npc 迭代数据与面积频率优化

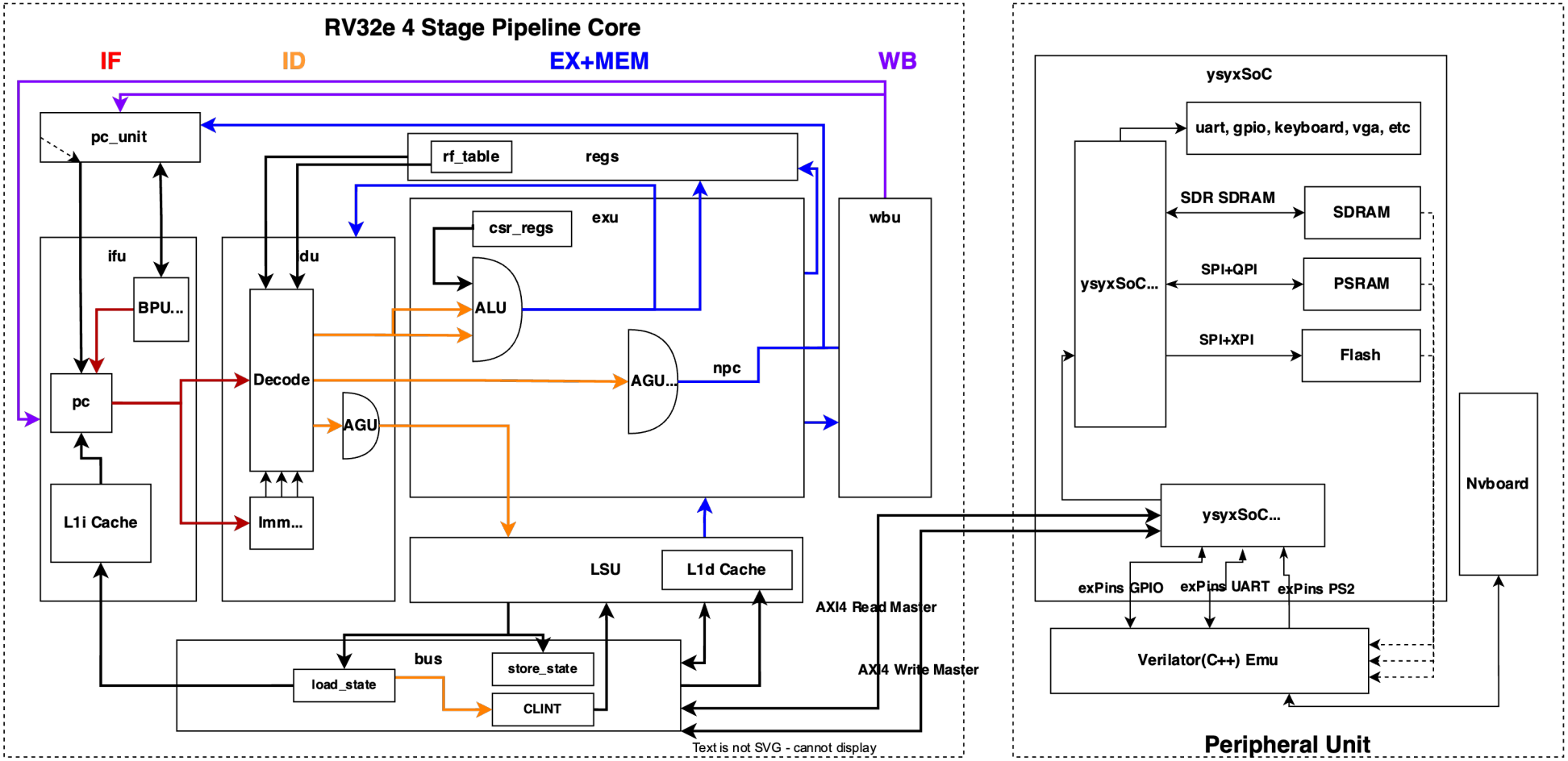
分享人：金宇

ysyx 学号：23060087

https://github.com/Kingfish404/ysyx-workbench

npc 当前状态介绍

- 4 级流水线 + L1i Cache + L1d Cache + 转发 + BPU + ysyxSoC



https://github.com/NJU-ProjectN/am-kernels

npc 迭代性能数据 | microbench of am-kernels

SoC Frequency: 100 MHz, IP Core Frequency: 500 MHz (for clint)
$ make perf: make ARCH=riscv32e-ysyxsoc NPCFLAGS='-b n' mainargs=test run

commit	comment	#Cycle	#Inst	IPC	Freq (MHz)	Area (um^2)
66b3344	perf	38309563	849339	0.022	466	23963
c1f2dc3	l1i:8x4B	26917653	848522	0.032	334	24760
dac9e4f	OPtiming	26917653	848522	0.032	532	23999
3e0f1ce	i8B,axi[1]	16970857	854729	0.050	560	26811
3e0f1ce	axi-delay	20720045	849533	0.041	560	26811
046eea8	l1i:4x8B	21867026	847917	0.039	566	22798
[2]	l1i:4x8B	22646833	848317	0.037	566	22798
569dd1c	add wbu[3]	23519169	848435	0.036	575	23553
f7dd997	pipeline	22312119	848260	0.038	545	25919
f32a4a1	ppa optim	21974627	848206	0.039	564	24034
72541a3	optim[4]	20575381	849051	0.041	532	24946
92e723a	simple bpu	21763542	849279	0.039	534	24671
2ead5f6	fence.i	21760489	849588	0.039	599	25014
88a5e66	ppa optim	21741693	849723	0.039	540	24889

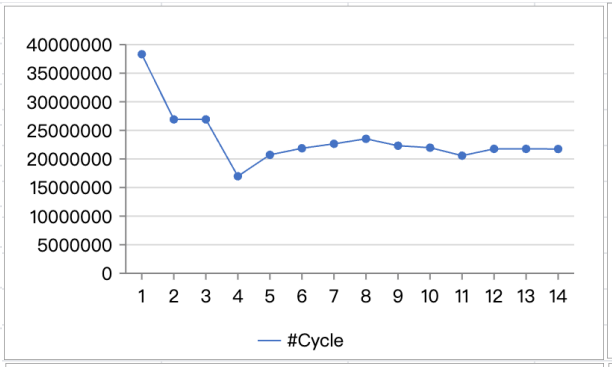
[1]: icache line size 修改为 8B，现在 icache size 为 8x8B，L1I Cache 的 SDRAM 部分修改为 arburst，一次传输 8B。
[2]: `046eea8e` 及之前的版本中，npc 的频率设置为 466 MHz，之后的版本中，npc 的频率设置为 500 MHz。
[3]: 从 3 周期 cpu 变为 4 周期 cpu，为之后的流水线设计做准备。
[4]: exu forward, fix pmu and axi store load separate

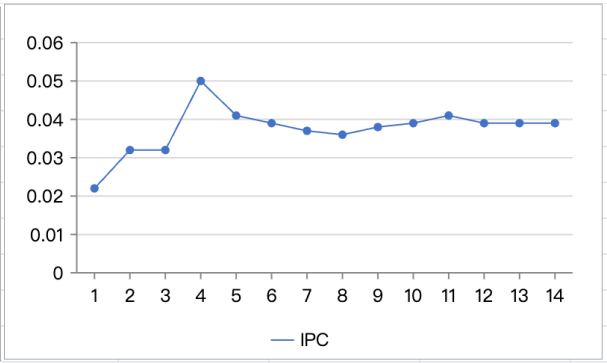
https://docs.qq.com/sheet/DRkdzQ0xsVFNvdEhF?tab=drxfto

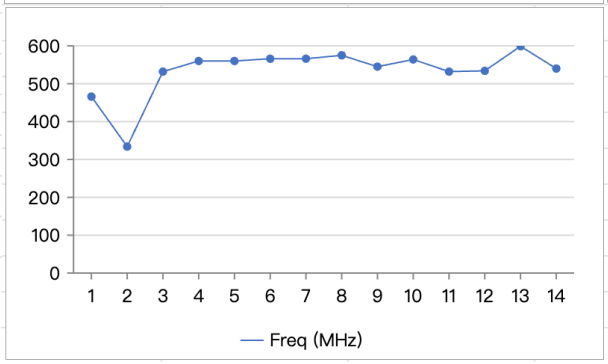
npc 迭代性能数据丨更可视化的版本

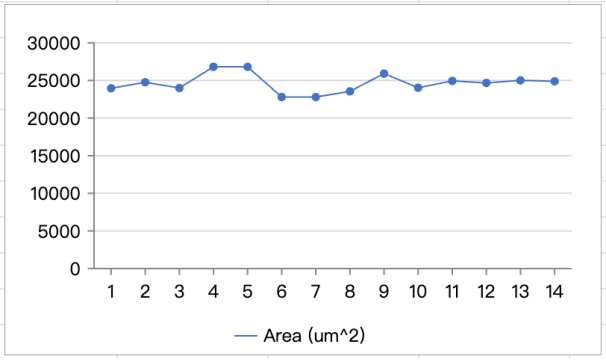
- 性能数据记录（ysyx学习记录子表）： https://docs.qq.com/sheet/DRkdzQ0xsVFNvdEhF?tab=drxfto
- 第2次迭代cache 加入后，IPC 提升巨大，第4次迭代增大 cache size 达成第二次大提升
- 第5次之后的迭代，microbench

	Tape Out Ready	ΔIPC（越大越好）	ΔFreq（越大越好）	ΔArea（越小越好）	comment
1	FALSE	-	-	-	perf
2	FALSE	45.45%	-28.33%	3.33%	I1i:8x4B
3	TRUE	0.00%	59.28%	-3.07%	OPtiming
4	FALSE	56.25%	5.26%	11.72%	i8B,axi[1]
5	FALSE	-18.00%	0.00%	0.00%	axi-delay
6	TRUE	-4.88%	1.07%	-14.97%	I1i:4x8B
7	TRUE	-5.13%	0.00%	0.00%	I1i:4x8B
8	TRUE	-2.70%	1.59%	3.31%	add wbu[3]
9	FALSE	5.56%	-5.22%	10.05%	pipeline
10	TRUE	2.63%	3.49%	-7.27%	ppa optim
11	TRUE	5.13%	-5.67%	3.79%	optim[4]
12	TRUE	-4.88%	0.38%	-1.10%	simple bpu
13	FALSE	0.00%	12.17%	1.39%	fence.i
14	TRUE	0.00%	-9.85%	-0.50%	ppa optim





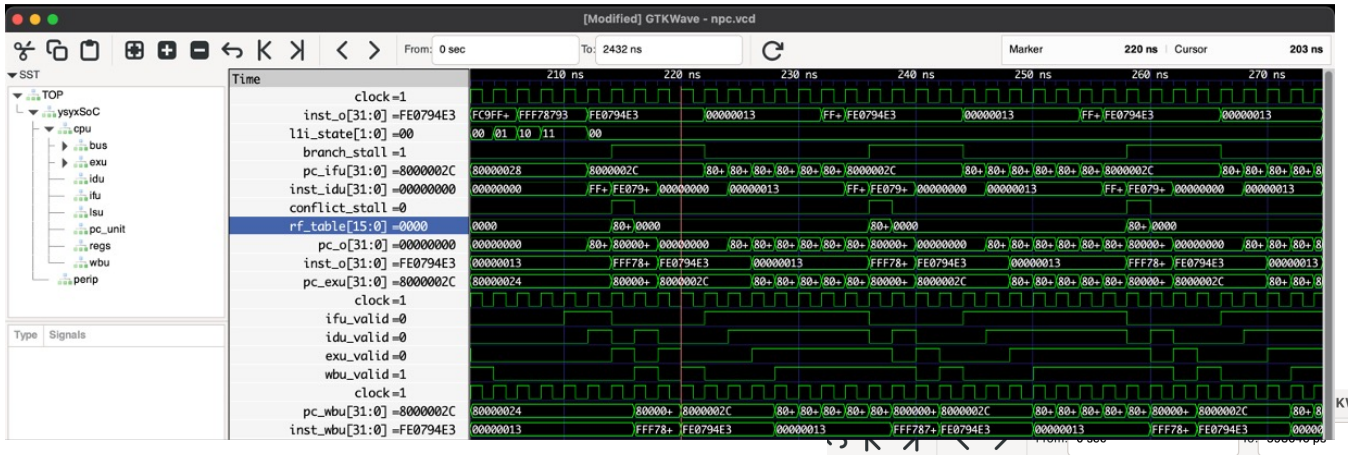


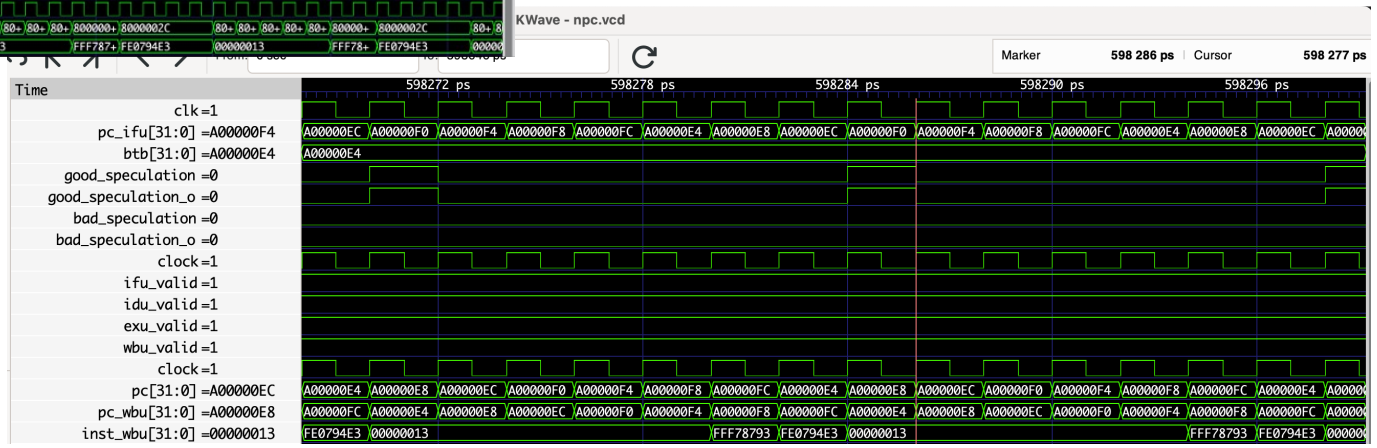


https://github.com/Kingfish404/ysyx-workbench

npc 迭代性能数据 | 理想情况

- 4 级流水线 + ysyxSoC -> exu 转发 idu + 简单分支预测。理想情况下：局部 IPC 为 1






```
80000010: 06400793 li a5,100
80000014: 00000013 nop
...
80000024: 00000013 nop
80000028: fff78793 add a5,a5,-1
8000002c: fe0794e3 bnez a5,80000014 <main+0x4>
80000030: 00000513 li a0,0
```

面积频率优化方法丨相关资料

- 适用于 FPGA 和 SoC 的 UltraFast 设计方法指南 (UG949)
  - https://docs.amd.com/r/zh-CN/ug949-vivado-design-methodology
  - 尽管是 FPGA，但相同的理念也可以用于 ASIC
  - 参考之前分享： NPC 的 FPGA 部署经验 - 叶永檬 - 一生一芯双周分享会
- 手把手教你设计 CPU——RISC-V 处理器篇. 胡振波. 2018
- CPU 设计实战. 汪文祥 / 邢金璋. 2021

https://github.com/OSCPU/yosys-sta

面积频率优化方法 | 自动化评估脚本

- Dockerfile + 参数化shell 脚本
- 实现快速 yosys-sta 快速环境创建和 npc 频率评估
- 本机 yosys 优化编译，发送到服务器 sta 评估，接收结果并过滤显示（grep）

```
FROM ubuntu:22.04

ENV DEBIAN_FRONTEND=noninteractive

RUN apt-get update && \
apt-get install -y \
yosys \
libunwind-dev \
libyaml-cpp-dev \
libgomp1 \
libtcl8.6 \
make

CMD ["bash"]

# docker build -t ubuntu-env .
# docker run -it -v.:/app ubuntu-env
```

```
#!/bin/bash
local_base_dir=$YSYX_HOME/yosys-sta/ysyx
chip_name=ysyx
top_name=ysyx
# chip_name=lltrisc-v
# top_name=datapath
# chip_name=hazard3
# top_name=hazard3_core
# chip_name=picorv32
# top_name=picorv32
mkdir -p $local_base_dir
cp vsrc/ysyx.v \
  vsrc/*.v \
  vsrc/include/ysyx_macro.vh \
  vsrc/include/*.vh
$local_base_dir
cp vsrc/include/ysyx_macro_dpi_c.vh.mock $local_base_dir/ysyx_macro_dpi_c.vh

SED_CMD="sed -i"
if [[ "$OSTYPE" == "darwin"* ]]; then
SED_CMD="gsed -i"
fi

$SED_CMD -i "3c\DESIGN ?= $top_name" "$YSYX_HOME/yosys-sta/Makefile"
$SED_CMD -i "5c\RTL_FILES ?= \$(shell find \$(PROJ_PATH)/$chip_name -name \"*v\")" "$YSYX_HOME/yosys-sta/Makefile"

make -C $YSYX_HOME/yosys-sta/ syn

# remote machine name variable
machine_name=my_machine_ssh_name
base_dir=/data/jinyu/Developer/git

sftp $machine_name <<EOF
cd $base_dir/yosys-sta/
put Dockerfile
put -r $YSYX_HOME/yosys-sta/result
EOF

# ssh $machine_name "cd $base_dir/yosys-sta/ && docker build -t my-ubuntu-env ."
ssh $machine_name "cd $base_dir/yosys-sta/ && \
sed -i '1c\set clk_port_name clock' example/gcd.sdc && \
sed -i '17,20 { /^[[:space:]]*#/! s/^/#/ }' Makefile && \
sed -i '3c\DESIGN ?= $top_name' Makefile"
ssh $machine_name "cd $base_dir/yosys-sta && \
docker run -i -v.:/app ubuntu-env make -C /app sta" \
| grep -E 'core_clock|Clock Group'
cat $YSYX_HOME/yosys-sta/result/$top_name-500MHz/synth_stat.txt | grep 'Chip area'
```

https://github.com/OSCPU/yosys-sta

面积频率优化方法 | 快速评估结果

- 方便自行优化 npc 的设计流程，修改+评估+确认修改，确保满足约束要求（500Mhz，25000 nm^2)
- 快速完成第三方 ip 的性能评估（面积um^2,频率MHz)
  - - (29246.00, 371.09), https://github.com/Wren6991/Hazard3,
  - - (21707.00, 1021.97), https://github.com/YosysHQ/picorv32/
  - - (23781.73, 317.99), https://github.com/LoveLonelyTime/LLTRISC-V/
  - - (27644.58, 401.06), https://github.com/ultraembedded/biriscv

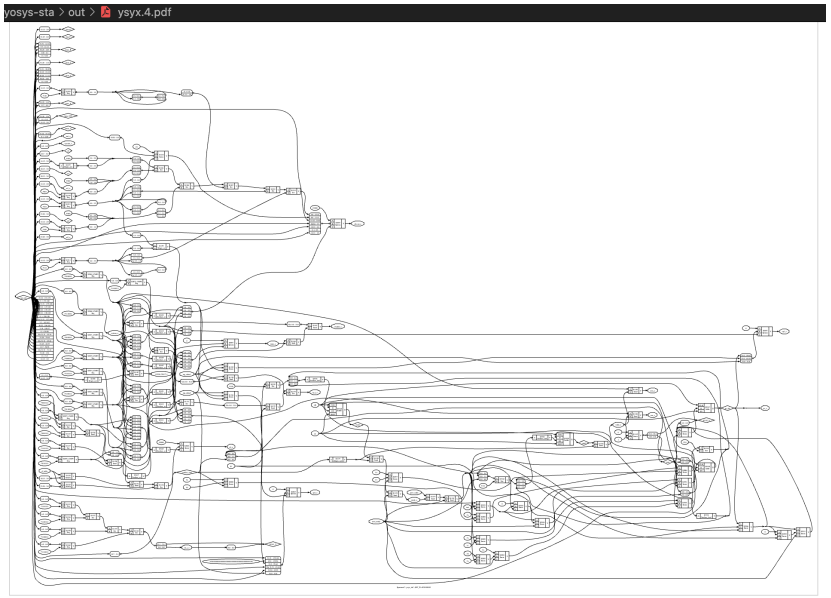
```
| Endpoint           | Clock Group | Delay Type | Path Delay | Path Required | CPPR  | Slack | Freq(MHz) |
| ifu/_3011_:D       | core_clock  | max        | 1.732f     | 1.961         | 0.000 | 0.229 | 564.589   |
| ifu/_3015_:D       | core_clock  | max        | 1.732f     | 1.961         | 0.000 | 0.229 | 564.589   |
| ifu/_3014_:D       | core_clock  | max        | 1.731f     | 1.961         | 0.000 | 0.230 | 565.038   |
| exu/_3588_:D       | core_clock  | min        | 0.121f     | 0.002         | 0.000 | 0.119 | NA        |
| bus/clint/_1076_:D | core_clock  | min        | 0.133r     | 0.005         | 0.000 | 0.128 | NA        |
| exu/_3588_:D       | core_clock  | min        | 0.139r     | 0.005         | 0.000 | 0.134 | NA        |
| core_clock | max        | 0.000 |
| core_clock | min        | 0.000 |
  Chip area for module '$paramod\ysyx_clint\ADDR_W=8'00100000\DATA_W=8'00100000': 977.816000
  Chip area for module '$paramod\ysyx_exu\BIT_W=8'00100000': 3630.900000
  Chip area for module '$paramod\ysyx_exu_alu\BIT_W=8'00100000': 1723.680000
  Chip area for module '$paramod\ysyx_idu\BIT_W=8'00100000': 2147.950000
  Chip area for module '$paramod\ysyx_ifu\ADDR_W=8'00100000\DATA_W=8'00100000': 3325.798000
  Chip area for module '$paramod\ysyx_reg\REG_ADDR_W=8'00000100\DATA_W=8'00100000': 6924.512000
  Chip area for module '\ysyx': 70.224000
  Chip area for module '\ysyx_bus': 1080.226000
  Chip area for module '\ysyx_exu_csr': 2177.210000
  Chip area for module '\ysyx_lsu': 1106.826000
  Chip area for module '\ysyx_pc': 601.426000
  Chip area for module '\ysyx_wbu': 15.162000
  Chip area for top module '\ysyx': 23781.730000
```

```
| Endpoint           | Clock Group | Delay Type | Path Delay | Path Required | CPPR  | Slack | Freq(MHz) |
| exu/_3708_:D       | core_clock  | max        | 1.774r     | 1.969         | 0.000 | 0.194 | 553.809   |
| exu/_3710_:D       | core_clock  | max        | 1.774r     | 1.969         | 0.000 | 0.194 | 553.809   |
| exu/_3711_:D       | core_clock  | max        | 1.774r     | 1.969         | 0.000 | 0.194 | 553.809   |
| exu/_3725_:D       | core_clock  | min        | 0.121f     | 0.002         | 0.000 | 0.119 | NA        |
| bus/clint/_1077_:D | core_clock  | min        | 0.133r     | 0.005         | 0.000 | 0.128 | NA        |
| exu/_3725_:D       | core_clock  | min        | 0.139r     | 0.005         | 0.000 | 0.134 | NA        |
| core_clock | max        | 0.000 |
| core_clock | min        | 0.000 |
  Chip area for module '\ysyx': 79.002000
  Chip area for module '\ysyx_bus': 1066.660000
  Chip area for module '\ysyx_clint': 977.816000
  Chip area for module '\ysyx_exu': 3675.322000
  Chip area for module '\ysyx_exu_alu': 1723.680000
  Chip area for module '\ysyx_exu_csr': 2268.980000
  Chip area for module '\ysyx_idu': 2484.174000
  Chip area for module '\ysyx_ifu': 3047.296000
  Chip area for module '\ysyx_lsu': 1809.332000
  Chip area for module '\ysyx_pc': 646.380000
  Chip area for module '\ysyx_reg': 6795.768000
  Chip area for module '\ysyx_wbu': 315.210000
  Chip area for top module '\ysyx': 24889.620000
```

https://github.com/YosysHQ/yosys

面积频率优化方法 | 寻找最长拓扑路径，优化主频

- yosys 使用 show 和 ltp 分别进行 module 可视化和寻找最长拓扑路径
  - yosys show: 可视化电路，辅助快速找到无用变量，不平衡逻辑
  - yosys ltp: 输出 module 最长拓扑路径，方便找到关键路径




```
set chip_name ysyx
set mod_name ysyx
# set chip_name lltrisc-v
# set mod_name datapath

yosys -import # import all yosys commands directly as tcl commands to the tcl shell
yosys read_verilog -sv ${chip_name}/*
yosys hierarchy -top ${mod_name}

#yosys synth_xilinx -top ${mod_name} # synthesis for Xilinx FPGAs
yosys proc
yosys opt
yosys ltp -noff
yosys show -format dot -prefix ./out/${mod_name} -viewer none
```

yosys-sta > result > ysyx-500MHz > gen_syn_graph.tcl.log

Longest Aa ab .* 9 of 13

```
1042497     489: \io_master_araddr [0]
1042498
1042499   Longest topological path in ysyx_bus (length=10):
1042500       0: \lsu_araddr [31]
1042501       1: $eq$ysyx/ysyx_bus.v:161$14_Y (via $eq$ysyx/ysyx_bus.v:161$14)
1042502       2: \clint_en (via $or$ysyx/ysyx_bus.v:161$15)
1042503       3: $logic_not$ysyx/ysyx_bus.v:164$17_Y (via $logic_not$ysyx/ysyx_bus.v:188$48)
1042504       4: $and$ysyx/ysyx_bus.v:188$49_Y (via $and$ysyx/ysyx_bus.v:188$49)
1042505       5: $or$ysyx/ysyx_bus.v:187$50_Y (via $or$ysyx/ysyx_bus.v:187$50)
1042506       6: \io_master_arvalid (via $and$ysyx/ysyx_bus.v:186$51)
1042507       7: $and$ysyx/ysyx_bus.v:131$5_Y (via $and$ysyx/ysyx_bus.v:131$5)
1042508       8: $procmux$3181_Y [0] (via $procmux$3181)
1042509       9: $procmux$3186_Y [0] (via $procmux$3186)
1042510      10: $procmux$3176_Y [0] (via $procmux$3176)
1042511      ff: \state [2] (via $auto$ff.cc:266:slice$3651)
1042512
1042513   Longest topological path in ysyx_exu_csr (length=7):
1042514       0: \waddr_add1 [11]
```