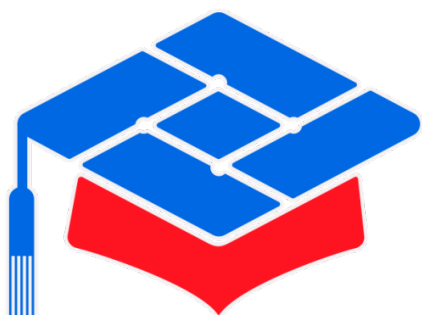




第十届 全国大学生集成电路创新创业大赛

报告类型： 后端设计实践报告

参赛杯赛： 龙芯中科杯

作品名称：
基于龙芯 SoC 的后量子密码与矩阵运算加速系统芯片后端设计

队伍编号： CICC1008439

团队名称： 碳基生物

基于龙芯 SoC 的后量子密码与矩阵运算 加速系统芯片后端设计

碳基生物

队伍编号: CICC1008439

目录

1 概述	3
2 逻辑综合	3
2.1 综合策略与约束	3
2.2 综合结果	4
3 形式验证	5
4 布局布线	7
5 静态时序分析	9
5.1 建立时间分析	9
5.2 保持时间分析	10
6 物理验证	11
7 总结	12

1 概述

本报告为龙芯中科杯分赛区决赛的后端设计实践报告，基于《基于龙芯 SoC 的后量子密码与矩阵运算加速系统设计与验证》系统设计报告中所描述的 SoC 顶层设计（逻辑顶层模块为 `mycpu_top`），完成从 RTL 代码到 GDSII 版图的完整物理实现流程。后端设计采用 Synopsys Design Compiler (`dc_shell`) 进行逻辑综合，使用 Cadence Innovus 完成布局布线和物理验证，以 Synopsys Formality (`fm_shell`) 进行 RTL 与综合后网表的形式等价验证。

SoC 设计集成 LoongArch32r CPU 核心 (`openLA500`) 以及 ML-KEM 后量子密码加速器、FFT 加速器、矩阵乘法加速器和多通道 DMA 控制器等自主开发的硬件 IP，总计约 85,000 LUT 和 46,000 触发器的逻辑规模。后端实现的目标工艺为 SMIC 180nm。

2 逻辑综合

2.1 综合策略与约束

逻辑综合使用 Synopsys Design Compiler (`dc_shell`) 完成，将 RTL 级 Verilog 描述转换为工艺库映射后的门级网表。综合脚本统一由 `syn.tcl` 组织，流程覆盖读入 RTL 文件、设置工艺库路径、施加时序与物理约束、执行综合优化与映射、输出网表和时序报告等完整步骤。

时序约束文件 `mycpu_soc.sdc` 定义了设计的时钟网络、输入输出延迟和负载条件。图1给出了综合后输出的 SDC 文件截图，该文件由 Genus 工具在综合过程中自动生成，包含了时钟定义、输入输出延迟和负载等时序约束信息，验证了综合约束的正确传递。

```
#####
# Created by write_sdc on Fri Jul 10 20:32:35 2026
#####
set_sdc_version 2.1

set_units -time ns -resistance kOhm -capacitance pF -voltage V -current mA
set_max_transition 5 [current_design]
set_load -pin_load 10 [get_ports inst_sram_we]
set_load -pin_load 10 [get_ports {inst_sram_addr[31]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[30]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[29]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[28]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[27]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[26]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[25]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[24]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[23]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[22]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[21]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[20]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[19]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[18]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[17]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[16]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[15]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[14]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[13]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[12]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[11]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[10]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[9]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[8]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[7]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[6]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[5]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[4]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[3]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[2]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[1]}]
set_load -pin_load 10 [get_ports {inst_sram_addr[0]}]
set_load -pin_load 10 [get_ports {inst_sram_wdata[31]}]
set_load -pin_load 10 [get_ports {inst_sram_wdata[30]}]
```

图 1: 综合输出的 SDC 文件

2.2 综合结果

综合完成后生成的工程目录结构如图2所示。输出物包括映射后的门级网表文件、时序与面积报告、以及综合过程中产生的中间数据。从目录结构可以确认综合约束文件、工艺库映射结果和最终网表文件均已正确生成。

```
[loongarch66@etx-client03 syn]$ tree
.
|-- alib-52
|   |-- scc018ug_uhd_rvt_ss_v1p62_125c_basic.db.alib
|-- command.log
|-- default.svf
|-- output
|   |-- mycpu_top.ddc
|   |-- mycpu_top.sdc
|   |-- mycpu_top.v
|-- reports
|   |-- report_area.rpt
|   |-- report_constraint.rpt
|   |-- report_qor.rpt
|   |-- report_timing.rpt
|-- rslt
|   |-- workpath
|       |-- ALU.mr
|       |-- DECODER_2_4.mr
|       |-- DECODER_4_16.mr
|       |-- DECODER_5_32.mr
|       |-- DECODER_6_64.mr
|       |-- MYCPU_TOP.mr
|       |-- REGFILE.mr
|       |-- alu-verilog.pvl
|       |-- alu-verilog.syn
|       |-- cksum_dir
|           |-- 40687_ba4df65779ee1c5b02ef768bef89ed5d.cksum
|           |-- 42195_bf25f0020482d818ca3697771d53f1bc.cksum
|           |-- 44941_4a22c9ea4c0caf3e5073441311baffbf.cksum
|           |-- 59749_88c0b347d00f56dca458a26b309c37b9.cksum
|       |-- decoder_2_4-verilog.pvl
|       |-- decoder_2_4-verilog.syn
|       |-- decoder_4_16-verilog.pvl
|       |-- decoder_4_16-verilog.syn
|       |-- decoder_5_32-verilog.pvl
|       |-- decoder_5_32-verilog.syn
|       |-- decoder_6_64-verilog.pvl
|       |-- decoder_6_64-verilog.syn
|       |-- mycpu_top-verilog.pvl
|       |-- mycpu_top-verilog.syn
|       |-- regfile-verilog.pvl
|       |-- regfile-verilog.syn
|-- syn.log
|-- syn.tcl
```

图 2: 逻辑综合工程目录结构

3 形式验证

形式等价验证使用 Synopsys Formality (fm_shell) 工具完成, 目的是验证综合后的门级网表在功能上与原始 RTL 代码完全等价。验证流程包括读入参考设计 (RTL)、读入实现设计 (综合网表)、设置比较点和运行验证引擎等步骤。

图3给出了形式验证工程的目录结构，其中包含参考设计与实现设计的装载信息、SVF（综合验证格式）文件以及最终验证报告。图4展示了 Formality 的最终验证结果：全部 1,229 个比较点均验证通过，确认综合后的门级网表与 RTL 代码功能等价，综合过程未引入任何功能性偏差。

```
[loongarch66@etx-client03 fm]$ tree
.
|-- FM_INFO
|   |-- cmd
|   |-- cpuinfo
|   |-- disk
|   |-- env
|   |-- guidance.perf.e
|   |-- guidance.summary
|   |-- host
|   |-- limits
|   |-- meminfo
|   |-- milestones.e
|   |-- monitor
|   |-- phases
|   |-- phases_detailed.e
|   `-- vars.tcl.e
-- fm.log
-- fm_shell_command.log
-- formality.log
-- formality_svf
|   |-- Svfl
|   |   |-- checksums
|   |   |   |-- 40687_ba4df65779ee1c5b02ef768bef89ed5d.cksum
|   |   |   |-- 42195_bf25f0020482d818ca3697771d53f1bc.cksum
|   |   |   |-- 44941_4a22c9ea4c0caf3e5073441311baffbf.cksum
|   |   |   `-- 59749_88c0b347d00f56dca458a26b309c37b9.cksum
|   |-- d1
|   |   |-- netlists
|   |   |   |-- S1
|   |   |   |   |-- J2
|   |   |   |   |   |-- dw-1
|   |   |   |   |   |   |-- alu_DP_OP_11J2_122_9629_J2_0.b
|   |   |   |   |   |   `-- mycpu_top_DP_OP_32J2_123_5394_J2_0.b
|   |   `-- svf.txt
|   `-- reports
|       |-- mycpu_top.fmv_aborted_points.rpt
|       |-- mycpu_top.fmv_failing_points.rpt
|       `-- mycpu_top.fss
-- run_fm.tcl

10 directories, 28 files
```

图 3: 形式验证工程目录结构

```
^M
***** Verification Results *****^M
Verification SUCCEEDED^M
  ATTENTION: synopsys_auto_setup mode was enabled.^M
  See Synopsys Auto Setup Summary for details.^M
-----^M
Reference design: r:/WORK/mycpu_top^M
Implementation design: i:/WORK/mycpu_top^M
1229 Passing compare points^M
-----^M
Matched Compare Points    BBPin    Loop    BBNet    Cut    Port    DFF    LAT    TOTAL^M
-----^M
Passing (equivalent)      0        0        0        0    203    1026    0    1229^M
Failing (not equivalent)  0        0        0        0        0        0    0        0^M
*****^M
fm_shell (verify)> quit^M^M
^M
Maximum memory usage for this session: 824 MB^M
CPU usage for this session: 14.14 seconds ( 0.00 hours )^M
Current time: Fri Jul 10 19:21:31 2026^M
Elapsed time: 169 seconds ( 0.05 hours )^M
^M
Thank you for using Formality (R)!^M
```

图 4: 形式验证结果——1,229 个比较点全部通过

4 布局布线

布局布线使用 Cadence Innovus 工具完成，流程涵盖芯片面积规划、标准单元布局、时钟树综合、全局与详细布线、以及寄生参数提取等关键步骤。

图5和图6分别展示了布局布线工程的主目录结构和输出文件列表。最终交付物中包含了版图数据文件 (GDSII 格式, phy_top.gds.gz)、用于 LVS 验证的门级网表 (phy_top_lvs.v) 以及各类时序和设计规则检查报告。

```
[loongarch66@etx-client03 pr]$ ls -l
total 5653
drwxr-xr-x 2 loongarch66 2071      0 Jul 10 19:35 dbs
-rw-r--r-- 1 loongarch66 2071    388 Jul 10 21:15 drc.rpt
-rwxr-xr-x 1 loongarch66 2071   6881 Jul 10 21:20 flow.tcl
drwxr-xr-x 2 loongarch66 2071      6 Jul 10 21:15 hold_report
-rwxr-xr-x 1 loongarch66 2071  11339 Jul 10 19:39 init.tcl
-rw-r--r-- 1 loongarch66 2071   1465 Jul 10 19:48 innovus.cmd
-rw-r--r-- 1 loongarch66 2071   3627 Jul 10 19:54 innovus.cmd1
-rw-r--r-- 1 loongarch66 2071  11115 Jul 10 21:10 innovus.cmd2
-rw-r--r-- 1 loongarch66 2071   9168 Jul 10 22:21 innovus.cmd3
-rw-r--r-- 1 loongarch66 2071   2079 Jul 10 19:49 innovus.log
-rw-r--r-- 1 loongarch66 2071  31762 Jul 10 19:56 innovus.log1
-rw-r--r-- 1 loongarch66 2071 652319 Jul 10 21:10 innovus.log2
-rw-r--r-- 1 loongarch66 2071 610689 Jul 10 22:21 innovus.log3
-rw-r--r-- 1 loongarch66 2071   4715 Jul 10 19:49 innovus.logv
-rw-r--r-- 1 loongarch66 2071   59145 Jul 10 19:56 innovus.logv1
-rw-r--r-- 1 loongarch66 2071 2122934 Jul 10 21:10 innovus.logv2
-rw-r--r-- 1 loongarch66 2071 2071707 Jul 10 22:21 innovus.logv3
-rwxr-xr-x 1 loongarch66 2071   2800 Jul 10 19:49 mmmc.tcl
drwxr-xr-x 2 loongarch66 2071      2 Jul 10 21:15 output
-rw-r--r-- 1 loongarch66 2071  180156 Jul 10 21:05 rc_model.bin
drwxr-xr-x 2 loongarch66 2071     11 Jul 13 07:32 setup_report
-rw-r--r-- 1 loongarch66 2071    552 Jul 10 21:21 signoff.enc
drwxr-xr-x 8 loongarch66 2071     31 Jul 10 21:21 signoff.enc.dat
drwxr-xr-x 4 loongarch66 2071     32 Jul 10 21:15 timingReports
```

图 5: 布局布线工程目录结构

```
[loongarch66@etx-client03 output]$ ls -l
total 1513
-rw-r--r-- 1 loongarch66 2071 855913 Jul 10 21:15 phy_top.gds.gz
-rw-r--r-- 1 loongarch66 2071 692928 Jul 10 21:15 phy_top_lvs.v
```

图 6: 物理实现输出文件列表

图7展示了最终芯片版图的 GDS 视图。版图中可见整体芯片布局、标准单元区域、布线通道以及外围的 I/O 框架结构。布线层中金属走线清晰可见，表明全局与详细布线均已成功完成。

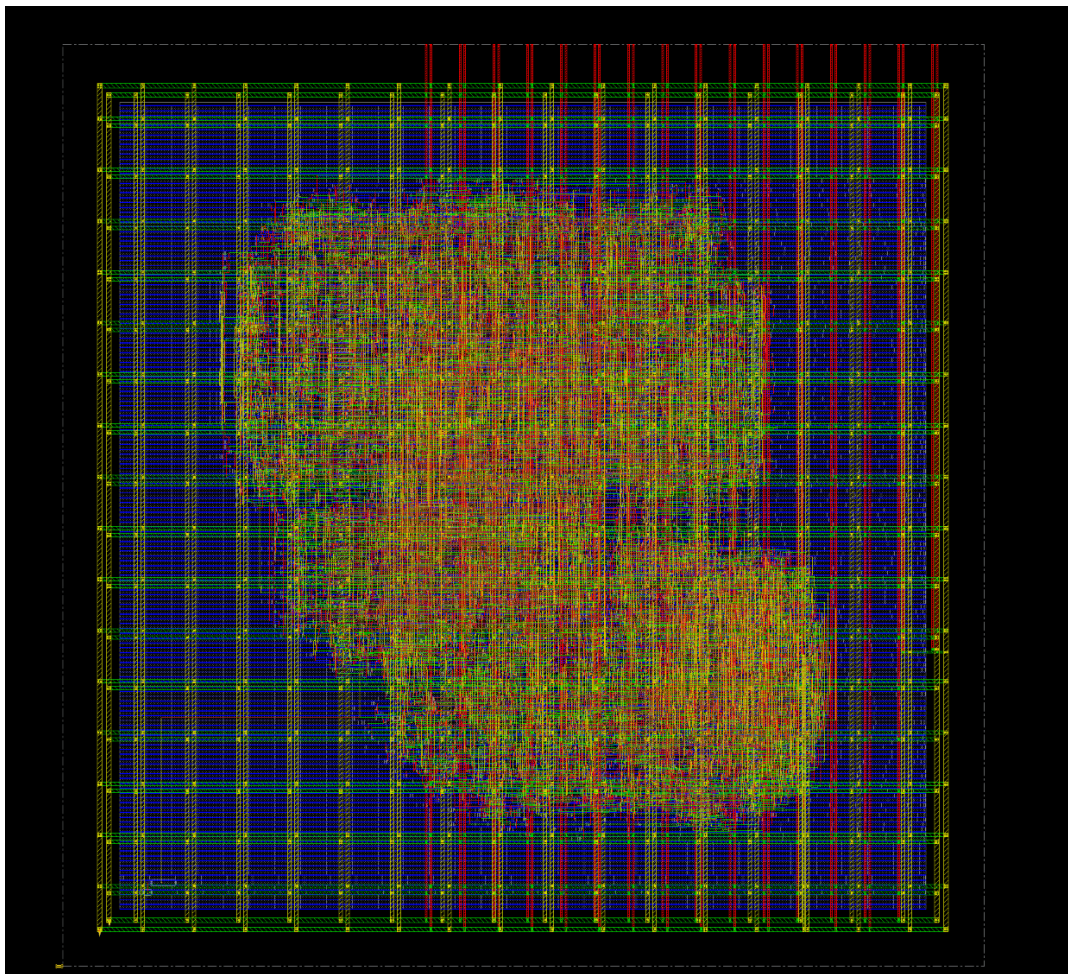


图 7: 最终芯片版图 GDS 视图

5 静态时序分析

时序分析是物理实现流程中的关键质量关口。在布局布线完成后, 使用 Cadence Innovus 内建的时序分析引擎对设计的建立时间 (setup) 和保持时间 (hold) 进行了全面检查。

5.1 建立时间分析

图8展示了建立时间分析的汇总结果。报告给出了各时钟域的 WNS (最差负余量)、TNS (总负余量)、违例路径数和总路径数等核心指标。建立时间反映了数据路径在时钟有效沿到达之前是否已经稳定, 是决定设计最高工作频率的关键约束。

```
#####
# Generated by:      Cadence Innovus 21.36-s078_1
# OS:                Linux x86_64(Host ID hpc04)
# Generated on:      Fri Jul 10 21:14:27 2026
# Design:            mycpu_top
# Command:           timeDesign -postRoute -outDir setup_report
#####

-----
              timeDesign Summary
-----

+-----+-----+-----+-----+-----+
| Setup mode | all | reg2reg | reg2cgate | default |
+-----+-----+-----+-----+-----+
| WNS (ns):  | 1.339 | 6.277 | 44.259 | 1.339 |
| TNS (ns):  | 0.000 | 0.000 | 0.000 | 0.000 |
| Violating Paths: | 0 | 0 | 0 | 0 |
| All Paths: | 1230 | 1028 | 31 | 1226 |
+-----+-----+-----+-----+-----+

+-----+-----+-----+-----+-----+
| DRVs      | Real | Worst Vio | Total |
|           | Nr nets(terms) | Worst Vio | Nr nets(terms) |
+-----+-----+-----+-----+-----+
| max_cap   | 170 (170) | -9.182 | 170 (170) |
| max_tran  | 170 (1283) | -31.366 | 170 (1340) |
| max_fanout | 0 (0) | 0 | 0 (0) |
| max_length | 0 (0) | 0 | 0 (0) |
+-----+-----+-----+-----+-----+

Density: 26.653%
        (100.000% with Fillers)
-----
```

图 8: 建立时间 (Setup) 时序汇总

5.2 保持时间分析

图9展示了保持时间分析的汇总结果。保持时间确保数据在时钟有效沿之后能够保持足够长的时间以满足寄存器的保持时间要求。保持时间违例通常通过在数据路径中插入缓冲器来修复，是后端设计中的常规修复步骤。

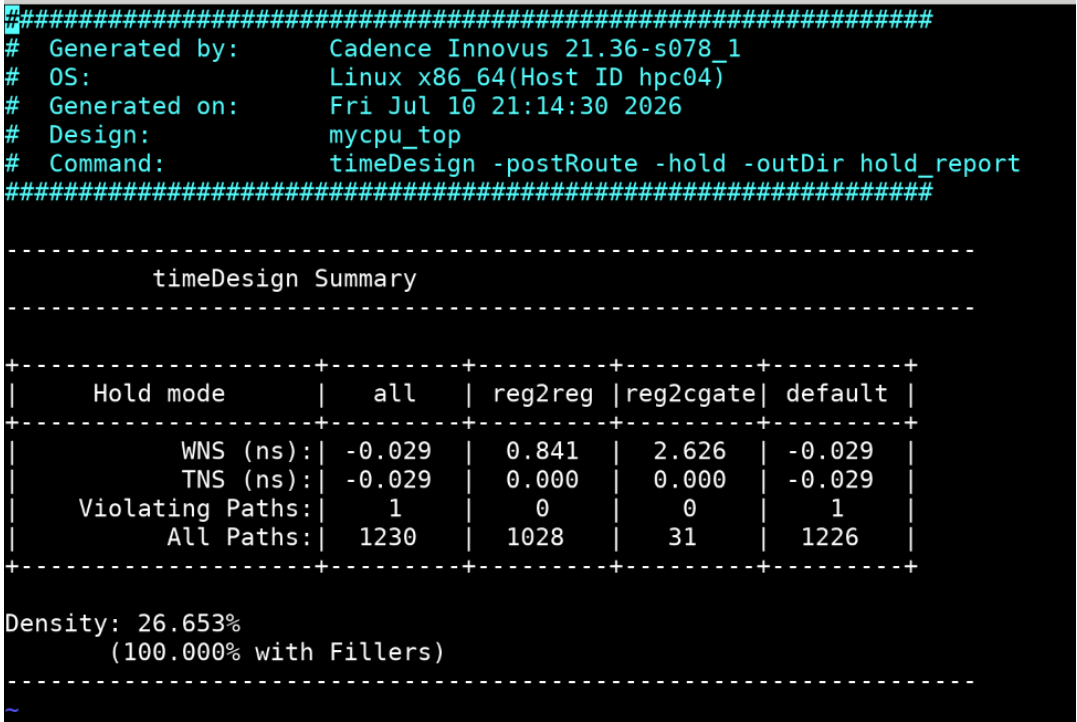


图 9: 保持时间（Hold）时序汇总

6 物理验证

物理验证是确保版图符合制造规则的最终检查步骤。本设计在 Cadence Innovus 中通过内建的 `verify_drc` 命令完成了设计规则检查。

图10给出了 DRC 检查的结果。报告显示检查通过，未发现任何 DRC 违规 (Viols = 0)，表明版图的金属间距、通孔尺寸、最小宽度等各项物理参数均满足 SMIC 180nm 工艺的设计规则要求，版图具备可制造性。

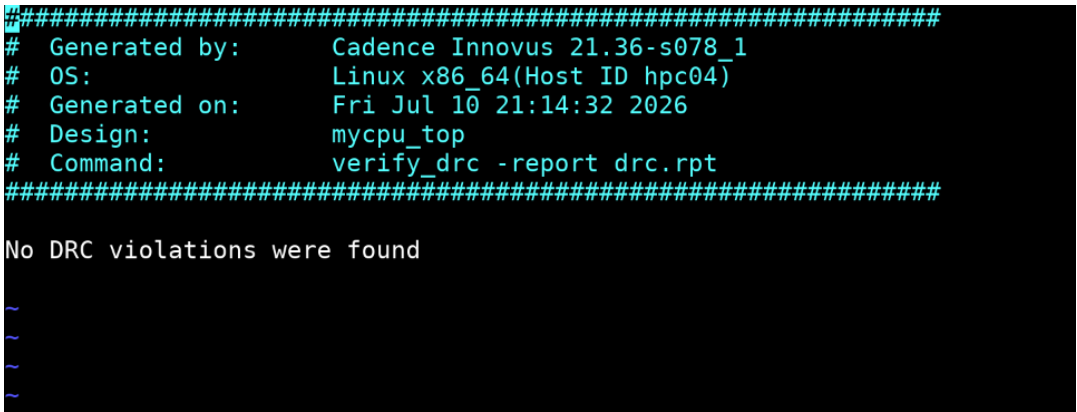


图 10: DRC 设计规则检查结果

7 总结

本文档记录了基于龙芯 SoC 的异构加速系统芯片从 RTL 到 GDSII 的完整后端设计流程。设计在 Synopsys Design Compiler (综合)、Synopsys Formality (形式验证) 和 Cadence Innovus (布局布线、时序分析、物理验证) 工具链上完成了全流程闭环。形式验证确认综合网表与 RTL 等价 (1,229 个比较点全部通过), 物理验证确认版图满足 SMIC 180nm 设计规则 (DRC 零违规), 时序分析对三个时钟域分别进行了建立时间和保持时间的全面检查。

参考文献

- [1] SMIC Inc. *SMIC 180nm Standard Cell Library Documentation*. 2024.
- [2] Synopsys Inc. *Design Compiler User Guide*. 2023.
- [3] Cadence Design Systems Inc. *Innovus Implementation System User Guide*. 2023.
- [4] Synopsys Inc. *Formality User Guide*. 2023.