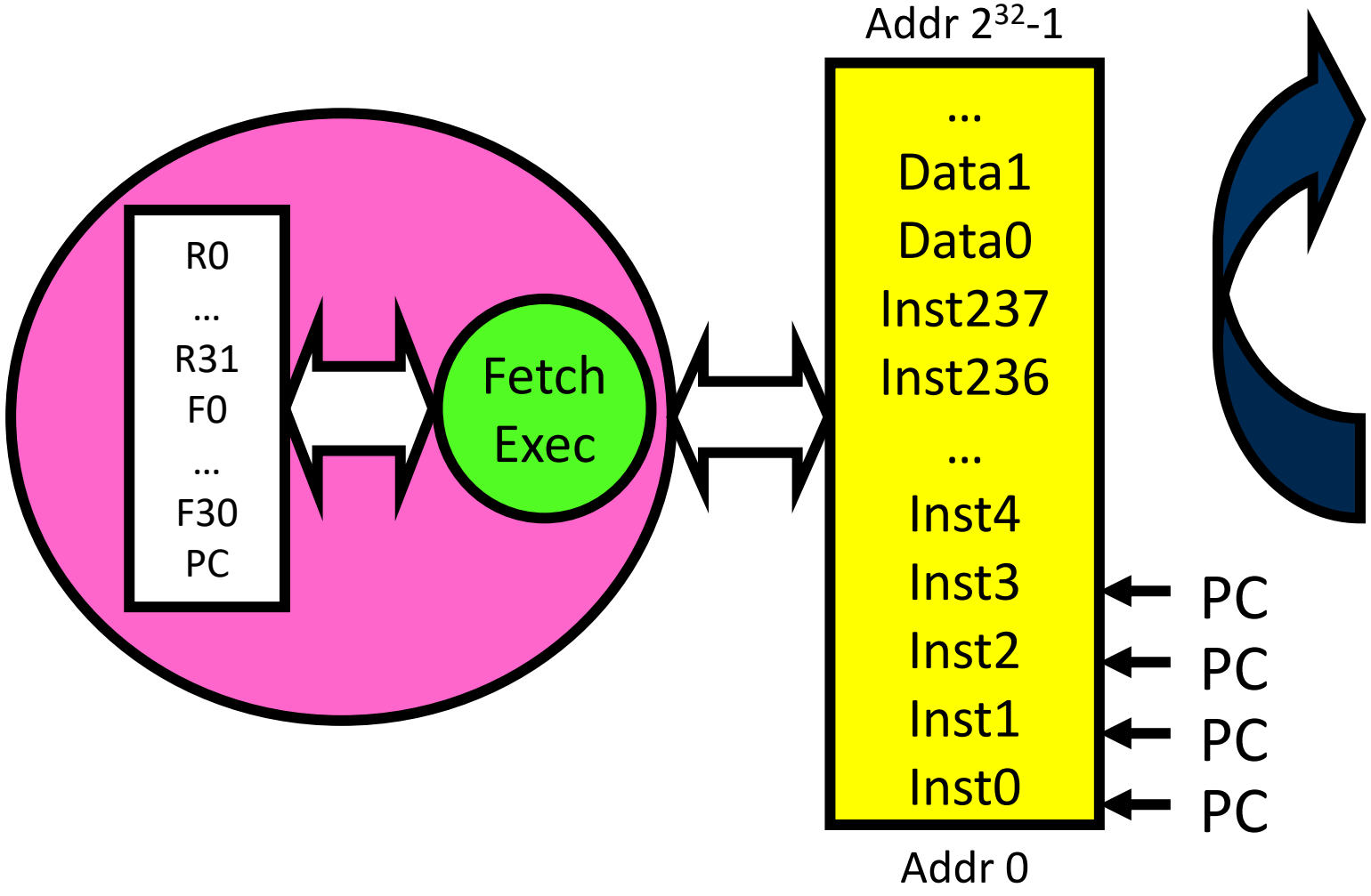


计算机的组成部件



计算机的工作原理示意图

指令执行的过程

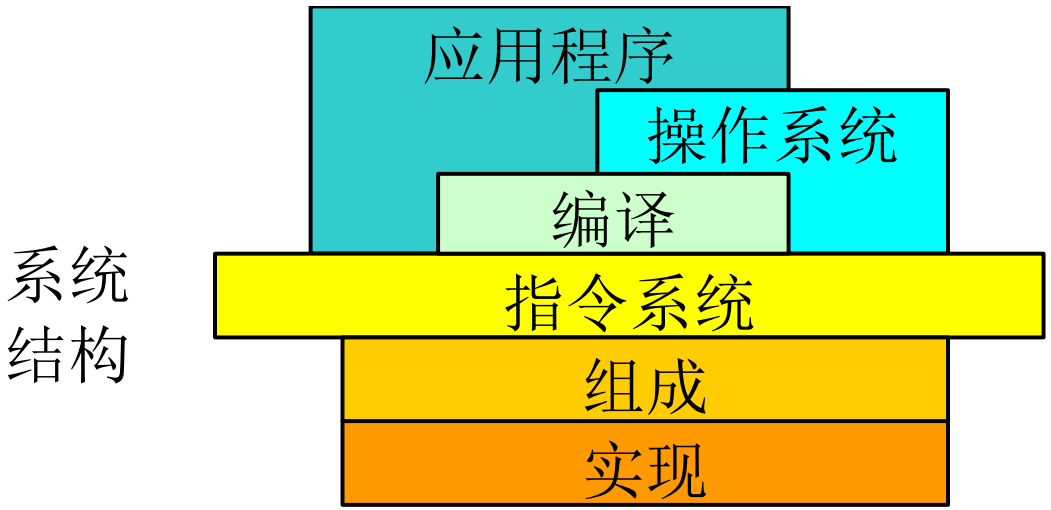


指令执行过程:

- 取指：根据PC，从存储器中取指令到IR（指令寄存器）
- 解码：解译IR中的指令
- 执行：控制CPU部件，执行运算，产生结果并写回，同时在标志寄存器里设置运算结果标志；跳转指令根据跳转地址修改PC，其他指令递增PC值
- 重复上述过程

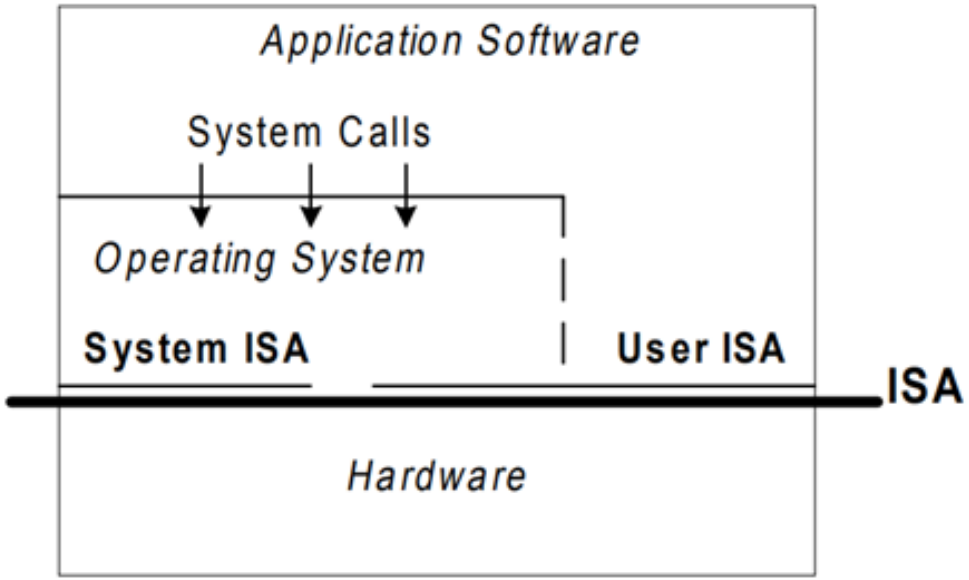
计算机系统结构与组成的区别与联系

- 计算机体系结构：程序员所看到的计算机的外属性
- 计算机组成：系统结构的具体实现
- 计算机的五个部分：
  - CPU：控制器，运算器，存储器
  - 输入设备/输出设备



指令集体系结构（ ISA ）

- 一个ISA包括：用户级指令+特权级指令
  - 特权级ISA 适用于（操作系统）对硬件资源的管理
  - 用户级ISA 适用于应用程序和操作系统



用户级指令架构

- 应用程序可见的指令集

Overview of User ISA

Integer	Memory	Control Flow	Floating Point
Add	Load byte	Jump	Add single
Sub	Load word	Jump equal	Mult. double
And	Store multiple	Call	Sqrt double
Compare	Push	Return	…
…	…	…	

“特权级”／“系统级”指令架构

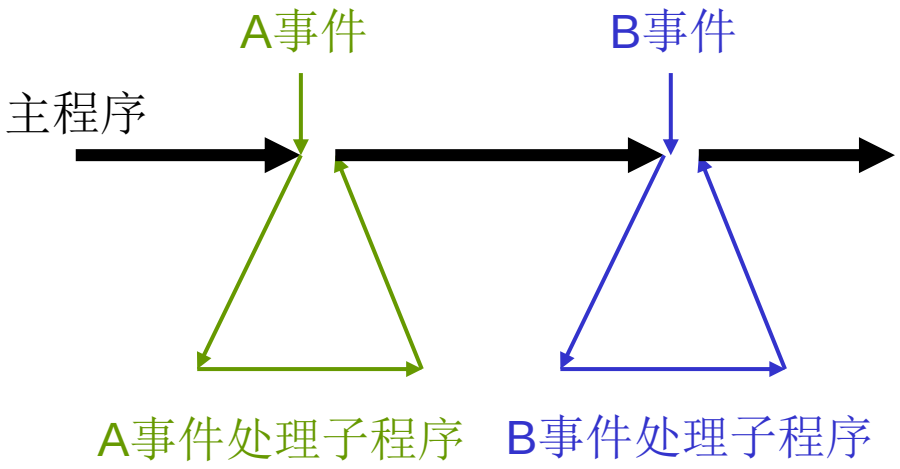
- 特权指令：用于管理系统硬件资源
- 系统级软件、如操作系统可见的指令
- “系统级”指令架构概貌
  - 特权级别（Privilege levels ）
  - 控制寄存器（Control registers）
  - 管理关键资源的指令
    - Processor （scheduling, time-sharing）
    - Memory （e.g., isolated address spaces）
    - I/O （e.g., isolated disk storage space）

Chapter 1: Introduction

- 什么是操作系统?
- Operating-System Basic Operations
  - Virtualization
  - Resource Management
  - Security and Protection …
- 计算机的基本组成结构
- 计算机的指令系统
- 中断&系统调用的概念
- 用户模式、系统模式的概念
- 其他：虚拟机、分布式操作系统、开源操作系统等

如何支持中断与异常处理？

- 指令系统必须考虑的重要内容
- 中断/异常处理
  - 程序执行过程中遇到需处理的事件
  - 暂时中止现行程序的运行
  - 转去执行相应的事件处理程序
  - 待处理完成后再返回原程序被中断处、或调度其他程序执行的过程。



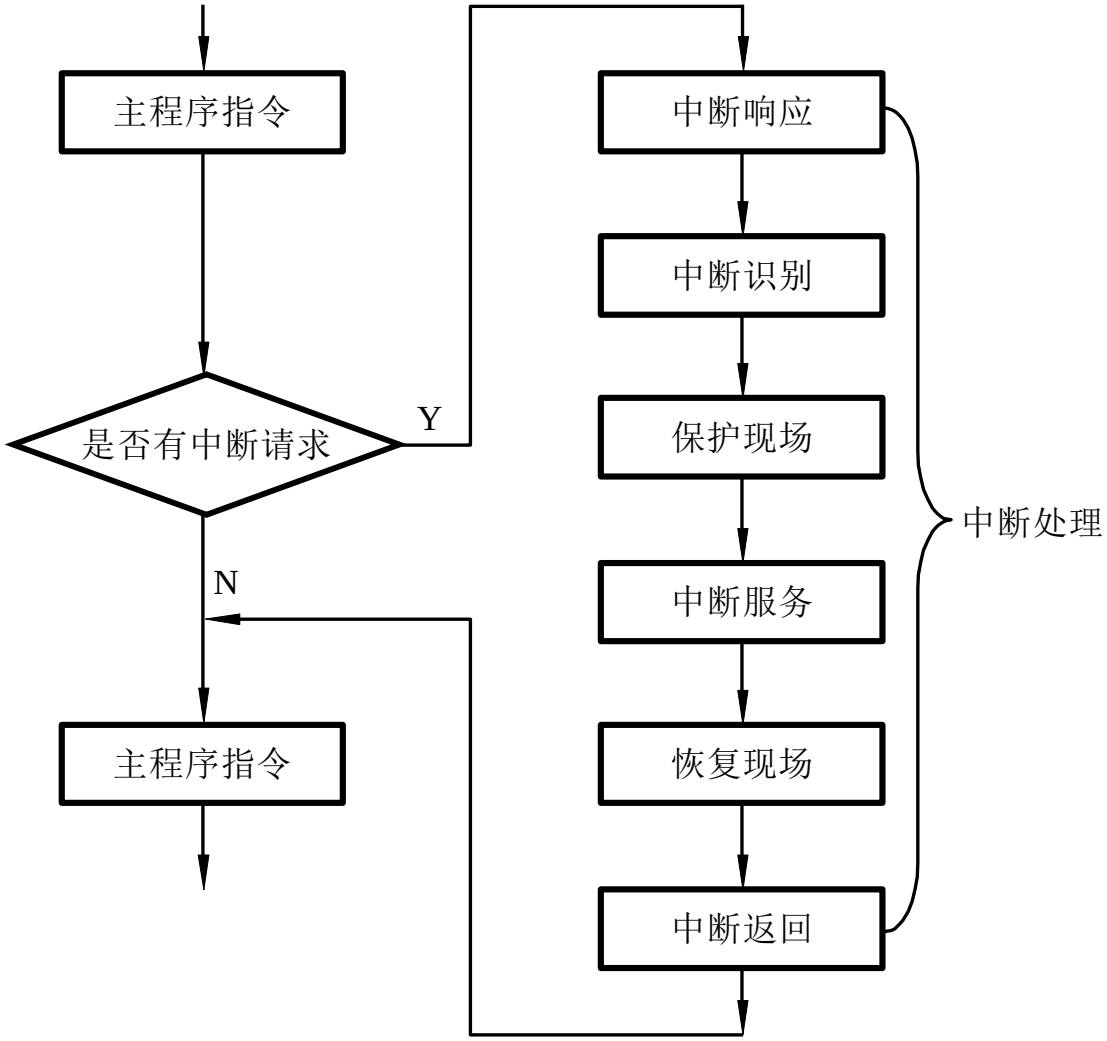
异常/中断的类型

- 中断
  - 异常
    - 自愿中断
      - 程序主动请求操作系统服务如：分配外设
      - 执行指令：int type（中断类型号）
    - 强迫中断
      - 软件出错
        - 除数为零、操作数溢出
        - 非法指令、地址越界
        - 虚拟地址异常
      - 硬件故障
        - 处理器、内存储器、总线等硬件故障
        - 例如：内存校验错
  - 外中断（强迫中断 ）
    - 时钟中断、间隔时钟
    - 设备报到与传输结束
    - 键盘/鼠标信号
    - 关机/重启动

指令系统如何支持中断与异常处理?

指令系统应该作出的规定:

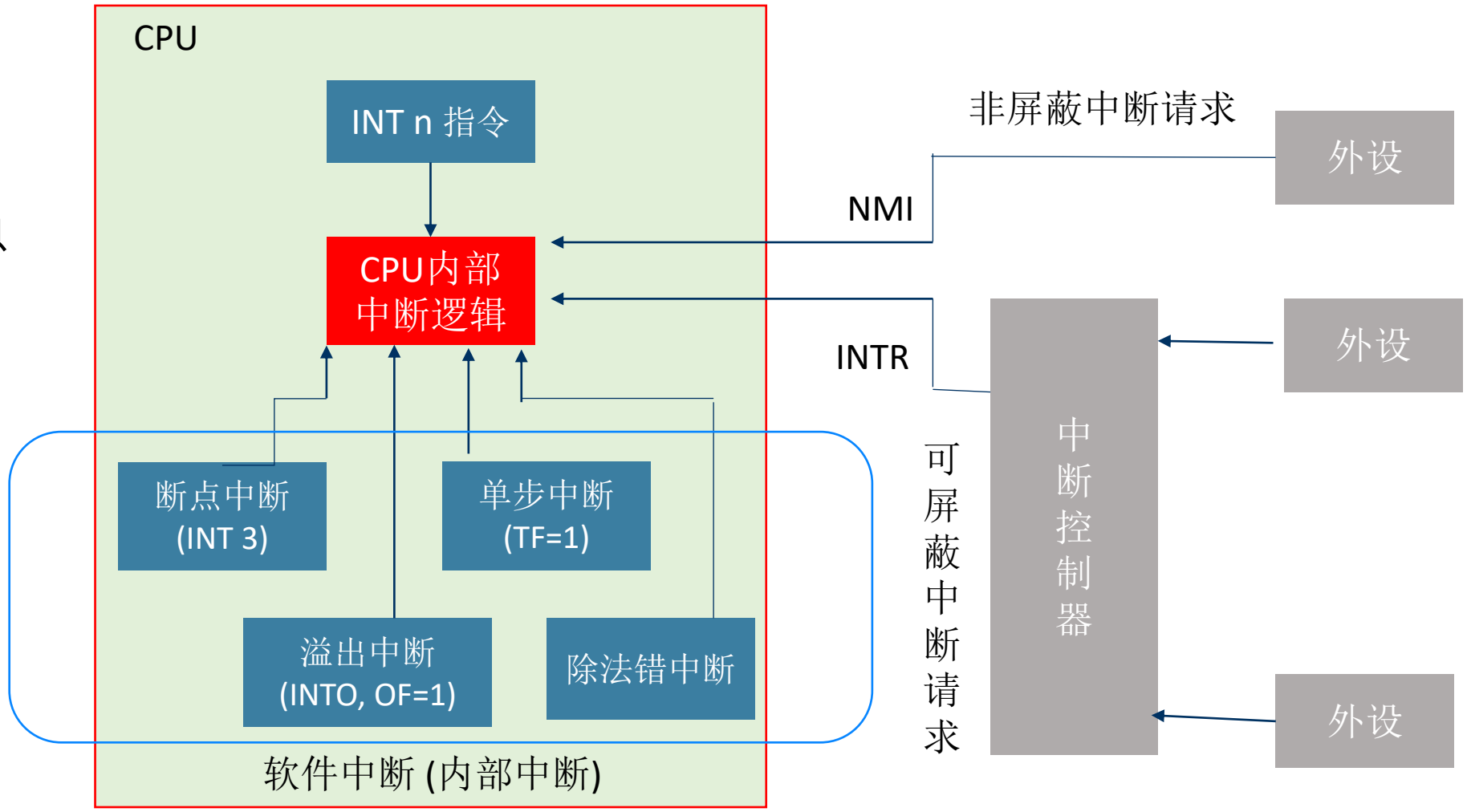
- 异常和中断类型的定义
- 自陷指令
- 中断允许位
- 开中断、关中断
- 异常/中断原因的识别和记录
- 断点信息的保存
- 中断处理时软硬件之间的协同
- …



中断源识别：中断向量

中断向量:
由设备主动向CPU发出识别信息

无论是内部中断，还是外部中断，都将自己的识别码输入到CPU内部的中断逻辑。识别码是一个向量号，所以这种方式被称为中断向量方式



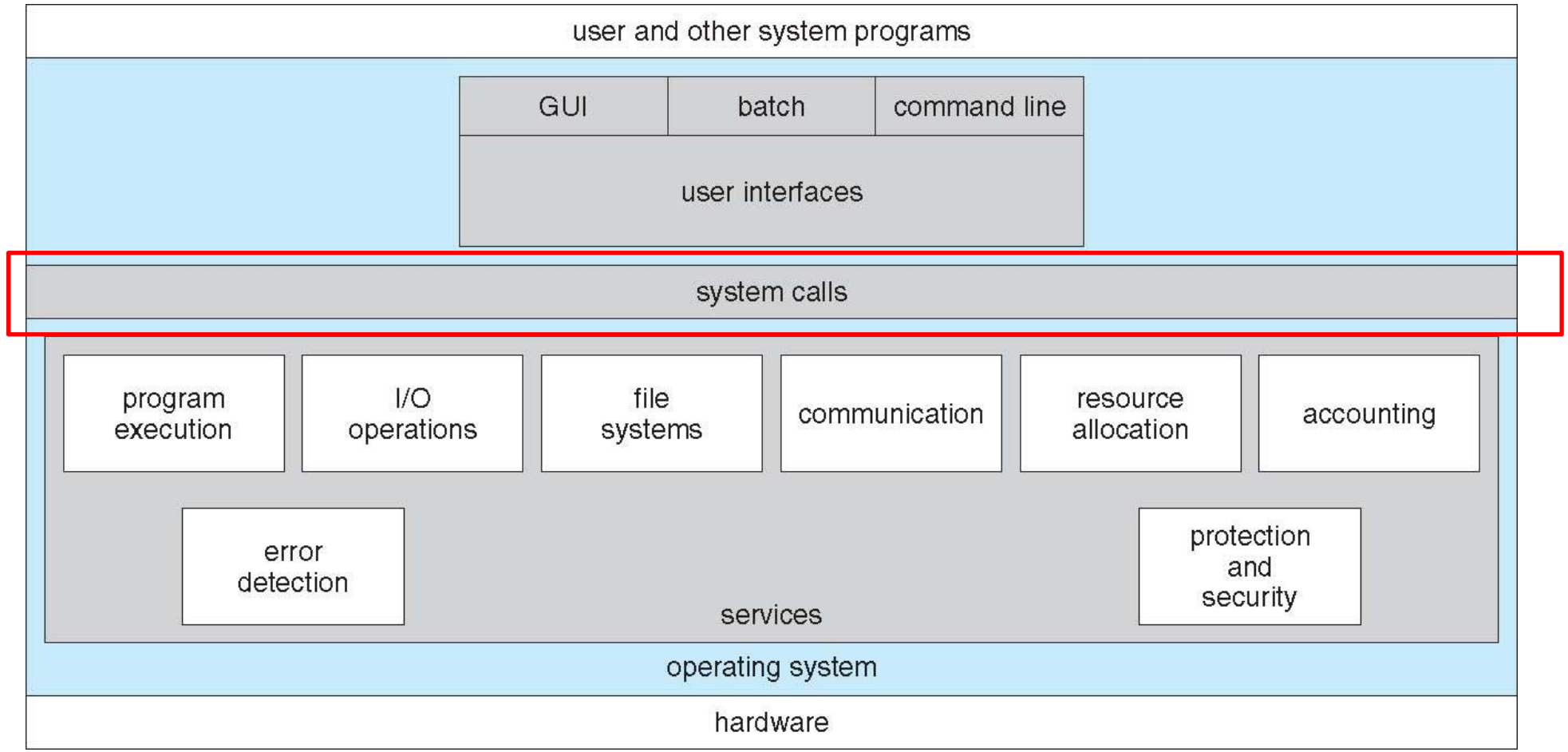
Chapter 1: Introduction

- 什么是操作系统？
- Operating-System Basic Operations
  - Virtualization
  - Resource Management
  - Security and Protection …
- 计算机的基本组成结构
- 计算机的指令系统
- 中断 & 系统调用
- 用户模式、系统模式的概念
- 操作系统引导的过程
- 其他：分布式操作系统、开源操作系统

系统调用是通过中断实现的！

Operating System Services：系统调用

程序员角度：
编程接口、
系统调用



系统调用

- 系统调用：用户通过操作系统提供的接口（API）获得内核提供的一些服务
- Typically written in a high-level language (C or C++)
- 种类：
  - 进程管理、信号管理
  - 文件管理、目录管理
  - 权限管理、 时间管理

系统调用 System Calls

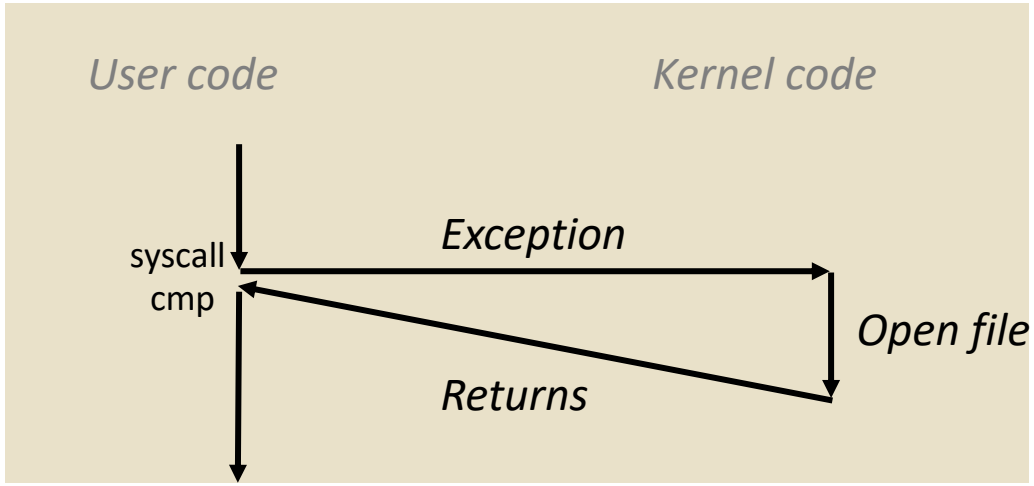
- **Each x86-64 system call has a unique ID number**
- **Examples:**

Number	*Name*	*Description*
0	`read`	Read file
1	`write`	Write file
2	`open`	Open file
3	`close`	Close file
4	`stat`	Get info about file
57	`fork`	Create process
59	`execve`	Execute a program
60	`_exit`	Terminate process
62	`kill`	Send signal to process

System Call Example: Opening File

- User calls: `open(filename, options)`
- Calls `__open` function, which invokes system call instruction `syscall`

```
00000000000e5d70 <__open>:
...
e5d79:   b8 02 00 00 00      mov  $0x2,%eax  # open is syscall #2
e5d7e:   0f 05               syscall         # Return value in %rax
e5d80:   48 3d 01 f0 ff ff   cmp  $0xfffffffffffff001,%rax
...
e5dfa:   c3                  retq
```



- `%rax` contains syscall number
- Other arguments in `%rdi`, `%rsi`, `%rdx`, `%r10`, `%r8`, `%r9`
- Return value in `%rax`
- Negative value is an error corresponding to negative `errno`

API与系统调用之间的关系

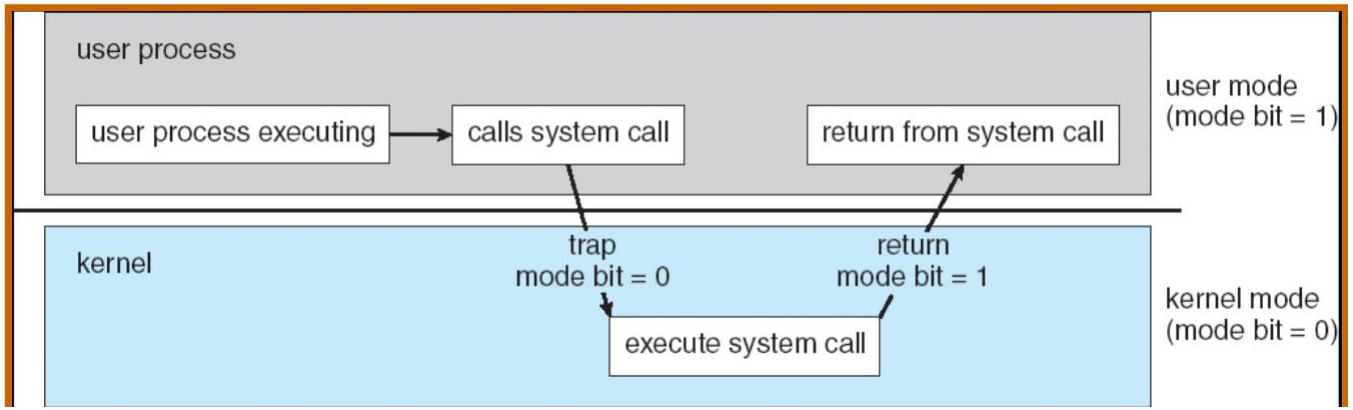
- API(Application Programming Interface)应用程序接口， 是一些预定义的函数。跟内核没有必然的联系。
- 不是所有的API函数都对应一个系统调用，
  - 有时，一个API函数会需要几个系统调用来共同完成函数的功能
  - 也有一些API函数不需要调用系统调用（因此它所完成的不是内核提供的服务，例如： abs(); sqrt()等函数）
- 程序员只能使用API与系统交互， 不能直接使用系统调用
- 系统调用不与程序员进行交互，它根据API函数，通过一个软中断机制向内核提交请求，以获取内核服务的接口

Chapter 1: Introduction

- 什么是操作系统？
- Operating-System Basic Operations
  - Virtualization
  - Resource Management
  - Security and Protection …
- 计算机的基本组成结构
- 计算机的指令系统
- 中断 & 系统调用
- 用户模式、系统模式的概念
- 操作系统引导的过程
- 其他：虚拟机、分布式操作系统、开源操作系统等

Protects Processes and the Kernel: Dual-Mode Operation

- **Hardware provides at least two modes** （硬件支持的工作模式）
  1. Kernel Mode (or “supervisor” mode)
  2. User Mode
- Certain operations are **prohibited** when running in user mode
  - Changing the page table pointer, disabling interrupts, interacting directly w/ hardware, writing to kernel memory
- Carefully controlled transitions between user mode and kernel mode
  - System calls, interrupts, exceptions



用户态与内核态

- 计算机系统中，通常运行着两类程序：
  - 系统程序和应用程序

- 计算机设置了两种状态：
  - 系统态(也称为管态或核心态)，操作系统在系统态运行——运行操作系统程序
  - 用户态，应用程序只能在用户态运行——运行用户程序在实际运行过程中，处理机会在系统态和用户态间切换。

X86 CPU工作模式

- CPU提供了四个工作模式，从Ring0-Ring3，
- Ring0是最高权限模式，可以执行所有指令。
- Ring3是最低权限模式，只能执行指令集中的一部分指令。
- 操作系统的核心代码，CPU执行的时候是工作在Ring0模式，又叫内核态模式
- 应用程序的代码，CPU执行的时候是工作在Ring3模式，又叫用户态模式。
- 进入内核态不是说CPU进入了某个地方，而是一种工作模式的切换。
  - 模式位是硬件提供的
  - 可以不止两种模式，例如增加其他模式以支持虚拟化

用户态切换到内核态

用户态切换到内核态的3种方式

1）系统调用

用户态进程主动要求切换到内核态的一种方式。用户态进程通过系统调用申请使用操作系统提供的服务程序完成工作。例如fork（）、如Linux的int 80h中断。

2）异常

当cpu在执行运行在用户态下的程序时，发生了一些没有预知的异常，这时会触发由当前运行进程切换到处理此异常的内核相关进程中，也就是切换到了内核态，如缺页异常

3）外围设备的中断

当外围设备完成用户请求的操作后，会向CPU发出相应的中断信号，这时CPU会暂停执行下一条即将要执行的指令而转到与中断信号对应的处理程序去执行，如果前面执行的指令时用户态下的程序，那么转换的过程自然就会是由用户态到内核态的切换。

课堂练习

下列选项中，会导致用户进程从用户态切换到内核态的操作是

I. 设置系统时钟

II. sin（ ）函数调用

III. read系统调用

A 仅I、II

B. 仅I、III

C. 仅II、III

D. I、II和III

Chapter 1: Introduction

- 什么是操作系统？
- Operating-System Basic Operations
  - Virtualization
  - Resource Management
  - Security and Protection …
- 计算机的基本组成结构
- 计算机的指令系统
- 中断 & 系统调用
- 用户模式、系统模式的概念
- 操作系统引导的过程
- 其他：虚拟机、分布式操作系统、开源操作系统等

Linux0.11 的引导启动程序

列表 6-1 linux/boot/目录

	文件名	长度(字节)	最后修改时间(GMT)	说明
	bootsect.s	5052 bytes	1991-12-05 22:47:58	
	head.s	5938 bytes	1991-11-18 15:05:09	
	setup.s	5364 bytes	1991-12-05 22:48:10	

boot 目录中含有 3 个汇编语言文件，是内核源代码文件中最先被编译的程序。这 3 个程序完成的主要功能是当计算机加电时引导内核启动，将内核代码加载到内存中，并做一些进入 32 位保护运行方式前的系统初始化工作。

bootsect.s 程序是磁盘引导块程序，编译后会驻留在磁盘的第一个扇区中（引导扇区，0 磁道（柱面），0 磁头，第 1 个扇区）。在 PC 机加电 ROM BIOS 自检后，将被 BIOS 加载到内存 0x7C00 处进行执行。

setup.s 程序主要用于读取机器的硬件配置参数，并把内核模块 system 移动到适当的内存位置处。

head.s 程序会被编译连接在 system 模块的最前部分，主要进行硬件设备的探测设置和内存管理页面的初始设置工作。

Linux0.11引导启动程序

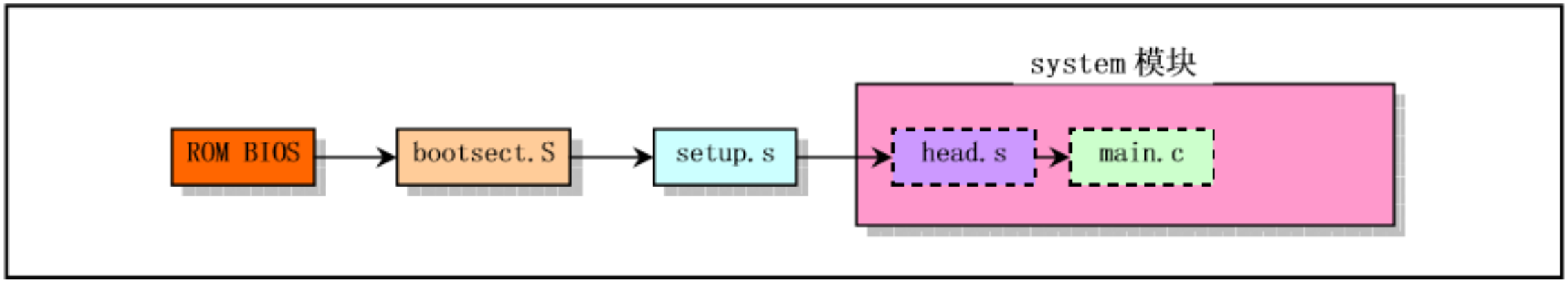


图 6-1 从系统加电起所执行程序的顺序

个人计算机的引导（装载）过程：

BIOS中有一个简单的引导程序

从磁盘调入一个较复杂的引导程序

后者再装载内核

bootsect.s程序功能描述

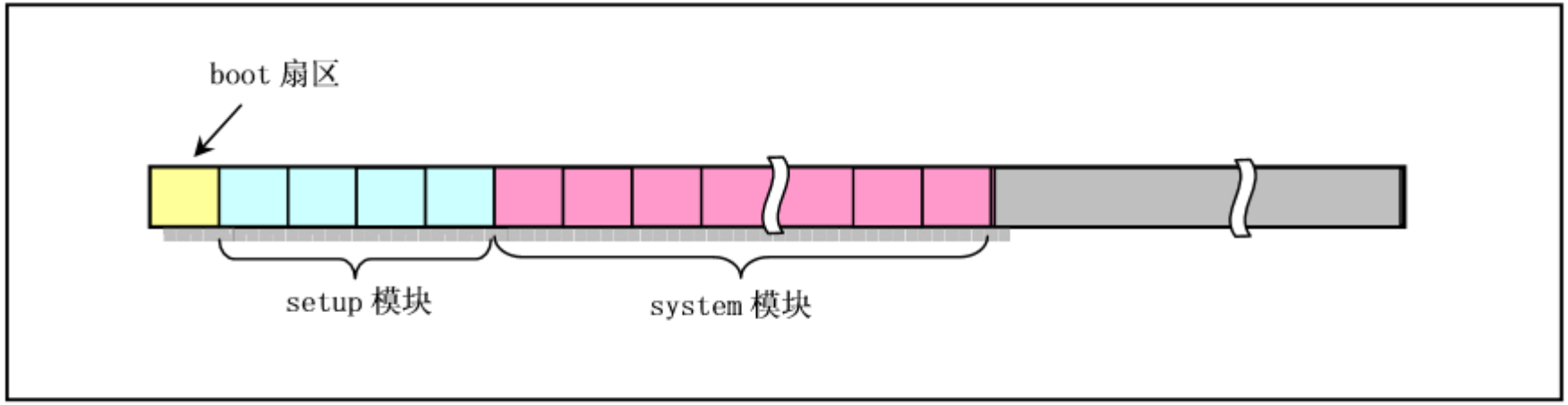


图 6-3 Linux 0.11 内核在 1.44MB 磁盘上的分布情况

bootsect.s 代码是磁盘引导块程序，驻留在磁盘的第一个扇区中（引导扇区，0 磁道（柱面），0 磁头，第 1 个扇区）。在 PC 机加电 ROM BIOS 自检后，ROM BIOS 会把引导扇区代码 bootsect 加载到内存地址 0x7C00 开始处并执行之。在 bootsect 代码执行期间，它会将自己移动到内存绝对地址 0x90000 开始处并继续执行。该程序的主要作用是首先把从磁盘第 2 个扇区开始的 4 个扇区的 setup 模块（由 setup.s 编译而成）加载到内存紧接着 bootsect 后面位置处（0x90200），然后利用 BIOS 中断 0x13 取磁盘参数表中当前启动引导盘的参数，接着在屏幕上显示“Loading system...”字符串。再者把磁盘上 setup 模块后面的 system 模块加载到内存 0x10000 开始的地方。

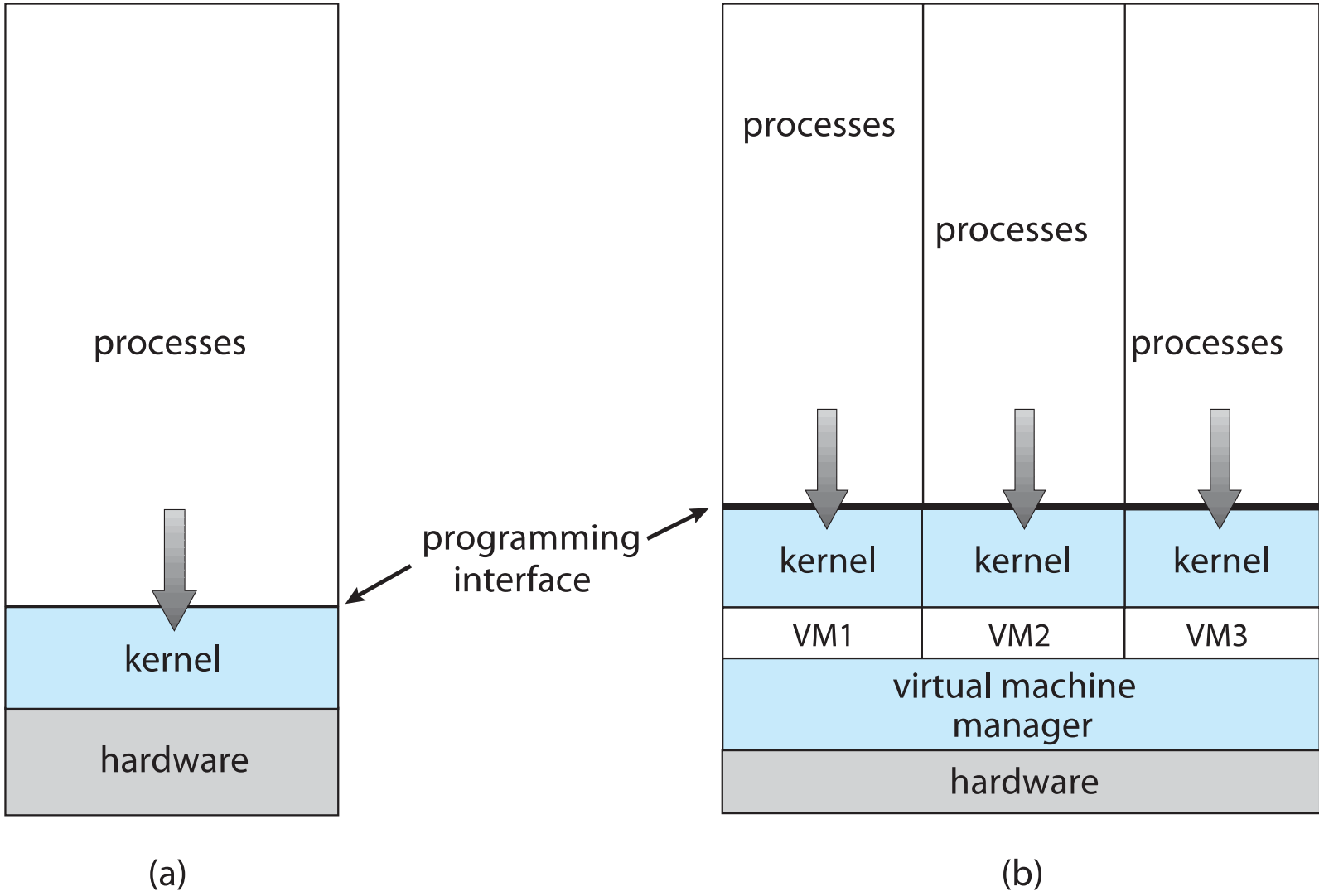
打开电源，计算机执行的第一句指令什么?

From: 哈工大 李治军 “打开钢琴的盖子”

Chapter 1: Introduction

- 什么是操作系统？
- Operating-System Basic Operations
  - Virtualization
  - Resource Management
  - Security and Protection …
- 计算机的基本组成结构
- 计算机的指令系统
- 中断 & 系统调用
- 用户模式、系统模式的概念
- 操作系统引导的过程
- 其他：虚拟机、分布式操作系统、开源操作系统等

Computing Environments - Virtualization



Distributed Systems

- Distributed computiing
  - Collection of separate, possibly heterogeneous, systems networked together
    - **Network** is a communications path, **TCP/IP** most common
      - **Local Area Network** (**LAN**)
      - **Wide Area Network** (**WAN**)
      - **Metropolitan Area Network** (**MAN**)
      - **Personal Area Network** (**PAN**)
  - **Network Operating System** provides features between systems across network
    - Communication scheme allows systems to exchange messages
    - Illusion of a single system

Computing Environments - Traditional

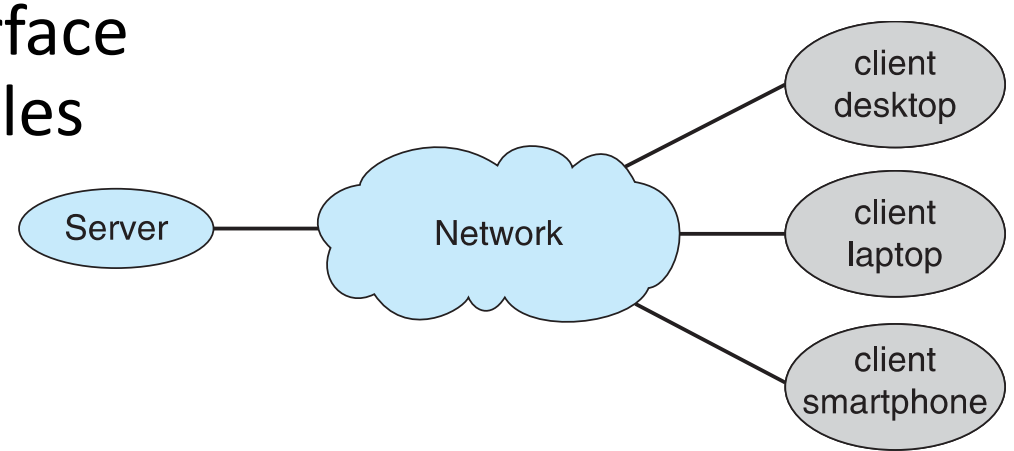
- Stand-alone general purpose machines
- But blurred as most systems interconnect with others (i.e., the Internet)
- **Portals** provide web access to internal systems
- **Network computers** (**thin clients**) are like Web terminals
- Mobile computers interconnect via **wireless networks**
- Networking becoming ubiquitous – even home systems use **firewalls** to protect home computers from Internet attacks

Computing Environments - Mobile

- Handheld smartphones, tablets, etc
- What is the functional difference between them and a “traditional” laptop?
- Extra feature – more OS features (GPS, gyroscope)
- Allows new types of apps like ***augmented reality***
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders are **Apple iOS** and **Google Android**

Computing Environments – Client-Server

- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now **servers**, responding to requests generated by **clients**
    - **Compute-server system** provides an interface to client to request services (i.e., database)
    - **File-server system** provides interface for clients to store and retrieve files

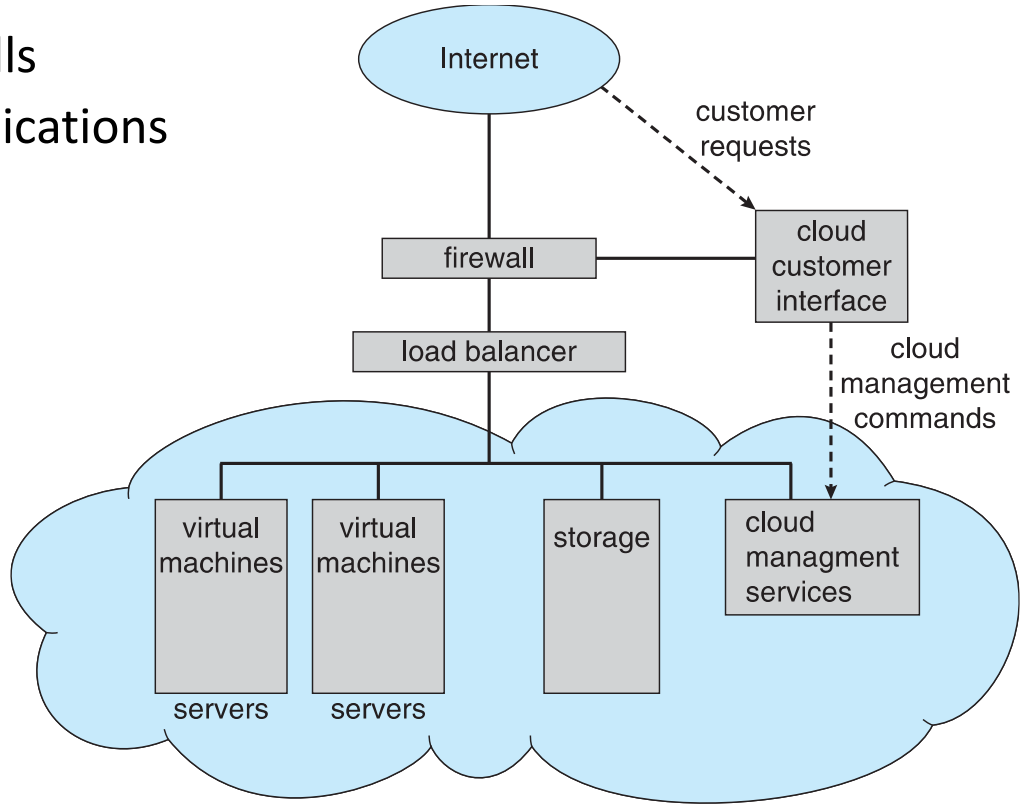


Computing Environments – Cloud Computing

- Delivers computing, storage, even apps as a service across a network
- Logical extension of virtualization because it uses virtualization as the base for it functionality.
  - Amazon **EC2** has thousands of servers, millions of virtual machines, petabytes of storage available across the Internet, pay based on usage
- Many types
  - **Public cloud** – available via Internet to anyone willing to pay
  - **Private cloud** – run by a company for the company's own use
  - **Hybrid cloud** – includes both public and private cloud components
  - Software as a Service (**SaaS**) – one or more applications available via the Internet (i.e., word processor)
  - Platform as a Service (**PaaS**) – software stack ready for application use via the Internet (i.e., a database server)
  - Infrastructure as a Service (**IaaS**) – servers or storage available over Internet (i.e., storage available for backup use)

Computing Environments – Cloud Computing

- Cloud computing environments composed of traditional OSes, plus VMMs, plus cloud management tools
  - Internet connectivity requires security like firewalls
  - Load balancers spread traffic across multiple applications



Computing Environments – Real-Time Embedded Systems

- Real-time embedded systems most prevalent form of computers
  - Vary considerable, special purpose, limited purpose OS, **real-time OS**
  - Use expanding
- Many other special computing environments as well
  - Some have OSes, some perform tasks without an OS
- Real-time OS has well-defined fixed time constraints
  - Processing ***must*** be done within constraint
  - Correct operation only if constraints met

Free and Open-Source Operating Systems

- Operating systems made available in source-code format rather than just binary **closed-source** and **proprietary**
- Counter to the **copy protection** and **Digital Rights Management (DRM)** movement
- Started by **Free Software Foundation (FSF)**, which has "copyleft" **GNU Public License (GPL)**
  - Free software and open-source software are two different ideas championed by different groups of people
    - http://gnu.org/philosophy/open-source-misses-the-point.html/
- Examples include **GNU/Linux** and **BSD UNIX** (including core of **Mac OS X**), and many more
- Can use VMM like VMware Player (Free on Windows), Virtualbox (open source and free on many platforms - http://www.virtualbox.com)
  - Use to run guest operating systems for exploration

总结

- 什么是操作系统？
- Operating-System Basic Operations
  - Virtualization
  - Resource Management
  - Security and Protection …
- 计算机的基本组成结构
- 计算机的指令系统
- 中断 & 系统调用
- 用户模式、系统模式的概念
- 操作系统引导的过程
- 其他：虚拟机、分布式操作系统、开源操作系统等