

Website: ysyx.org
Email: ysyx@bosc.ac.cn



WeChat
Account

“One Student One Chip” Initiative

Let Students Design Their Own Open Source Processor Chips

Yuyang Miao(缪宇飚)

2024.04

Institute of Computing Technology, Chinese Academy of Sciences



中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

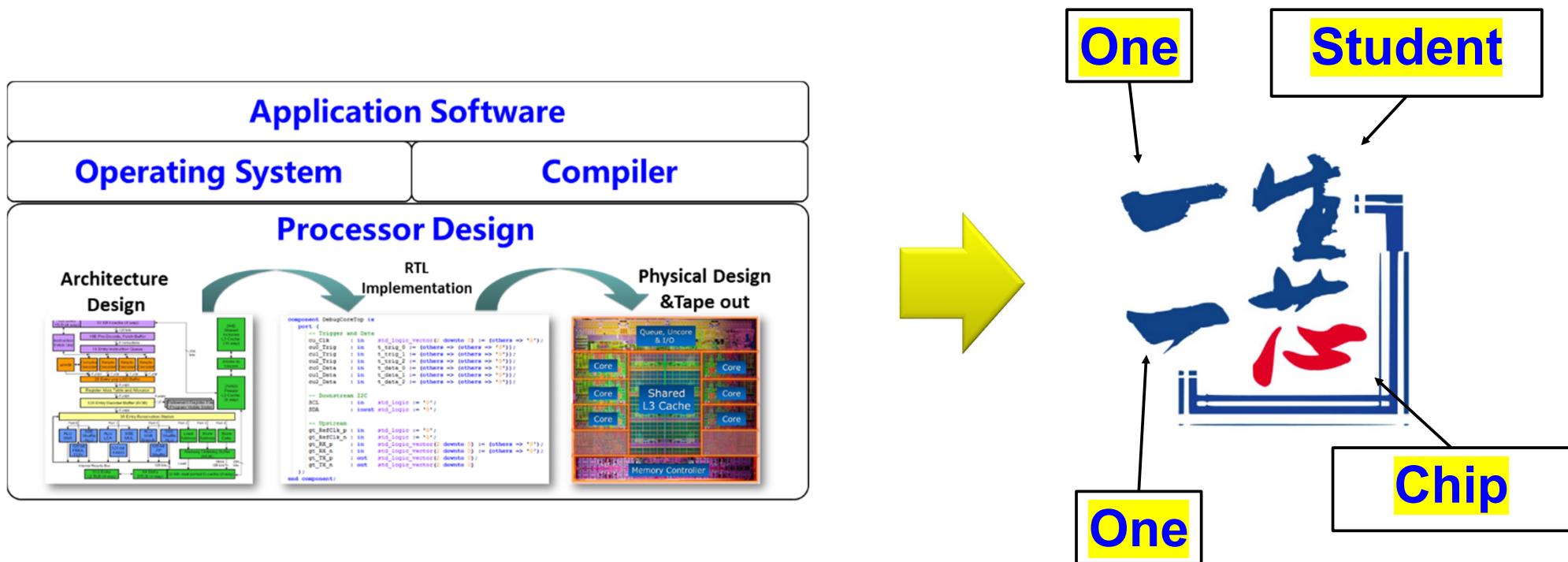


中国科学院大学
University of Chinese Academy of Sciences

1. Background

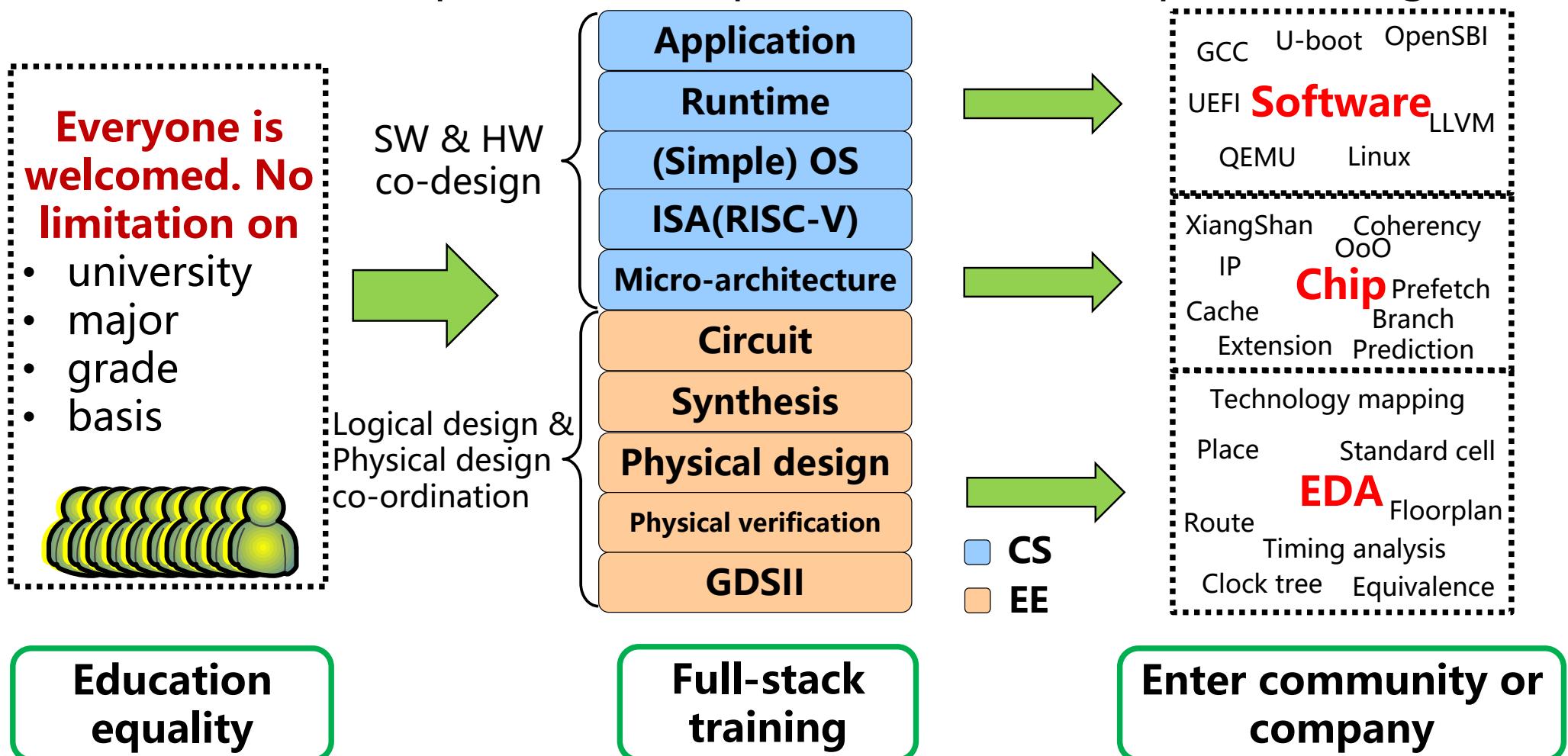
The One Student One Chip (OSOC) Initiative

- Learning-by-Doing: Teach students to build real chips
- Launched the OSOC Initiative in 2019



OSOC Initiative

Based on open-sourced, practice-oriented, open learning



> 6000 students participated in the OSOC Initiative

No.	Start Date	End Date	# of Applicants	# of Schools	#Stu. Learning	#Stu. Tapeout
1 st	Aug, 2019	-	5	1	5	5
2 nd	Aug, 2020	-	11	5	11	11
3 rd	Jul, 2021	Sep, 2021	760	168	215	51
4 th	Feb, 2022	Aug, 2022	1753	328	215	16
5 th	Aug, 2022	Jul, 2023	1881	379	155	13
6 th	July, 2023	In progress	2208	383	176	-

Updated: Jan 13, 2024

**2. 1st OSOC – Let students graduate with their own chips
(Aug to Dec, 2019)**

The 1st OSOC (2019)

- Five senior undergraduates participated
- Completed the design of a **64-bit RISC-V** processor in four months
- The chip was **taped out** with 110nm and ran Linux and a self-built UCAS-core OS



Yue Jin



Huangqiang Wang



Kaifan Wang

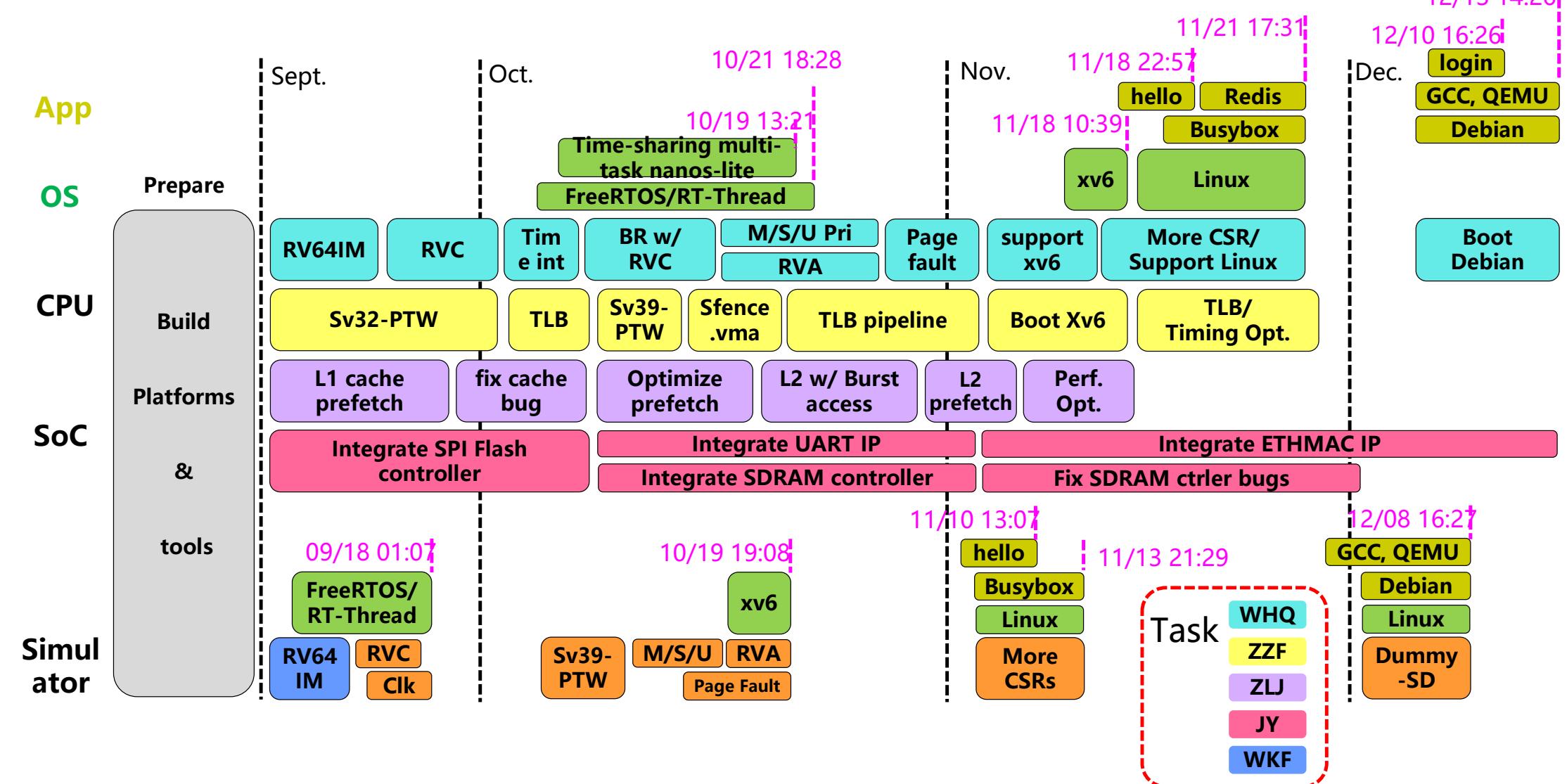


Linjuan Zhang



Zifei Zhang

Development Timeline

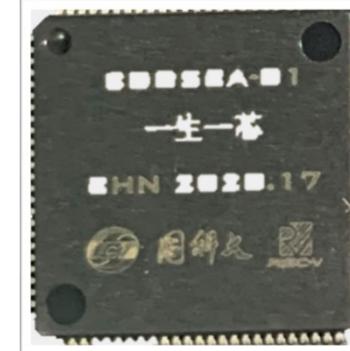
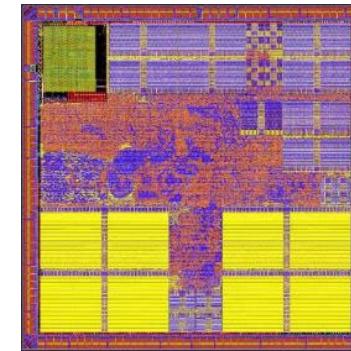


NutShell: A Linux-Compatible RISC-V Processor Designed by Undergraduates

A 64-bit RISC-V Processor

- Single-issue, 9-stage, in-order
- RV64IMAC, support M/S/U
- BPU with PHT, 512-entry BTB, 16-entry RAS
- Sv39, hardware TLB refill
- 32K L1I & L1D
- Read consistency for L1I & L1D
- 128K L2 cache, next line prefetch
- Develop with Chisel
- SDRAM, SPI flash, UART
- Support Linux 4.18.0 kernel
- Support Busybox
- Can boot Debian 11 on Emulator & FPGA

Tape-out w/ 110nm process



- 110nm process
- 10mm²
- 200mw@350MHz Typical
- TQFP100 package

Feedback from students

Self-exploring

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。

From user to creator

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**。

More confidence, more patience

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**。

Fulfillment

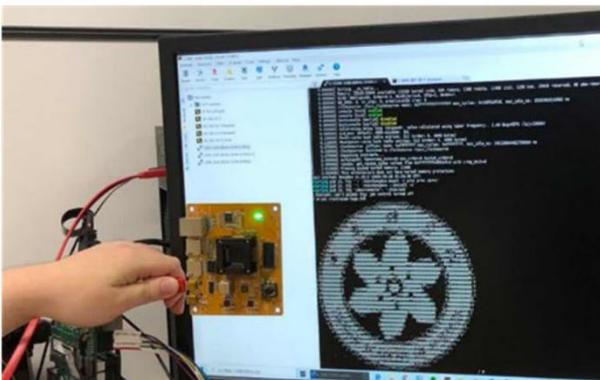
真正参与到项目中才知道课程作业就像直接给人采摘的果园一样，但项目却是**给一片荒地和几颗果树苗，从开垦种植和施肥都要自己动手**，并且还不知道这样能不能结出果实。不知为何，总觉得**从0开始种出的果实要更甜一些**。

Make students stronger!

Chips and Demos



Chips



Run Linux
Display CAS logo

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

Demo



Thesis Defense 2020.6.2

1st OSOC finished successfully

- Students graduated with their own processor chips.



Report Accepted by RISC-V Global Forum 2020

NutShell: A Linux-Compatible RISC-V Processor Designed by Undergraduates

Your submission was accepted for
RISC-V Global Forum 2020

The RISC-V Event Team

★ 详情

Dear Huaqiang,

It is our pleasure to inform you that we
have accepted your submission.

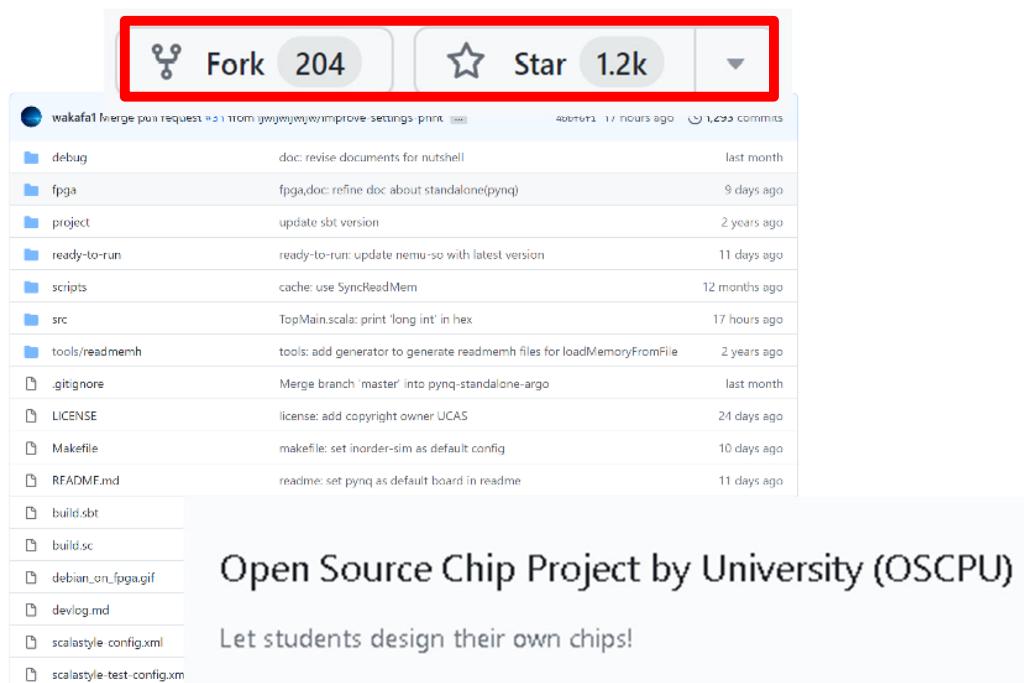
***Nutshell: A Linux-Compatible RISC-V
Processor Designed by Undergraduates,***
as a lightning talk, and would like to
welcome you as a speaker to the [RISC-V
Global Forum 2020](#), happening virtually
Thursday, September 3. Sessions will
overlap during US, Europe, and Asia
working hours. We ask that you please
review the information below and on the
[Speaker Guide](#) and complete all the
required items to confirm your speaking
engagement.

The event will take place on the virtual
event platform. MeetingPlav. The



Open-sourcing

- Open-sourced on **GitHub**
 - #Fork > 200



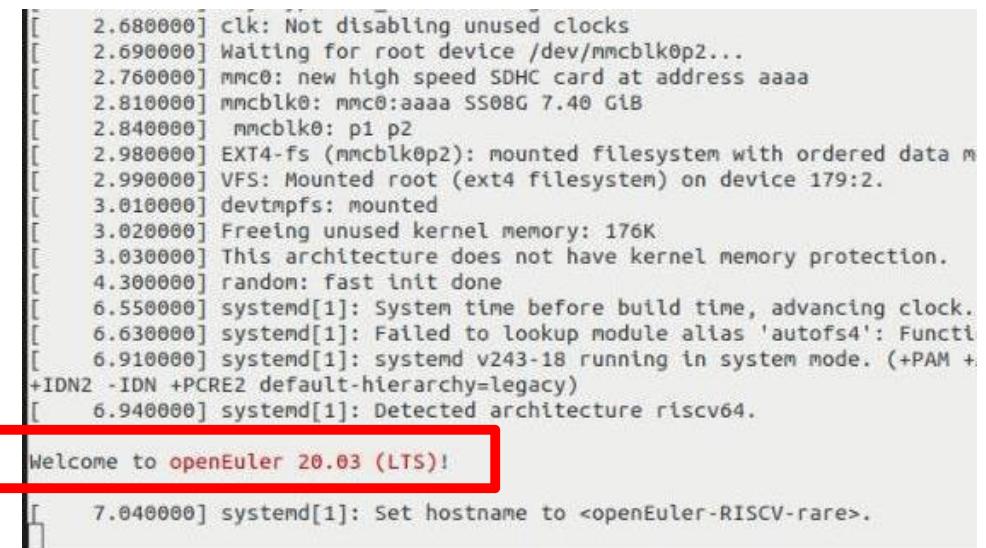
Fork 204 Star 1.2k

wakafat Merge pull request #5 from jwjjwjjw/improve-settings-print 4066761 17 hours ago 1,295 commits

File	Description	Time Ago
debug	doc: revise documents for nutshell	last month
fpga	fpga/doc: refine doc about standalone(pynq)	9 days ago
project	update sbt version	2 years ago
ready-to-run	ready-to-run: update nemu-so with latest version	11 days ago
scripts	cacher: use SyncReedMem	12 months ago
src	TopMain.scala: print 'long int' in hex	17 hours ago
tools/readmemh	tools: add generator to generate readmemh files for loadMemoryFromFile	2 years ago
.gitignore	Merge branch 'master' into pynq-standalone-argo	last month
LICENSE	license: add copyright owner UCAS	24 days ago
Makefile	makefile: set inorder-sim as default config	10 days ago
RFADME.md	readme: set pynq as default board in readme	11 days ago
build.sbt		
build.sc		
debian_on_fpga.gif		
devlog.md		
scalastyle config.xml		
scalastyle-test-config.xml		

Open Source Chip Project by University (OSCPU)
Let students design their own chips!

- **OpenEuler, an OS developed by Huawei, is booted successfully on NutShell**



```
[ 2.680000] clk: Not disabling unused clocks
[ 2.690000] Waiting for root device /dev/mmcblk0p2...
[ 2.760000] mmc0: new high speed SDHC card at address aaaa
[ 2.810000] mmcblk0: mmc0:aaaa SS08G 7.40 G18
[ 2.840000] mmcblk0: p1 p2
[ 2.980000] EXT4-fs (mmcblk0p2): mounted filesystem with ordered data mode
[ 2.990000] VFS: Mounted root (ext4 filesystem) on device 179:2.
[ 3.010000] devtmpfs: mounted
[ 3.020000] Freeing unused kernel memory: 176K
[ 3.030000] This architecture does not have kernel memory protection.
[ 4.300000] random: fast init done
[ 6.550000] systemd[1]: System time before build time, advancing clock.
[ 6.630000] systemd[1]: Failed to lookup module alias 'autofs4': Function not found
[ 6.910000] systemd[1]: systemd v243-18 running in system mode. (+PAM +IDN +IDN +PCRE2 default-hierarchy=legacy)
[ 6.940000] systemd[1]: Detected architecture riscv64.

Welcome to openEuler 20.03 (LTS)!

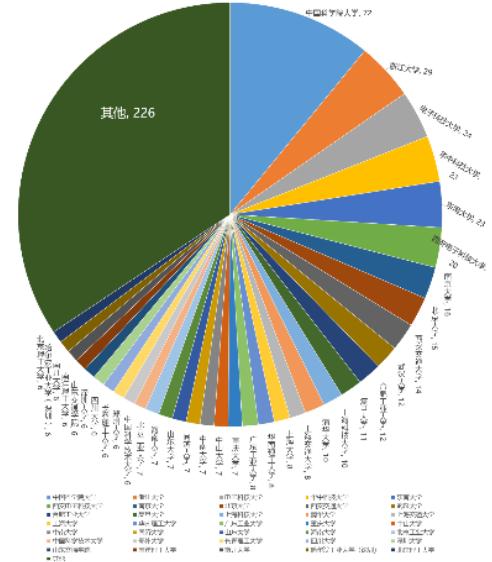
[ 7.040000] systemd[1]: Set hostname to <openEuler-RISCV-rare>.
```

<https://github.com/OSCPU/NutShell>

3. 3rd OSOC – Exploring large-scale learning (Jul to Dec, 2021)

3rd One Student One Chip

- Bring chip talent cultivation and open-source chip community together, while paying special attention to whole-flow chip design.
 - **760 students from 168 colleges (30 oversea)**
 - Undergraduate students: 50%
 - Students-in-school: 82%



中国科学院大学



ByteDance
字节跳动



中国科学院计算技术研究所 INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES



 上海处理器技术创新中心
SHANGHAI INNOVATION CENTER FOR COMPUTING TECHNOLOGY



中国开放指令生态(RISC-V)联盟
China RISC-V Alliance

Breakdown of colleges

■ Total : 760 students from 168 colleges (including 30 oversea)

■ Students in school: 625 (82%)

■ China

- 72 : University of Chinese Academy of Sciences
- 29 : Zhejiang University
- 24 : University of Electronic Science and Technology of China
- 23 : Huazhong University of Science and Technology
- . . .

■ Oversea

- Many students are from the USA and Europe
 - Most of them are Chinese students
 - We are translating the Chinese materials into English version.

Georgia Institute of Technology
University of Toronto
The University of Edinburgh
KTH Royal Institute of Technology
Nanyang Technological University
The University of Melbourne
University of Michigan
Monash University
University College London
Clemson University
University of California, Los Angeles
University of California, Riverside
University of California, Davis
University of Waterloo
University of Virginia
Technische Universiteit Delft
Dartmouth College
Dalhousie University
Paris-Saclay University
Royal Melbourne Institute of Technology

Teaching Assistants

- Selected from students

- Inspire others to think, instead of giving the answer
- Leave the training to students

List of TA (Jun, 2023)

	Name	University	Grade
1	苗金标	中国科学技术大学	Grade 2 master
2	段震伟	中国科学技术大学	Grade 3 master
3	刘汉章	太原理工大学	Junior
4	曹勋	中国科学技术大学	Grade 2 master
5	杨海帆	浙江工商大学	Senior
6	曹世洋	中国科学技术大学	Grade 2 master
7	倪仁涛	东北大学	Grade 1 master
8	魏人	兰州大学	Senior
9	吴佳宾	青岛大学	Senior
10	陈璐	中国科学院大学	Ph.d Candidate
11	栗金伦	太原理工大学	Senior



苗金标



段震伟



刘汉章



曹勋



杨海帆



曹世洋



倪仁涛



魏人



陈璐



栗金伦



吴佳宾

Students' learning note and meeting attendance

Learning note of a student

日期	计划任务	总时长	任务完成情况		卡了一段时间的bug
			开始时间	结束时间	
第一周 (5天)	安装Verilator	2月25日	尝试完成MIPS的模拟	5h	基础部分MIPS的SISCU-V差距不大，实现起来的思路也大同小异
	调试	2月26日	完成MIPS的浮点计算	5h	完成浮点寄存器，可以将32位和64位浮点数在寄存器中
	完成实验报告	2月27日	完成MIPS的浮点	5h	貌似MIPS的JISA没有更新，直接rate合作错了。重看了报告，是自己没有看清楚
	完成	3月1日	完成MIPS的浮点	5h	
	完成	3月3日	完成MIPS的浮点	5h	
	准备	3月5日	完成C3R指令	5h	
	完成	3月7日	完成C3R指令	5h	
	修正	3月9日	修正	5h	
	完成	3月10日	引入PA	5h	
	完成	3月11日	完成PA的内存	5h	
第二周 (7天)	进一步	2月12日	完成PA的内存	5h	
	完成	2月14日	使用SMP和PCI作为PA的外设	5h	
	修正	2月16日	修正上电时序	5h	
	完成	2月18日	修正上电时序	5h	
	完成	2月20日	完成PA的内存	5h	
	完成	2月22日	测试中断	5h	
	完成	2月24日	完成PA的内存	5h	
	完成	2月26日	完成PA的内存	5h	
	完成	2月28日	完成PA的内存	5h	
	完成	3月1日	完成PA的内存	5h	
第三周 (7天)	完成	3月3日	完成PA的内存	5h	
	完成	3月5日	完成PA的内存	5h	
	完成	3月7日	完成PA的内存	5h	
	完成	3月9日	完成PA的内存	5h	
	完成	3月11日	完成PA的内存	5h	
	完成	3月13日	完成PA的内存	5h	
	完成	3月15日	完成PA的内存	5h	
	完成	3月17日	完成PA的内存	5h	
	完成	3月19日	完成PA的内存	5h	
	完成	3月21日	完成PA的内存	5h	
第四周 (7天)	完成	3月23日	完成PA的内存	5h	
	完成	3月25日	完成PA的内存	5h	
	完成	3月27日	完成PA的内存	5h	
	完成	3月29日	完成PA的内存	5h	
	完成	3月31日	完成PA的内存	5h	
	完成	4月2日	完成PA的内存	5h	
	完成	4月4日	完成PA的内存	5h	
	完成	4月6日	完成PA的内存	5h	
	完成	4月8日	完成PA的内存	5h	
	完成	4月10日	完成PA的内存	5h	
第五周 (6天)	完成	4月12日	布线	5h	
	完成	4月14日	布线	5h	
	完成	4月16日	配置Verdi环境 & 学习	5h	
	完成	4月18日	修正布线	5h	
	完成	4月20日	配置Vivado环境 & 学习	5h	
	完成	4月22日	测试中断	5h	
	完成	4月24日	完成取指&译码	5h	
	完成	4月26日	完成译码	5h	
	完成	4月28日	修改译码	5h	
	完成	4月30日	修改译码	5h	
第六周 (6天)	完成	5月1日	学习LoongArch <1>	5h	
	完成	5月3日	学习LoongArch <1>	5h	
	完成	5月5日	学习LoongArch <1>	5h	
	完成	5月7日	学习LoongArch <1>	5h	
	完成	5月9日	学习LoongArch <1>	5h	
	完成	5月11日	学习LoongArch <1>	5h	
	完成	5月13日	学习LoongArch <1>	5h	
	完成	5月15日	学习LoongArch <1>	5h	
	完成	5月17日	学习LoongArch <1>	5h	
	完成	5月19日	学习LoongArch <1>	5h	
第七周 (7天)	完成	5月21日	学习LoongArch <1>	5h	
	完成	5月23日	学习LoongArch <1>	5h	
	完成	5月25日	学习LoongArch <1>	5h	
	完成	5月27日	学习LoongArch <1>	5h	
	完成	5月29日	学习LoongArch <1>	5h	
	完成	5月31日	学习LoongArch <1>	5h	
	完成	6月2日	学习LoongArch <1>	5h	
	完成	6月4日	学习LoongArch <1>	5h	
	完成	6月6日	学习LoongArch <1>	5h	
	完成	6月8日	学习LoongArch <1>	5h	
第八周 (6天)	完成	6月10日	学习LoongArch <1>	5h	
	完成	6月12日	学习LoongArch <1>	5h	
	完成	6月14日	学习LoongArch <1>	5h	
	完成	6月16日	学习LoongArch <1>	5h	
	完成	6月18日	学习LoongArch <1>	5h	
	完成	6月20日	学习LoongArch <1>	5h	
	完成	6月22日	学习LoongArch <1>	5h	
	完成	6月24日	学习LoongArch <1>	5h	
	完成	6月26日	学习LoongArch <1>	5h	
	完成	6月28日	学习LoongArch <1>	5h	
第九周 (4天)	完成	6月30日	学习LoongArch <1>	5h	
	完成	7月2日	学习LoongArch <1>	5h	
	完成	7月4日	学习LoongArch <1>	5h	
	完成	7月6日	学习LoongArch <1>	5h	
	完成	7月8日	学习LoongArch <1>	5h	
	完成	7月10日	学习LoongArch <1>	5h	
	完成	7月12日	学习LoongArch <1>	5h	
	完成	7月14日	学习LoongArch <1>	5h	
	完成	7月16日	学习LoongArch <1>	5h	
	完成	7月18日	学习LoongArch <1>	5h	
第十周 (7天)	完成	7月20日	学习LoongArch <1>	5h	
	完成	7月22日	学习LoongArch <1>	5h	
	完成	7月24日	学习LoongArch <1>	5h	
	完成	7月26日	学习LoongArch <1>	5h	
	完成	7月28日	学习LoongArch <1>	5h	
	完成	7月30日	学习LoongArch <1>	5h	
	完成	7月32日	学习LoongArch <1>	5h	
	完成	7月1日	学习LoongArch <1>	5h	
	完成	7月3日	学习LoongArch <1>	5h	
	完成	7月5日	学习LoongArch <1>	5h	
第十一周 (7天)	完成	7月7日	学习LoongArch <1>	5h	
	完成	7月9日	学习LoongArch <1>	5h	
	完成	7月11日	学习LoongArch <1>	5h	
	完成	7月13日	学习LoongArch <1>	5h	
	完成	7月15日	学习LoongArch <1>	5h	
	完成	7月17日	学习LoongArch <1>	5h	
	完成	7月19日	学习LoongArch <1>	5h	
	完成	7月21日	学习LoongArch <1>	5h	
	完成	7月23日	学习LoongArch <1>	5h	
	完成	7月25日	学习LoongArch <1>	5h	
第十二周 (7天)	完成	7月27日	学习LoongArch <1>	5h	
	完成	7月29日	学习LoongArch <1>	5h	
	完成	7月31日	学习LoongArch <1>	5h	
	完成	8月2日	学习LoongArch <1>	5h	
	完成	8月4日	学习LoongArch <1>	5h	
	完成	8月6日	学习LoongArch <1>	5h	
	完成	8月8日	学习LoongArch <1>	5h	
	完成	8月10日	学习LoongArch <1>	5h	
	完成	8月12日	学习LoongArch <1>	5h	
	完成	8月14日	学习LoongArch <1>	5h	
第十三周 (4天)	完成	8月16日	学习LoongArch <1>	5h	
	完成	8月18日	学习LoongArch <1>	5h	
	完成	8月20日	学习LoongArch <1>	5h	
	完成	8月22日	学习LoongArch <1>	5h	
	完成	8月24日	学习LoongArch <1>	5h	
	完成	8月26日	学习LoongArch <1>	5h	
	完成	8月28日	学习LoongArch <1>	5h	
	完成	8月30日	学习LoongArch <1>	5h	
	完成	8月1日	学习LoongArch <1>	5h	
	完成	8月3日	学习LoongArch <1>	5h	
第十四周 (7天)	完成	8月5日	学习LoongArch <1>	5h	
	完成	8月7日	学习LoongArch <1>	5h	
	完成	8月9日	学习LoongArch <1>	5h	
	完成	8月11日	学习LoongArch <1>	5h	
	完成	8月13日	学习LoongArch <1>	5h	
	完成	8月15日	学习LoongArch <1>	5h	
	完成	8月17日	学习LoongArch <1>	5h	
	完成	8月19日	学习LoongArch <1>	5h	
	完成	8月21日	学习LoongArch <1>	5h	
	完成	8月23日	学习LoongArch <1>	5h	
第十五周 (7天)	完成	8月25日	学习LoongArch <1>	5h	
	完成	8月27日	学习LoongArch <1>	5h	
	完成	8月29日	学习LoongArch <1>	5h	
	完成	8月31日	学习LoongArch <1>	5h	
	完成	9月2日	学习LoongArch <1>	5h	
	完成	9月4日	学习LoongArch <1>	5h	
	完成	9月6日	学习LoongArch <1>	5h	
	完成	9月8日	学习LoongArch <1>	5h	
	完成	9月10日	学习LoongArch <1>	5h	
	完成	9月12日	学习LoongArch <1>	5h	
第十六周 (7天)	完成	9月14日	学习LoongArch <1>	5h	
	完成	9月16日	学习LoongArch <1>	5h	
	完成	9月18日	学习LoongArch <1>	5h	
	完成	9月20日	学习LoongArch <1>	5h	
	完成	9月22日	学习LoongArch <1>	5h	
	完成	9月24日	学习LoongArch <1>	5h	
	完成	9月26日	学习LoongArch <1>	5h	
	完成	9月28日	学习LoongArch <1>	5h	
	完成	9月30日	学习LoongArch <1>	5h	
	完成	10月1日	学习LoongArch <1>	5h	
第十七周 (4天)	完成	10月3日	学习LoongArch <1>	5h	
	完成	10月5日	学习LoongArch <1>	5h	
	完成	10月7日	学习LoongArch <1>	5h	
	完成	10月9日	学习LoongArch <1>	5h	
	完成	10月11日	学习LoongArch <1>	5h	
	完成	10月13日	学习LoongArch <1>	5h	
	完成	10月15日	学习LoongArch <1>	5h	
	完成	10月17日	学习LoongArch <1>	5h	
	完成	10月19日	学习LoongArch <1>	5h	
	完成	10月21日	学习LoongArch <1>	5h	
第十八周 (7天)	完成	10月23日	学习LoongArch <1>	5h	
	完成	10月25日	学习LoongArch <1>	5h	
	完成	10月27日	学习LoongArch <1>	5h	
	完成	10月29日	学习LoongArch <1>	5h	
	完成	10月31日	学习LoongArch <1>	5h	
	完成	11月1日	学习LoongArch <1>	5h	
	完成	11月3日	学习LoongArch <1>	5h	
	完成	11月5日	学习LoongArch <1>	5h	
	完成	11月7日	学习LoongArch <1>	5h	
	完成	11月9日	学习LoongArch <1>	5h	
第十九周 (7天)	完成	11月11日	学习LoongArch <1>	5h	
	完成	11月13日	学习LoongArch <1>	5h	
	完成	11月15日	学习LoongArch <1>	5h	
	完成	11月17日	学习LoongArch <1>	5h	
	完成	11月19日	学习LoongArch <1>	5h	
	完成	11月21日	学习LoongArch <1>	5h	
	完成	11月23日	学习LoongArch <1>	5h	
	完成	11月25日	学习LoongArch <1>	5h	
	完成	11月27日	学习LoongArch <1>	5h	
	完成	11月29日	学习LoongArch <1>	5h	
第二十周 (7天)	完成	11月30日	学习LoongArch <1>	5h	
	完成	12月2日	学习LoongArch <1>	5h	
	完成	12月4日	学习LoongArch <1>	5h	
	完成	12月6日	学习LoongArch <1>	5h	
	完成	12月8日	学习LoongArch <1>	5h	
	完成	12月10日	学习LoongArch <1>	5h	
	完成	12月12日	学习LoongArch <1>	5h	
	完成	12月14日	学习LoongArch <1>	5h	
	完成	12月16日	学习LoongArch <1>	5h	

3rd One Student One Chip

■ Tape out

- 12/2021: 39 cores
- 02/2022: 9 cores

■ Breakdown

- EE/IC: 25; CS/SE: 13; Others: 4
- Freshman (2) , Sophomore (3) , Junior (11) , Senior (3)
- Grade 1 master (8), Grade 2 master (11), Grade 3 master (1)
- Ph.d Candidate(4)

■ Micro-architecture Design

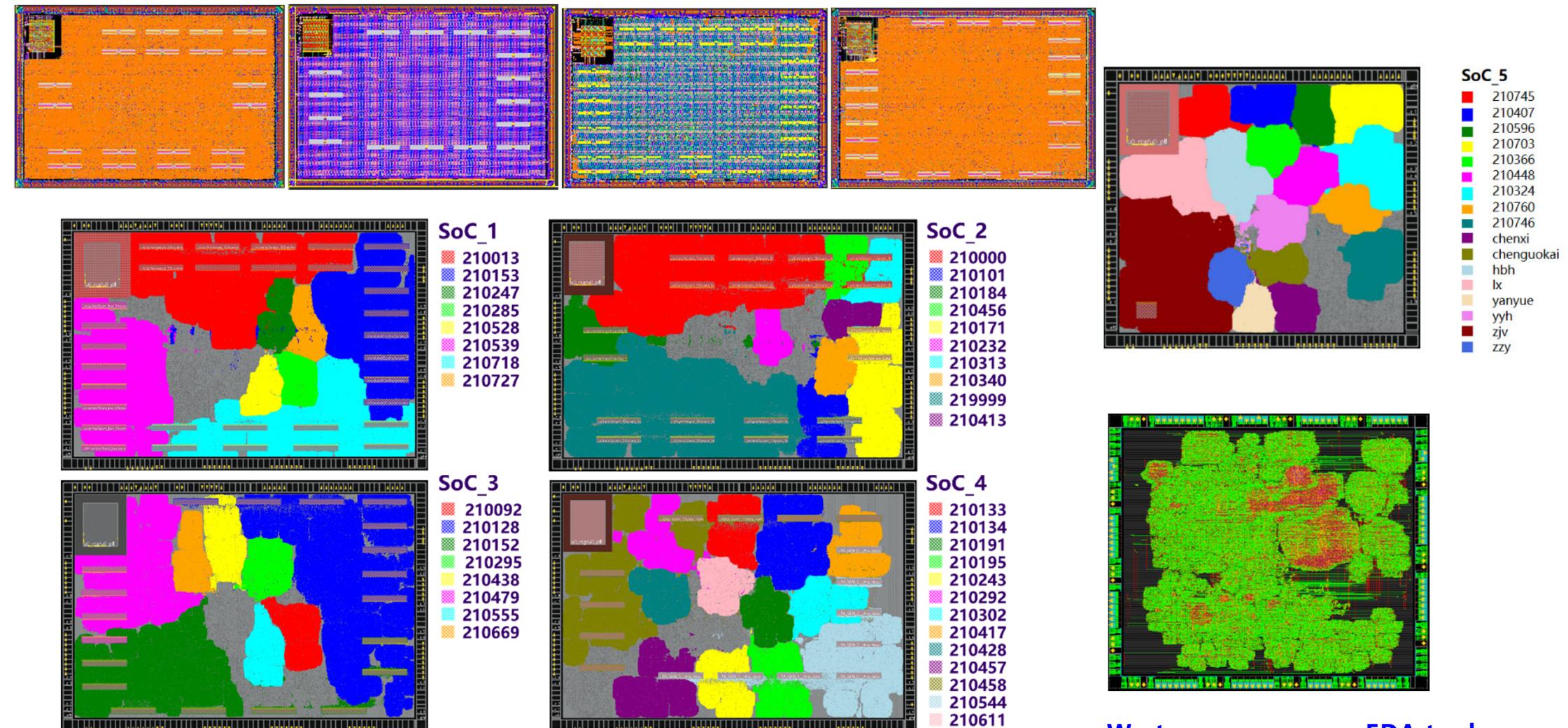
- Branch Prediction: 12;
- OoO: 3;
- Chisel: 16; Verilog/sv: 31
- 5-stage pipeline with Cache: 12

Water Conservancy

University	Grade	Major
ysyx_210092 西安电子科技大学	大三 (即将升计算机科学与技术专业硕)	
ysyx_210456 电子科技大学	研一	电子信息科学与技术
ysyx_210247 南京理工大学	研一	电子信息
ysyx_210243 华中科技大学	大三	电子信息与通信工程
ysyx_210544 南京航空航天大学	博士一年级	软件工程
ysyx_210232 青岛科技大学	大三	集成电路设计与集成系统
ysyx_210295 华东师范大学	研一	集成电路设计与集成系统
ysyx_210457 山东交通学院	大一	电子信息工程
ysyx_210458 太原理工大学	大二	水利
ysyx_210611 南京大学	大一	计算机科学与技术
ysyx_210285 南京大学	大二 (准大三)	计算机科学与技术
ysyx_210128 上海交通大学	大四	电子与计算机工程
ysyx_210727 华中科技大学	研二	计算机科学与技术
ysyx_210718 深圳大学	研二	电子信息
ysyx_210133 电子科技大学	研二	电子科学与工程学院

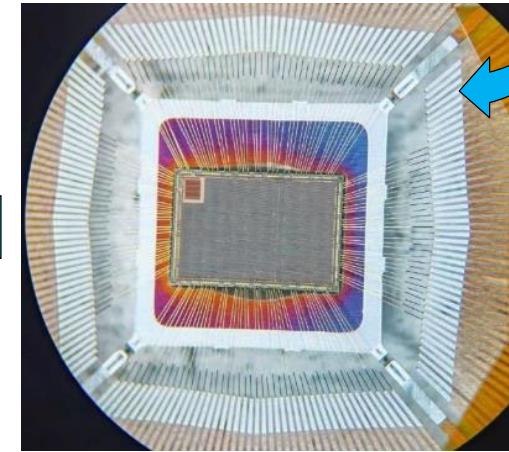
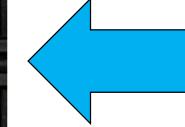
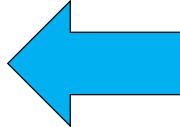
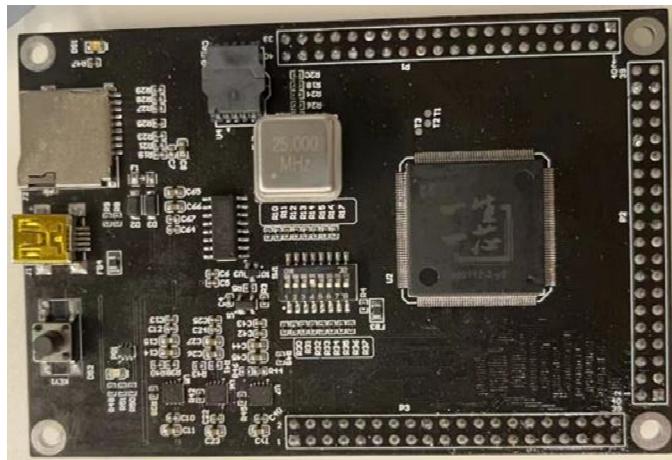
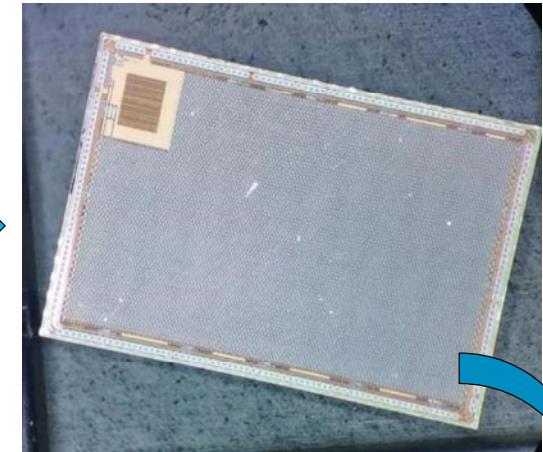
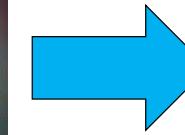
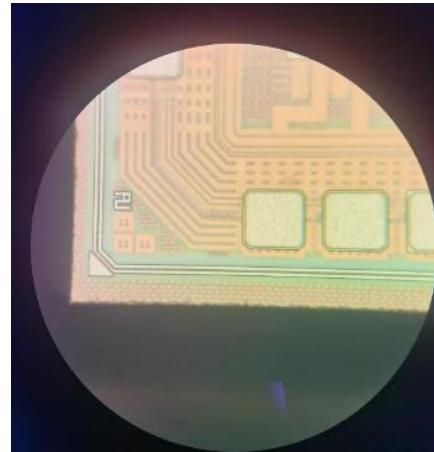
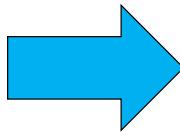
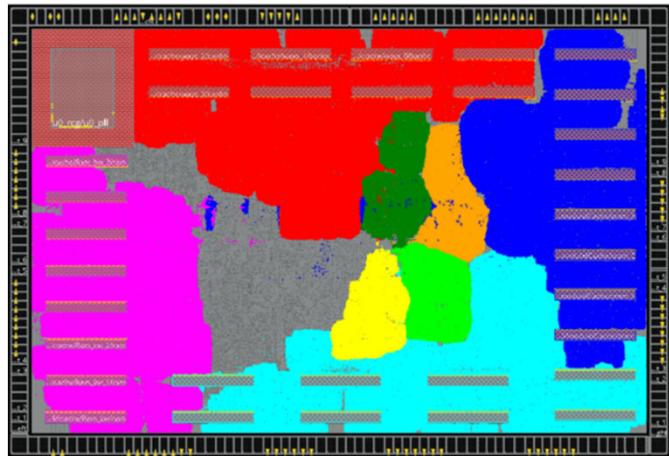
ysyx_210292 集美大学	大四	计算机科学与技术
ysyx_210191 南京理工大学	大三	计算机科学与技术
ysyx_210195 西安电子科技大学	硕士二年级	电子科学与技术
ysyx_210413 大连理工大学	研一	软件工程
ysyx_210428 沈阳工业大学	研二	电子科学与技术
ysyx_210313 电子科技大学	大三	微电子
ysyx_210302 复旦大学	研一	微电子学与固体电子学
ysyx_210184 清华大学	研二	集成电路工程
ysyx_210479 太原理工大学	大三	计算机科学与技术
ysyx_210013 西安交通大学	研二	微电子学与固体电子学
ysyx_210438 南京大学	直博二年级	电子信息技术
ysyx_210555 南京大学	研一	集成电路工程
ysyx_210528 中国农业大学	大三	电子信息工程
ysyx_210669 北京工业大学	研二	计算机技术
ysyx_210417 中国科学技术大学	研一	集成电路工程
ysyx_210134 浙江大学	大三	计算机科学与技术
ysyx_210152 重庆邮电大学	大三	电子信息工程

Taping out of 3rd OSOC



We try open-source EDA tools

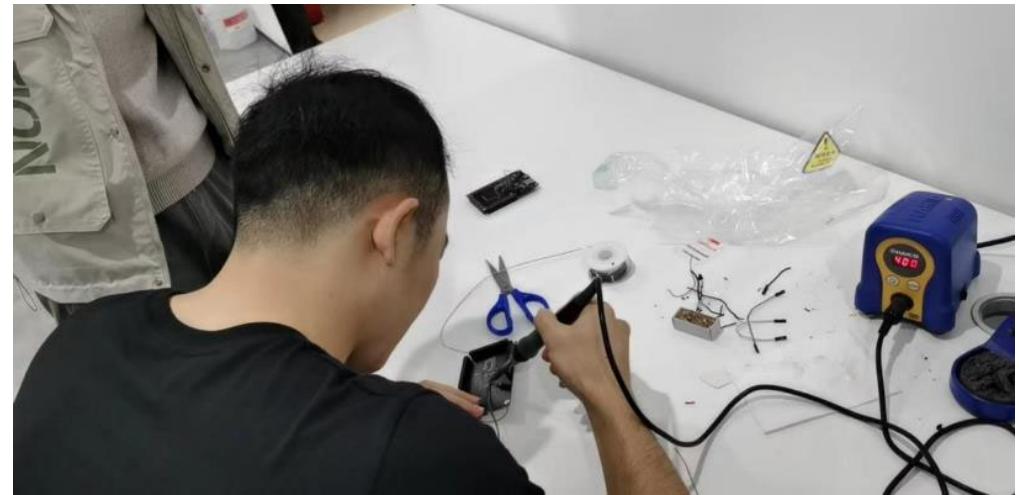
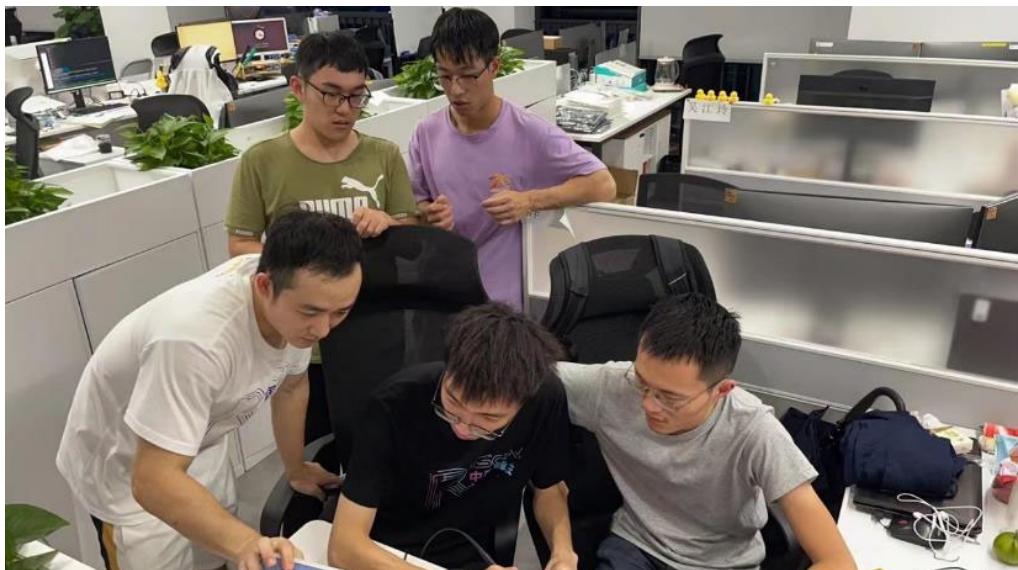
Chip & Board



PCB Testing

Testing team

- 黄健明(海南大学)
- 卢非凡(西安财经大学)
- 马壮(中国科学技术大学)
- 缪宇驰(鹏城实验室)
- 许立达(中科院微电子所)



Software Testing

	hello world	memtest
flash	[10:59:26.279]收←◆Hello World! [11:04:04.430]收←◆Hello World! [11:04:06.275]收←◆Hello World!	[14:51:55.890]收←◆start test... mem tests prepared mem tests passed!!
mem	[14:19:33.762]收←◆Loading program of size: 208 bytes, expect 128 '# Loading.... ##### Load finished Exec app... Hello World!	[15:06:04.482]收←◆Loading program of size: 3840 bytes, expect 128 '# Loading.... ##### [15:06:04.745]收←◆##### Load finished Exec app... start test... mem tests prepared mem tests passed!!

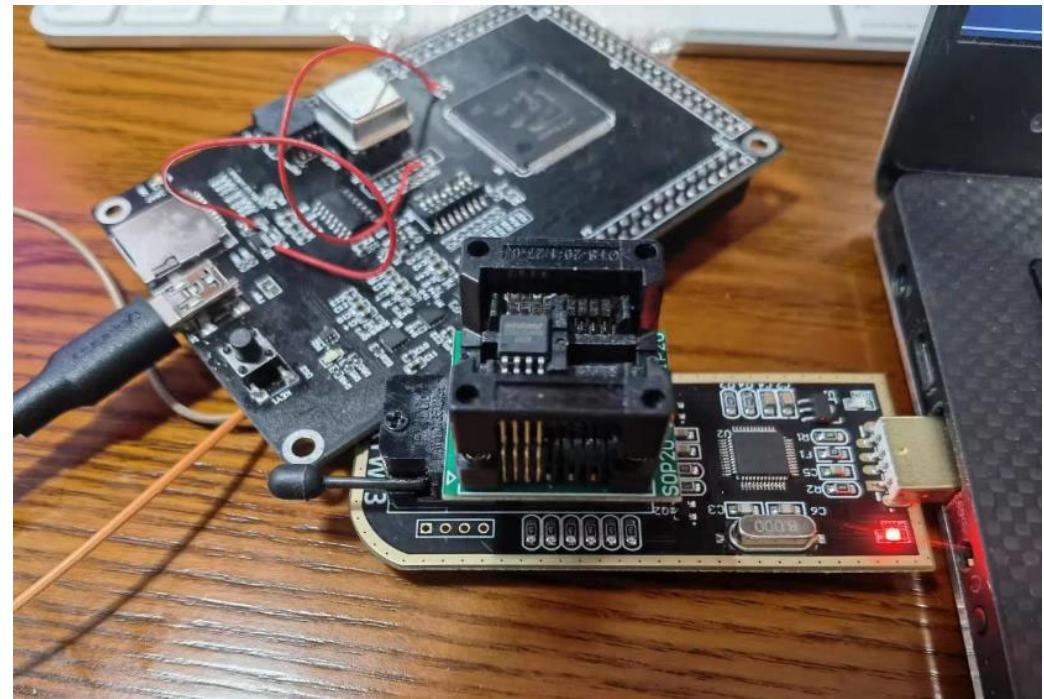
Memory access

```
[10:31:49.356]收←◆[mem data] cnt: 65929216(3ee0000), addr: 0x9f900000
[10:31:49.416]收←◆[mem data] cnt: 65994752(3ef0000), addr: 0x9f980000
[10:31:49.476]收←◆[mem data] cnt: 66080288(3f00000), addr: 0x9fa00000
[10:31:49.535]收←◆[mem data] cnt: 66125824(3f10000), addr: 0x9fa80000
[10:31:49.594]收←◆[mem data] cnt: 66191360(3f20000), addr: 0x9fb00000
[10:31:49.653]收←◆[mem data] cnt: 66256896(3f30000), addr: 0x9fb80000
[10:31:49.712]收←◆[mem data] cnt: 66322432(3f40000), addr: 0x9fc00000
[10:31:49.772]收←◆[mem data] cnt: 66387968(3f50000), addr: 0x9fc80000
[10:31:49.830]收←◆[mem data] cnt: 66453504(3f60000), addr: 0x9fd00000
[10:31:49.889]收←◆[mem data] cnt: 66519040(3f70000), addr: 0x9fd80000
[10:31:49.949]收←◆[mem data] cnt: 66584576(3f80000), addr: 0x9fe00000
[10:31:50.007]收←◆[mem data] cnt: 66650112(3f90000), addr: 0x9fe80000
[10:31:50.067]收←◆[mem data] cnt: 66715648(3fa0000), addr: 0x9ff00000
[10:31:50.126]收←◆[mem data] cnt: 66781184(3fb0000), addr: 0x9ff80000
[10:31:50.185]收←◆mem tests passed!!
[10:32:25.838]收←◆\0\0
```

Booting
RT-Thread

```
Load finished
Exec app...
heap: [0x80022590 - 0x86422590]
\ | /
- RT - Thread Operating System
/ | \ 4.0.4 build Nov 29 2022
2006 - 2021 Copyright by rt-thread team
Hello RISC-V!
thread1 count: 0
thread2 count: 0
msh />
[10:41:52.743]收←◆thread1 count: 1
thread2 count: 1
[10:41:53.385]收←◆thread1 count: 2
thread2 count: 2
```

Student tries at home



Hello, YSYX!

Have a good luck

Have a good luck!

Have a good luck

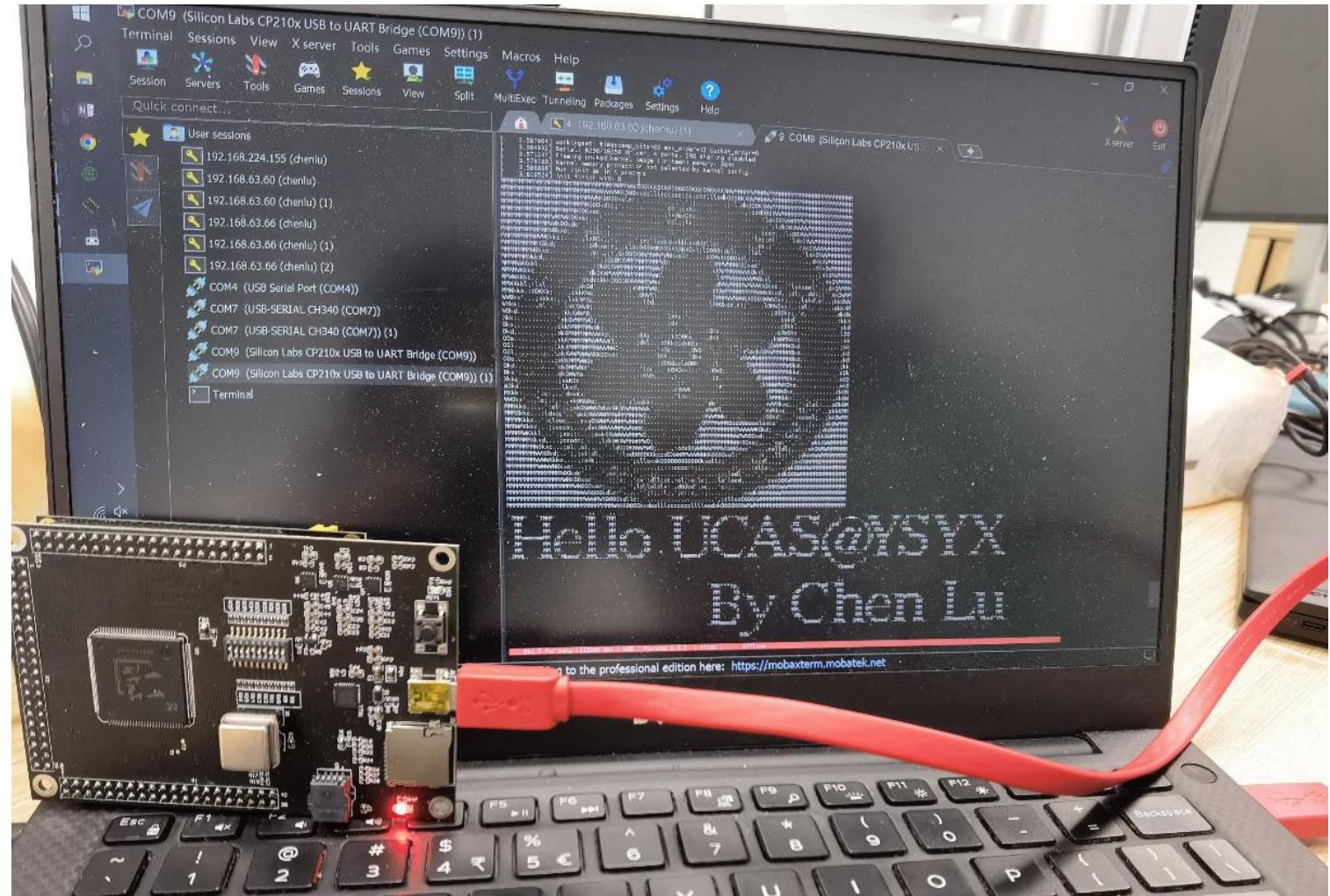
Have a good luck!

Have a good luck!

Have a good L

Demo by single student

- Lu Chen@NJU,
computer science,
learning when
junior
 - Load Linux from
flash and boot
successfully,
showing the CAS
logo



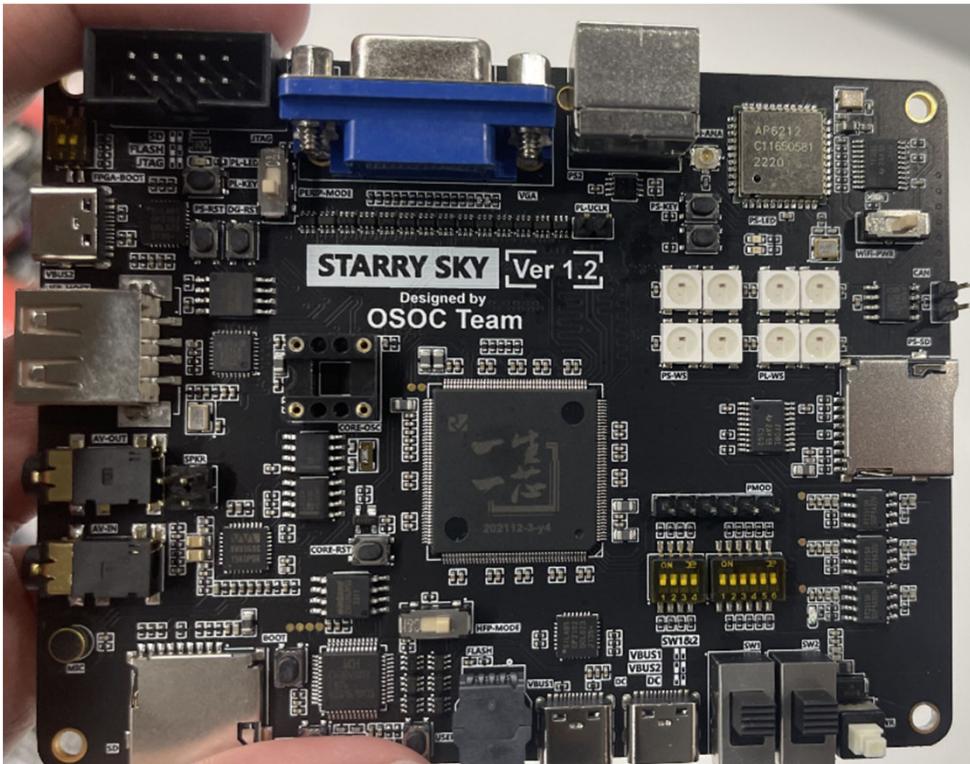
Demo by another single student

- Haojin Tang@UCAS,
Electronic engineering,
learning when junior
 - Boot Debian and run
games



<https://www.bilibili.com/video/BV1CL411X7wV>

Delivering PCB to students

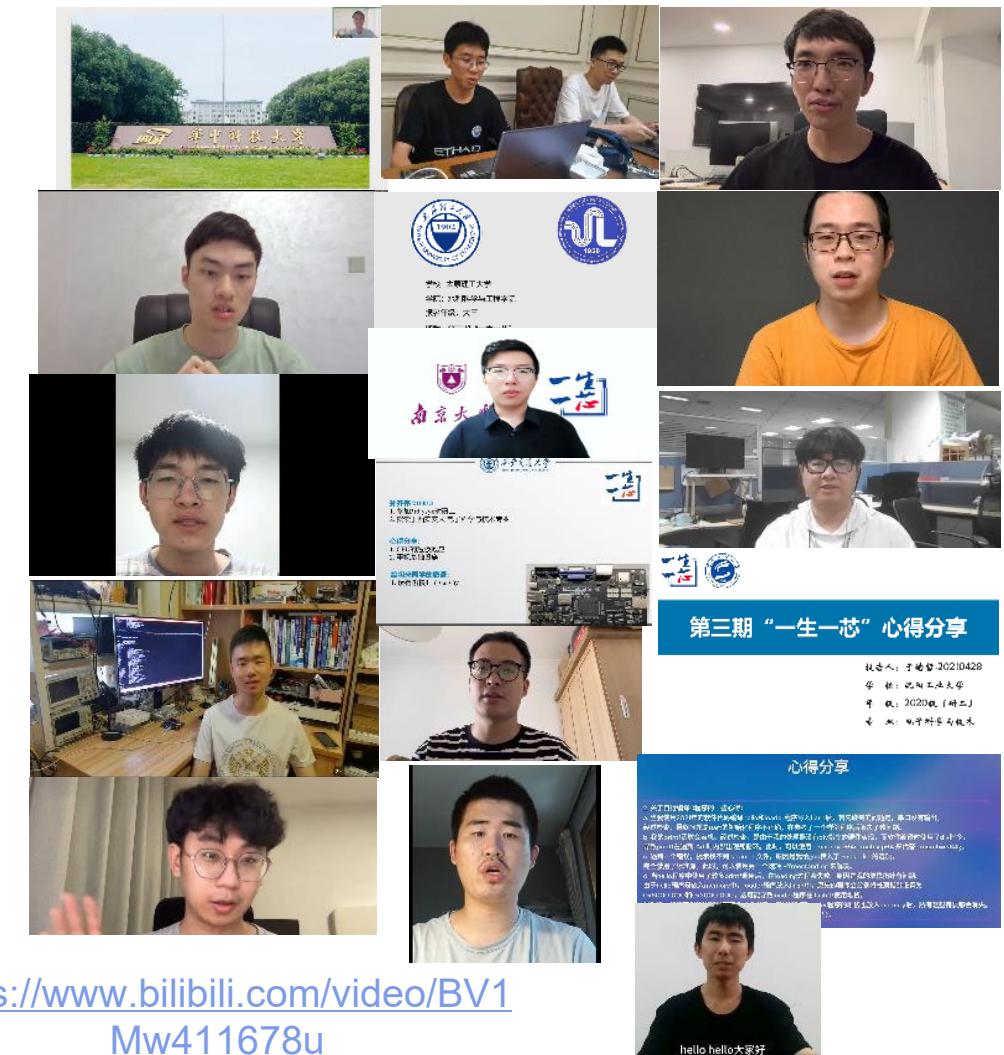
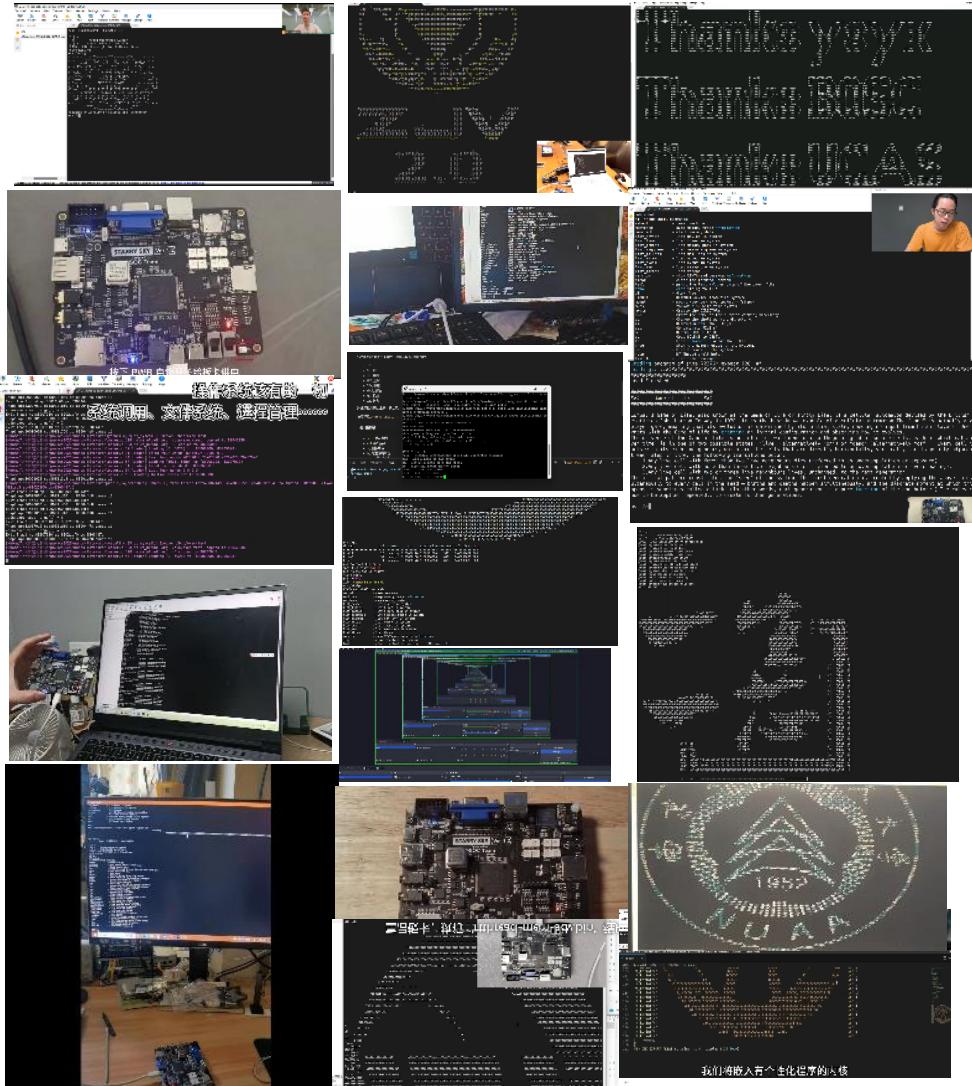


PCB designed by OSOC



Before delivering

Brought up Chips, Ran OS & Applications



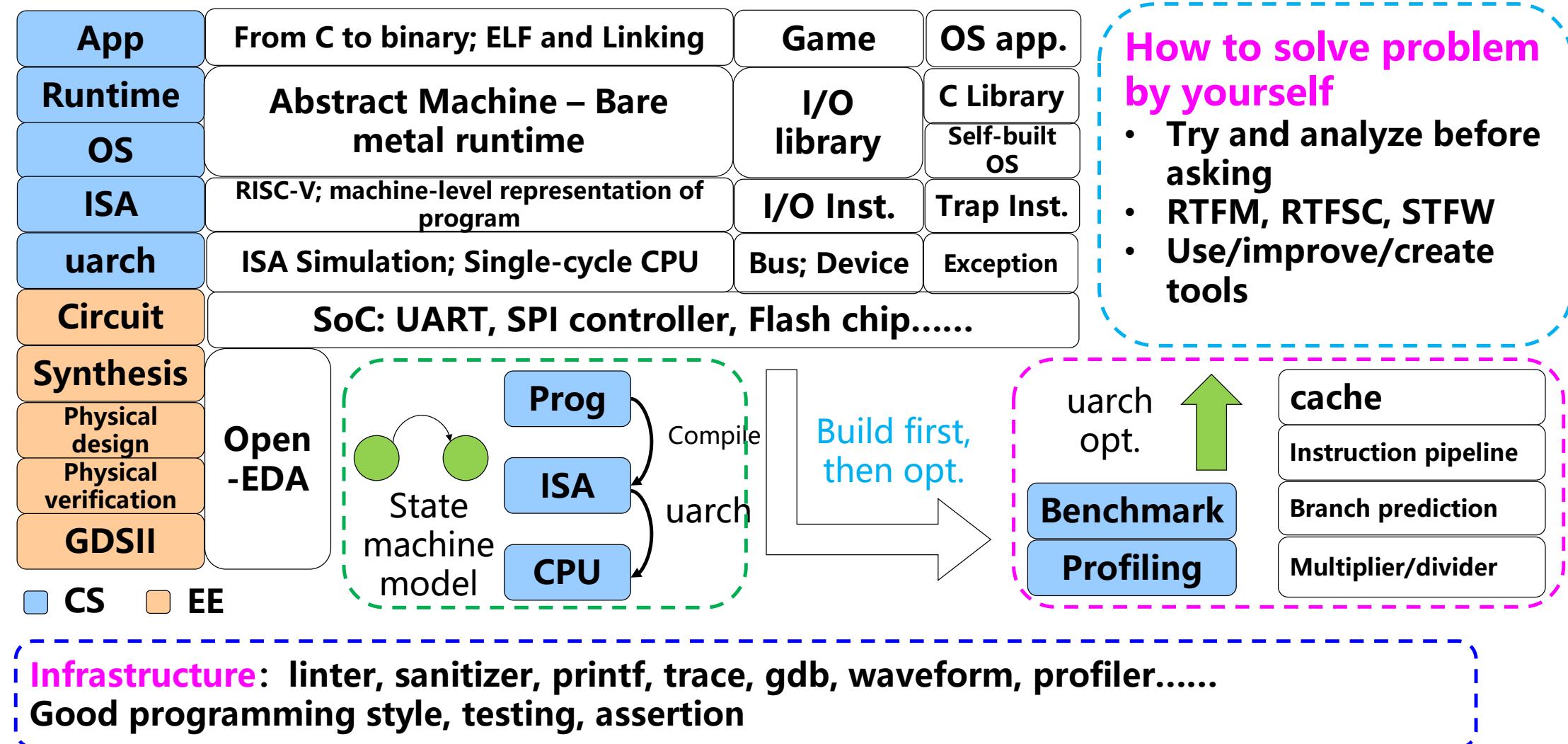
<https://www.bilibili.com/video/BV1Mw411678u>

4. 4th OSOC and after – Learning more (Feb 2022 to now)

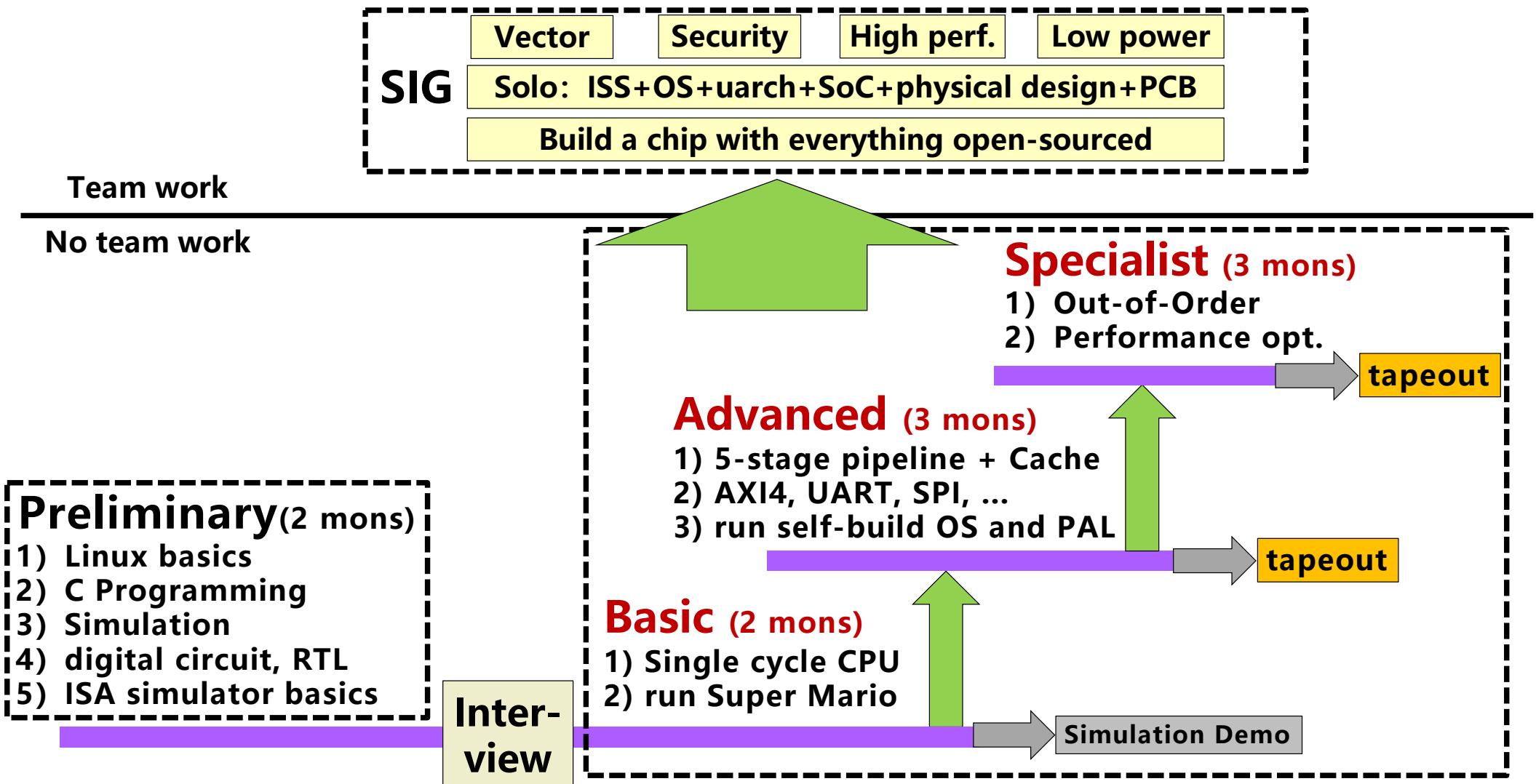
4th and 5th OSOC

- Higher goal: RV64IM, self-build OS, run non-trivial games
- Add self-build simulator and system software **practicing software-hardware co-design**
 - Build an ISA simulator, compile programs, develop a small OS and runtime
- **Better lecture note**
 - Preliminary stage/Basic stage/Advanced stage
 - Building a RISC-V system from scratch
- No team work any more
- **Videos and slides** in 5th OSOC

Knowledge Diagram



Learning stages



Learning Schedule

C = C language(sw)

R = RISC-V(ISA)

P = Processor(hw)

T = Tools(infrastructure)

RV32E
system

Intro
Simple system

阶段	序号	任务	讲义	课件	视频	C	R	P	T
预学习阶段	1	如何科学地提问	书	文	影				
	2	Linux系统安装和基本使用	书	文	影				✓
	3	计算机系统的状态机模型	-	文	影	✓	✓	✓	
	4	复习C语言	书	文	影	✓			✓
	5	程序的执行和模拟器	-	文	影	✓	✓		
	6	搭建verilator仿真环境	书	-	-	✓		✓	
	7	数字电路基础实验	书	文	影			✓	
	8	完成PA1	书	文	影	✓			✓
■申请入学答辩									

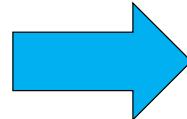
9	支持RV32IM的NEMU	书	文	影	✓	✓	✓		
10	程序的机器级表示(上)	-	文	影	✓	✓			
11	程序的机器级表示(下)	-	文	影	✓	✓			
12	用RTL实现最简单的处理器	书	-	-					✓
13	AM运行时环境	书	文	影	✓				
14	工具和基础设施	-	文	影					✓
15	支持RV32E的单周期NPC	书	文	影				✓	✓
16	ELF文件和链接	-	文	影	✓	✓			
17	设备和输入输出	书	文	影	✓	✓	✓		✓
18	调试技巧	-	文	影					✓
19	异常处理和RT-Thread	书	文	影	✓	✓	✓		
20	总线	书							✓
21	SoC计算机系统	书							✓
22	性能和简易缓存	书							✓
23	流水线	书							✓
■达成B阶段流片指标									

Advanced stage: M extension, S mode & U mode, A extension, xv6, Linux, uarch opt.

Demo in 4th OSOC

- Adding VGA controller

Display by UART in 3rd OSOC



Basic stage: run Super Mario (released in 1980s)

Demo in 4th OSOC

- Adding VGA controller



Advanced stage: run PAL
(a famous Chinese RPG,
released in 1990s)



Challenge task: run CLANNAD
(a Japanese game,
released in 2000s)

Learning resources are opened

第五期“一生一芯”课程主页

- 课时: 每周六19:00~21:00
 - B站直播  | 录播链接 
 - 答疑: 每周日19:00~20:00 (通过预学习答辩后由助教通知)
 - 报名流程 | 报名常见问题

课件和讲义



计算机系统的状态机模型

余子濠



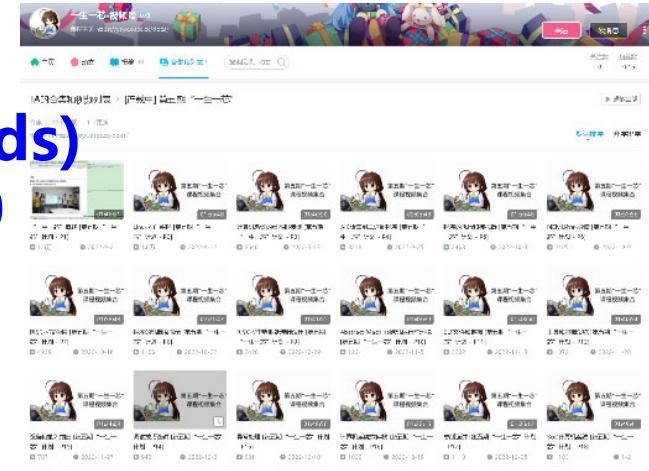
中国科学院
计算技术研究所



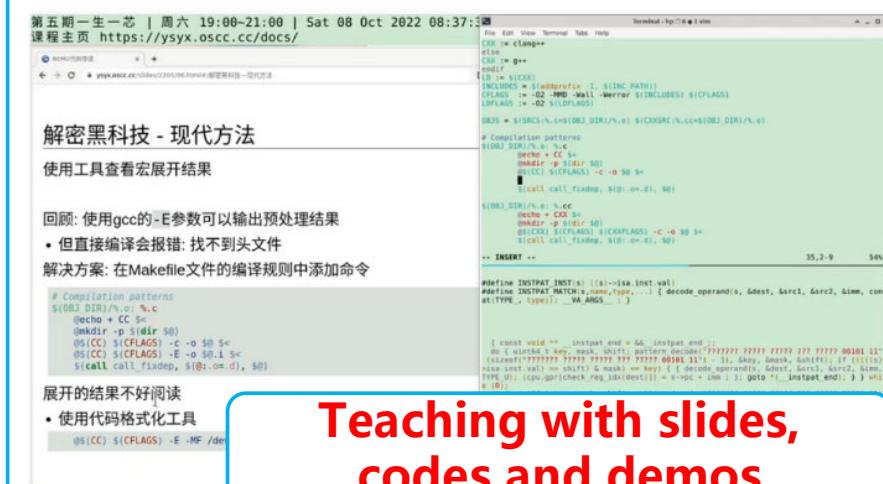
计算机系统与处理器
芯片课程虚拟教研室

中国科学院大学

- Course Website
 - Lecture note(260,000 words)
 - Slides(>800 pages, 85,000 words)
 - Videos(> 40 hours)



Account in Bilibili: 一生一芯-视频号



English version of learning resources

The screenshot shows two adjacent web pages from a platform titled "The 6th 'One Student One Chip' Program".

Left Page: The 6th "One Student One Chip" Program Home Page

- Navigation Bar:** Course Home, Study Handouts, Other Materials, Select Languages.
- Section: Learning Objectives**
 - Time: Every Saturday 15:00~17:00 China Standard Time
 - [Bilibil Live](#) | [recording](#)
- Section: Learning Objectives**

"One Student One Chip" will develop your general skills. At the end of the course, you will have a better understanding of the following questions:

 1. how processors are designed?
 2. how programs run on computers?
 3. how to optimize the performance of a processor?
 4. how to use/design the right tools for efficient debugging?
 5. how to write your own test cases for unit testing?
 6. how does an RTL design generate a flowable layout?

We will guide you to design a RISC-V pipeline processor. Run an operating system on your processor. Run a real game on the OS. The processor that achieves the target will be connected to the SoC and will be given the opportunity to tapeout.
- Section: Learning Resource**
 - Icons can be clicked to jump to the appropriate resource
 - Complete handouts can be viewed via the navigation bar at the top right of the page
 - The content of the Stage 5 handout is still available

C = C language (program/simulator/system software) | R = RISC-V instruction set | P = processor design | T = tools

Right Page: How to ask smart questions

- Prestudy Stage**
 - Prestudy overview
 - How to ask smart questions**
 - Linux system installation and basic usage
 - Reviewing the C language
 - Build a verilator simulation environment
 - Basic Digital Circuit Lab
 - Complete PA1
 - Submission of pre-study defense application
- Task 1: Fill in the general education questionnaire**

Before starting the first task of pre learning, please carefully read the content in sections "[Signup](#)" and "[FAQ](#)" on the official website and fill in "[The general education questionnaire of 'One Student One Chip'](#)". Note: The general education questionnaire can be repeated many times, and only those who score 100 can apply for admission defense.
- Task 2: Read "How To Ask Questions The Smart Way" and "Stop Ask Questions The Stupid Ways", and write an essay of your thoughts on them**

Your first task in the prestudy is to read the articles "[How To Ask Questions The Smart Way](#)" and "[Stop Ask Questions The Stupid Ways](#)" and "[Don't Ask Like a Retard](#)", and write an 800-word essay about your experience of asking and being asked questions, and what you think about "good questioning" and "Independent problem solving through STFW and RTFM".
- Task 3: STFW, RTFM, RTFSC**

Try to find and understand the meanings of the three acronyms in the above article.

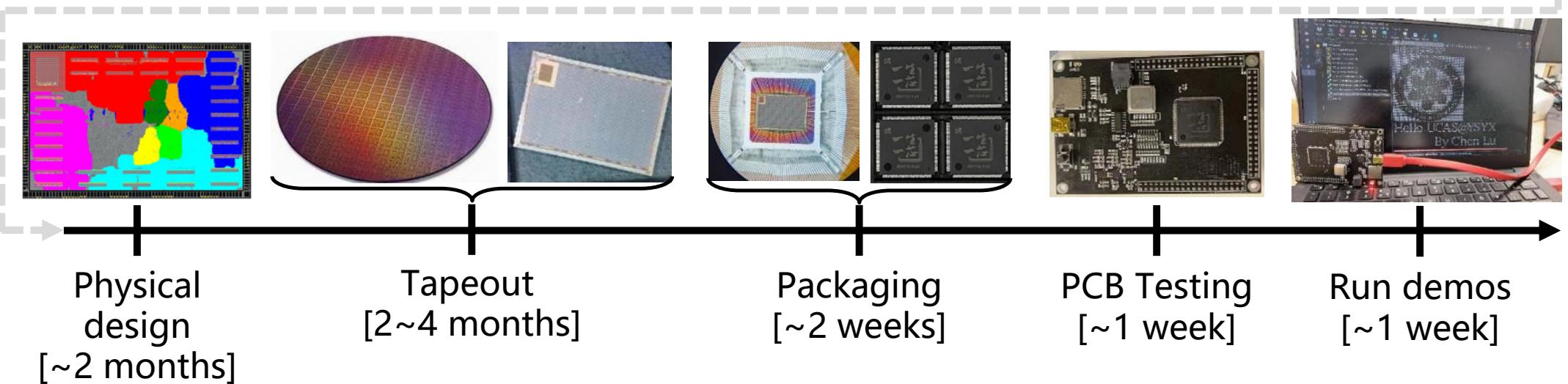
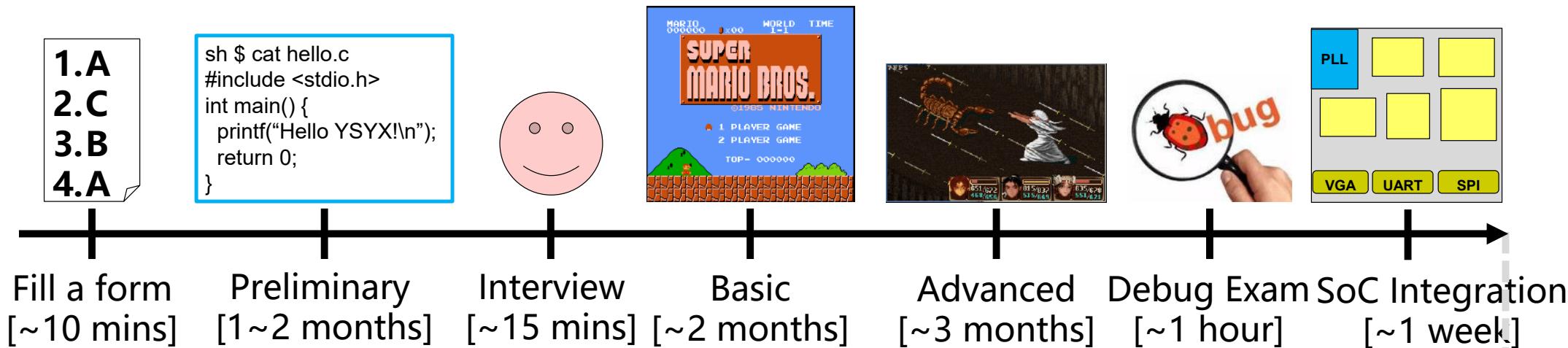
You may feel offended by the F word, but in fact the meaning of the F word is never the point, it just reflects the legend behind the three acronyms and makes them easier to remember. For example, RTFSC originated with the first words of Linus Torvalds, the father of Linux, in a reply to an email dated April 1, 1991, which is still available on the Internet mailing list. Interestingly,

The translation work is still on-going.

Online-debug exam, instead of paper exam

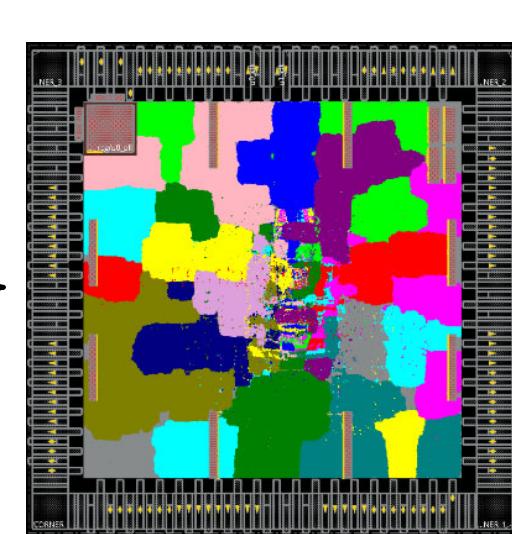
- When students apply for tapeout, TA will inject some bugs randomly in students' project
 - Including hardware, software, building and simulation system
- Students are required to debug within 1 hour in an online meeting
 - Students should share the screen
- According to the debugging process, TA will evaluate whether
 - Student knows enough details in the whole project
 - Student can analyze problem from the respect of sw-hw co-operation
 - Student knows when to use the right tools during debugging
 - Student can solve new problems by himself

Learning Roadmap



Students finish in 4th OSOC

	Student #	University	Major	Grade
1	ysyx_22040053	南京大学	CS	Sophomore
2	ysyx_22040066	南京大学	CS	Sophomore
3	ysyx_22040656	中国计量大学	CS	Sophomore
4	ysyx_22040163	南通大学	CS	Sophomore
5	ysyx_22040091	中国科学院大学	CS	Junior
6	ysyx_22040596	华南理工大学	EE	Senior
7	ysyx_22041812	南方科技大学	EE	Grade 1 master
8	ysyx_22040127	东南大学	AI	Grade 1 master
9	ysyx_22040654	福州大学	EE	Grade 1 master
10	ysyx_22040978	中国科学院大学	EE	Grade 1 master
11	ysyx_22041514	杭州电子科技大学	CS	Grade 1 master
12	ysyx_22040213	中国科学院大学	EE	Grade 2 master
13	ysyx_22040561	北京大学	EE	Grade 2 master
14	ysyx_22041461	四川大学	EE	Sophomore
15	ysyx_22040886	北京理工大学	EE	Junior
16	ysyx_22050228	东北大学	EE	Junior
17	ysyx_22050920	杭州电子科技大学	EE	Senior
18	ysyx_22040501	哈尔滨工业大学	Information and Communication Engineering	Grade 1 master
19	ysyx_22050133	北京大学	Machinery	Grade 2 master



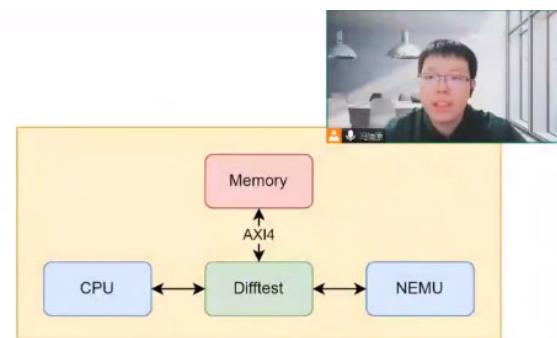
- ysyx_040053
- ysyx_040066
- ysyx_040091
- ysyx_040127
- ysyx_040163
- ysyx_040213
- ysyx_040561
- ysyx_040596
- ysyx_040654
- ysyx_040656
- ysyx_040978
- ysyx_041514
- ysyx_041812

Learning Experiment Shared by Students

- Haoyuan Feng@UCAS, computer science, learning when junior
 - Later became one of the key member of the "XiangShan" team
 - <https://www.bilibili.com/video/BV1C54y1T7hw>

心得分享①：理解系统

- “一生一芯” 仿真系统
 - 逐步建立了对系统的认识，增加调试的信心与效率
 - 仔细阅读源码和手册说明
 - 通过阅读源码，了解 NEMU 和 Spike 的对比机制
 - 接入 ysyxSoC 后，为什么无法从 SRAM 中读出数据
 - “香山” 团队中的学习、工作
 - 处理器架构和验证框架复杂很多
 - 主要负责 MMU 模块的开发
 - 涉及到和前端取指、后端访存、内存访问的交互
 - 包含 L1 TLB、L2 TLB、PMP、PMA 等子模块
 - 实践一生一芯的系统观念
 - 从整体入手，关注前端取指和后端访存对 MMU 的查询行为
 - 关注细节，理清 MMU 对于页表查询请求的处理过程
 - 仔细阅读源码，不放过每一处内容，对模块充分掌握



“一生一芯” 仿真系统示意图

端口	说明
output [127:0] Q	读数据
input CLK	时钟
input CEN	使能信号, 低电平有效
input WEN	写使能信号, 低电平有效
input [5:0] A	读写地址
input [127:0] D	写数据

YsyxSoC 中对 SRAM 行为的描述

OSOC Forum@2nd RISC-V China Summit



High school student shares
learning experience

The FIRST high school student who
passes the interview in OSOC!

RVFA certification exam translation

consider some words



Chapter 1: RISC-V Overview

Conventions used in this file:

- Heading 1 is used to mark the beginning of a course page.
- Heading 2 is used for subtitles within a page.
- Bold for references to buttons or menu options, and first sentences in bullet points.
- Bold *Italics*** for introducing new terms.

According to LF Author's Guide:

- Bold Consolas Font (black)** for source code.
- Bold Consolas Font (Dark Blue)** for file names and folder names.
- Bold Consolas Font (Dark Blue)** Text typed at the command line.
- Bold Consolas Font (Green)**: Output.
- Hyperlinks** are left in GoogleDocs' format.

Learning Objectives

- By the end of this chapter, you should be able to:
- Understand what RISC-V is and what it is not.
 - Identify the characteristics of an ISA to decide whether it is a CISC or a RISC type ISA.
 - Describe the history of RISC-V.
 - Identify the most notable differences between RISC-V and the leading commercial ISAs, like x86 and ARM.
 - Understand the structure and operation of RISC-V International.
 - Analyze the documentation of RISC-V specifications.
 - Explore ways to contribute to the RISC-V effort.

Chapter Introduction

In this chapter, you will get acquainted with RISC-V. Please note that you are expected to already be familiar with computer architecture and to have some exposure to some specific ISA. We will provide some refreshers on basic terms, but this is certainly not an introduction to computer architecture.

Among the topics we will cover, you will learn about the history of RISC-V and how it differs from today's dominating ISAs. You will also get the chance to browse through the vast documentation of RISC-V, and you will learn how the documents are organized.

You will be introduced to RISC-V International and the RISC-V ecosystem, and you will learn about quite a few different ways you may contribute to the RISC-V community.

Let us get started!

History of RISC-V: The Free and Open ISA

RISC-V (pronounced "risk-five") is an open standard instruction set architecture principle, enabling a new era of processor innovation through open standard co

RISC stands for Reduced Instruction Set Computer, a computer architecture principle based on simplicity, as opposed to current microprocessors at the time, dubbed Computers, or CISC. The RISC architecture was born in an academic environment for simplicity and efficiency, proposing a series of features that dramatically improved what was motivated by commercial interests at the time. RISC is the opposite of CISC in many ways:

original & translation

1: 自由开放的 ISA

是一种基于 RISC 规范的开放标准指令集架构 (ISA)，通过开放标准协作促进

几，这是一种 1980 年代初期提出的基于简单性的计算机体系结构，与当时的 CISC 的现代处理器设计。RISC 架构诞生于学术环境，因此在设计上力求简洁、高效且容易使用。CISC 架构则更侧重于复杂性和多样性。RISC 在许多方面与 CISC 截然不同，例如它的指令集更小，其大部分指令可以访问内存，而 RISC CPU 有很多内存访问仅限于加载和存储指令。

RISC-V翻译工作组 (18)

2023-6-7日 1

刘汉章
这种语序的调整是可以的，但是忽略了它原本的一些连词 (thus) 但是更符合中国人的语言好像

刘汉章
因此，一个 I

刘汉章
针去正确地

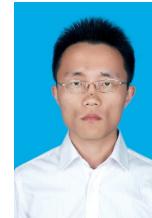
我觉得没问题，在不丢失原本语义的情况下，可以按照中文习惯进行适当地修改。

刘汉章
好

中英文对照表已经整理好了
https://docs.google.com/spreadsheets/d/1TgFLAT4hb3sC37cvUWBeoSU75iQLP9q73WcaWtdZB/edit#heading=h.32of465ab4d3），但是由于缺乏具体的语境，有一些单词没办法识别是动词还是名词，所以中文只能作为参考，要具体问题具体分析（翻译有问题可直接在线修改）。

2. 昨天提到的每篇文档中都有的内容或者跟格式相关的标准我统一放到这个文档里面 (<https://docs.google.com/document/d/1TgFLAT4hb3sC37cvUWBeoSU75iQLP9q73WcaWtdZB/edit#heading=h.32of465ab4d3>)，如果大家没有异议的话，可以直接复制到自己的文档中（翻译有问题可直接在线修改）。

3. 像rs1, jar, opcode, qemu这种专有名词我觉得保持原样就好，像ISA, IPC这种由几个单词首字母拼接在一起的专有名词，可以像昨天讨论的那样翻译时保持原样。关于是否在第一次出现的地方添加括号解释，我觉得可以最后等到Review时由一个人统一添加一下会比较好。



缪宇飚



苗金标



段震伟



刘汉章



曹勋



杨海帆



曹世洋



倪仁涛



魏人



陈璐



粟金伦



吴佳宾

Name	University	Grade
缪宇飚	中科院计算所	- (组长)
苗金标	中国科学技术大学	研二
段震伟	中国科学技术大学	研三
刘汉章	太原理工大学	大三
曹勋	中国科学技术大学	研二
杨海帆	浙江工商大学	大四
曹世洋	中国科学技术大学	研二
倪仁涛	东北大学	研一
魏人	兰州大学	大四
吴佳宾	青岛大学	研一
陈璐	中国科学院大学	博一
粟金伦	太原理工大学	大四

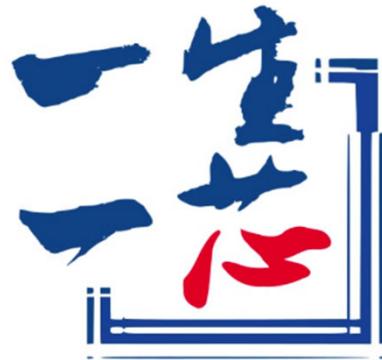
Campus lecture tour

- From March 2023 to now, OSOC has gone deep into the campus and carried out many lecture tours
- In 2024, the journey will continue!

Data	University	Contents
3月19日	北京工业大学	宣讲
4月13日	北京科技大学	宣讲+教学交流
5月17日	东北大学（秦皇岛分校）	宣讲
5月25日	青岛大学	宣讲+教学交流
6月10日	天津工业大学	宣讲
6月14日	天津理工大学	宣讲+教学交流
6月16日	太原理工大学	宣讲



Thank you!



website
ysyx.org



WeChat official account



计算机系统与处理器
芯片课程虚拟教研室

北京开源芯片研究院
BEIJING INSTITUTE OF OPEN SOURCE CHIP



有道
youdao