

中期进度汇报

COD19GRP4

概览

▶ 成果

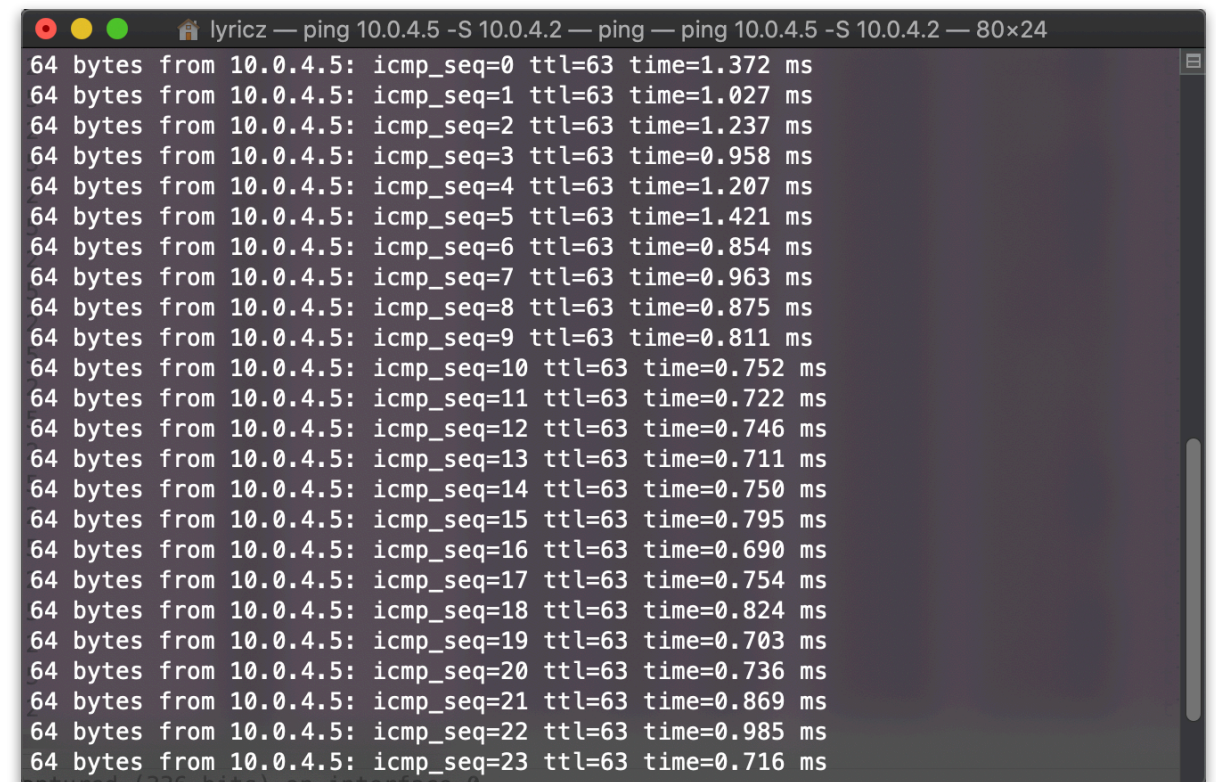
- ▶ 把 MAC & IP 写死，可以 4 个网口连接相互 ping 通
- ▶ （尚未连接 ARP 表和路由表）

▶ 简约的 ARP 表

- ▶ 线性算法

▶ 高效的路由表

- ▶ 32 拍 Trie 树(第 5 周)
- ▶ 1/2/4/8/16/32 步长可参数化多路 Trie 树(第 7 周)



```
lyricz — ping 10.0.4.5 -S 10.0.4.2 — ping — ping 10.0.4.5 -S 10.0.4.2 — 80x24
64 bytes from 10.0.4.5: icmp_seq=0 ttl=63 time=1.372 ms
64 bytes from 10.0.4.5: icmp_seq=1 ttl=63 time=1.027 ms
64 bytes from 10.0.4.5: icmp_seq=2 ttl=63 time=1.237 ms
64 bytes from 10.0.4.5: icmp_seq=3 ttl=63 time=0.958 ms
64 bytes from 10.0.4.5: icmp_seq=4 ttl=63 time=1.207 ms
64 bytes from 10.0.4.5: icmp_seq=5 ttl=63 time=1.421 ms
64 bytes from 10.0.4.5: icmp_seq=6 ttl=63 time=0.854 ms
64 bytes from 10.0.4.5: icmp_seq=7 ttl=63 time=0.963 ms
64 bytes from 10.0.4.5: icmp_seq=8 ttl=63 time=0.875 ms
64 bytes from 10.0.4.5: icmp_seq=9 ttl=63 time=0.811 ms
64 bytes from 10.0.4.5: icmp_seq=10 ttl=63 time=0.752 ms
64 bytes from 10.0.4.5: icmp_seq=11 ttl=63 time=0.722 ms
64 bytes from 10.0.4.5: icmp_seq=12 ttl=63 time=0.746 ms
64 bytes from 10.0.4.5: icmp_seq=13 ttl=63 time=0.711 ms
64 bytes from 10.0.4.5: icmp_seq=14 ttl=63 time=0.750 ms
64 bytes from 10.0.4.5: icmp_seq=15 ttl=63 time=0.795 ms
64 bytes from 10.0.4.5: icmp_seq=16 ttl=63 time=0.690 ms
64 bytes from 10.0.4.5: icmp_seq=17 ttl=63 time=0.754 ms
64 bytes from 10.0.4.5: icmp_seq=18 ttl=63 time=0.824 ms
64 bytes from 10.0.4.5: icmp_seq=19 ttl=63 time=0.703 ms
64 bytes from 10.0.4.5: icmp_seq=20 ttl=63 time=0.736 ms
64 bytes from 10.0.4.5: icmp_seq=21 ttl=63 time=0.869 ms
64 bytes from 10.0.4.5: icmp_seq=22 ttl=63 time=0.985 ms
64 bytes from 10.0.4.5: icmp_seq=23 ttl=63 time=0.716 ms
```

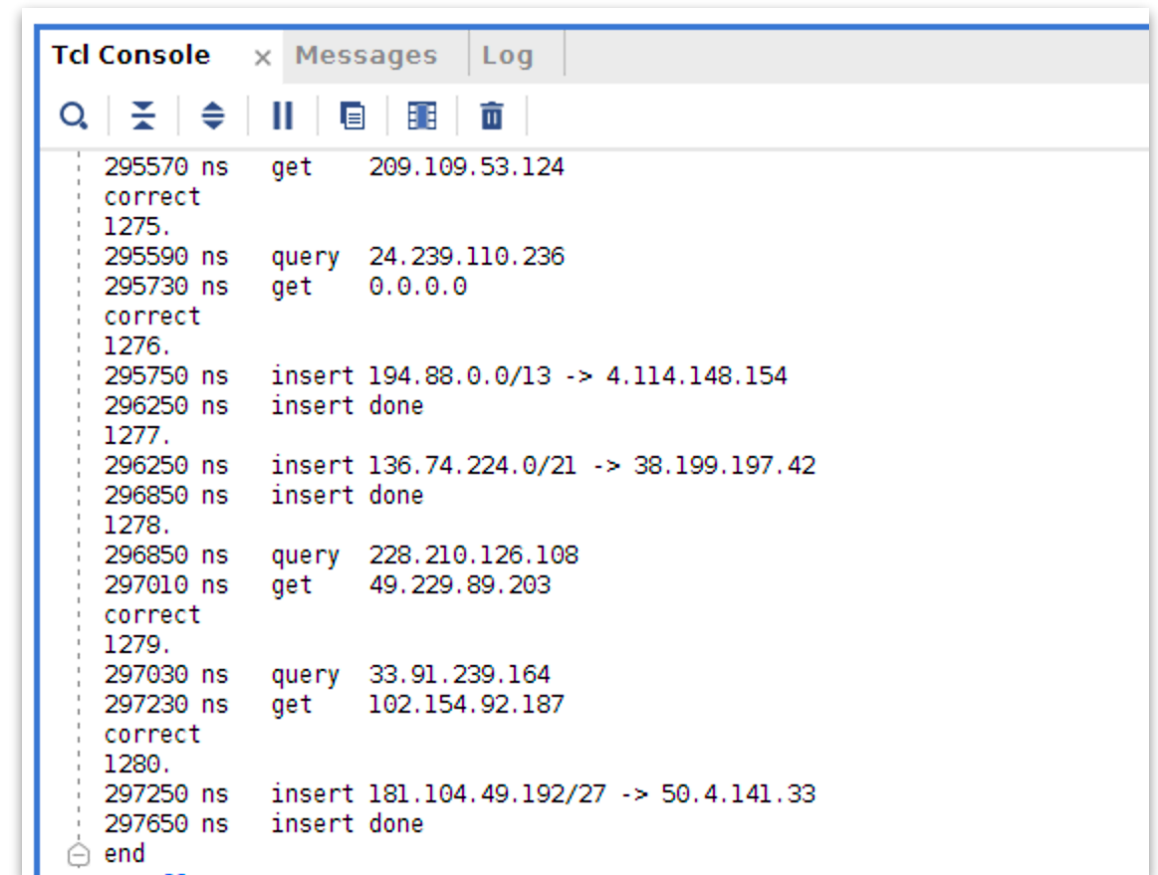
概览

▶ 智能的 Testbench

- ▶ Python 自动化生成 ARP/路由/以太网帧测试输入，可指定数量等参数
- ▶ Vivado 行为仿真可以显示每个测试输入预期结果 vs 实际结果

▶ 优美的 Debug 方式

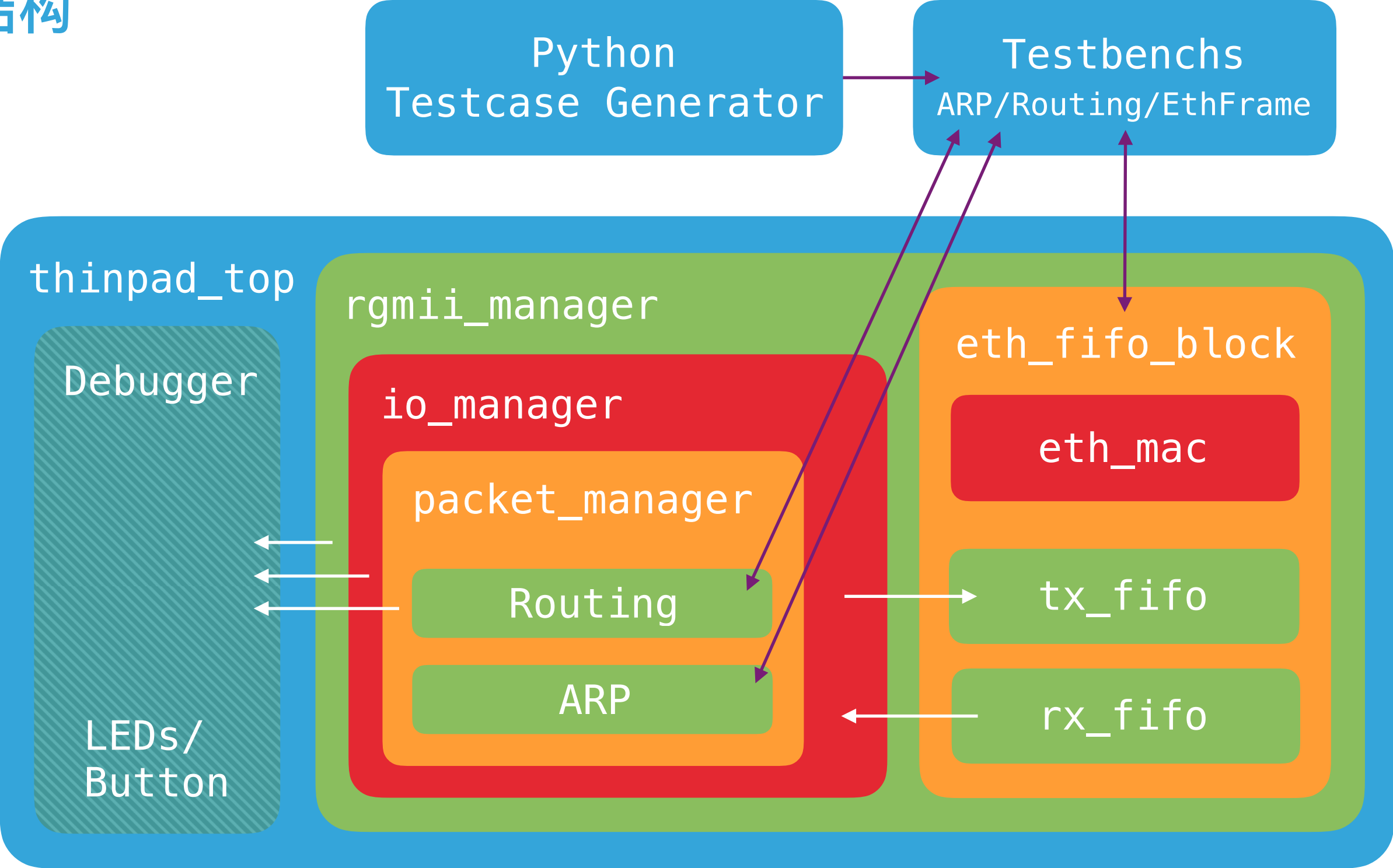
- ▶ LED/7 端数码管显示调试信息
- ▶ 通过硬件按钮控制调试
- ▶ 内嵌逻辑分析仪接到队列 RX 和 TX
- ▶ (基本) 规范的 git 版本控制
- ▶ 搭了一个简单的5级流水CPU框架
 - ▶ 还把 ALU/SRAM/UART 小作业写了



The screenshot shows a 'Tcl Console' window with a toolbar and a list of simulation messages. The messages are organized into columns: time, namespace, command, and result. The results show various network operations like 'get', 'query', 'insert', and 'insert done' with corresponding IP addresses and status messages like 'correct'.

Time	Namespace	Command	Result
295570	ns	get	209.109.53.124
			correct
			1275.
295590	ns	query	24.239.110.236
295730	ns	get	0.0.0.0
			correct
			1276.
295750	ns	insert	194.88.0.0/13 -> 4.114.148.154
296250	ns	insert	done
			1277.
296250	ns	insert	136.74.224.0/21 -> 38.199.197.42
296850	ns	insert	done
			1278.
296850	ns	query	228.210.126.108
297010	ns	get	49.229.89.203
			correct
			1279.
297030	ns	query	33.91.239.164
297230	ns	get	102.154.92.187
			correct
			1280.
297250	ns	insert	181.104.49.192/27 -> 50.4.141.33
297650	ns	insert	done
			end

结构



实操事项

- ▶ 对于 ping 请求 Windows 有防火墙，要先关了
- ▶ 如何使用有 bug 的 macOS Catalina 连接 Thinpad
 - ▶ 使用最新版的 Parallel Desktop **15**，把 Thinpad 分配给虚拟机
- ▶ 如何用一台电脑的两个网口测试
 - ▶ ping -S 或者 ping -I 指定网口
 - ▶ 或者一个网口分配给虚拟机
- ▶ 12V 电源线转 USB 淘宝有卖，再连一个 USB/RJ45 的 Hub 即可一根 Type-C 连接到电脑非常方便，熄灯也能调试

一些坑

- ▶ 夯实基础概念少走弯路（可能除了我们组大家都知道）
 - ▶ KSZ8795 是个交换机（一直以为它就是负责打 VLAN TAG）
 - ▶ 为了测试 ping 是不是通的，子网掩码可以设置成 255.255.255.255，不然直接会被交换机扔回去（交换机会学习到 MAC 和 Port 的映射）
- ▶ VLAN 从 1 开始编号不是 0
- ▶ ILA 监控两个 FIFO 的信号非常方便调试
 - ▶ Open Synthesis Design → Debug Set Up
 - ▶ 或者 IP Catalog 里面添加
- ▶ PLL 分频之后 locked 信号可以作为后面所有模块的复位
- ▶ 调自闭了问几位大哥或者出去换个脑子

更多坑

- ▶ 在 testbench 中，如何读取一个测例文件
 - ▶ 相对路径绝对路径都不好使，将 mem 文件加为 simulation source
 - ▶ \$fread 读会崩，要么报错要么读 0，可用 \$fscanf
- ▶ 寄存器可以用上电初始化 logic foo = 1，不用 rst
 - ▶ 遇到过手动 rst 没能重置一个库提供的模块，然后 gg
- ▶ 注意一下 implementation 之后有没有报 timing 的错
 - ▶ 时钟频率对于一些模块可能过高，导致上板子无法工作

后面打算

- ▶ 把ARP表和路由彻底连上
- ▶ 更快的ARP算法
- ▶ 改成流水
- ▶ 造CPU

DEMO

谢谢