操作系统中的
复杂问题和工程思维

南开大学 宫晓利
2019.11.22 杭州

目录

复杂工程问题

工程教育认证标准中提到的“复杂工程问题”，指的是复杂的工程问题，而不是复杂工程的问题。“复杂工程问题”必须具备下述特征(1)，同时具备下述特征(2)—(7)的部分或全部：

(1)必须运用深入的工程原理，经过分析才可能得到解决；

(2)涉及多方面技术、工程和其它因素，并可能相互有一定冲突；

(3)需要通过建立合适的抽象模型才能解决，在建模过程中需要体现出创造性；

(4)不是仅靠常用方法就可以完全解决的；

(5)问题中涉及的因素可能没有完全包含在专业工程实践的标准和规范中；

(6)问题相关各方的利益不完全一致；

(7)具有较高的综合性，包含多个相互关联的子问题。

操作系统正是一个复杂的工程问题

- 操作系统有一整套复杂的理论
  - 进程、同步、虚拟地址、文件……
- 操作系统的中涉及多种复杂的工程细节
  - 启动上电的前向兼容，第一个进程加载前的页表
- 操作系统的实现，需要多门相关学科的知识
  - 数据结构、C语言、汇编、编译等
- 操作系统的设计和实现，对应了模型和实战的多种因素
  - 调度算法、换页算法、空页分配算法、文件系统……
- 操作系统中总有“按下葫芦浮起瓢”的情况
  - 吞吐与延迟，空间与时间，优先级与公平……

教学目标

- 必要的知识点讲解
- 用课程实验对照相应的知识点
- 融入Linux操作系统的原理和使用
- 培养学生解决复杂工程问题的能力
- 吸引并支撑学生参加系统相关的竞赛和科研活动

实验课程的演变

- 基于JOS操作系统的补全任务
- 优点：知识详实具体，难度适中
- 缺点：代码过于简单，工程思维无从谈起
- 改进1：增大工作量，由3人一组，变为一人一组，进度压缩
- 改进2：进行Tizen移植实验做为补充

第一阶段：Tizen移动计算平台的移植实验

Tizen移动计算平台的移植实验

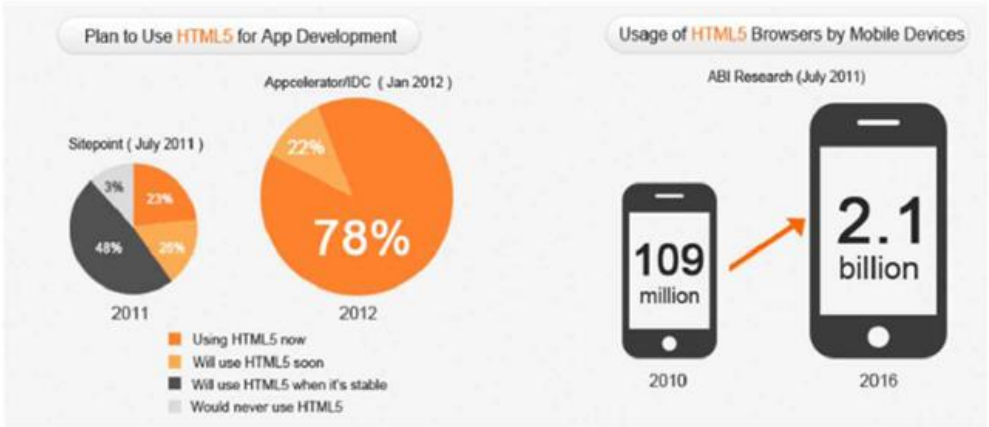
- Tizen: 全开源的，基于C/C++的移动计算框架
- Odroid-U3：基于ARM的开发板
- 熟悉Linux操作系统的各类控制命令
- 熟悉开源代码的管理方法
- 了解一个操作系统的部署过程
- 了解操作系统中启动、进程、驱动等概念在实际系统中的应用

Tizen 平台在开发板上的移植

- **Standard-based, cross category platform**
  - Provide common & multiple categories of compliances
- **Open Source Platform**
  - Tizen project resides within the Linux Foundation
- **Provides a robust and flexible environment for HTML5 based application**

Tizen is W3C Standard-Based

- HTML5 is being adopted rapidly, especially for mobile Web app development
- Tizen has the top score in html5test.com



[1] http://www.dotcominfoway.com/blog/dot-com-infoway-releases-html5-infographic

Industry Support

- Guiding the industry roles of Tizen

TIZEN™ ASSOCIATION

- Gathering
  - Requirements
  - Identification and Facilitation of service models

- **NEC, Panasonic, and Telefonica leaved Tizen Association.**
- **Telefonica: Firefox Phone**

Tizen Open Source Information

Visit

- http://www.tizen.org
- http://developer.tizen.org/sdk
- http://source.tizen.org/
- https://developer.tizen.org/documentation

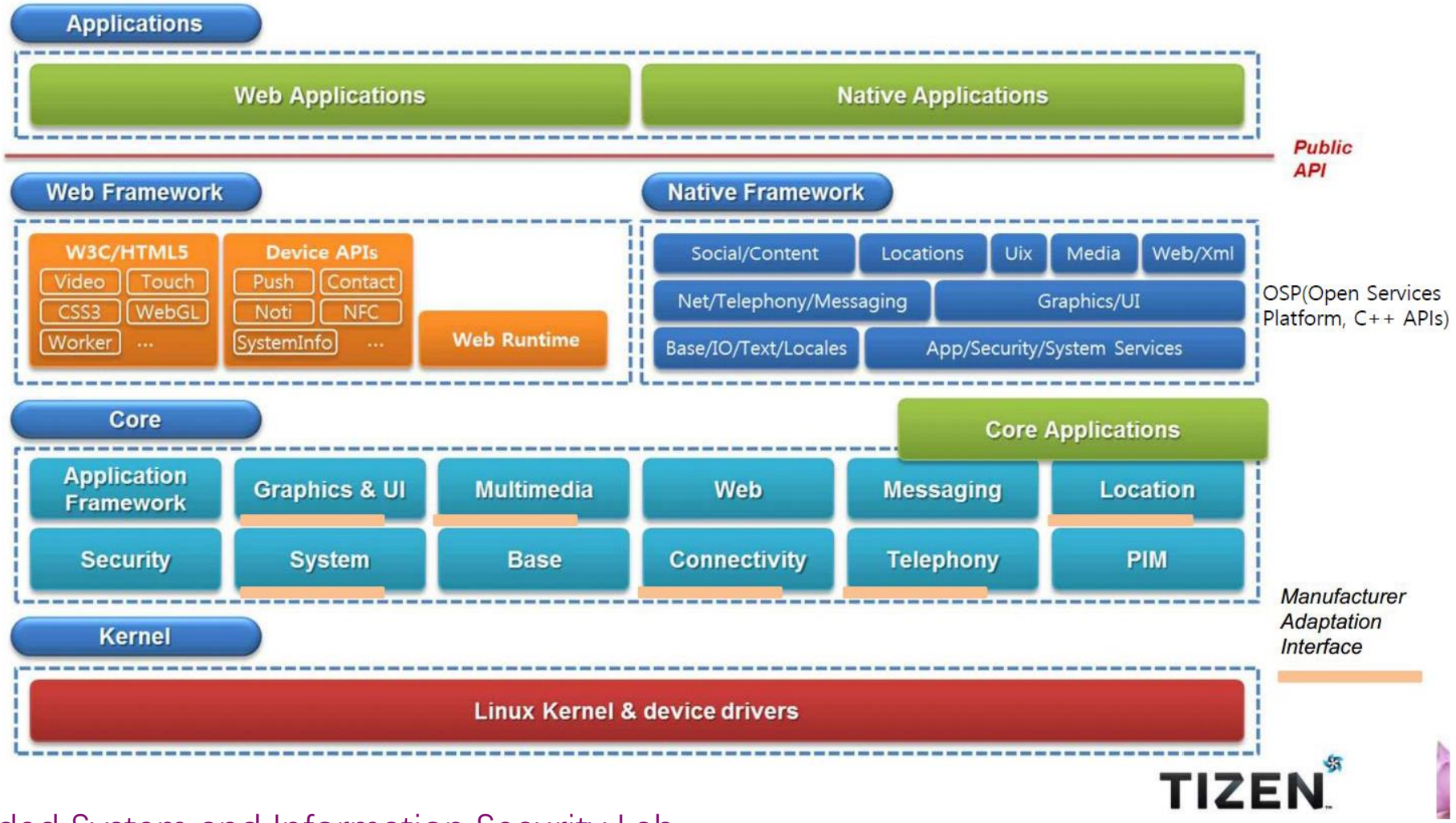
Community

- Mailing lists: http://www.tizen.org/community/mailing-lists
- IRC Channel: #tizen
- Wiki: https://www.tizen.org/community/wiki
- JIRA: http://bugs.tizen.org

Tizen Developer Conference

- https://www.tizen.org/conference

Tizen v2.3 Architecture Overview

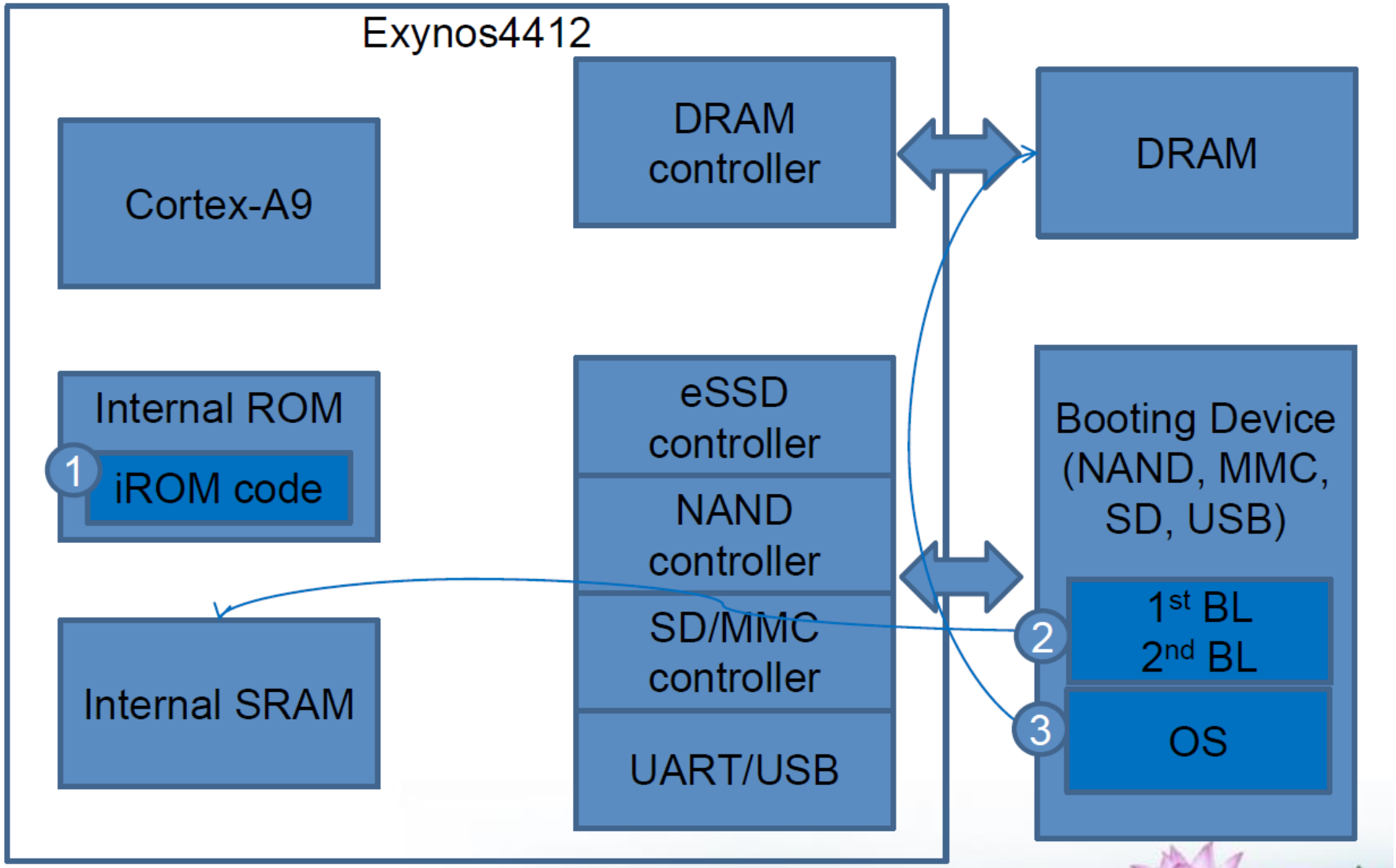


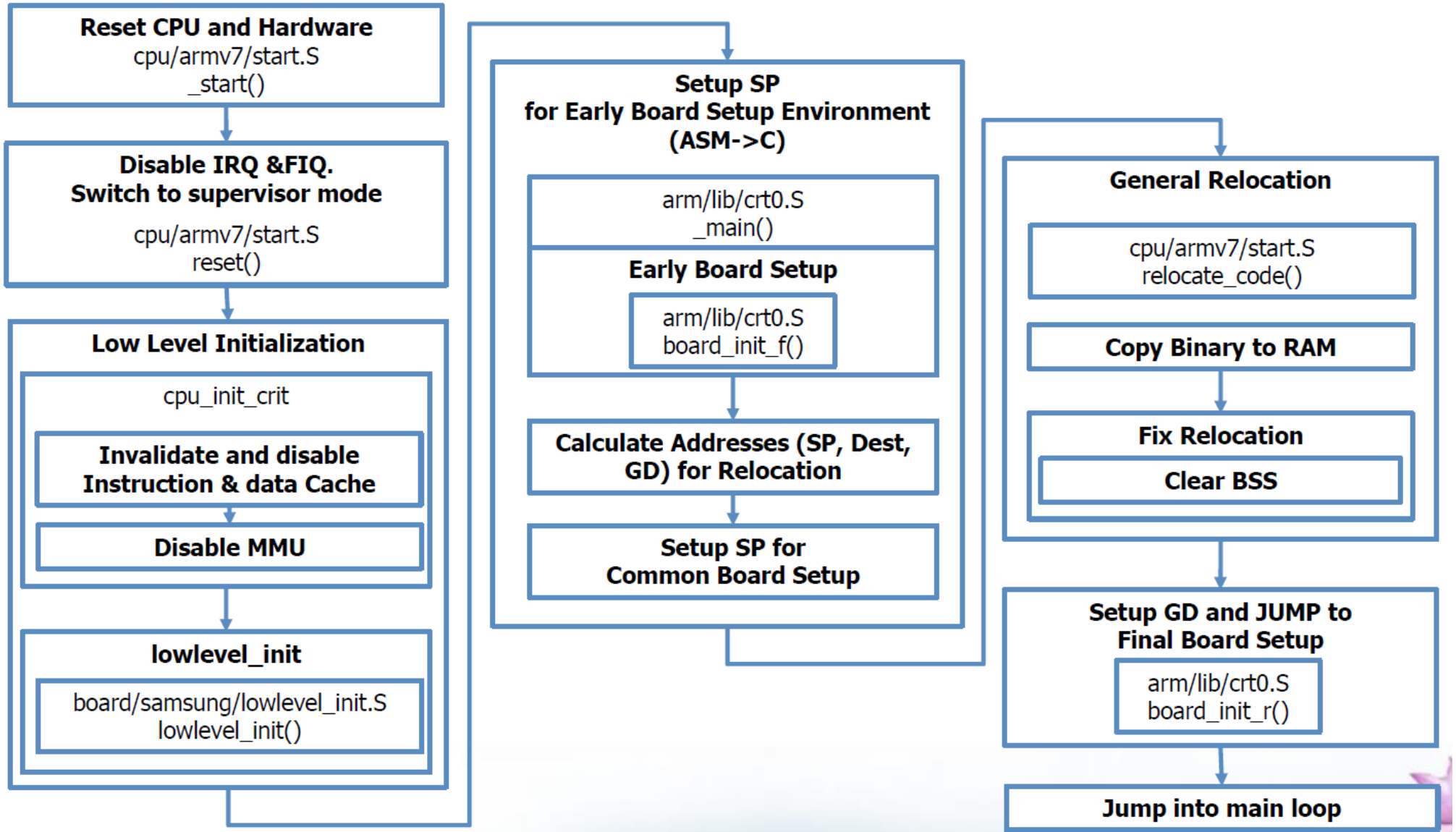
Tizen移植实验的目标平台

- Odroid U3
- Samsung Exynos4412 Prime Cortex-A9 Quad Core 1.7Ghz with 1MB L2 cache
- 2048MB(2GB) LP-DDR2 880Mega data rate
- Mali-400 Quad Core 440MHz
- micro HDMI connector
- LAN 10/100Mbps Ethernet with RJ-45 Jack (Auto-MDIX support)
- High speed standard A type connector x 3 ports
- HDMI monitor

Tizen实验设计

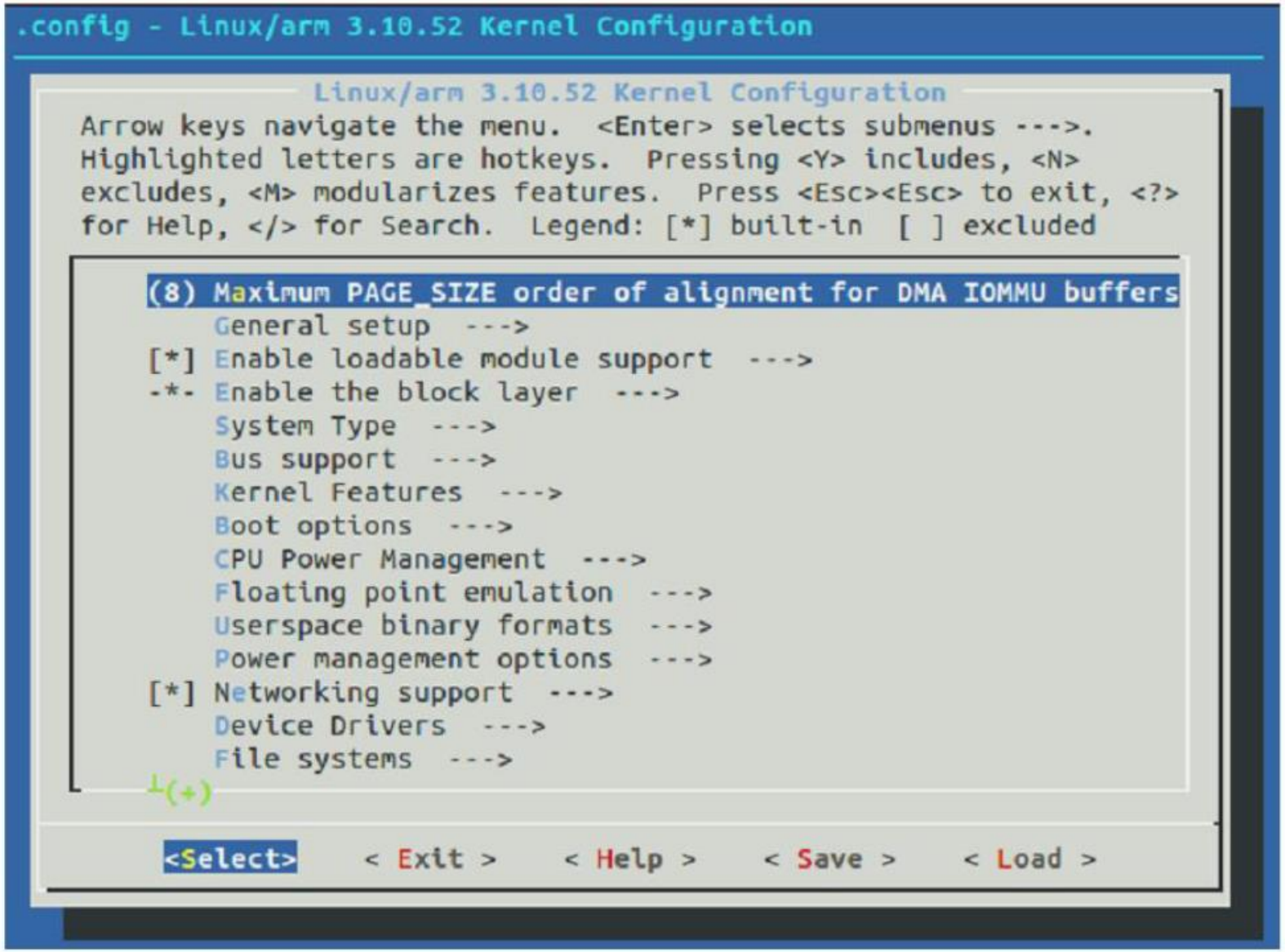
- 开源软件的使用
  - vim, git, latex
  - Github, overleaf, GBS repo
- 系统的上电过程分析
  - uboot 和 Linux kernel加载


Exynos4412
Cortex-A9
DRAM controller
DRAM
Internal ROM
1
iROM code
eSSD controller
NAND controller
SD/MMC controller
UART/USB
Booting Device (NAND, MMC, SD, USB)
2
1st BL
2nd BL
3
OS
Internal SRAM


Reset CPU and Hardware
cpu/armv7/start.S
_start()
Disable IRQ &FIQ.
Switch to supervisor mode
cpu/armv7/start.S
reset()
Low Level Initialization
cpu_init_crit
Invalidate and disable
Instruction & data Cache
Disable MMU
lowlevel_init
board/samsung/lowlevel_init.S
lowlevel_init()
Setup SP
for Early Board Setup Environment
(ASM->C)
arm/lib/crt0.S
_main()
Early Board Setup
arm/lib/crt0.S
board_init_f()
Calculate Addresses (SP, Dest,
GD) for Relocation
Setup SP for
Common Board Setup
General Relocation
cpu/armv7/start.S
relocate_code()
Copy Binary to RAM
Fix Relocation
Clear BSS
Setup GD and JUMP to
Final Board Setup
arm/lib/crt0.S
board_init_r()
Jump into main loop

Tizen实验设计

- 开源软件的使用
  - vim, git, latex
  - Github, overleaf, GBS repo
- 系统的上电过程分析
  - uboot 和 Linux kernel加载
- 系统软件的配置与编译
  - Linux Kernel编译
  - Framework 编译

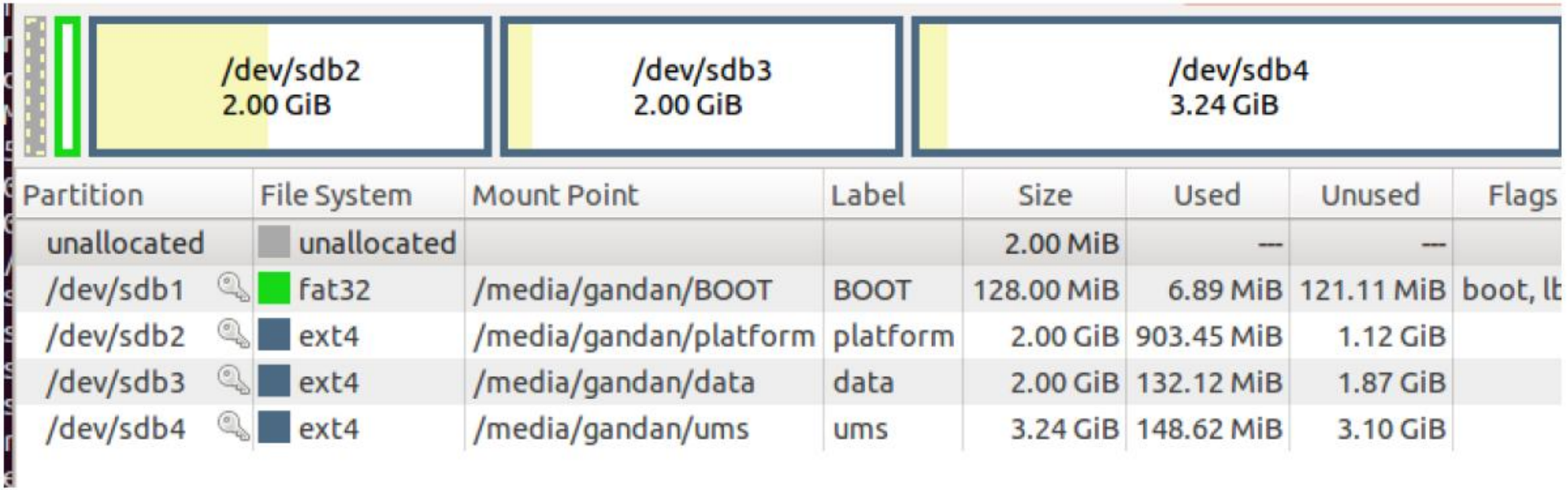

.config - Linux/arm 3.10.52 Kernel Configuration
Linux/arm 3.10.52 Kernel Configuration
Arrow keys navigate the menu. <Enter> selects submenus --->.
Highlighted letters are hotkeys. Pressing <Y> includes, <N>
excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?>
for Help, </> for Search. Legend: [*] built-in [ ] excluded
(8) Maximum PAGE_SIZE order of alignment for DMA IOMMU buffers
General setup --->
[*] Enable loadable module support --->
-*- Enable the block layer --->
System Type --->
Bus support --->
Kernel Features --->
Boot options --->
CPU Power Management --->
Floating point emulation --->
Userspace binary formats --->
Power management options --->
[*] Networking support --->
Device Drivers --->
File systems --->
<Select> < Exit > < Help > < Save > < Load >

~/tizen-platform/profile/mobile/model/config-odroid-u3/model-config.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<model-config version="2.2.0" model="ODROID-U3-REF">
 <platform>
  <!-- Model Name, Platform Name/Version, Processor Name -->
  <key name="tizen.org/system/model_name" type="string">ODROID-U3</key>
  <key name="tizen.org/system/platform.name" type="string">Tizen</key>
  <key name="tizen.org/feature/platform.version" type="string">2.2</key>
  <key name="tizen.org/system/platform.processor" type="string">exynos4412</key>

  <!-- Specification of Features -->
  <key name="tizen.org/feature/camera" type="bool">false</key>
  <key name="tizen.org/feature/camera.back" type="bool">false</key>
  </platform>
  <custom>
  </custom>
</model-config>
```

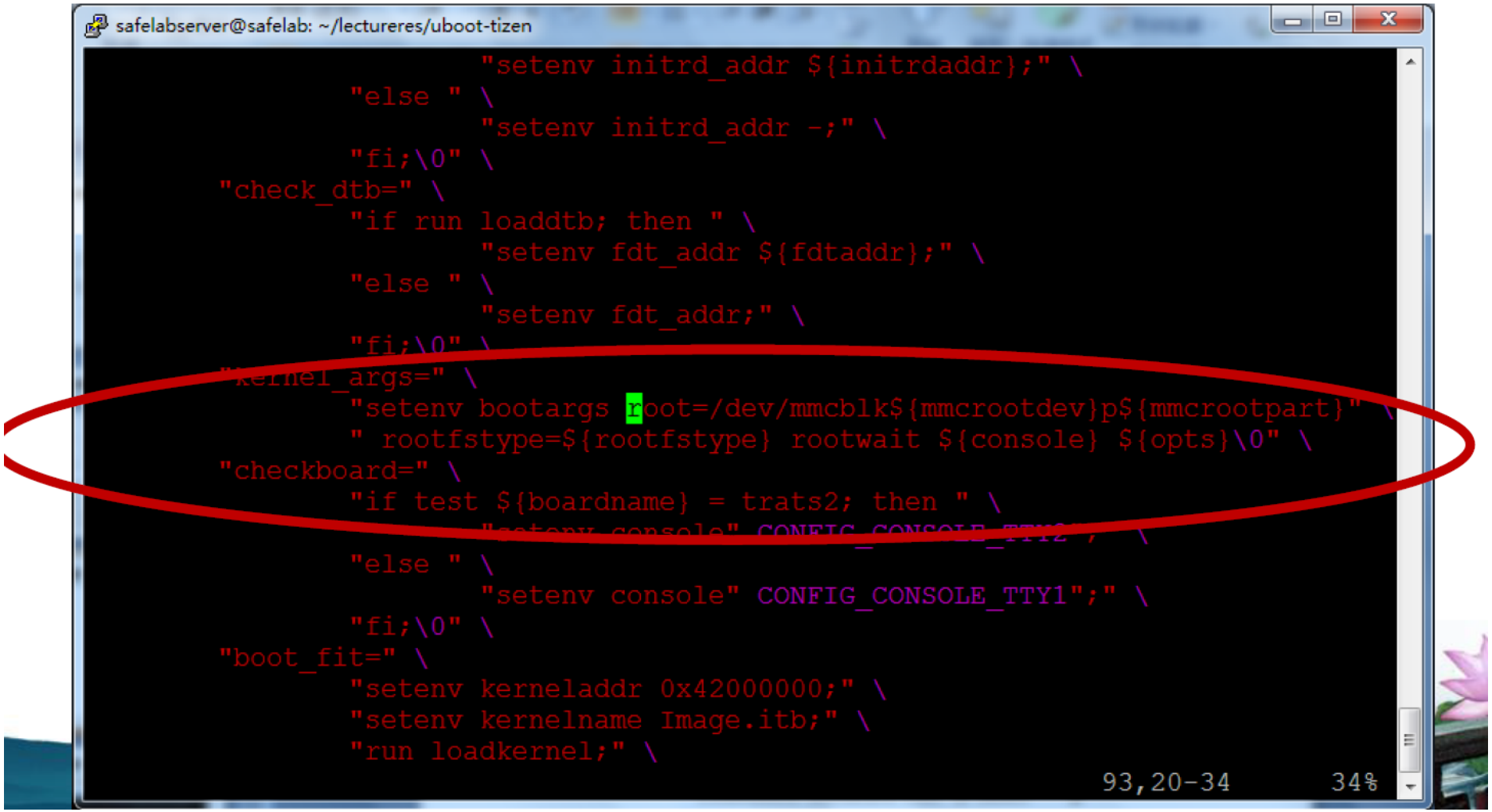
Tizen实验设计



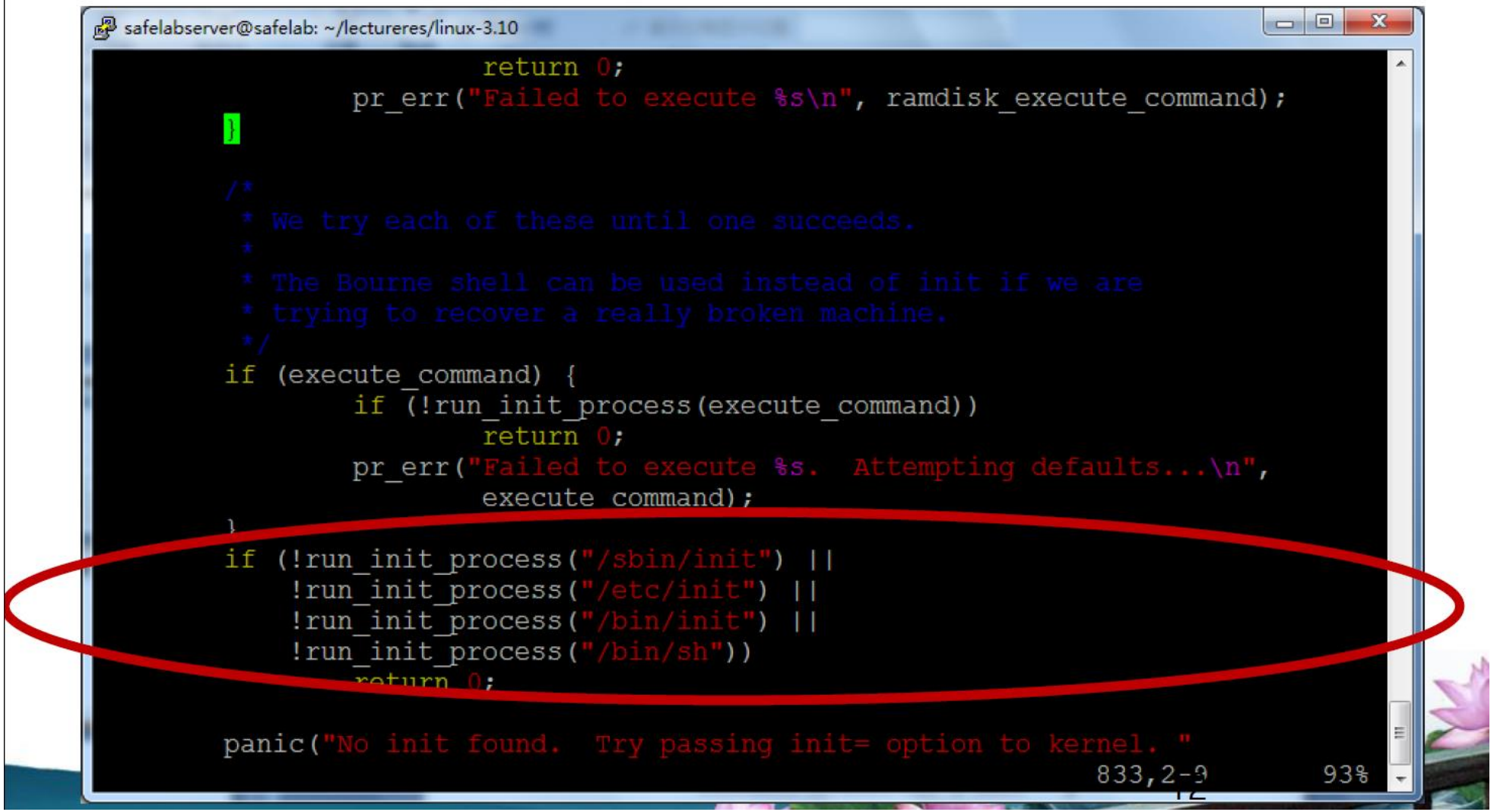
Partition	File System	Mount Point	Label	Size	Used	Unused	Flags
unallocated	unallocated			2.00 MiB	---	---	
/dev/sdb1	fat32	/media/gandan/BOOT	BOOT	128.00 MiB	6.89 MiB	121.11 MiB	boot, lb
/dev/sdb2	ext4	/media/gandan/platform	platform	2.00 GiB	903.45 MiB	1.12 GiB	
/dev/sdb3	ext4	/media/gandan/data	data	2.00 GiB	132.12 MiB	1.87 GiB	
/dev/sdb4	ext4	/media/gandan/ums	ums	3.24 GiB	148.62 MiB	3.10 GiB	

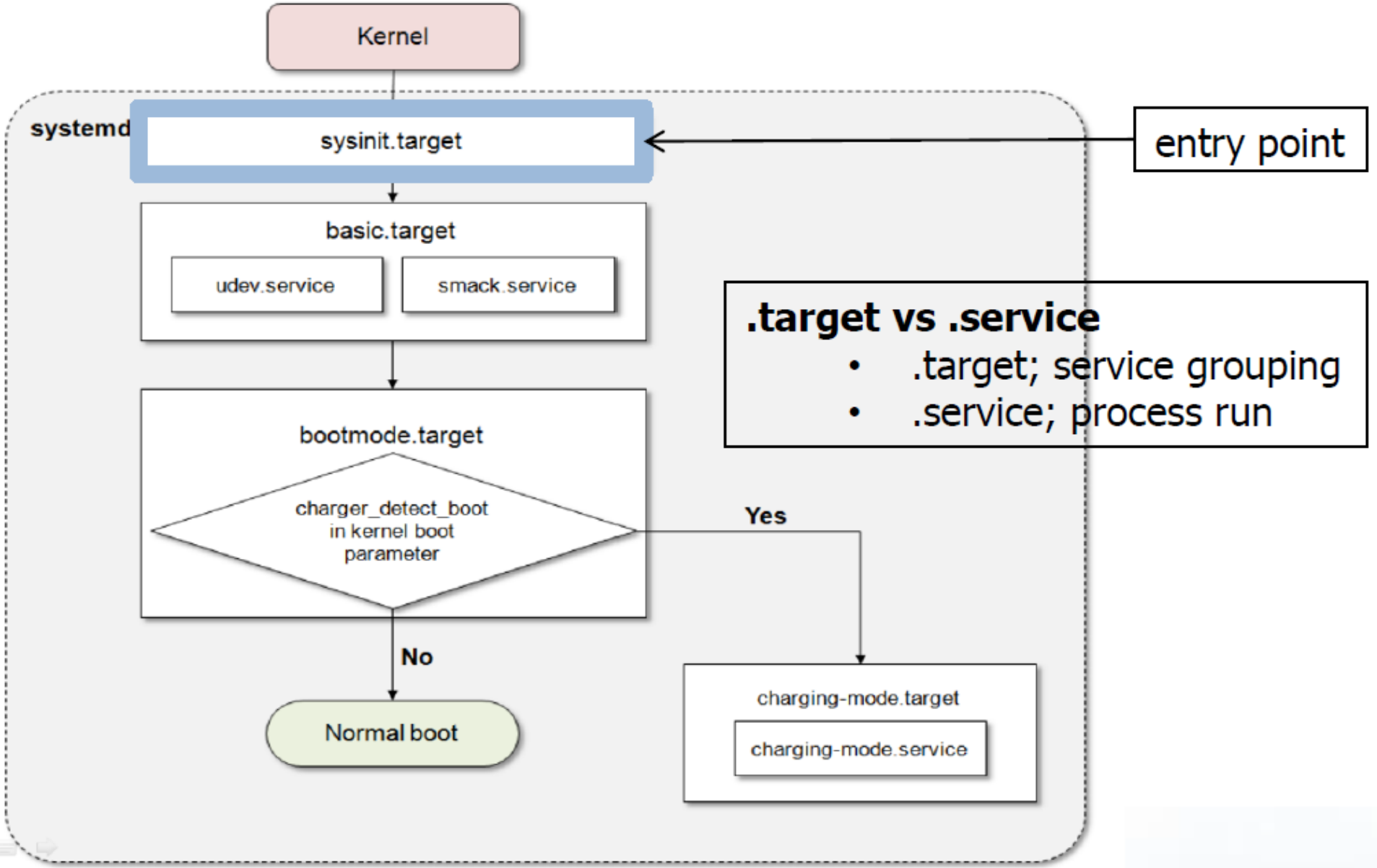
- 存储分区管理和使用
- 系统加载和进程创建的过程分析

In u-boot source code, root file system type is specified: include/configs/tizen.h



Kernel source : init/main.c kernel_init




Kernel
systemd
sysinit.target
entry point
basic.target
udev.service
smack.service
.target vs .service
• .target; service grouping
• .service; process run
bootmode.target
charger_detect_boot
in kernel boot
parameter
Yes
No
Normal boot
charging-mode.target
charging-mode.service

Tizen实验收获（2015年-2018年）

- 教学项目不能依赖开源社区
  - 活跃的项目常常不能相互匹配
  - 不活跃的项目下载不了
- 太难的项目学生做不了
  - Tizen文档不足，遇到的未知问题无从下手
  - 学生水平参差不齐，部分学生收获很多，部分学生怨声载道
- 非教学类装备在教学过程中损耗严重
  - 13套开发板，在第三年后能够正常使用的不足5套

第二阶段：基于ucore的操作系统工程实践

工程思维
Design Concepts for Engineers

- 设计
  - 如何做头脑风暴和记录
- 团队工作
  - 如何组织团队讨论、分工
- 制定清单和进度
  - 进度管理的工具
- 撰写过程文档
  - 文档的重要性、规范的必要性
- 演讲表达
  - 如何交流、如何演讲、如何回答问题
- ……

ucore vs JOS

□JOS:

- 来自MIT 6.828课程的实验，一个非常简易的OS，能够实现启动、内存管理、进程、中断、系统调用等基本功能的学习
- 存在问题：代码太过简单，没有兼容和扩展性的考虑；网上的低质答案太多

□ucore:

- 是一个由清华大学开发的教学用操作系统。它参考了UNIX,xv6的代码设计，支持系统引导、中断异常处理、基于页的虚拟内存管理、进程管理、进程间通信、文件系统等功能。保留了许多核心数据结构和编码方式。
- 该系统可以运行于x86、RISC-V、ARM、MIPS等多个平台。

ucore on X86 & mips & risc-v

- ucore代码相对简单，也可以支持多种体系结构
- 以X86为主体，同时对比其他多种体系结构
- 主要完成X86的代码补全，有能力的学生同时推进其他结构
- 定期讨论交流
- 体现工程思维：
  - 如何划分面向体系结构的代码和面向OS流程的代码
  - 操作系统如何利用体系结构的新特性

实验内容

- 01 操作系统兼容体系结构的差异，即ucore在risc-v上的移植
  - 系统启动
  - 中断管理
  - 基于页面的虚拟内存管理
  - 内核线程与用户进程
  - 进程调度通信与文件系统
- 02 体系结构的特性在操作系统中的使用
  - PUM模式
  - Cache模式

01 操作系统兼容体系结构的差异

1.1 系统启动

- x86：BIOS实现
- RISC-V：Berkeley Boot-loader，模拟了一个可以实现部分I/O功能的RISC-V实例。
  - Supervisor Binary Interface：通过ecall调用实现

提供配置好的环境以及文档供学生直接使用，做原理讲解，理解x86的复杂所在

01 操作系统兼容体系结构的差异

1.2 中断管理

- RISC-V：众多CSR寄存器

Name	Description
sstatus/mstatus	各种状态信息
sie/mie	处理器目前能处理和必须忽略的中断
stvec/mtvec	PC被设置为stvec
sscratch/mscrach	暂时存放一个字大小的数据
sepc/mepc	发生例外的指令被存入sepc
scause/mcause	表示异常类型
sbadaddr/mbadaddr	出错地址（这个CSR的功能在Pv1.10中被合并到另外一个寄存器里）
sip/mip	指出目前正准备处理(pending)的中断
sptbr	Page-table base register
medeleg	控制哪些异常（同步）委托给S模式
mideleg	控制哪些中断（异步）委托给S模式

01 操作系统兼容体系结构的差异

1.2 中断管理——S-MODE下的中断管理

- 硬件自动对S-MODE下的CSR进行一些处理
  - 此时的PC存入sepc，PC则被设为stcev中的值，也就是切入通用中断处理函数
  - scause会根据异常类型存入相应的错误码
  - sstatus中的SIE位的值存入SPIE中后置0全局屏蔽中断
  - 异常触发时所在的特权级放入sstatus中的SPP域，然后把当前特权级设为S-MODE
- 处理中断
  - 将所有会用到的整数寄存器和CSR的值保存起来
  - 进入trap函数
- 中断返回
  - 用之前保存的寄存器值恢复中断触发现场。

01 操作系统兼容体系结构的差异

1.2 中断管理——中断管理的比较

- x86：中断向量　中断描述符表
- RISC-V：为中断提供了多个CSR寄存器帮助处理，能够很快识别包括中断使能、中断委托、异常类型、出错地址、异常处理程序入口地址等在内的多个寄存器内容

寄存器的多少，是RISC与CISC的主要差别之一，也是影响性能的主要因素之一

01 操作系统兼容体系结构的差异

1.3 虚拟内存管理

- X86：段页式内存管理
- RISC-V：页式内存管理
  - 页表格式：

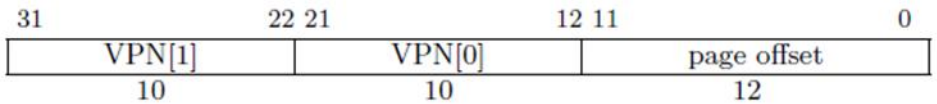


Figure 4.11: Sv32 virtual address.

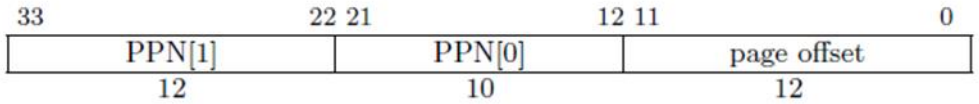
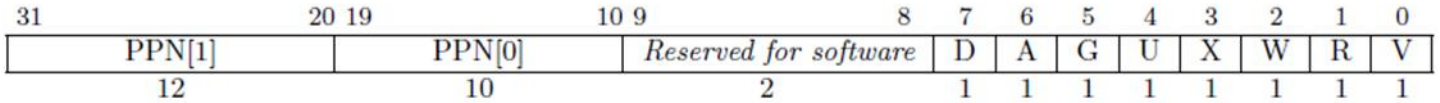


Figure 4.12: Sv32 physical address.



01 操作系统兼容体系结构的差异

1.3 虚拟内存管理——虚拟内存映射机制 （以SV32为例）

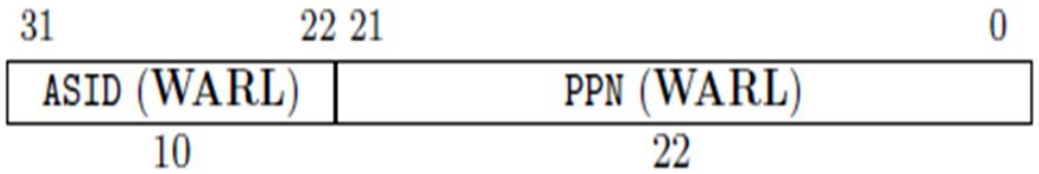
- 页表项中权限位编码

X	W	R	Meaning
0	0	0	Pointer to next level of page table.
0	0	1	Read-only page.
0	1	0	*Reserved for future use.*
0	1	1	Read-write page.
1	0	0	Execute-only page.
1	0	1	Read-execute page.
1	1	0	*Reserved for future use.*
1	1	1	Read-write-execute page.

01 操作系统兼容体系结构的差异

1.3 虚拟内存管理——虚拟内存映射机制 （以SV32为例）

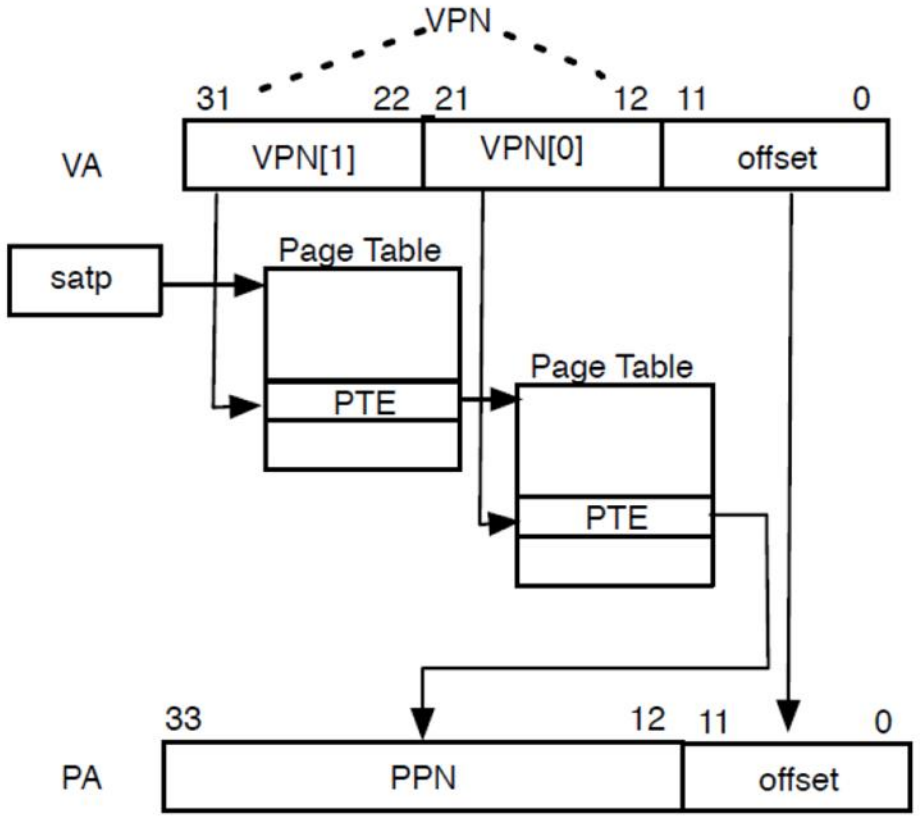
- Sptbr寄存器
  - 类似x86中cr3寄存器
  - 保证取ASID和PPN为一个原子操作



01 操作系统兼容体系结构的差异

1.3 虚拟内存管理——虚拟内存映射机制 （以SV32为例）

- 地址转换过程



01 操作系统兼容体系结构的差异

1.4 进程管理

- ucore在RISC-V平台上的相关数据结构和执行逻辑与X86类似。
- 两个体系结构上的差异主要表现为两种体系结构寄存器名称、功能、设置的不同。

如何划分和融合面向体系结构的代码和面向OS流程的代码?

01 操作系统兼容体系结构的差异

1.5 进程调度 进程同步

- 进程调度
  - 缺省的Round-Robin 调度算法
  - Stride Scheduling调度算法
- 进程同步
  - 信号量机制
  - 管程机制

X86和RISC-V两种体系架构上面的实现相同，让学生体会平台无关代码的优势。

02 体系结构的特性在操作系统中的使用

2.1 虚拟内存管理的新特性：PUM模式

内存保护措施：PUM模式

- PUM模式是在虚拟内存模式下提供的一种内存保护措施。
- 通过设置mstatus 中的PUM 位。PUM=0 的时，一切访问和内存保护照常进行，如果PUM=1，则S_MODE 下不可以访问U_MODE（PTE中U=1）下可以访问的内存。

讨论

- 原有操作系统特权模式的安全隐患
- 如何应用这一机制到OS中？哪些部分会受到影响？

02 体系结构的特性在操作系统中的使用

2.1 Cache模式：Cache的设计和管理

◘具体表现

- 传统80386平台：不带有cache
- ARM920T平台：cache设置成VIVT虚地址索引，通过协处理器进行cache的刷新
- RISC-V：L1cache和L2cache可以采用PIPT寻址模式

有助于学生加深对虚拟地址概念的理解

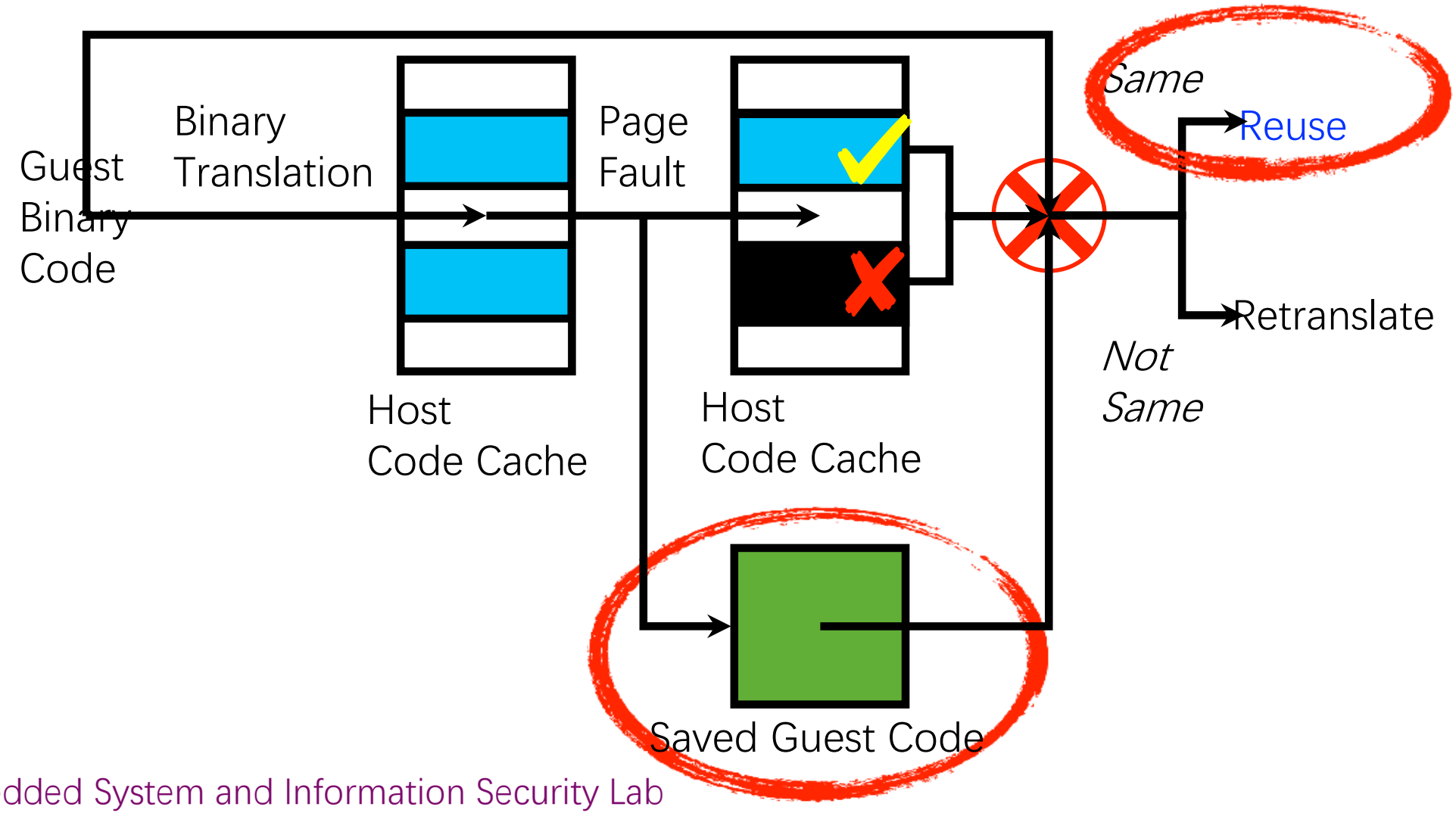
作业模式的变化

- 一人一组变回三人一组
- 时间点越来越松散，从lab3开始需要进度规划
- 项目文档从原理剖分变成工作日志和“吐槽集锦”
- 官方支持的本科生助教
- 为Linux相关内容重新开设课程

科研引导

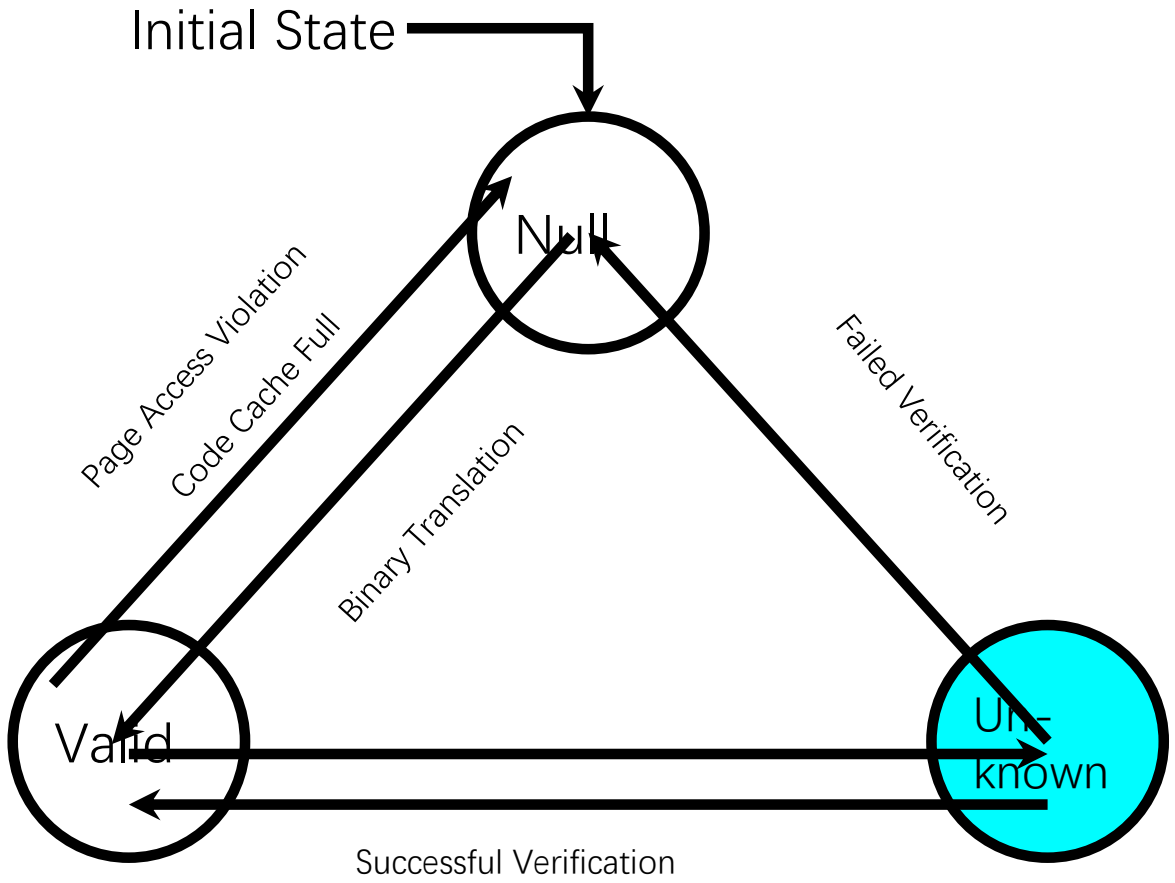
- 龙芯大赛
- 九五之芯奖学金

VEE论文： Host Code Reuse

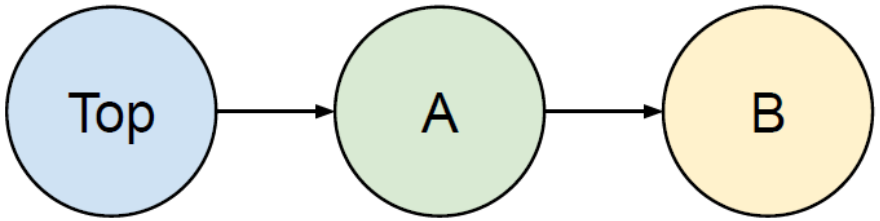


Reusing Host Binary Code

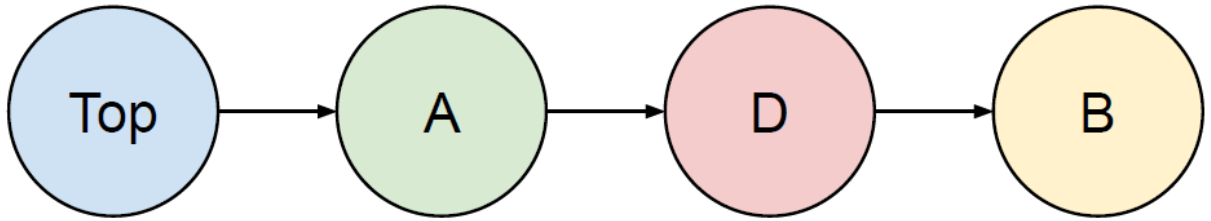
- Reuse after verification
  - Comparing the saved vs. new guest code
- A new state of host binary code



ABA问题

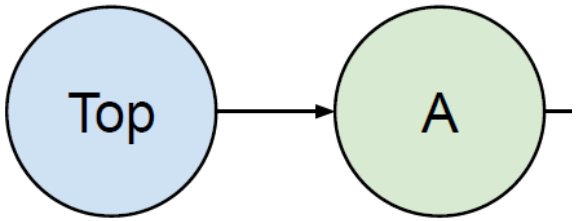


Thread i calls method POP that reads Top(pointer to A) and saves Top_next(pointer to B). Thread 1 gets preemted before doing Top.CAS(A, B).

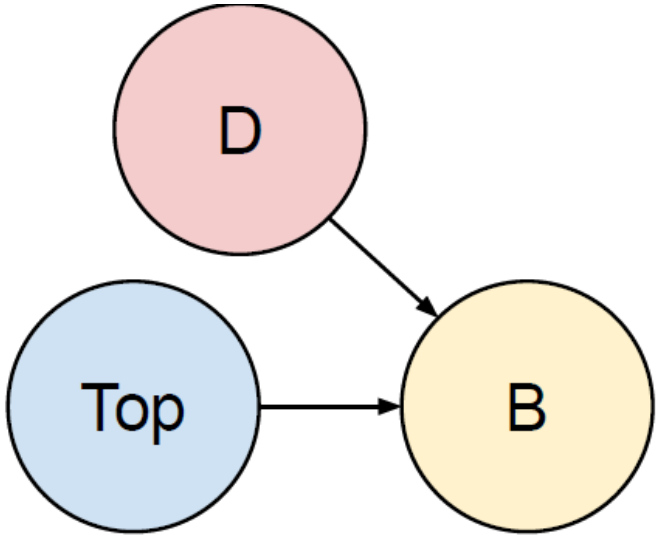


Thread j alse does POP by doing Top.CAS(A, B). After that Thread j calls PUSH with argument D by doing Top.CAS(B, D) and PUSH with A by doing Top.CAS(D, A).

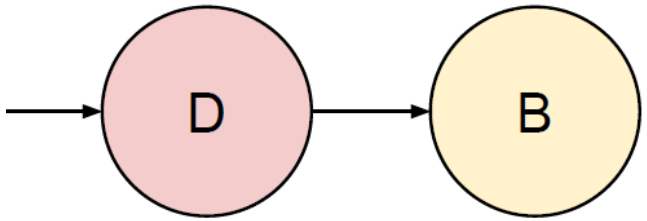
ABA问题



Thread i calls method PO
Top(pointer to A) and sav
Top_next(pointer to B). T
preemted before doing To



Thread i resumes and complete its CAS(A, B), leaving D hanging.



by doing Top.CAS(A, B).
PUSH with argument D
) and PUSH with A by

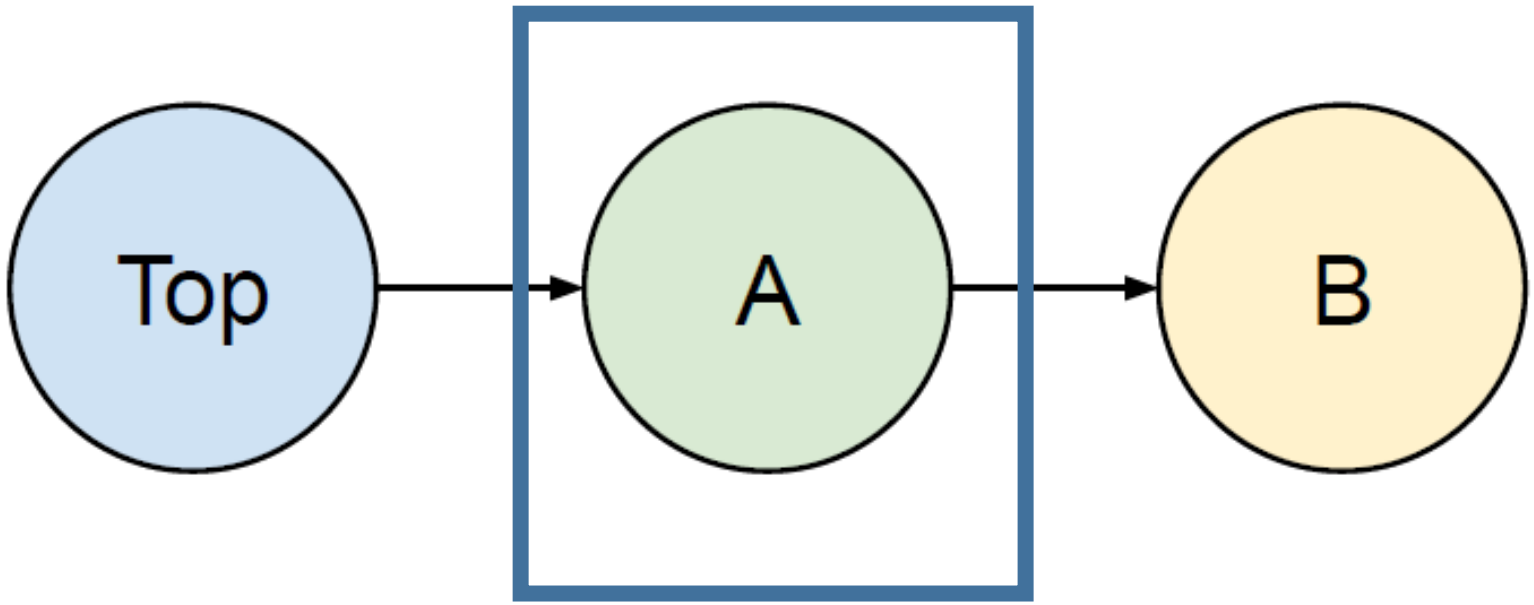
ABA问题

```
tryPush:

        ldr             r1, =p_top              @r1 = &p_top
        ldrex   r2, [r1]                @r2 = p_top
        str             r2, [r0, #4]    @r0 = obj_ptr, p_top -> obj_ptr->next
        strex   r2, r0, [r1]    @store obj_ptr -> *&p_top
        cmp             r2, #0                  @success?
        bne             tryPush

        mov             pc, lr
        .fnend
```

ABA问题的解决方案



总结

- 操作系统是一个复杂的课程，实验课程尤其重要
- 为学生增加工作量要量力而行，大部分学生需要循序渐进
- 适宜的工作量，加上针对性的引导，辅助学生建立工程思维
- 大工作量的课程，需要额外的方式促进学生的动力
- 学生的创造潜力巨大，正确的引导就可以取得优异的成果

Thank you!

Embedded System and Information Security Lab

南开大学 宫晓利　2019.11.22 杭州