

2026年南京大学开源软件论坛



FPGAOL Next

下一代FPGA实验平台

罗嘉宏

中国科学技术大学化学物理系

luojh@mail.ustc.edu.cn

目录

contents

1 为何需要FPGAOL?

2 设计与实现：EtherFPGA架构

3 我们的FPGA确无问题！
——开源FPGA工具链的现状

4 教学 or 科研，差别究竟在哪？

5 结语

1

为何需要FPGAOL?



为何需要FPGAOL?

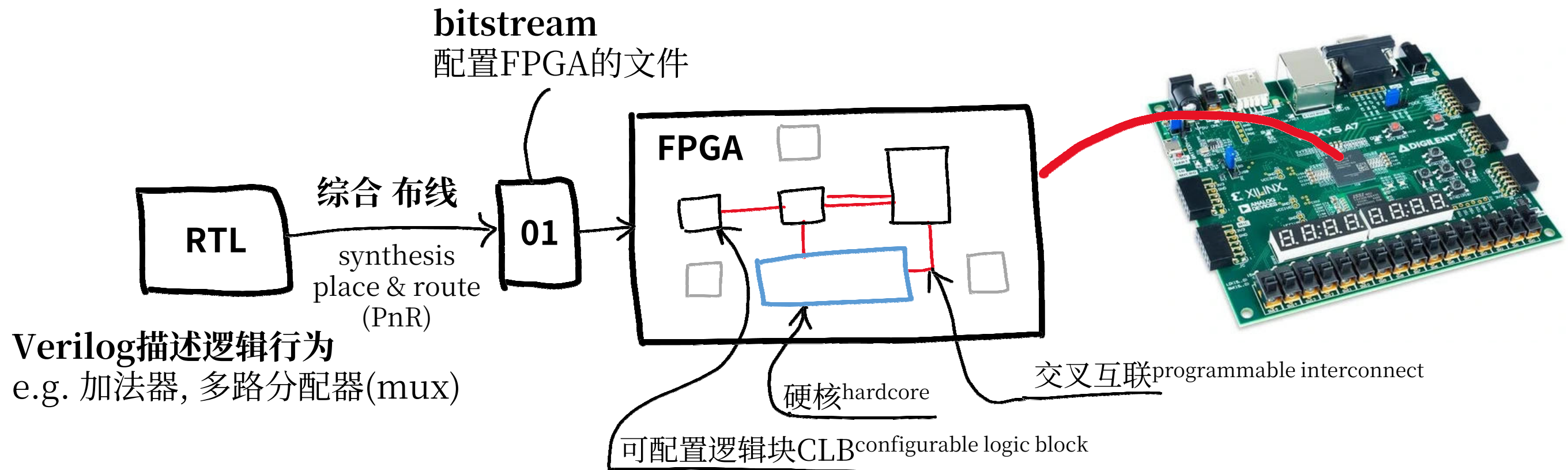
- FPGA(field programmable gate array)作为可编程逻辑(programming logic)的代表器件，已经用在各种场景



图：一块FPGA加速卡，用于计算加速辅助处理
使用Xilinx的Virtex-6系列FPGA

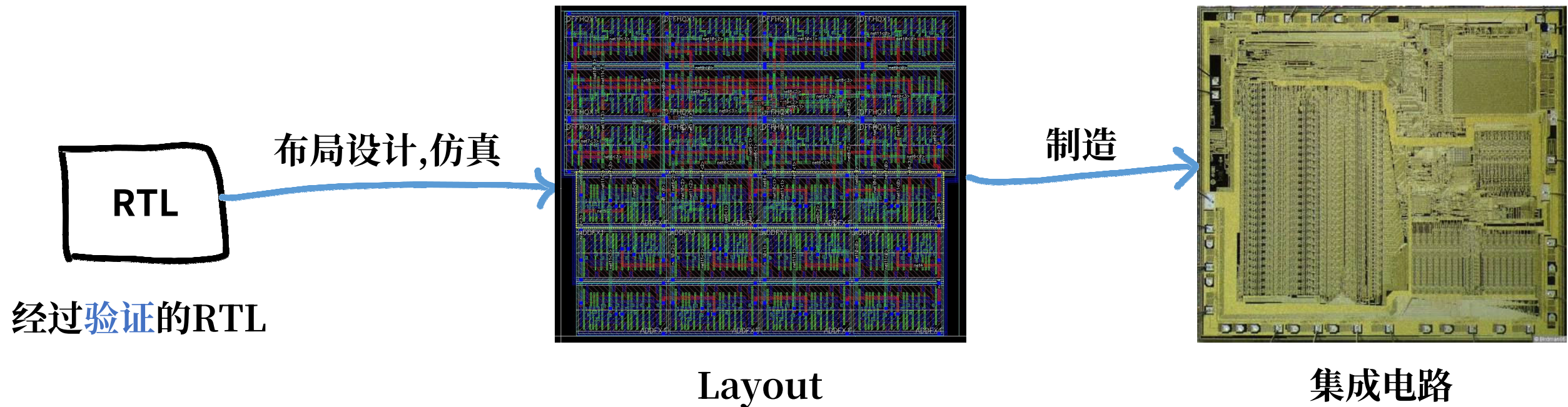
为何需要FPGAOL?

- 在数字电路设计(这里特别是ASIC设计)中, FPGA是一种重要的**原型验证**器件。



为何需要FPGAOL?

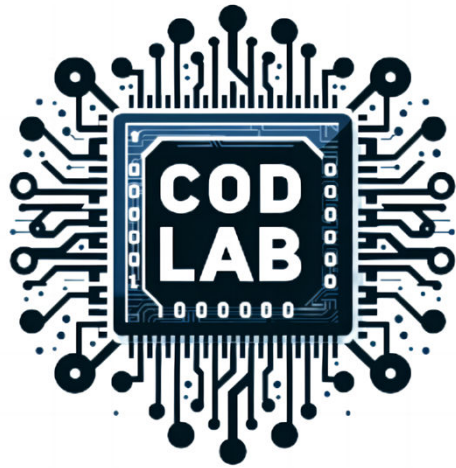
- 在数字电路设计(这里特别是ASIC设计)中, FPGA是一种重要的**原型验证**器件。



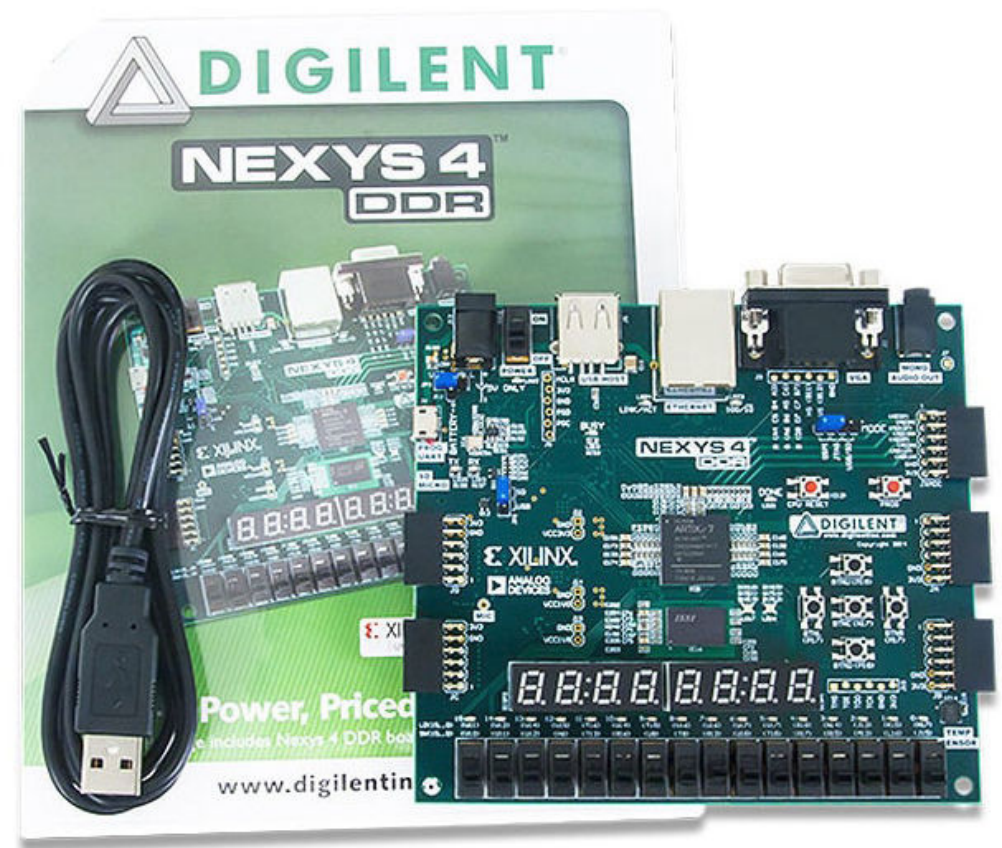
为何需要FPGAOL?

- 在教学中，FPGA广泛用于**数字电路**、**计算机组成原理**(实际上就是设计处理器)的教学。

soc.ustc.edu.cn



👉 图：Digilent NEXYS 4 DDR FPGA开发板
我们曾经的计算机组成原理教学实验平台



为何需要FPGAOL?

- **但是!**
- FPGA开发板直接交给学生做实验，存在一些问题：
 - FPGA板卡价格高，需求数量大(每人一块)但利用率低
 - 板卡可能本身有一些故障，并且可能容易损坏
 - 开发环境不统一，因为环境差别造成实验困难
 - 靠人观察外设(例如数码管)表现来判断实验成功与否，~~耗费助教~~

为何需要FPGAOL?

“已经生成了一个正确的比特流，然后只修改了 .coe 文件，并没有改动其他设置，但生成的比特流始终不正确。后来重新创建了一个 Vivado 工程，再次生成比特流，结果就正常了。助教老师说这可能是 Vivado 的问题，在上一次生成比特流后，没有完全清理干净缓存。如果有遇到类似问题的朋友，可以先尝试重新创建一个新工程再生成，看看能否解决问题 😞” (DragonKnighttt)



420实验室中至少已经看到两块损坏的Nexys开发板，这是总数的大约1%到2%

助教检查实验的方式是，(数字电路实验)比对波形图和正确的参考波形，(在SoC实验中)通过串口输出和数码管状况确定实验是否完成。

为何需要FPGAOL?

- 解决上面的问题?
 - 数量少&利用率低 → 时分复用(time sharing)
 - 板卡损坏 → 机房固定环境，远程访问
 - 开发环境不统一 → 预配置的开发环境
 - 人工判断实验完成 → 自动评测，online judge 思路

“**FaaS**” —— **FPGA as a Service**

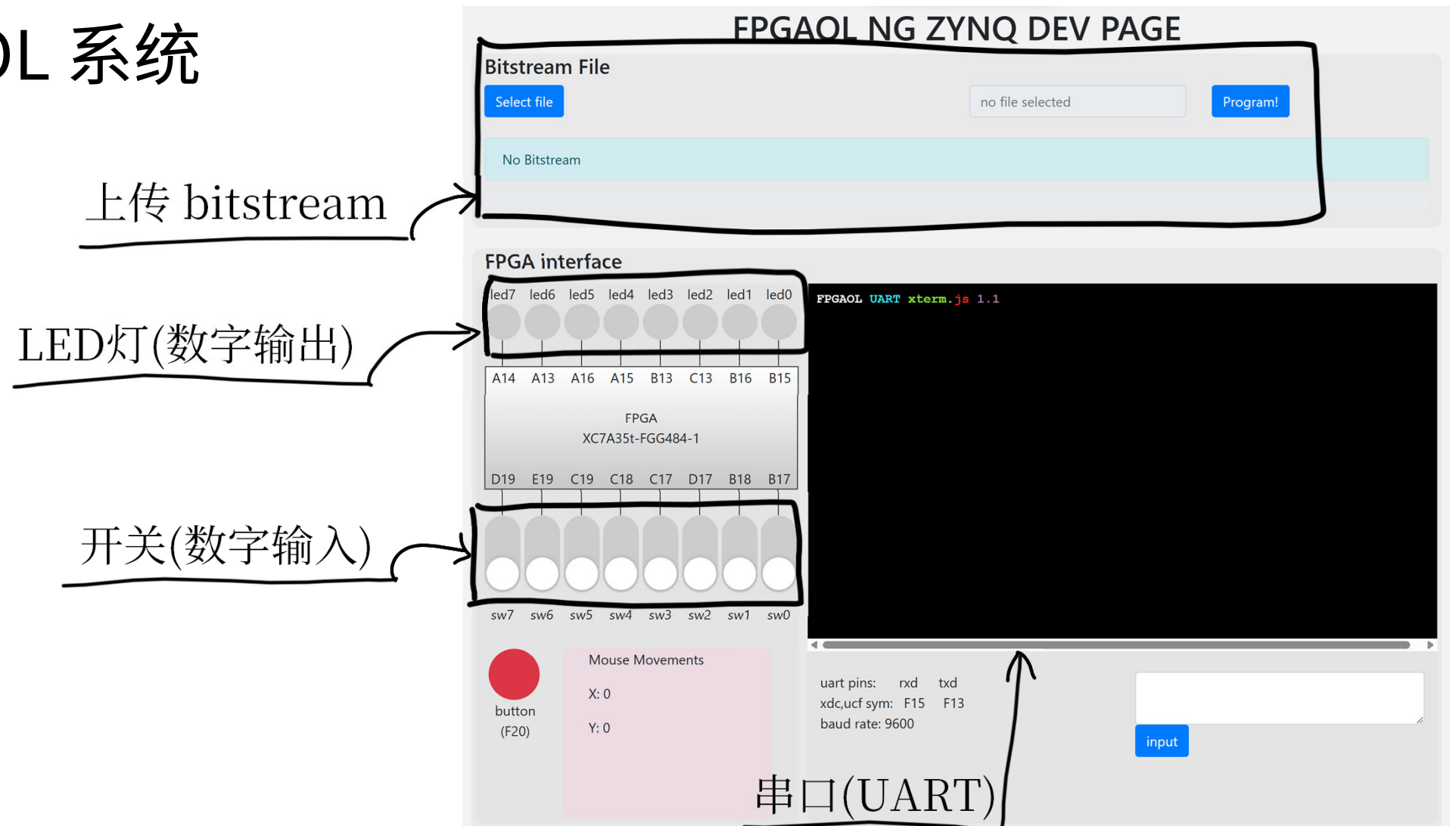
为何需要FPGAOL?

- 已经工作的 FPGAOL 系统

fpgaol.ustc.edu.cn

目标

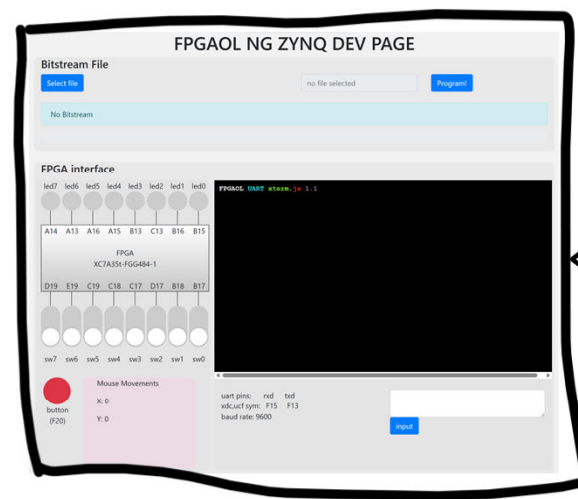
浏览器中模拟一块现实的FPGA板卡，用户与板卡上的外设交互



为何需要FPGAOL?

- 已经工作的 FPGAOL 系统

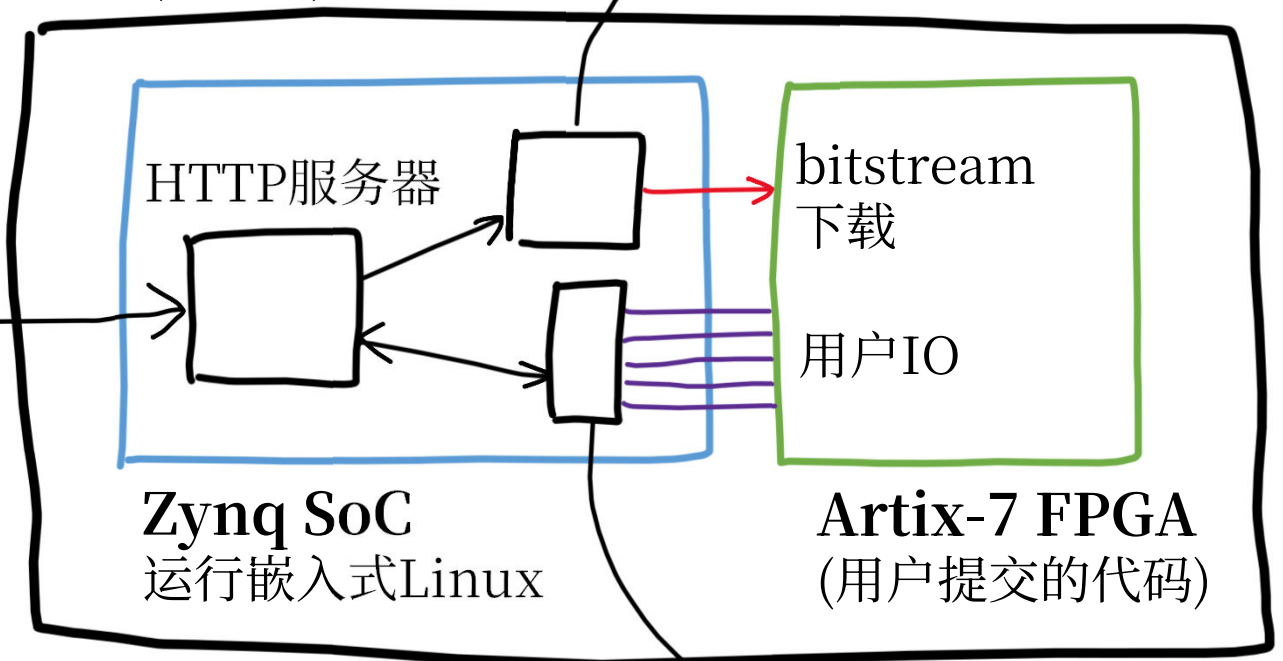
fpgaol.ustc.edu.cn



网页端



板卡(node)



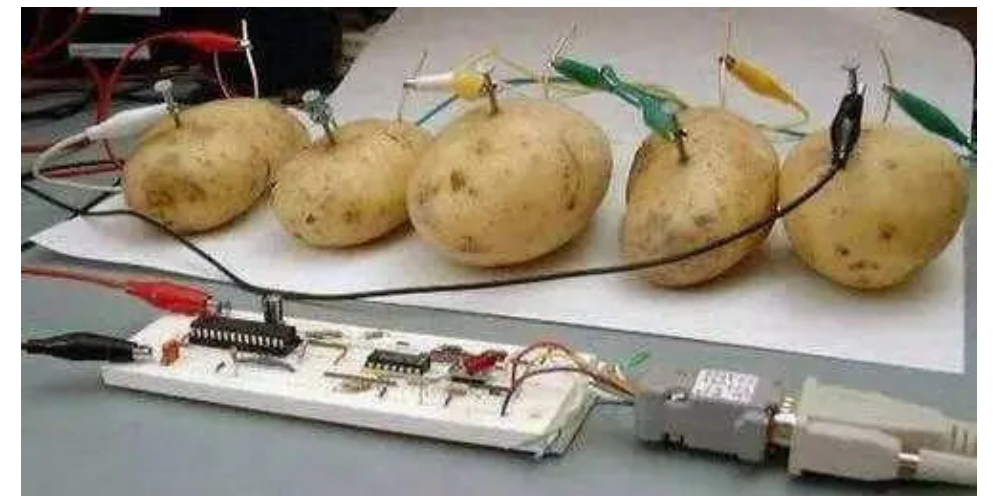
I/O监控和写入

根据网页交互，控制和读取对应的IO线

为何需要FPGAOL?

- 已经工作的 FPGAOL 系统——**好处在哪?**
 - 通过FPGA板卡集群解决了上述各种问题
 - 通过Ethernet连接板卡、服务器等，成熟可靠
 - Nginx 只做反向代理(revproxy)，降低服务器的负担
 - 具有足够的FPGA资源(35k LUT)和DDR3

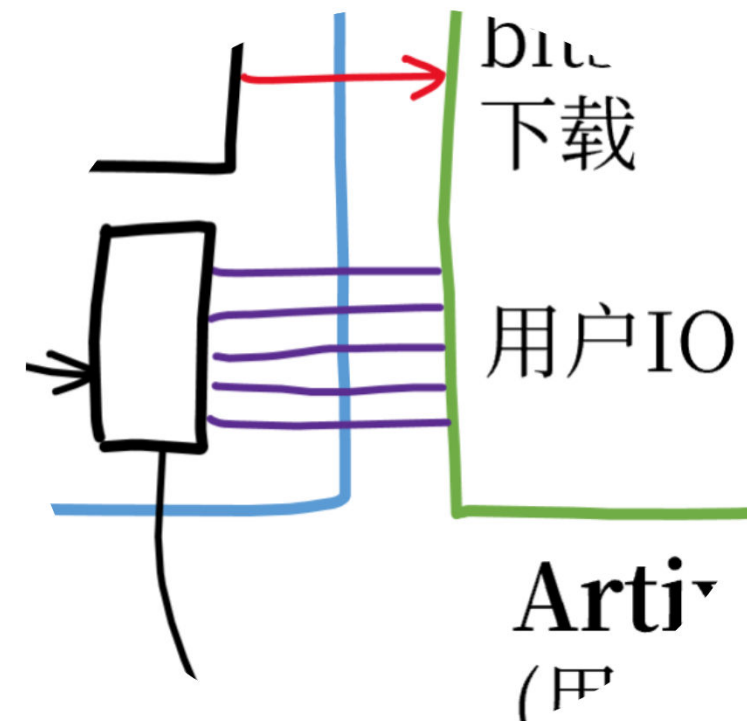
👉 图：FPGAOL运行在一台i3 CPU，6GB内存的主机上
这何尝不是一种土豆服务器



为何需要FPGAOL?

- 已经工作的 FPGAOL 系统——问题又在哪?
 - 最大输入输出端口数量决定于**物理连线**（Zynq 和 Artix FPGA 之间）
至多不超过100条线，可能不到这个数量

如果需要模块化测试（例如一个ALU，输入输出一共不止100位宽），或者需要debug的时候查看FPGA内部的信号，100条信号线的限制就会成为一个阻碍。



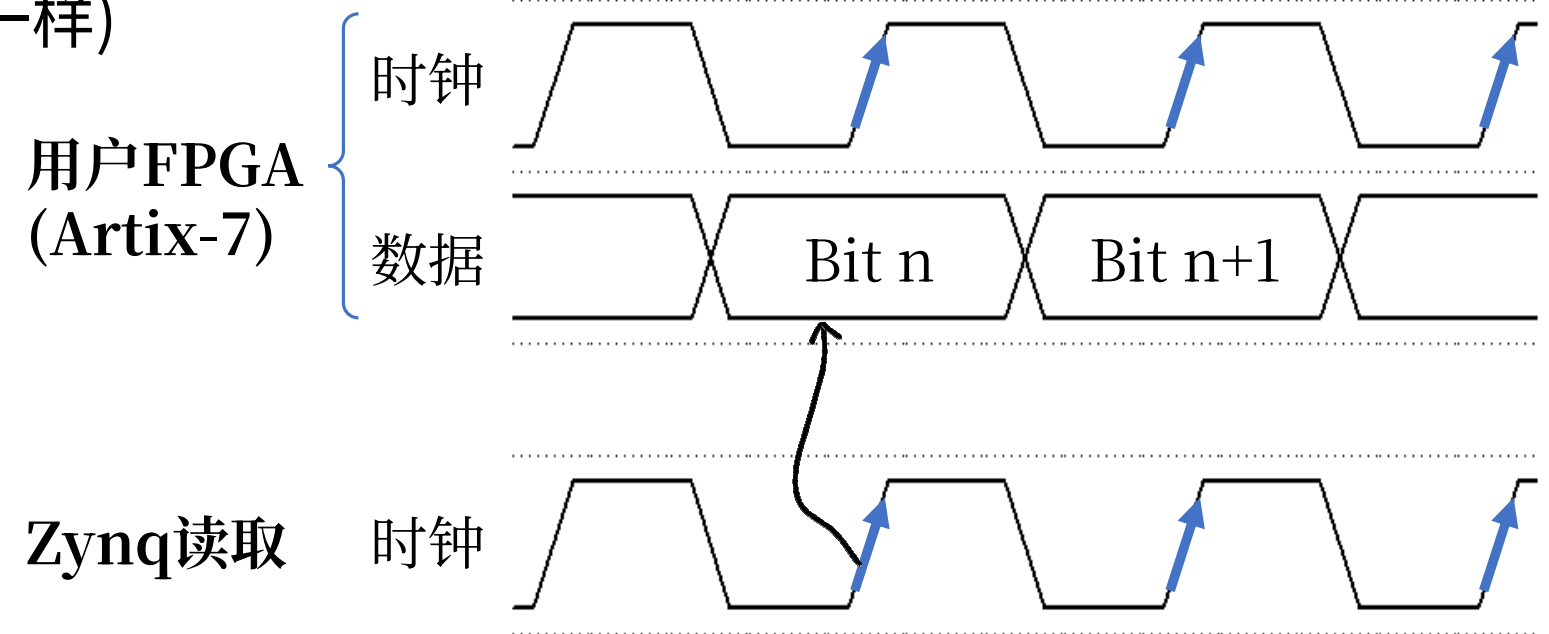
为何需要FPGAOL?

- 已经工作的 FPGAOL 系统——问题又在哪?

- 非相同时钟域(clock domain)内的采样

造成非常严重的采样率降低和潜在的逻辑冲突

(这里是展示了输出, 输入情况一样)



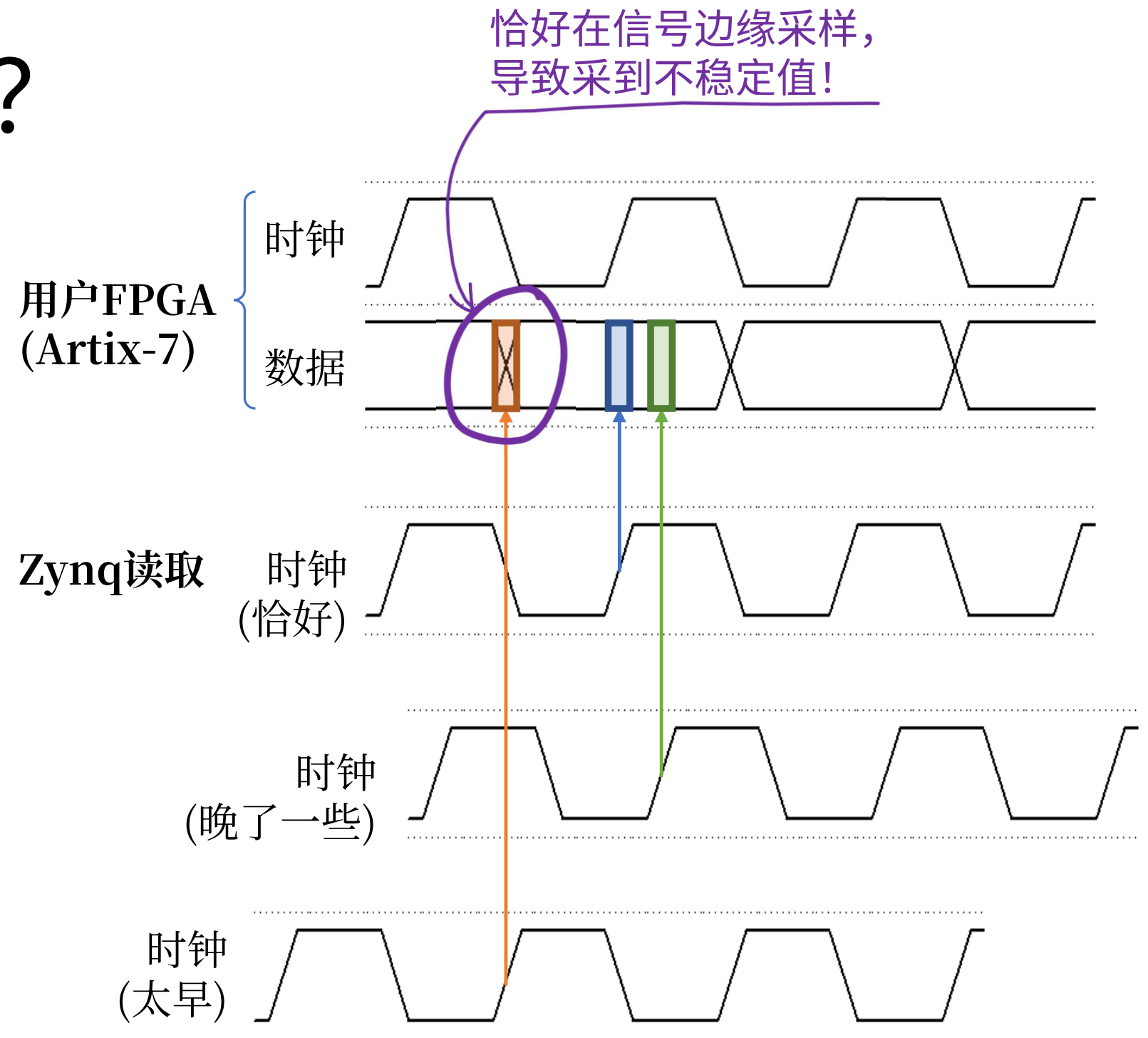
在每个时钟上升沿读入(采样)

为何需要FPGAOL?

- 已经工作的 FPGAOL 系统——问题又在哪?

- 非相同时钟域(clock domain)内的采样

用户FPGA和Zynq读取的节奏(时钟) **不一定对齐**, 是**随机**的



为何需要FPGAOL?

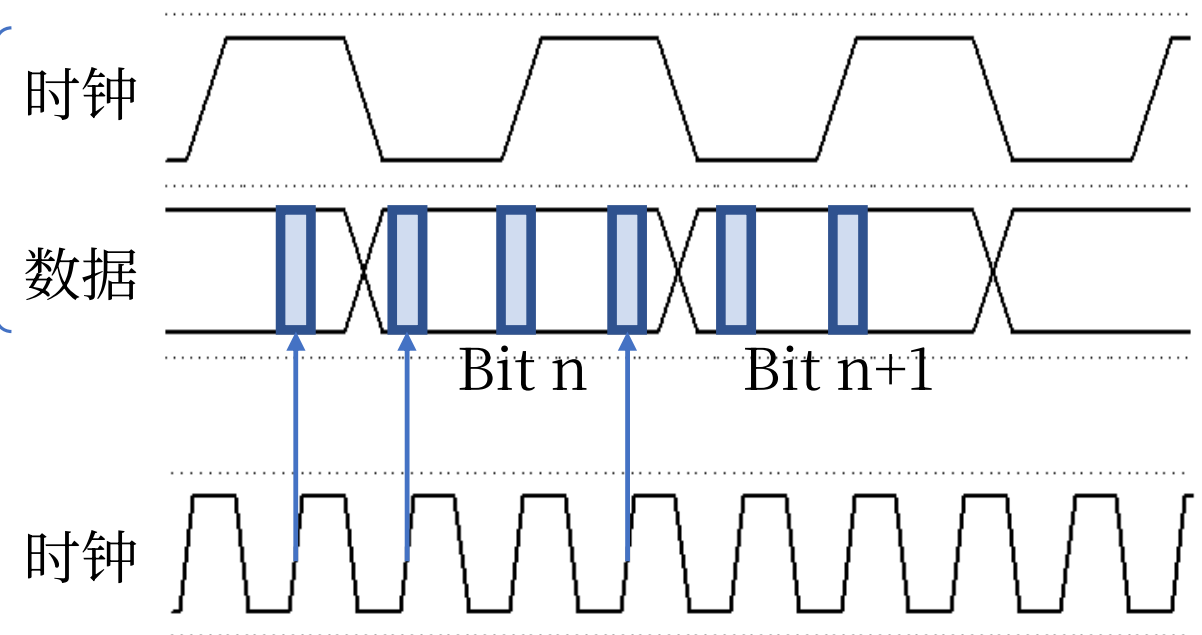
- 已经工作的 FPGAOL 系统——问题又在哪?

- 非相同时钟域(clock domain)内的采样

walkaround: 高频采样
显著降低了采样速度(100kHz?)
只能用于和用户视觉交互, 无法做精确的采样

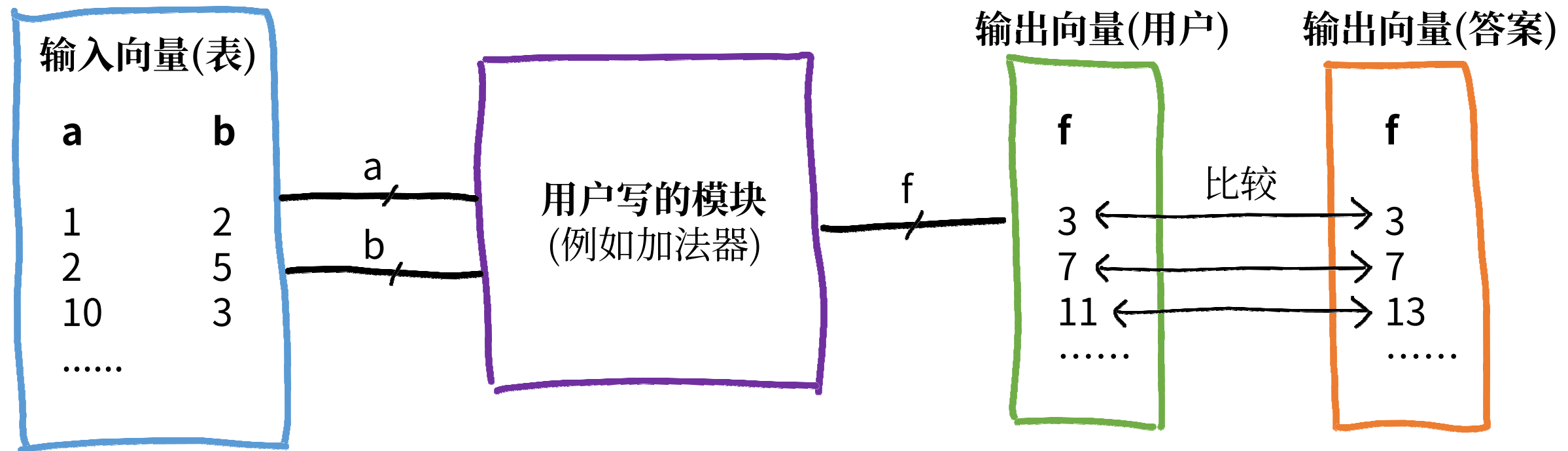
用户FPGA
(Artix-7)

Zynq读取



为何需要FPGAOL?

- 已经工作的 FPGAOL 系统——问题又在哪?
 - 无法做**模块/自动化测试**，仍然是某种“人工观察”



图：自动化测试的一种实现

2

设计与实现 EtherFPGA架构



EtherFPGA架构

- 需要解决上面列出的各种问题
 - 新的架构，我们称之为**EtherFPGA**

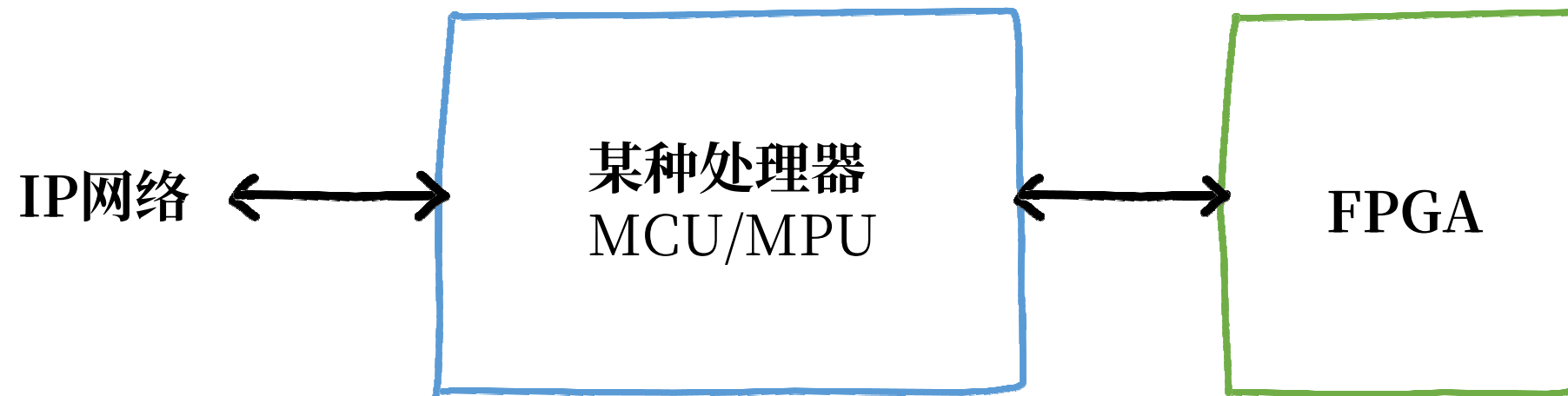
Ether/FPGA

Ethernet
从以太网启动和工作

硬件目标是**FPGA**

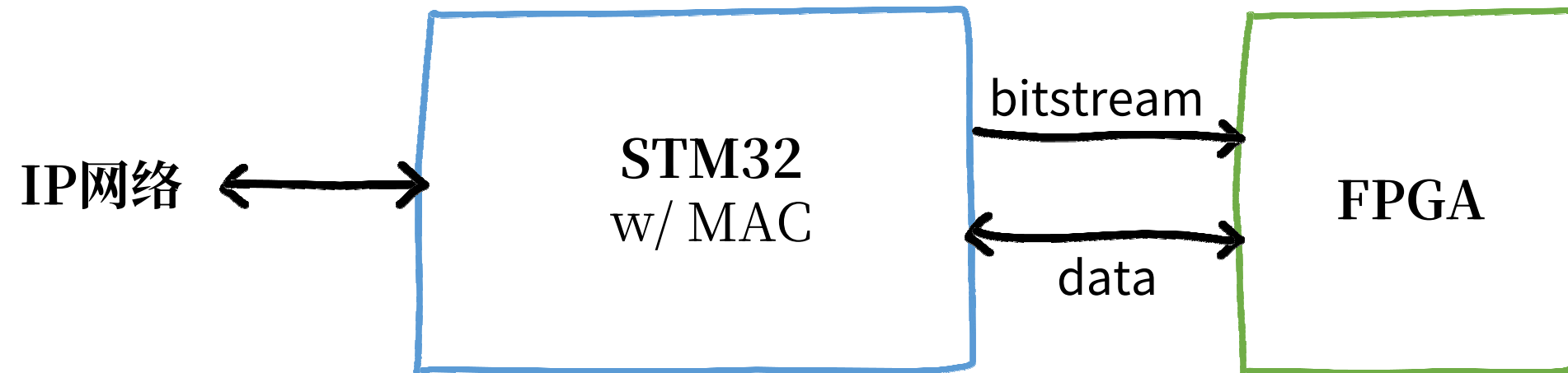
EtherFPGA架构

- EtherFPGA
 - 沿用了**处理器+FPGA**的架构
 - 同样采用**网络通信**



EtherFPGA架构

- EtherFPGA
 - **处理器**：为了简单，使用**STM32**(Cortex-M3内核)，含有MAC，以便与网络通信，有成熟的[arm-non-eabi-gcc](#)和[libopenm3](#)支持；
 - 与FPGA之间至少需要两个流：**配置流**bitstream和**用户数据流**data



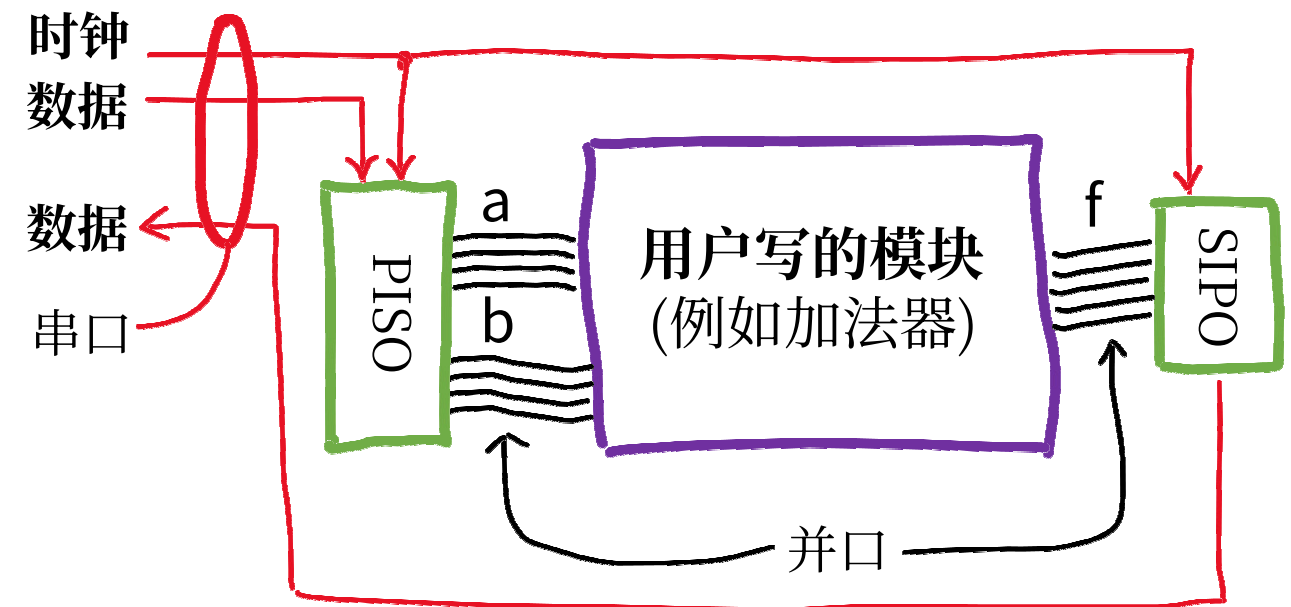
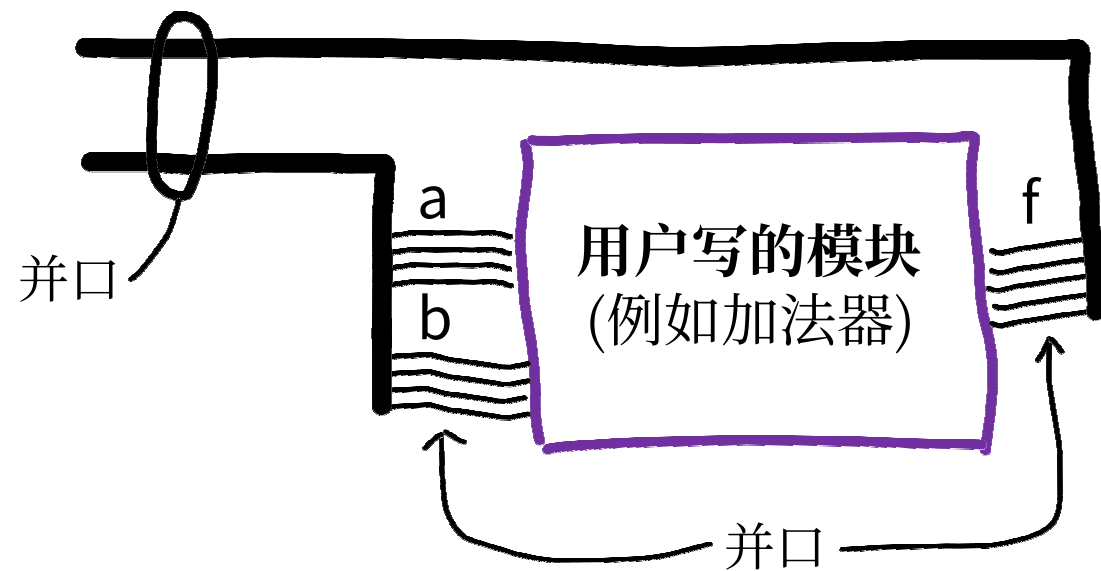
EtherFPGA架构

- **data流**如果是**异步并口**，则会带来前述的各种问题(采样相关)
- 因此，应该使用**同步(高速)串口**！
- 解决了两个问题
 - 采样是必然发生在同步时钟边沿上
 - 不受到并口大小的限制



EtherFPGA架构

- **同步(高速)串口!**
- 对用户模块(不改变输入输出的定义)的控制(输入/输出)必然发生在FPGA内部。
- FPGA内部需要有一个模块，完成串(Serial)-并(Parallel)转换

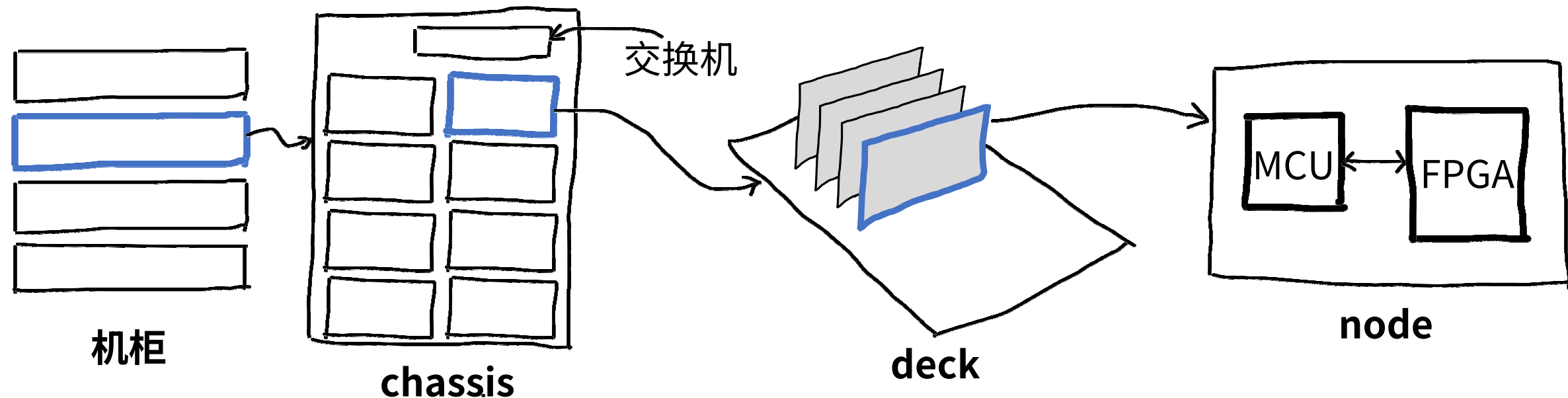


EtherFPGA架构

- 除此之外？

一个笑话：旧FPGAOL的特性是，如果需要写入固件或者修改配置，就需要把数十个节点卡的TF卡拔出来，插在读卡器里面写入然后装回去…

- 我们需要良好的**可维护性**！



EtherFPGA架构

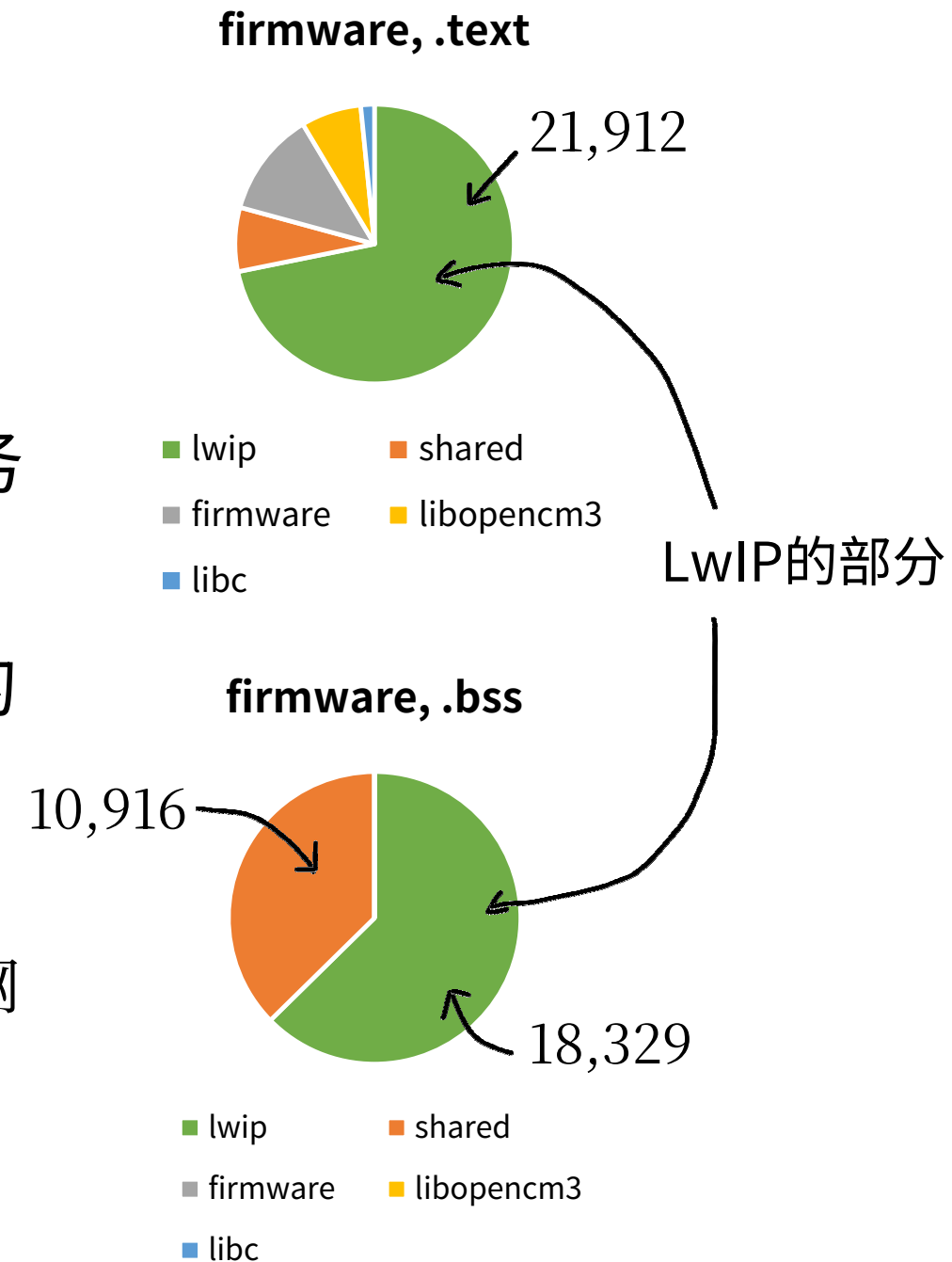
- 可维护性?

- 需要一个**PXE**类似物——**从网络启动**，方便从服务器固件更新。

- **in-memory execution**，在SRAM里面运行主要的业务程序。

★ Flash内的bootloader需要访问网络(下载 app image)，因此已经包含了**LwIP** (巨大，~20kB + ~20kB)，即所谓“网络驱动”，app可以复用，极大降低app image的大小。

-Wl,--just-symbols=\$(FIRMWARE_ELF)



EtherFPGA架构

- 可维护性?

- 需要一个PXE类似物——**从网络启动**，方便从服务器固件更新。
- SRAM内更新是容易的，那么Flash如何更新——**左脚踩右脚?**
- Flash需要两份(A-side, B-side)，防止当前更新的这一份炸了之后就无法启动，由两个配置位确定(**side, failsafe**)

A or B 禁止下次更新



EtherFPGA架构

- 可维护性?
 - 另一方面是，如何知道板卡的位置：物理位置？网络位置？
 - 八元组(chassis, deck, slot, fw_ver, app_ver, uniqid, ip, mac)
 - 盘坏了得更换，节点卡也是一样
 - 需要在机箱中识别板卡的插槽位置



3

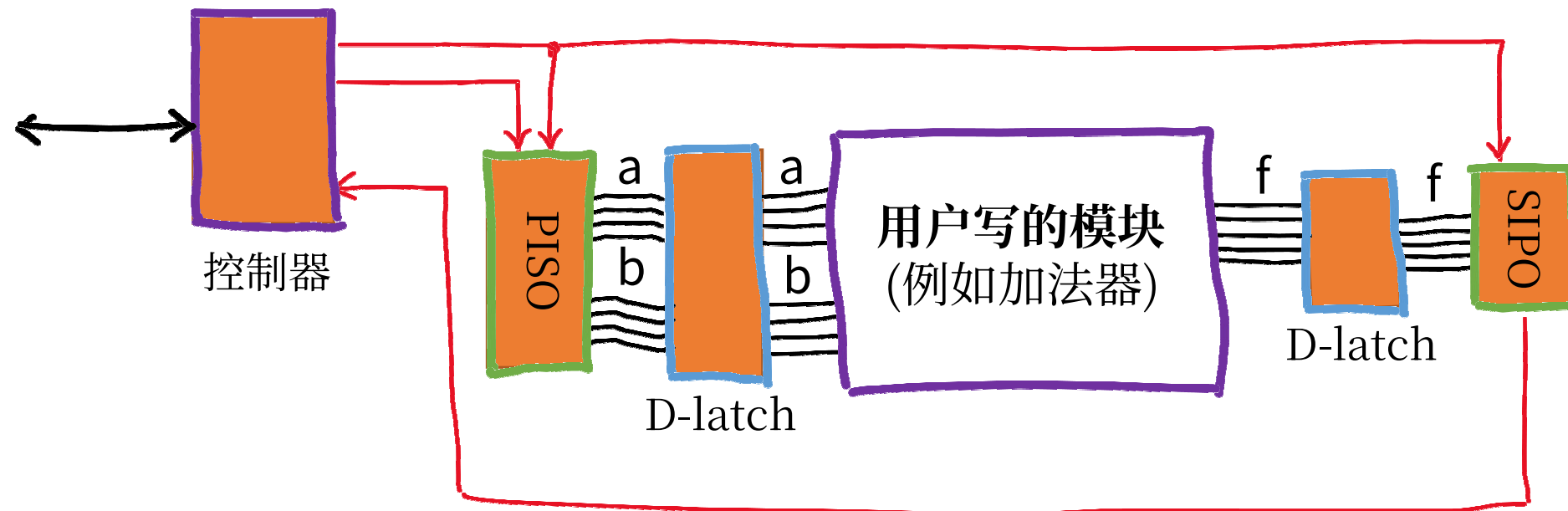
我们的FPGA确无问题！

——开源FPGA工具链的现状



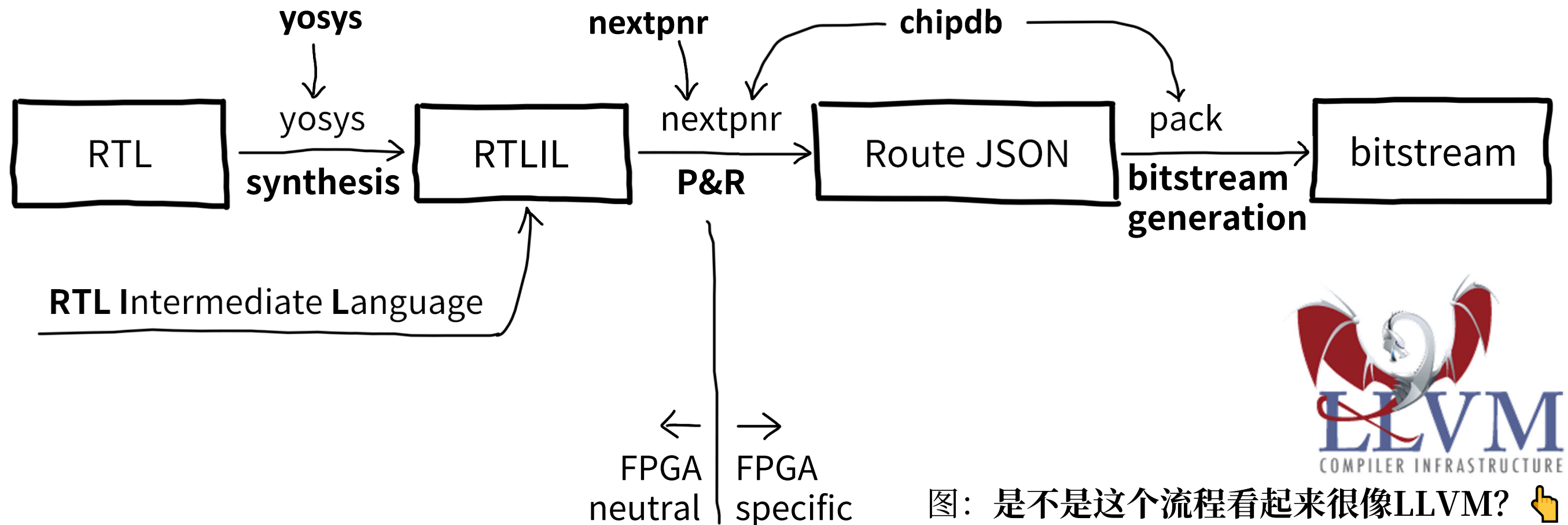
开源FPGA工具链

- 为何需要开源FPGA工具链?
- FPGA难以增量编译，而有一些模块不是用户写的（），需要在服务器上完成编译(综合、布线等)。——许可证限制，vendor的商业EDA无法运行，并且EDA软件价格也非常高，还可能受到出口管控。



开源FPGA工具链

- **yosys + nextpnr + (FPGA特定的某种chipdb) + openFPGALoader**



开源FPGA工具链

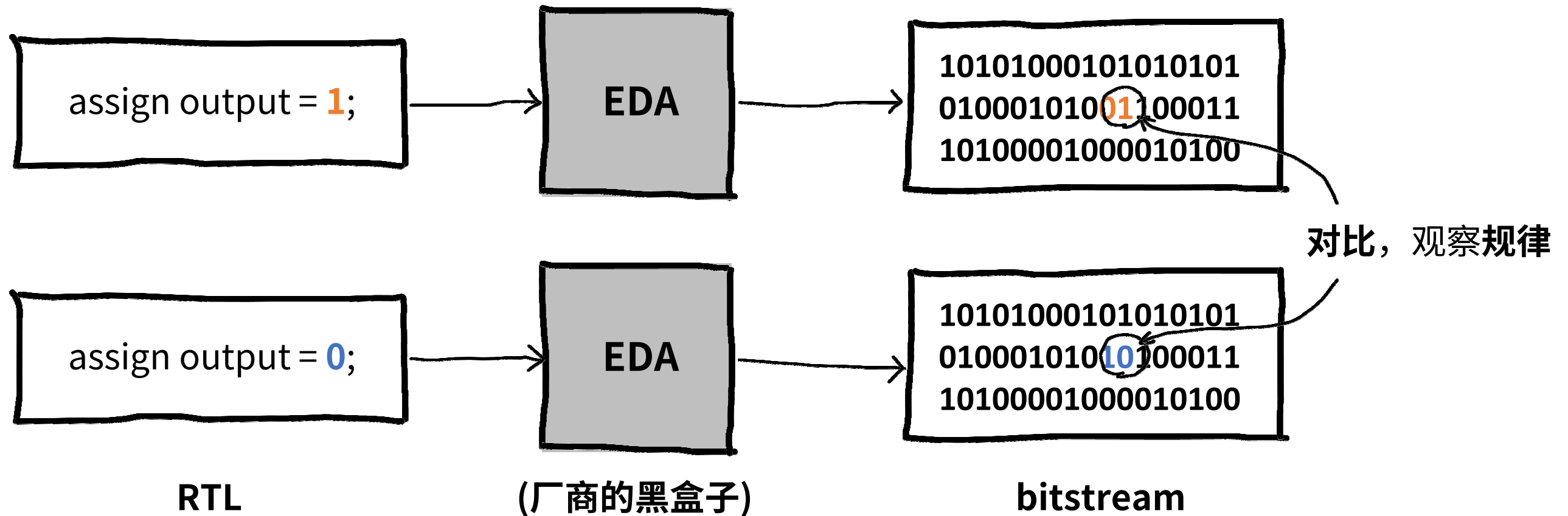
• **yosys** + **nextpnr** + (FPGA特定的某种**chipdb**) + **openFPGALoader**

综合工具 (造一个大象) 布线工具 (设计如何将大象放进冰箱) 芯片数据 (冰箱的图纸, 描述冰箱是什么样子) 刷写工具 (把大象真正放进冰箱)

FPGA&厂家	yosys	nextpnr	chipdb	openFPGALoader
Lattice iCE40	yosys	nextpnr-ice40	icestorm	openFPGALoader
Lattice ECP5		...	trellis	
Lattice Nexus		...	oxide	
Gowin LittleBee & Aurora		nextpnr-himbaechel	apicula	
Xilinx 7-series		...	openxc7	
...		...	...	

开源FPGA工具链

- 开源FPGA工具链是怎么来的? ——fuzzing工程, 靠发现规律



开源FPGA工具链

- FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了**这条路**？
- 各种私有化、商业化的非标结构，导致，除了 Verilog 几乎没有什么东西是靠谱中性的（其实 Verilog 本身也好不到哪里去）。
- 从工业系统发展和教育的角度来看，延续一代代设计人员的“**集体智慧**”是至关重要的。然而，硬件领域高度私有化的工具链正在极大地破坏这一点。
- 如果我们不能找到一种载体来传承这些智慧，工程师们的智力付出就会变得像现在 LLM（大语言模型）生成的代码一样，沦为按年(或者产品系列)计算的一次性消耗品。

开源FPGA工具链

- FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了**这条路**？
- 集成电路（IC）设计领域，情况也是如此吗？很多设计人员对 Cadence 等 EDA 软件颇有微词。
- 软件的数据形式完全无法被现代版本管理工具所接纳。git diff 往往只能变成对一个巨大二进制文件的粗暴覆盖。
- 极其恐怖的系统资源消耗——动辄数 GB 甚至上百 GB 的仿真数据，我们甚至根本无从知晓黑盒里面具体发生了什么。
- 设计人员的智力付出，必须依赖这样一个商业闭源且私有的“载体”才能可读或运作。

开源FPGA工具链

- FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了**这条路**？
- 集成电路设计是因为极其昂贵的流片成本，导致其生态无法被完全开放，那么 FPGA 呢？
- 本质上，FPGA 设计的失败成本接近于零，它就像是一种针对特殊硬件的二进制软件，和现代 CPU 里面的 microcode 是同一类东西。但在这种零成本试错的背景下，开源和标准化的进展依然举步维艰：目前在开源工具链支持上，Lattice 的器件相对好用，Gowin 有一些可用，Xilinx 还处于试验阶段，而 Altera 则几乎完全不可用……实在糟糕！
- 软件行业之所以能蓬勃壮大，核心原因在于它使用了人可读的格式（源代码）来记载人类的智力产出，并且借助 GitHub 这样的平台，让技术资料能够以极低甚至零成本传播。尊重人类智力产出的第一步，就是确保它具有**足够长的寿命和独立性**。

开源FPGA工具链

- FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了**这条路**？
- 关于 Fuzzing 工程，在一定程度上它的方向是对的，但是现实是骨感的。现代 FPGA 内部的状态空间大得惊人，而且无法观测。一个 Gowin 器件两个细微的版本（silicon revision）变化，就能导致 nextpnr 生成完全无法使用的 Bitstream。天知道内部究竟是哪出了问题，或许只有厂家知道，甚至可能连厂家自己都搞不明白（这也催生了像 yosys 这样的开源综合工具）。
- 这个领域从 1980 年代开发出 XC2064 至今，仅过去了不到 50 年，却和同时期（甚至更早）诞生的 C 语言走向了两个截然不同的极端——只要有硬件的地方，就能移植 C 语言的执行环境，清晰且普适。而 **FPGA** 彻底丢掉了其前身 CPLD 和 PLD 时代的清晰透明，选择**退化**成一个完全“**黑盒化**”的模糊环境——我们不知道里面具体是什么，只知道它恰好能运行。

开源FPGA工具链

- FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了**这条路**？
- 当我们从 XC2064 详尽的文档和 Fuzzing 经验中走出来，面对现代 FPGA 时，却发现自己对它的内部结构一无所知。
- XC2064 的文档会清晰地指出：这是一个 CLB，那是一个连接单元。但现代器件（在不靠盲猜的情况下）连这些最基本的信息都没有——**我们无法获知，自己的代码究竟跑在怎样的硬件架构上。**
- 万一这个 FPGA 实际上只是一个多 I/O 的 CPU，里面跑着个 Verilator 呢？在 FPGA 里硬加载一个 CPU 软核（类似带有 Flash LUT 的 MachXO 系列），分配好引脚后当成 CPU 卖出去，本质上和把一个内部完全不可知的所谓“可编程逻辑”器件卖给用户，难道不是同一种行为吗？

开源FPGA工具链

- **FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了这条路？**
- 这就暴露了一个致命的大问题：我花钱买到了一个器件，但我并没有买到对它的“控制能力”。我对芯片的控制力，是需要厂商通过其私有工具链临时“授权”给我的。那么，我们这种行为究竟是在“购买” FPGA，还是在“租用” FPGA 的开发权？

开源FPGA工具链

- FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了**这条路**？
- FPGA和CPU，架构到底差别在哪？我们认为核心在**可组合(mixing)性**和**位置无关性**两点。

CPU
功能(指令)是**独立的**

```
add eax, ebx  
add ecx, 20h
```



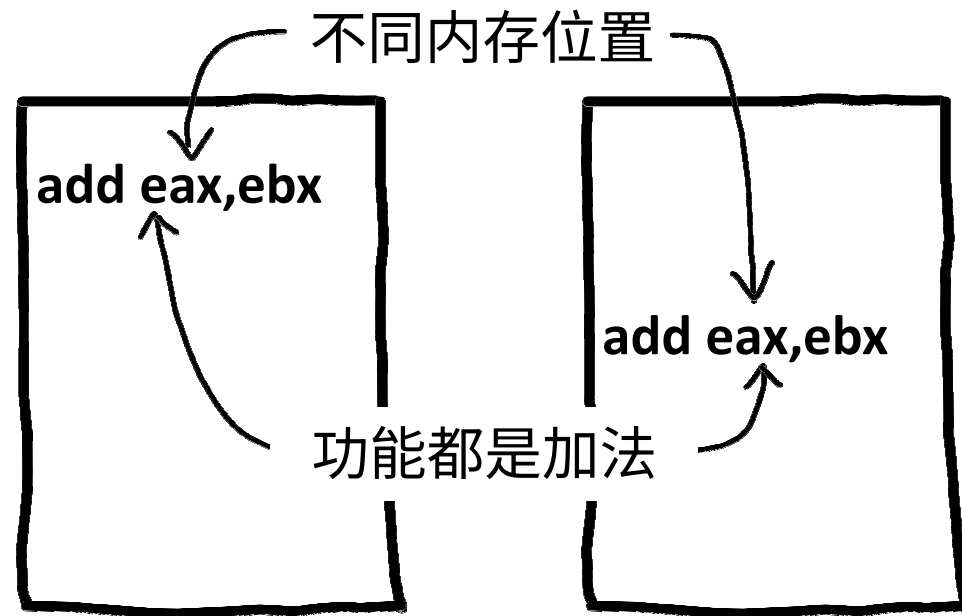
FPGA
功能是**混合的**



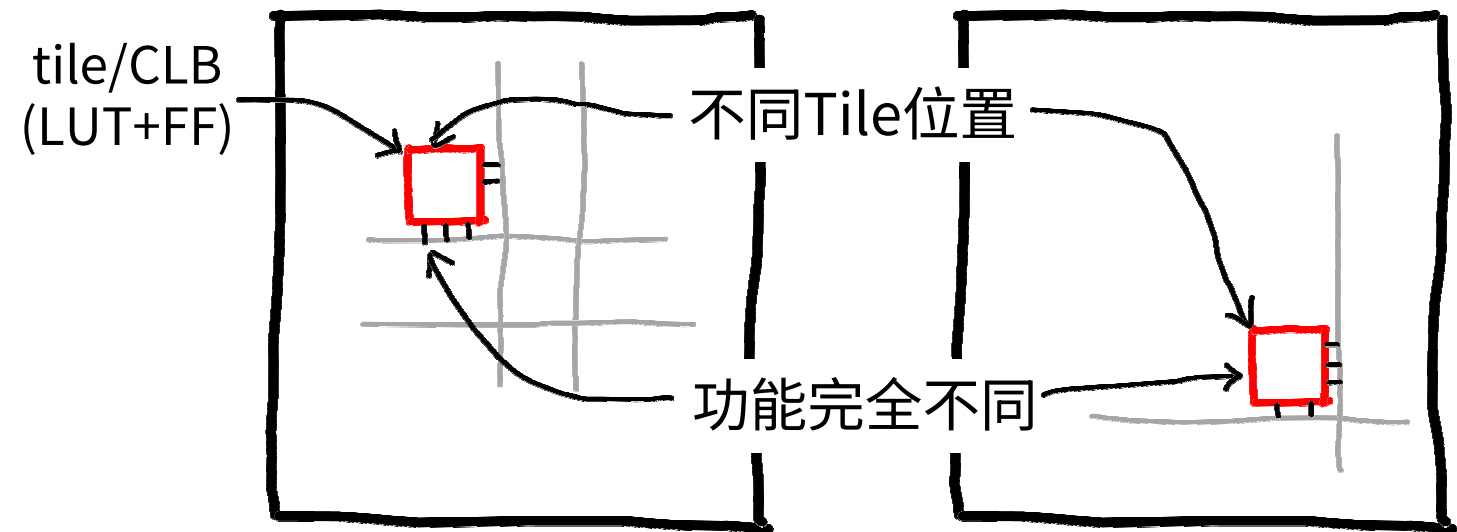
大毛线团子？

开源FPGA工具链

- FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了**这条路**?
- FPGA和CPU，架构到底差别在哪？我们认为核心在可组合(mixing)性和**位置无关性**两点。



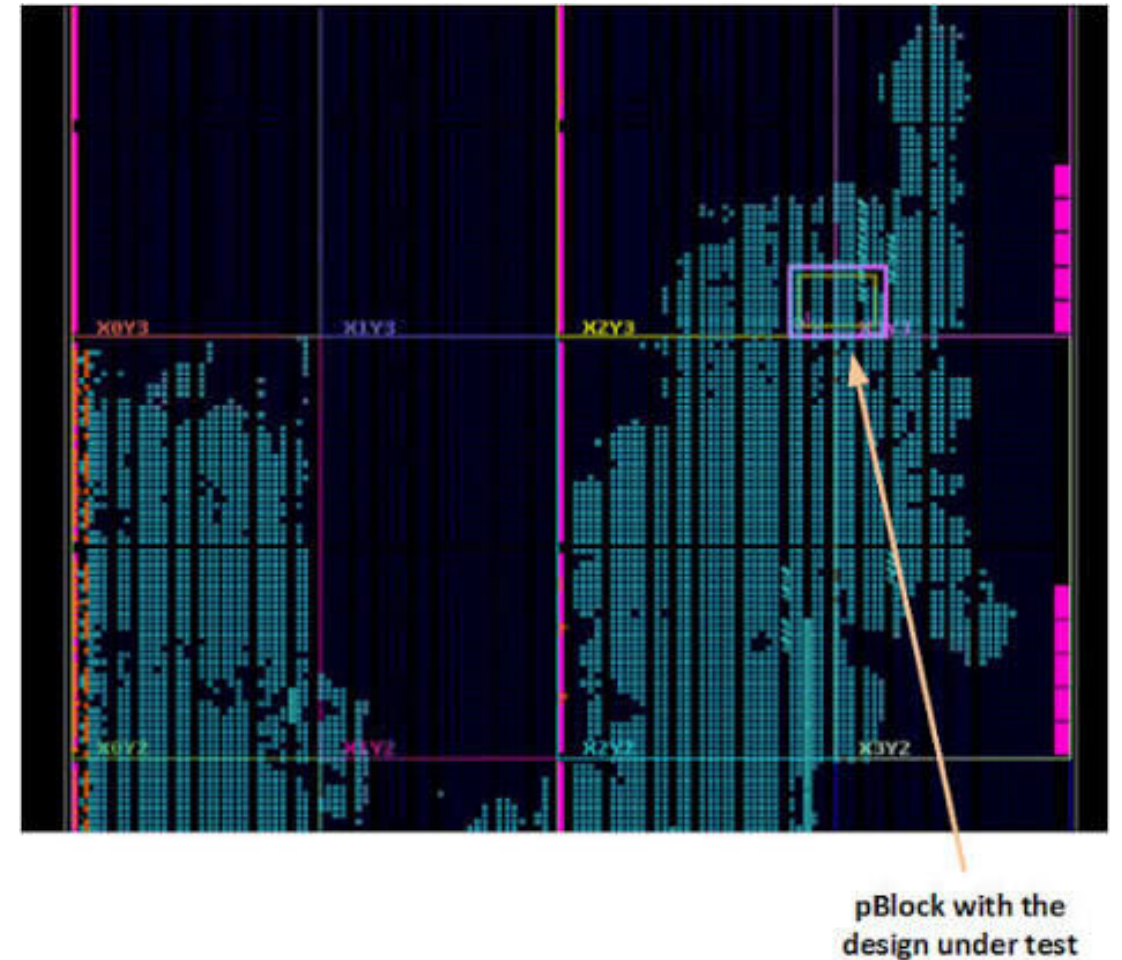
CPU (位置无关)



FPGA (位置有关)

开源FPGA工具链

- FPGA 简直是现代硬件生态系统里的一个巨大坑洞。但为何FPGA走上了**这条路**？
- FPGA和CPU，架构到底差别在哪？我们认为核心在可组合(mixing)性和位置无关性两点。
- 这直接导致了
 - 困难的增量综合（虽然我们有 partial reconfiguration）
 - 难以分割的设计——所有的设计被涂抹在 fabric 上
 - 时序控制——局部的小变化也可能导致全局的大改变
 - e.g. 被综合成LUT+FF的状态机和BRAM-based的状态机



图：Xilinx FloorPlanner展示一个FPGA的活跃CLBs👉

开源FPGA工具链

- 题外话
- Yosys还提供了形式化验证的工具，**SymbiYosys(sby)**
- 验证一个硬件电路(RTL设计)是否具有符合我们定义的性质?
 - **a_oflow: assert (count <= MAX_DATA);** ——计数不会超过最大值
 - **a_counts: assert (count == 0**
 - || count == \$past(count)**
 - || count == \$past(count) + 1**
 - || count == \$past(count) - 1);** ——计数每次是加一，减一或者不变(0是初始值)

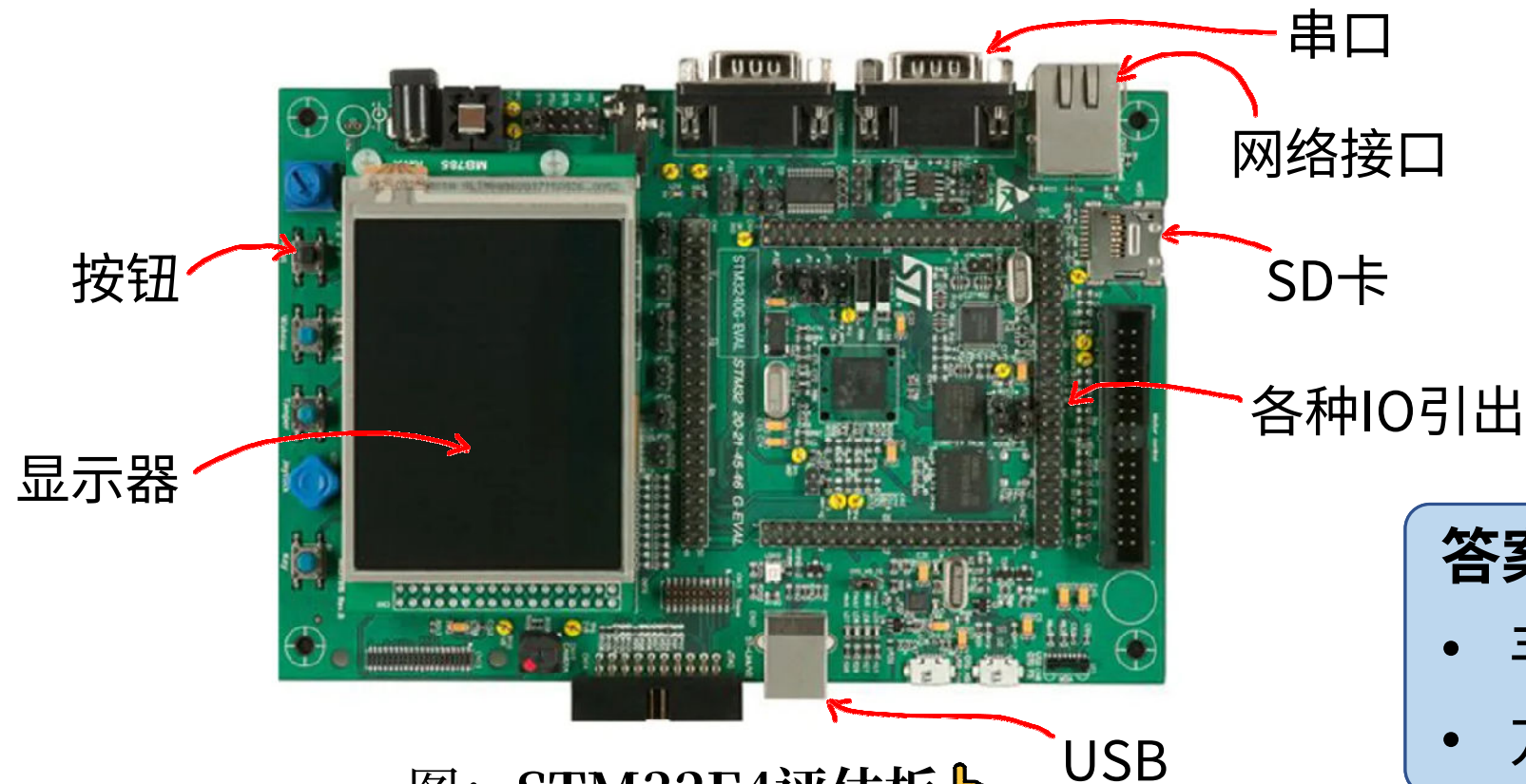
4

教学 or 科研，
差别究竟在哪？



教学 or 科研

- FPGA 加速卡早已有之，但是为何没人拿来做教学？
- 答案可以从**教学上常用的设备**中找到



图：STM32F4评估板👉



图：Xilinx ALVEO FPGA加速卡👉

答案

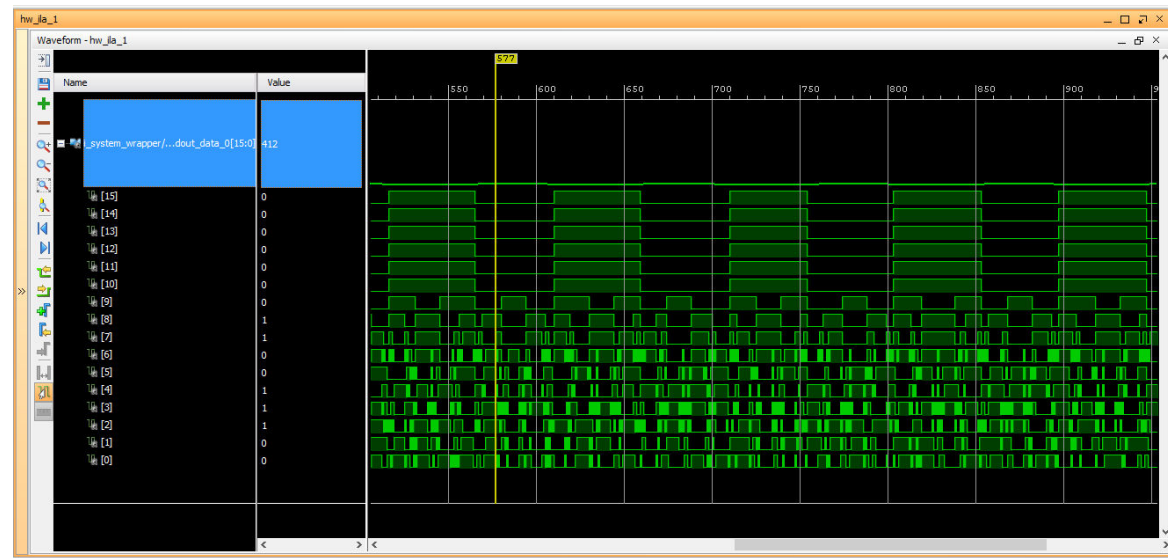
- 丰富且易用的外设/接口
- 方便写入程序和调试(JTAG/SWD)

教学 or 科研

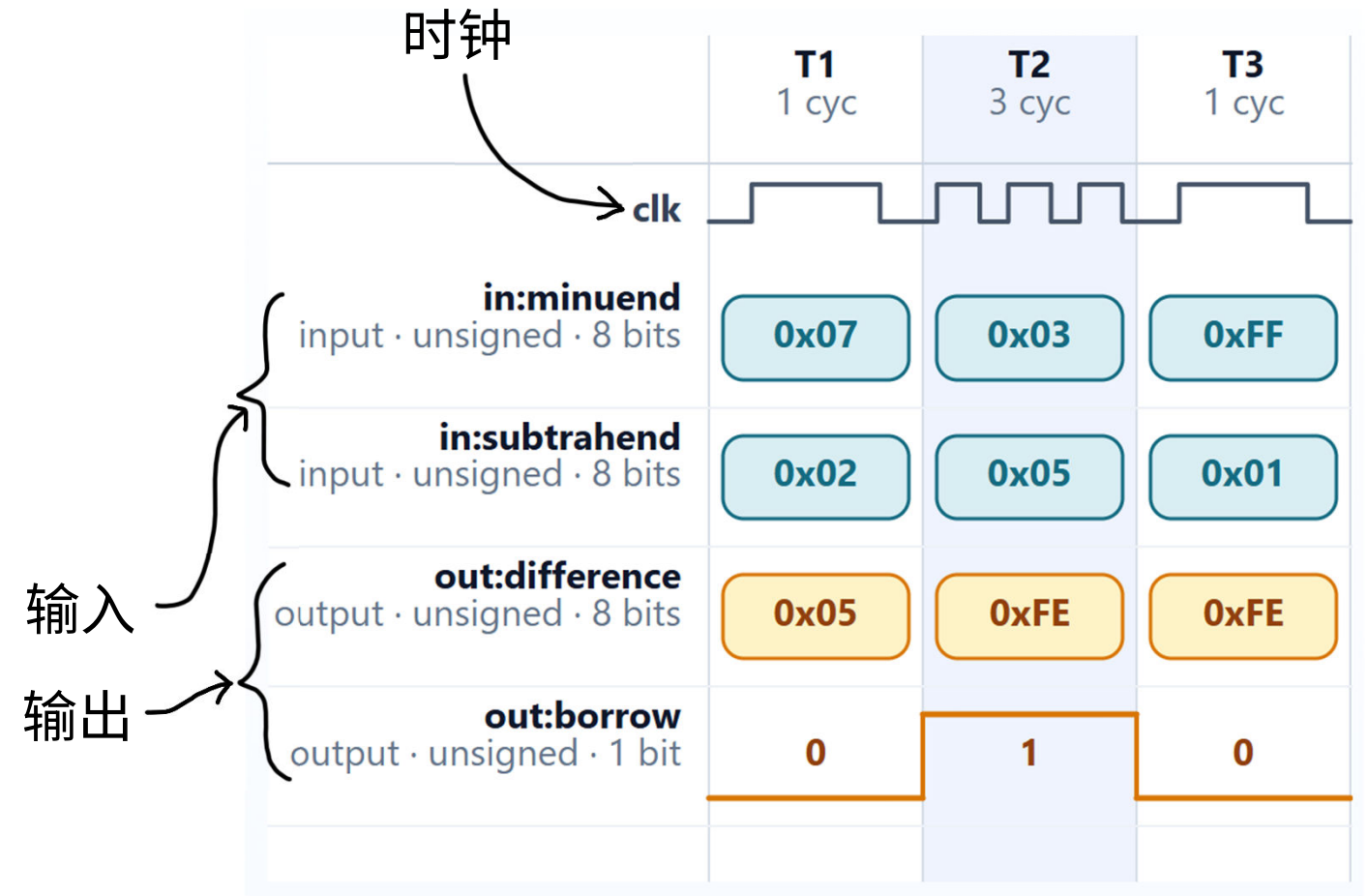
- FPGA 加速卡早已有之，但是**为何没人拿来做教学？**
- FPGA 加速卡适用于**科研/商业场景**，但是**教学场景**完全不同
 - 成熟的开发流程，熟悉FPGA编程的开发人员 → **学生不熟悉，需要调试和观察**
 - 机房固定的安装布置（服务器的slots）, FaaS → **开发板随身携带**
 - 不需要与物理世界交互（只需要PCIe传输数据） → **FPGA 嵌入式应用需要交互**
 - 依靠仿真等进行错误检查，无作业/实验任务等等 → **教师/助教方便检查作业实验**
- 需要一种既可以作为 **FaaS** 服务，又可以**离线使用**的开发板

教学 or 科研

- 学生不熟悉，需要调试和观察
- **Xilinx ILA**(Integrated Logic Analyzer) 类似物
- 需要输入注入、输出观察(即波形图)



图：Xilinx ILA的波形图(ILA无输入功能)👉



图：FPGAOL Next中的波形编辑器👉

5

结语



结语

- FPGAOL Next 的核心目标是
 - 用尽可能(目前确实是)完全开源和reproducible的软件
 - 验证 FPGA 的硬件设计及开发的可行性
 - 并就此开发用于教学(将来用于科研目的)的FaaS平台
 - 并在此期间对 FPGA 内部的设计细节进行研究

So again,
“**FaaS**” —— **FPGA as a Service**

结语

- 我们需要什么？

1/

稳定可复现的开发和构建环境

2/

社区的支持 & 对社区的贡献

3/

可靠的软硬件生产能力

4/

持续参与项目的学生团队

Thanks