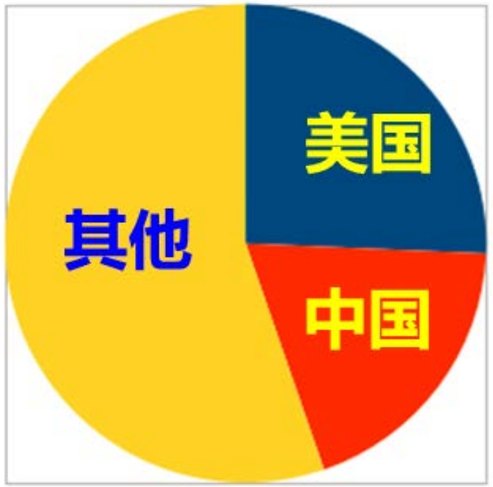
第二期“一生一芯”计划启动会

包云岗
中科院计算所
国科大计算机学院
2020.8.12

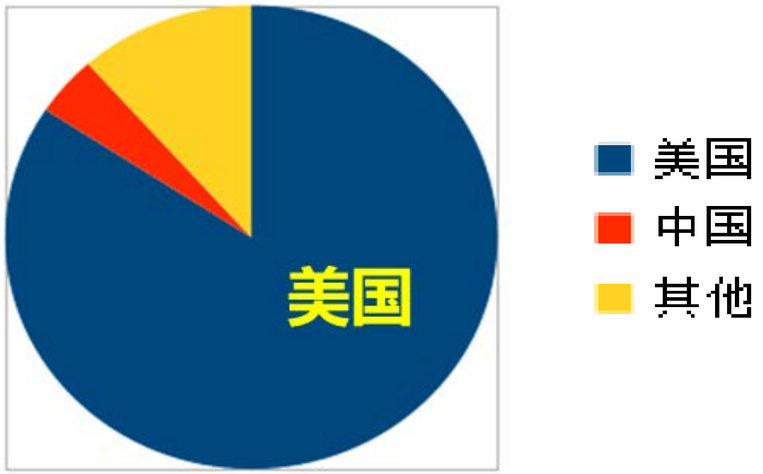
我国优秀处理器芯片人才储备严重不足

- **85% v.s. 4%**：2008-2017十年体系结构国际顶级会议ISCA论文的第一作者**85%**在美国，仅有**4%**在中国，不足美国的**1/20**，差距巨大
- 中国**加快**处理器芯片人才**培养规模与速度**，迫在眉睫

一作国籍



毕业去向



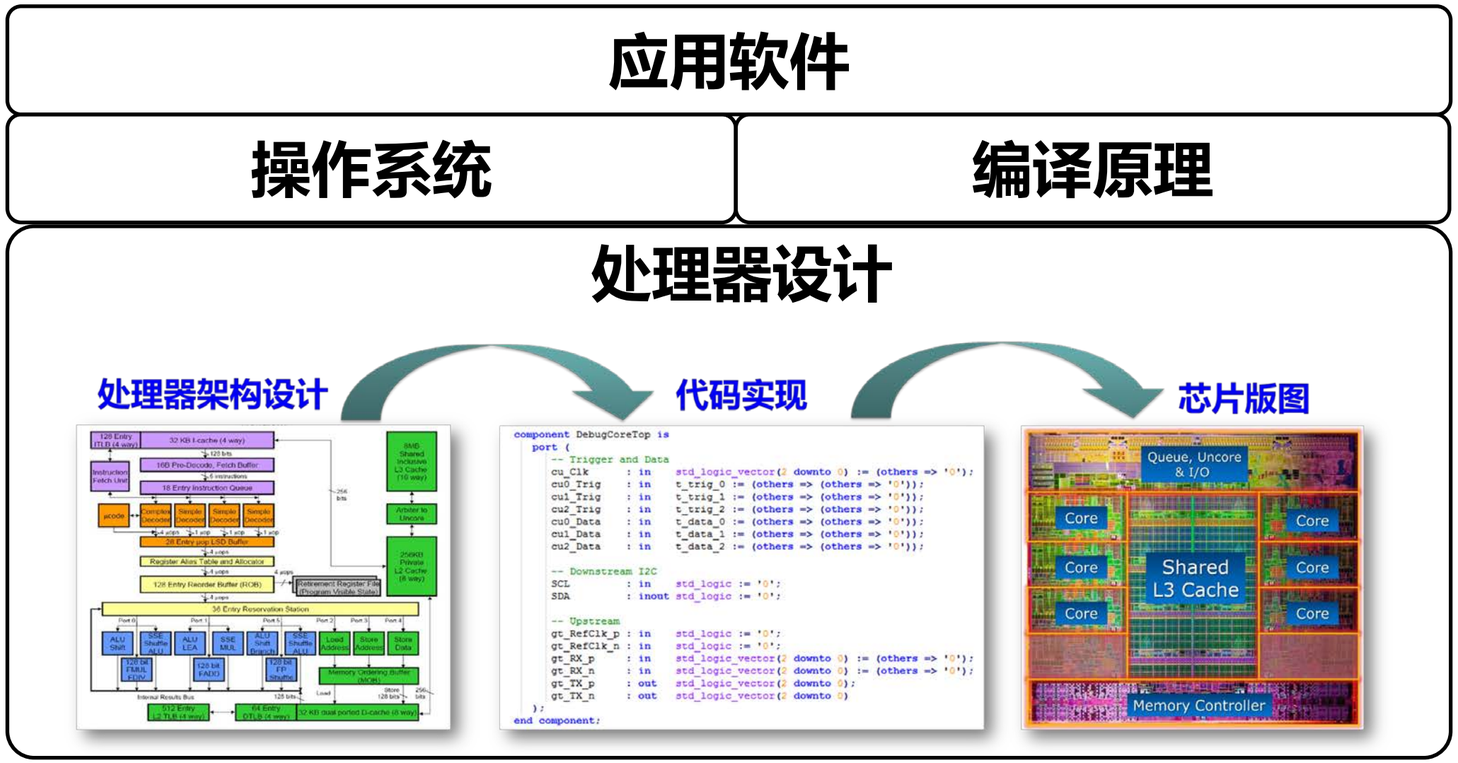
ISCA十年论文第一作者统计情况(2008-2017)

处理器芯片设计人才培养挑战大，需理论与实践并重

Image Source: https://dzone.com/articles/how-does-the-asic-design-flow-cycle-works

启动“一生一芯”计划，探索贯通性实践课程

- **硅上做教学**：“一生一芯”通过**让本科生设计处理器芯片并完成流片、运行操作系统**，贯通本科阶段**计算机系统课程知识点**，包括计算机组成原理、体系结构、操作系统、编译原理等



首期“一生一芯”计划
——让学生带着自己设计的处理器芯片毕业

- 2019年8月底，在国内**首次以流片为目标**，由**5位**2016级计算机学院本科生主导完成一款64位**RISC-V处理器SoC芯片**设计，于12月19日完成**流片**
- 芯片成功**运行Linux**操作系统与**国科大教学操作系统UCAS-Core**


金越


王华强

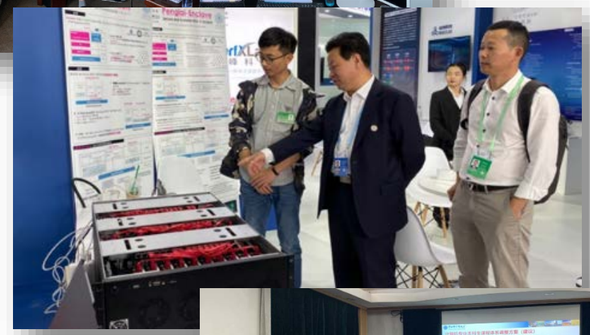

王凯帆


张林隽


张紫飞

首期“一生一芯”计划时间线

- **2019年8月：** 正式启动
- **2019年10月：** 李树深校长调研教学平台，
  听取“一生一芯”计划规划
- **2019年11月：** 向国科大校领导汇报阶段进展
- **2019年12月：** 64位RISC-V处理器“果壳”
  NutShell 完成设计，成功投片
- **2020年4月：** 果壳芯片返回，开始测试验证
- **2020年5月：** 芯片点亮，成功运行Linux
- **2020年6月：** 本科生毕业答辩展示



教学计划与实施

- 制定学习计划，循序渐进地安排实践任务
- 每周反馈进展与问题，安排新任务

学习计划

> 建议按照如下顺序学习:
> 1. [Chisel Bootcamp](https://github.com/freechipsproject/chisel-bootcamp)是一个很不错的chisel 教程, 还支持在线运行 chisel 代码,
> 你可以一边编写 chisel 代码一边学习. 其中
> * 第 1 章是 scala 入门
> * 第 2 章是 chisel 基础
> * 第 3 章是 scala 高级特性和 chisel 的混合使用
> * 第 4 章是 FIRRTL 后端相关内容
> 你需要完成前两章的学习, 同时我们强烈建议你学习第 3 章.
> 第 4 章和本课程没有直接关系, 可以作为课外阅读材料.

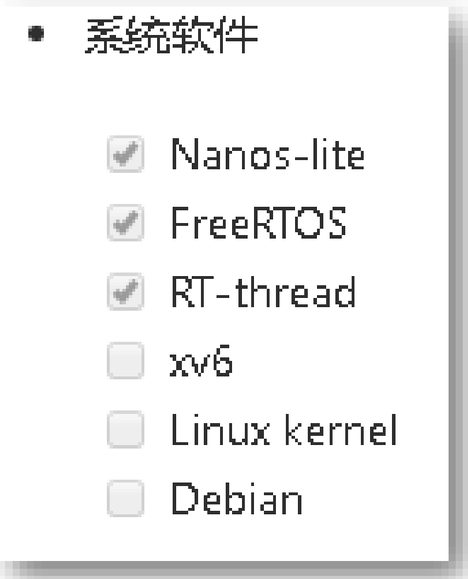
每周进展 & 任务

本周任务

- 华强
  - 补充RV64M的边界行为(mulhsu, 除0, 溢出), 跑riscv-tortrue
  - 添加CLINT(mtime和mtimecmp)和时钟中断, 分时跑仙剑和hello(hello是内核线程)
  - 添加RVC, 跑FreeRTOS和RT-thread
- 紫飞
  - 解决冲刷问题, 跑通ptwtest
  - 把页表放在trm.c, 跑通所有AM测试和应用
  - 在dcache前加入PTW, 跑通所有AM测试和应用
  - 加TLB(尽量做)
  - 完成下面的"理解项目框架"
- 林隽
  - 解决卡死问题, 下周展示性能提升
  - 完成下面的"理解项目框架"
- 金越 - 解决X信号

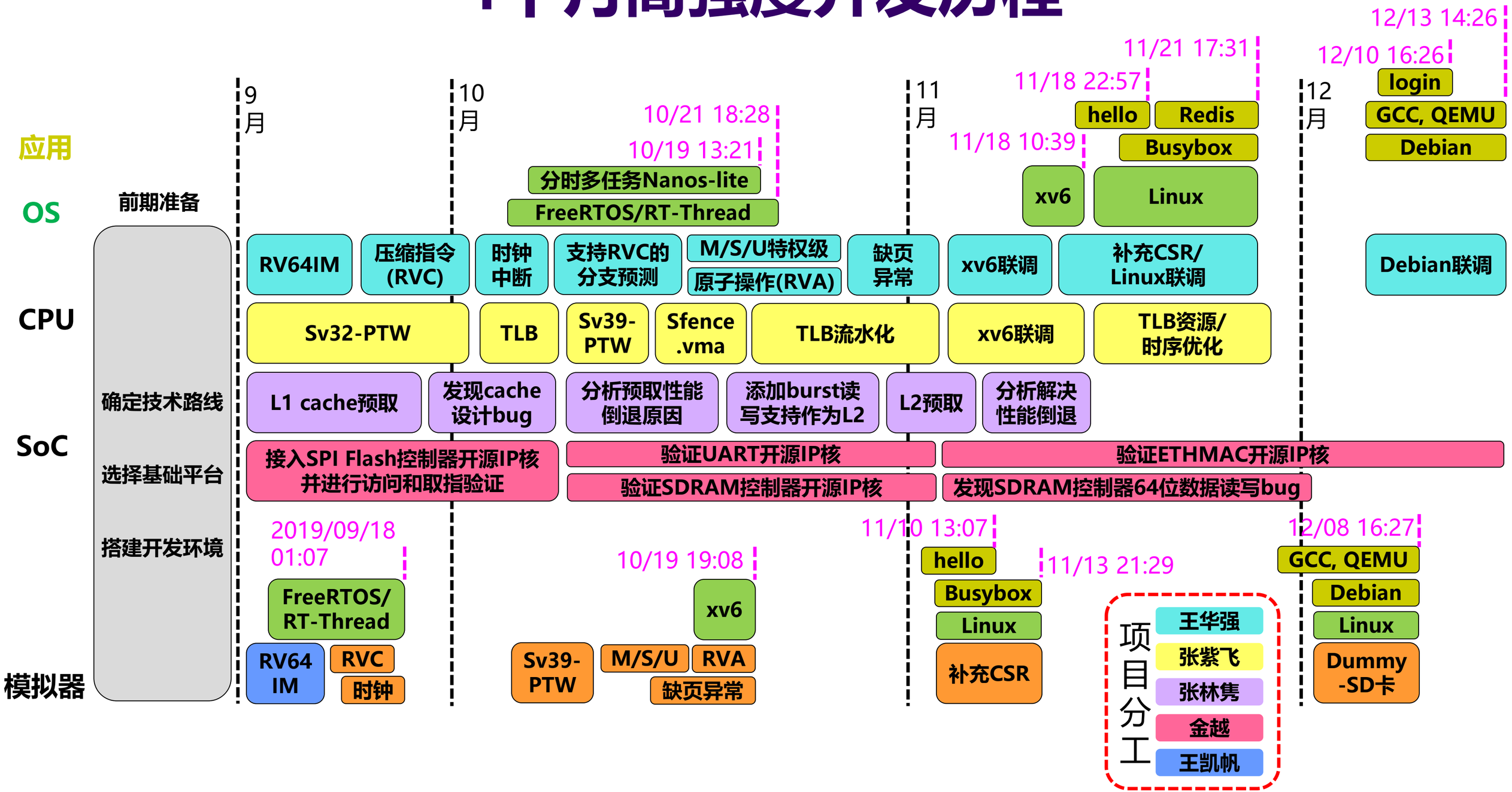
- core
  - RV64IM
  - fence.i
    - icache和dcache的一致性
    - icache冲刷
    - 流水线冲刷
  - M-mode CSR
    - 异常处理相关(mcause, mepc, mstatus)
    - 其它
  - 中断
  - S-mode 分页
  - A扩展
  - RVC(起Debian必须)

测试CPU功能



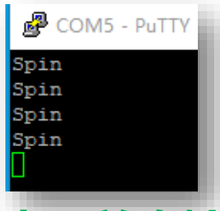
- 系统软件
  - Nanos-lite
  - FreeRTOS
  - RT-thread
  - xv6
  - Linux kernel
  - Debian

4个月高强度开发历程

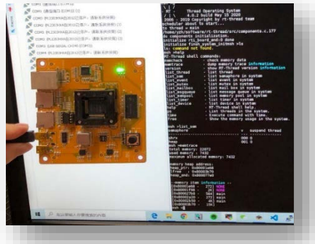

9月
10月
11月
12月
应用
OS
CPU
SoC
模拟器
前期准备
确定技术路线
选择基础平台
搭建开发环境
10/21 18:28
10/19 13:21
分时多任务Nanos-lite
FreeRTOS/RT-Thread
11/21 17:31
11/18 22:57
hello
Redis
11/18 10:39
Busybox
xv6
Linux
12/13 14:26
12/10 16:26
login
GCC, QEMU
Debian
RV64IM
压缩指令(RVC)
时钟中断
支持RVC的分支预测
M/S/U特权级
原子操作(RVA)
缺页异常
xv6联调
补充CSR/Linux联调
Debian联调
Sv32-PTW
TLB
Sv39-PTW
Sfence.vma
TLB流水化
xv6联调
TLB资源/时序优化
L1 cache预取
发现cache设计bug
分析预取性能倒退原因
添加burst读写支持作为L2
L2预取
分析解决性能倒退
接入SPI Flash控制器开源IP核并进行访问和取指验证
验证UART开源IP核
验证SDRAM控制器开源IP核
验证ETHMAC开源IP核
发现SDRAM控制器64位数据读写bug
2019/09/18 01:07
FreeRTOS/RT-Thread
RV64IM
RVC
时钟
10/19 19:08
xv6
Sv39-PTW
M/S/U
RVA
缺页异常
11/10 13:07
hello
11/13 21:29
Busybox
Linux
补充CSR
12/08 16:27
GCC, QEMU
Debian
Linux
Dummy-SD卡
项目分工
王华强
张紫飞
张林隽
金越
王凯帆

1个半月的芯片测试验证

封装后的芯片



串口首次输出



成功运行
RT-Thread

答辩演示

本科
毕业
答辩

6/2 13:40

5
月

5/15 22:48
5/15 10:00
串口测试
RT-Thread

2020/4/25 12:32
4/23 14:56
封装
完成

修复电源问题
生产第二版测试板

芯片装入测试板,
开始串口测试

SDRAM
测试

200MHz
测试

5/19 16:43
更换正确型号的
SDRAM颗粒

SDRAM
测试

5/28 19:54
5/27 21:00
hello
Busybox
xv6
修改版
Linux

6
月

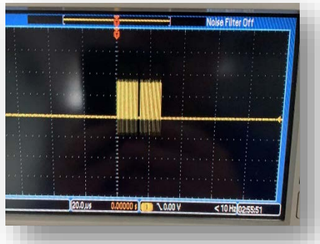
6/7 22:54
Busybox
原版
Linux

200MHz
测试

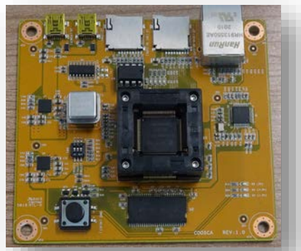
SD卡
测试

由于SD控制器的缺陷，无法在芯片上启动Debian

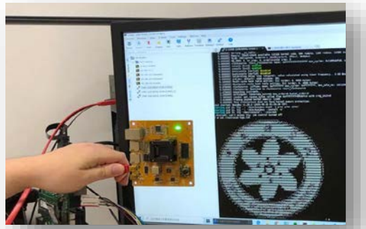
首次取指成功



第二版测试板



成功运行Linux
并展示中科院logo



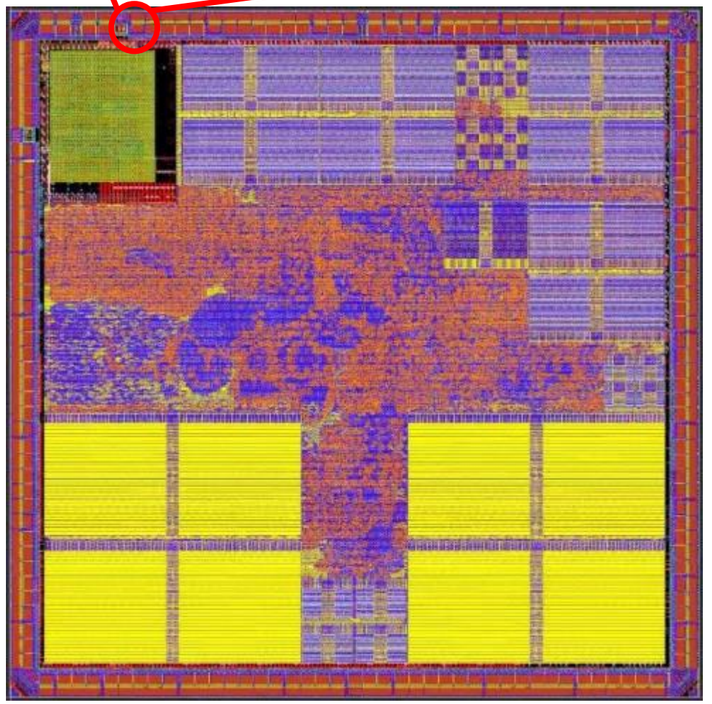
分工
YZH
CY
LT

学生因疫情
无法参与芯
片测试流程

开源SoC芯片——果壳（NutShell）

- 单发射**9级流水顺序核**
- RV64IMAC指令集，支持M/S/U特权级
- 两位饱和计数器分支预测，512项BTB，16项RAS
- 支持Sv39分页机制, 支持硬件TLB填充
- 32K指令/数据L1 cache
- 支持L1 指令/数据cache读一致性
- 128K L2 cache，支持next line预取
- **使用Chisel语言开发**
- 支持SDRAM 、SPI flash、UART等外设
- 支持启动**Linux 4.18.0内核**
- 支持运行Busybox套件
- 仿真/FPGA环境下可启动Debian 11/Fedora 32

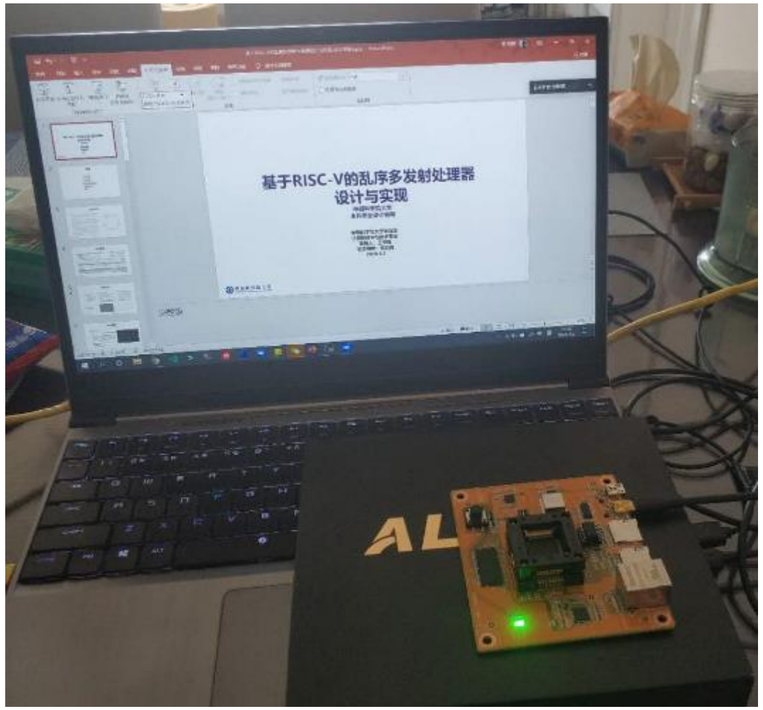
COOSCA (内部代号)



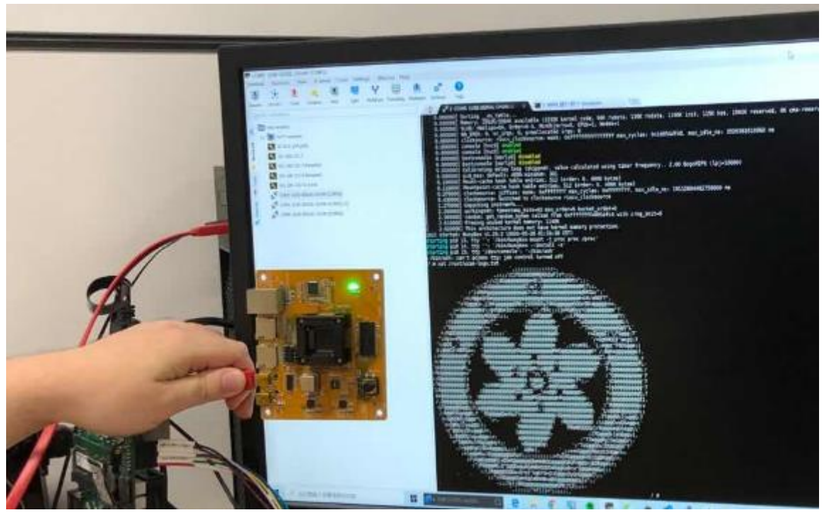
- SMIC 110nm工艺
- 10mm^2
- **200mw@350MHz** Typical
- TQFP100封装

果壳芯片展示

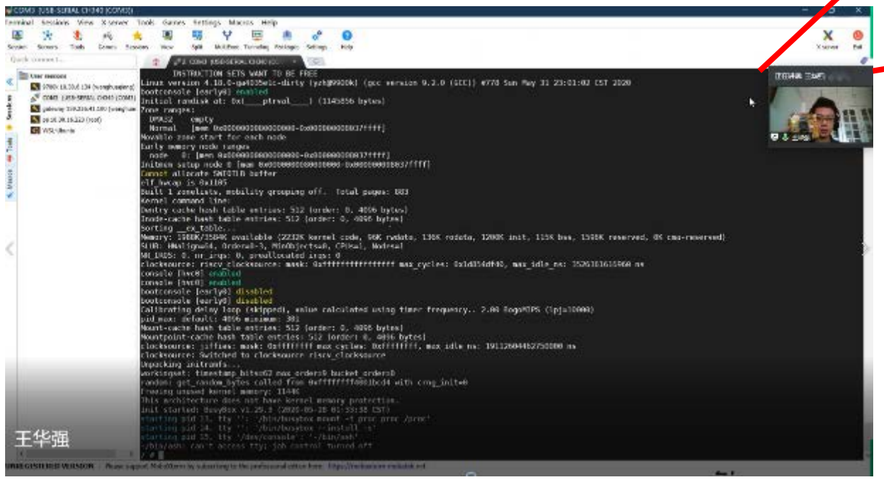
完成封装的芯片



学生的本科毕业设计题目
以及正在运行的芯片



在芯片上运行Linux系统
并展示中科院logo



学生在本科毕业远程
答辩中，代表一生一
芯团队向评委展示芯
片运行的过程

果壳性能展示

芯片工作频率达到350MHz

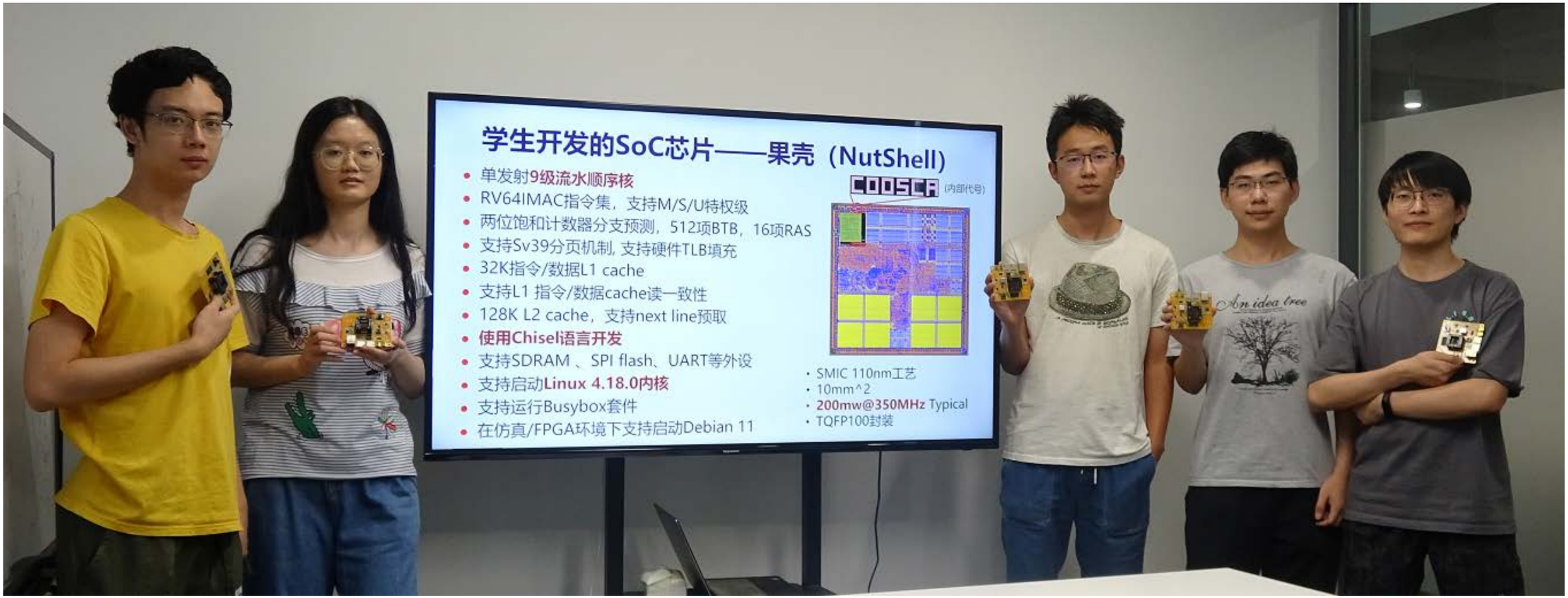
配置开关	倍率	50MHz晶振	100MHz晶振
000	1	50MHz	100MHz
001	1.5	75MHz	150MHz
010	2	100MHz	200MHz
011	2.5	125MHz	250MHz
100	2.75	137.5MHz	275MHz
101	3	150MHz	300MHz
110	3.5	175MHz	350MHz
111	4	200MHz	400MHz

业界标准测试CoreMark跑分为1.49/MHz

```
                    vs. 100000 Marks (i7-7700K @ 4.20GHz)
Exit with code = 0
freq = 200MHz
Hello world! Build time: May 29 2020 11:57:58
Filling RAS with legal address...
sp = 0x80300000
Hello world! Build time: May 29 2020 17:04:04
zeroing memory region [0x800056a0, 0x80005eb0)
jump to 0x80000000...
@@@
@@@
@@@
Running CoreMark for 1000 iterations
2K performance run parameters for coremark.
CoreMark Size    : 666
Total time (ms)  : 3344
Iterations       : 1000
Compiler version : GCC8.3.0
seedcrc          : 0xe9f5
[0]crclist       : 0xe714
[0]crcmatrix     : 0x1fd7
[0]crcstate      : 0x8e3a
[0]crcfinal      : 0xd340
Finised in 3344 ms.
CoreMark 1.0 : (29904 / 100) / GCC8.3.0
=================================================
CoreMark PASS       873 Marks
                vs. 100000 Marks (i7-7700K @ 4.20GHz)
Exit with code = 0
```

首期“一生一芯”计划取得成功

- 五位同学实现“**带着自己设计的处理器芯片毕业**”这一目标



“一生一芯”纪念文化衫

学生感悟

依赖指导 => 主动探索

与之前实验最大的不同……就是**没有先行者一步一步的详细指导**，而是要**自己寻找方法**，**独立实现**，然后进行验证甚至推倒重来。

使用者 => 创造者

胡伟武老师曾经说过，我们计算机系的同学应该学会怎么造计算机而不是怎么用计算机。我以前对这句话并不太有感触，相反曾经质疑国科大计算机系的课程设置这么多硬件的内容是否合理。但**真正参与到项目中才发现在大学里所学的知识和技能是真的有用**。

大部分知识在体系结构课程中...**工作原理也很简单**，只有短短的几行，但是**真正在代码中实现却比自己所想象的要困难得多**。

更自信、更有耐心

和4个月之前的自己相比……**最重要的就是这种观念上的转变**。遇到bug不再在一个地方上死磕，而是从心理上告诉自己bug都是人写出来的，**只要有耐心，只要挖得足够深就一定能找到问题所在**。

成就感

真正参与到项目中才知道课程作业就像直接给人采摘的果园一样，但项目却是**给一片荒地和几颗果树苗，从开垦种植和施肥都要自己动手**，并且还不知道这样能不能结出果实。不知为何，总觉得**从0开始种出的果实要更甜一些**。

提升专业知识，锤炼心理素质

第二期“一生一芯”计划正式启动

学员列表

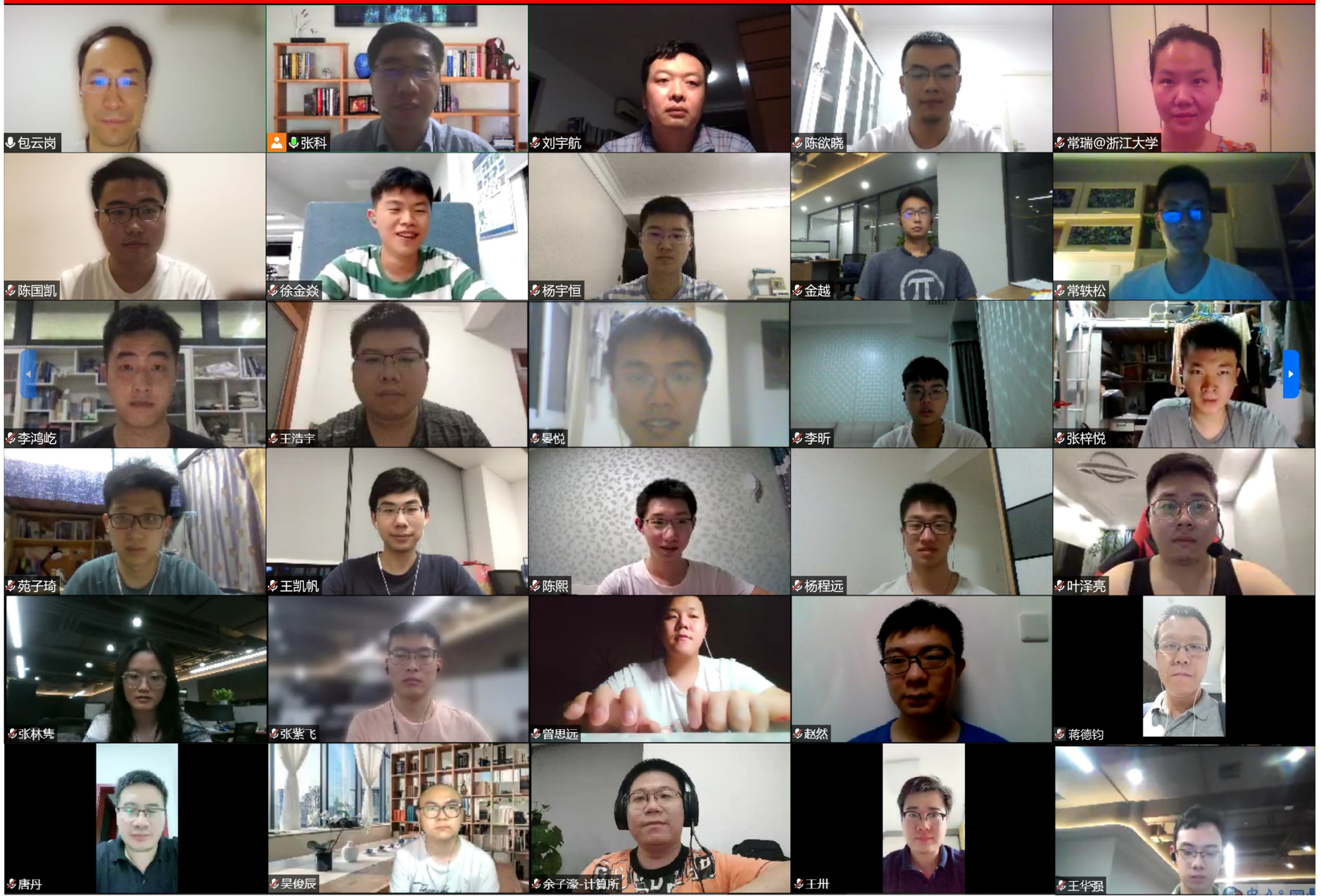
陈国凯	中国科学院大学
陈熙	中国科学院大学
李鸿屹	浙江大学
李昕	西北工业大学
徐金焱	浙江大学
晏悦	中国科学院大学
杨程远	中国科学院大学
杨宇恒	中国科学院大学
叶泽亮	美国密西根州立大学
苑子琦	浙江大学
曾思远	中国科学院大学
张梓悦	南京大学

教学团队

- 班主任
  - **包云岗**
- 辅导员
  - **张科**
- 总顾问
  - **余子濠**
- 导师
  - **常瑞、常轶松、蒋德钧、刘宇航、唐丹、王卅、解壁伟、赵然**
- 助教
  - **陈欲晓、金越、王凯帆、王华强、张林隽、张紫飞**

第二期“一生一芯”计划启动会

2020.8.12



时间节点

- SMIC 110nm MPW流片班车时间点：**2020年12月29日**
- 从现在开始，预计将有**4个多月**高强度的设计、逻辑实现、仿真、硅前验证等工作
- 今年年底投片后，需要等待**约4个月**时间才能回片
- 回片后还需**一些时间**（根据芯片状态，1～4+周均有可能）对芯片进行硅后调试和测试
- 关于毕业设计的两种选择
  - 一是基于“一生一芯”设计的处理器，对其中某个模块进一步优化
  - 二是在12月29日投片结束后在本课题组启动某项新任务，或回到所在学校/课题组开展工作

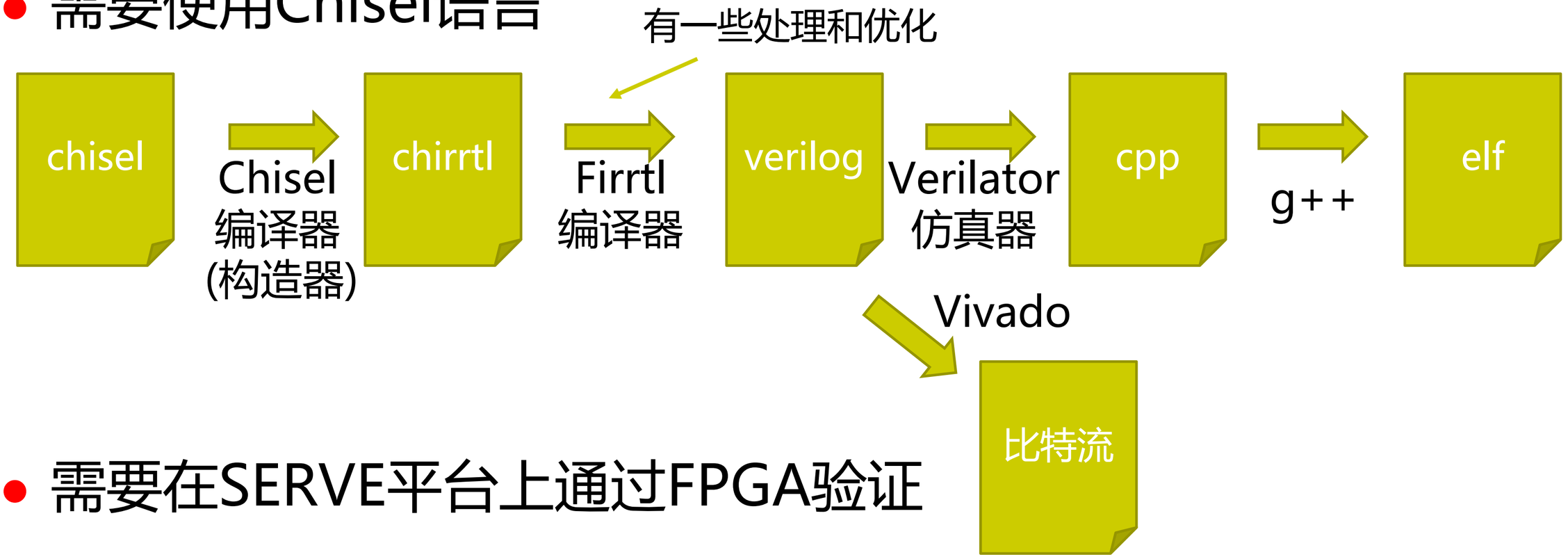
阶段安排

- 第一阶段: 9月13日前
  - 独立用chisel从零开始实现一个可以支持RT-Thread的RV64处理器, 支持串口和时钟中断
  - 正确性优先，有时间的话可以进行性能优化
  - 共同完善指导手册

- 第二阶段: 9月14日~11月31日(需要预留一个月时间进行后端物理设计)
- 各自选题，最后将大家的设计集成起来
- 鼓励大家不组队，独立完成
- 可选题目
  - 操作系统方向：添加TLB和CSR, 启动xv6, Linux, 或者是自己实现的操作系统
  - 流水线方向：性能优化, 包括分支预测, 多发射, 乱序执行等
  - 存储层次结构方向：添加cache, 尝试实现前沿文章的预取算法或替换算法等
  - SoC方向：实现多设计集成技术(MDD)和chiplink技术, 接入各种外设
  - 前沿探索方向：实现riscv中一些还没有冻结的规范特性, 如hypervisor扩展, 用户中断扩展

开发环境

- 需要使用Chisel语言



- 需要在SERVE平台上通过FPGA验证
- 使用git管理自己的代码, 上传到github上接受监督

一些要求

- 运行RT-Thread只需实现RV64IM + M-mode
  - 编译时可以去掉C(压缩指令)扩展
- 运行Linux需要支持RV64IMA和M/S/U
- 运行Debian/Fedora需要支持RV64IMAC和M/S/U
  - RV64FD可以由BBL进行软件模拟(已支持)
  - mips32起Debian要么改内核, 要么实现硬件FPU

- 为了连接外设(如串口), 至少需要支持AXI4-Lite总线协议
  - 如果想提高性能, 访存接口还需要支持AXI4协议
  - SoC内部可以支持采用其它协议(包括自定义)

一些资料

- Chisel
  - Chisel Bootcamp - 很不错的chisel教程, 支持在线运行chisel代码, 可以边写chisel代码边学习
    - https://github.com/freechipsproject/chisel-bootcamp
    - 第1章是scala入门, 第2章是chisel基础, 第3章是scala高级特性和chisel的混合使用, 第4章是FIRRTL后端相关内容
    - 学习前两章 = 入门, 学习第3章 = 提高, 第4章可以作为课外阅读材料
  - Chisel Users Guide - 比较系统地整理了chisel的特性, 也是不错的入门教程
    - https://github.com/freechipsproject/chisel3/wiki/Short-Users-Guide-to-Chisel
  - Chisel小抄 - 简明地列出了chisel语言的大部分用法
    - https://chisel.eecs.berkeley.edu/doc/chisel-cheatsheet3.pdf
  - Chisel API - 详细地列出了chisel库的所有API供参考
    - https://chisel.eecs.berkeley.edu/api/latest/index.html

一些资料

- Chisel示例项目
  - https://github.com/freechipsproject/chisel-template
- RISC-V指令集手册
  - https://github.com/riscv/riscv-isa-manual
- RISC-V各种资料(如ABI规范等)
  - https://github.com/riscv/
- 用Chisel开发的RISC-V处理器示例项目
  - https://github.com/ucb-bar/riscv-sodor
  - https://github.com/OSCPU/NutShell
  - https://github.com/freechipsproject/rocket-chip
    - 不适合入门阅读

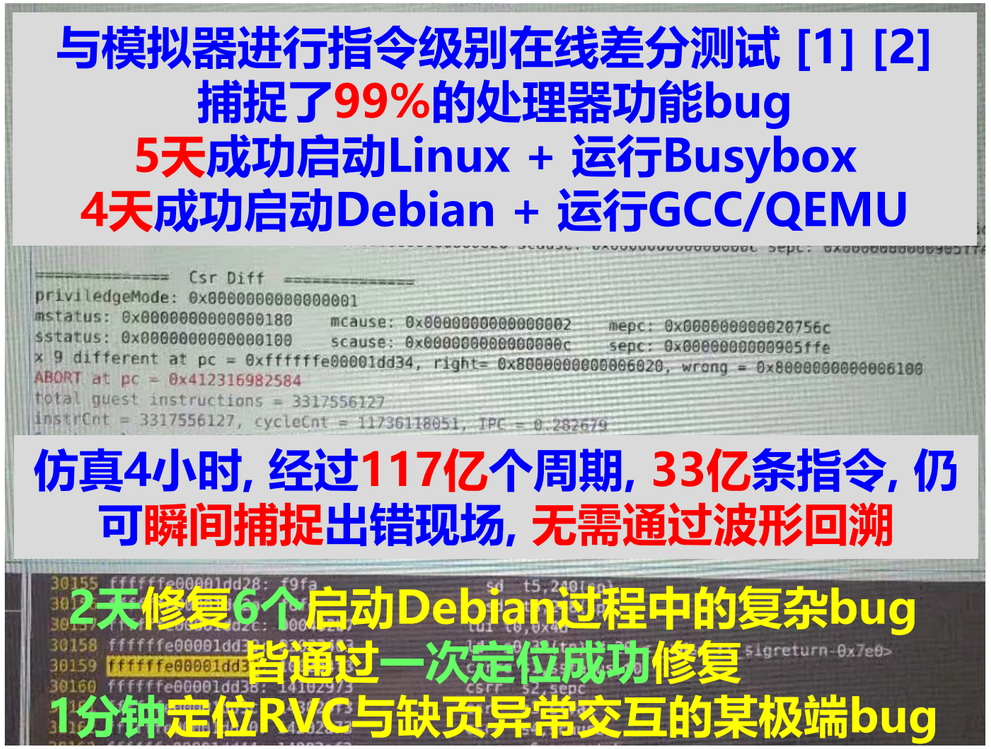
辅助工具和项目

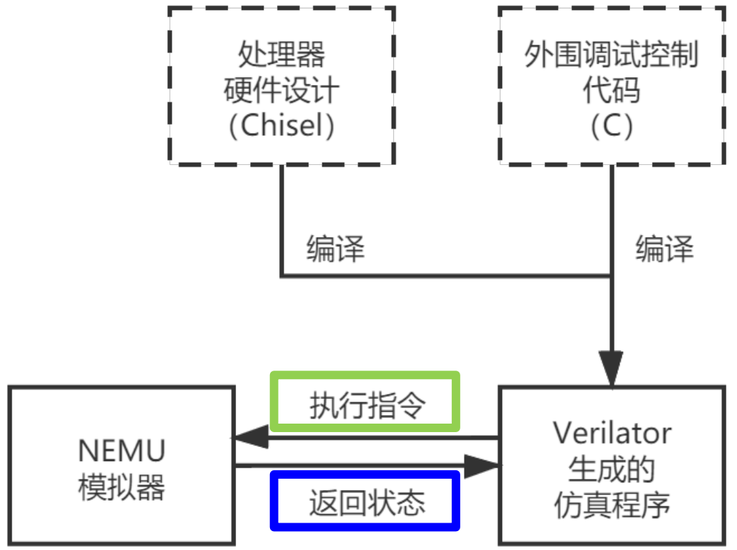
- 开源仿真器Verilator和riscv工具链可以通过apt-get一键安装

- 模拟器NEMU
  - https://github.com/NJU-ProjectN/nemu
- 裸机运行时环境AM
  - https://github.com/NJU-ProjectN/nexus-am
- 我们有这两个项目的参考代码, 但希望大家从框架开始锻炼系统能力
  - NEMU框架代码导读
    - https://nju-projectn.github.io/ics-pa-gitbook/ics2019/1.3.html
    - https://nju-projectn.github.io/ics-pa-gitbook/ics2019/2.2.html
  - AM框架代码导读
    - https://nju-projectn.github.io/ics-pa-gitbook/ics2019/2.3.html
  - 框架没有实现RV64的内容, 需要大家理解后自行添加

差分测试

借鉴软件工程领域的差分测试技术进行处理器的仿真验证





测试框架结构示意

```
while (1) {
    verilator_step();
    nemu_step(1);
    verilator_getregs(&r1);
    nemu_getregs(&r2);
    if (r1 != r2) { abort(); }
}
```

在线比对机制

[1] Yu, EasyDiff: An Effective and Efficient Framework for Processor Verification, CRVF 2019, https://crvf2019.github.io/pdf/14.pdf
[2] 2018年龙芯杯南京大学二队决赛答辩 报告 http://www.nscscc.org/a/wangjie/NSCSCC2018/2018/1010/46.html

遇到问题

- 我们的安排比较有挑战, 坚持独立完成, 能力会大幅提升

- 同学之间可以相互讨论, 但不要直接借鉴代码
- 可以咨询第一批一生一芯的学长学姐
- 以及助教老师