上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

# MHSA 加速器后端报告


学生姓名 张博文、黄超凡、汪子尧

学号 124039910094、124039910088、124039910018

学院 集成电路与学院


2025 年 6 月 19 日

# 目录

# 1 理论基础

## 1.1 模块功能简述

Multi-Head Self-Attention（MHSA，多头自注意力）模块是 Transformer 架构的关键组成部分，其核心功能在于通过多个并行的注意力头对输入序列进行建模，提取不同子空间下的特征表示。每个注意力头通过对输入特征的线性映射，生成 Query（Q）、Key（K）和 Value（V）矩阵，并通过点积注意力机制（Scaled Dot-Product Attention）计算权重，进而加权汇聚信息。多个头的结果经过拼接与线性变换后输出，为后续网络提供全局感知能力。

该模块在硬件实现中涉及大量矩阵乘法、向量加权、softmax 归一化等计算操作，计算密集且访存频繁，成为 Transformer 整体延迟和能耗的瓶颈之一。

在实际应用中，序列长度 $L$ 和特征维度 $C$ 通常都比较大。这使得多头自注意力机制的运算复杂度非常高，尤其是当需要处理长序列时，计算量会呈指数级增长。我们设计了专门针对多头自注意力机制的硬件加速器，并将其挂载在蜂鸟 E203 RISC-V 处理器上，以充分利用硬件的并行计算能力，显著降低运算复杂度，提高处理效率。

## 1.2 综合目标

MHSA 模块作为 Transformer 架构中的关键计算核心，在硬件实现中存在显著的结构复杂性与资源敏感性。前端 RTL 设计完成后，为评估设计的物理可实现性、资源消耗与性能指标，需对 MHSA 加速器进行综合。综合阶段不仅是从行为级到门级设计的桥梁，更是后端实现流程的前提，需要实现设计可实现性检查、资源评估、性能分析、功耗估算等。

异构加速系统架构如图 1所示。Acc Wrapper 模块为 MHSA 加速器顶层模块，加速器顶层模块内主要包含内部 Memory 、计算模块与其他相应的控制模块。计算流程中涉及大量矩阵乘法与向量变换操作，计算路径较长，逻辑深度大，因此，综合阶段的主要目标在于确保功能模块在满足设计正确性的前提下，具备良好的性能表现与物理可实现性。

综合阶段的具体目标与期望指标如下：

(1) **工作频率**：本设计期望综合后在 **150 MHz** 的频率下稳定运行，时序路径无 setup/hold 违例，满足深度管线下的延迟控制需求。

(2) **面积控制**：由于 MHSA 计算逻辑复杂，目标将总逻辑面积控制在 **1.5 $mm^2$** 以内（约合 1500k GE），兼顾算力与片上集成度。

(3) **功耗估计**：在典型工作电压（1.1V）下，期望**总功耗不超过 120 mW**，通过结构优化、低功耗策略确保整体能效水平。

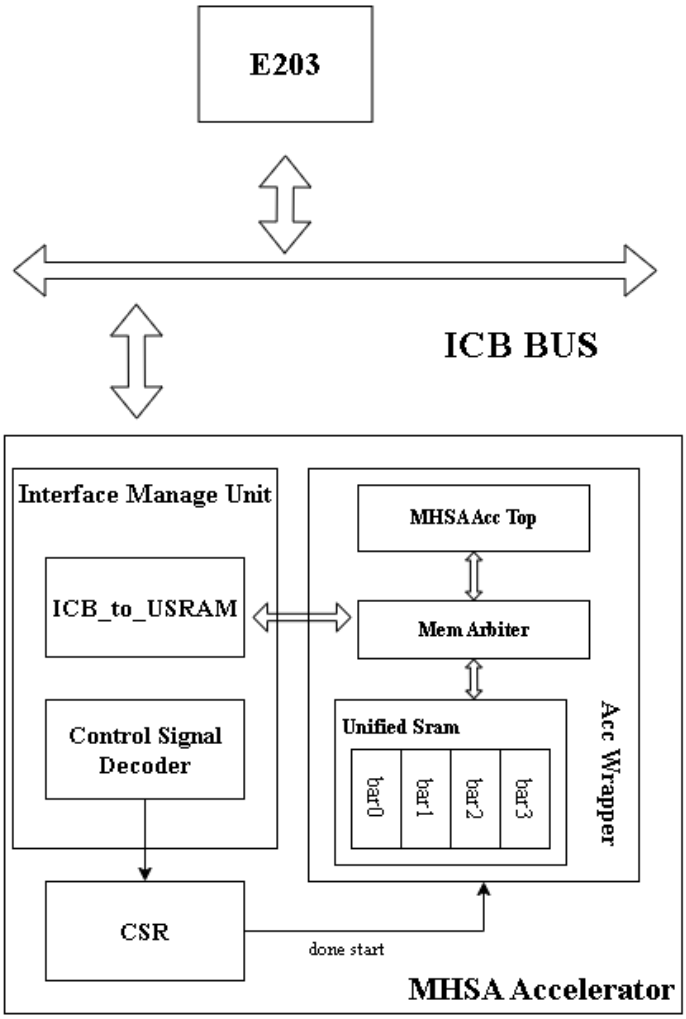


图 1: E203-MHSA-Accelerator 异构计算加速系统

# 2 平台与工具

为了完成 MHSA 加速器的 RTL 级综合、静态分析与物理设计的工作，本设计基于课程统一提供的软硬件平台展开。所采用的工具链、工艺库及平台环境配置如下：

## 2.1 操作系统与环境

相关实验流程均在一套配置完整的 Almalinux 虚拟机中进行。该虚拟机预安装了必要的综合工具、脚本支持环境以及相关依赖组件，便于在标准化环境中进行流程复现与调试。

## 2.2 工具

综合工具采用 Synopsys Design Compiler（DC），具备高效的静态时序分析（STA）、逻辑综合、门级网表生成与约束分析功能。本次综合过程中，主要使用 DC 的以下功能模块：

(1) **RTL 综合**：将 Verilog 描述的 MHSA 加速器模块转化为门级网表。

(2) **时序分析**：对关键路径进行静态分析，确认能否满足频率约束。

(3) **面积报告**：统计综合后各子模块标准单元与面积消耗。

(4) **功耗估算**：基于默认切换率，估算综合后模块的功耗表现。

物理设计采用 Innovus Implementation System，具备布局规划、时钟树综合、布线等物理设计功能。

## 2.3 工艺库

本次综合和物理设计基于 GSCL 45nm 工艺库，包括以下关键库文件：

- **gscl45nm.db**：是 45nm 标准单元工艺库的数据库格式文件，定义了所有门级标准单元的面积、延迟、功耗等信息。
- **SRAM.db**：为课程提供的片上 SRAM（4K×64）模块的门级建模库，包含读写端口、时序参数等信息，支持对 Memory 模块进行准确综合。
- **gscl45nm.lef IO.lef SRAM.lef**：45nm 标准单元工艺库、SRAM、IO 接口的版图层面抽象描述的文件。

# 3 综合流程

## 3.1 文件整理

在综合流程开始前，首先对所有设计源码（包含顶层模块 mhsa_acc_wrapper.sv 及其下属计算与控制模块）进行统一整理，并列入工程文件清单（filelist）中，确保所有模块均可被综合工具识别并正确分析。

设计文件清单保存在 filelist.f 文件中：

```
...
/home/MicroE/project/DC/rtl/linear.sv
/home/MicroE/project/DC/rtl/qkmm.sv
/home/MicroE/project/DC/rtl/mhsa_acc_top.sv
/home/MicroE/project/DC/rtl/mhsa_acc_wrapper.sv
```

文件路径需与实际文件结构保持一致，便于 Design Compiler 在读取时无需手动逐个添加。

## 3.2 SRAM 接口适配

MHSA 加速器顶层模块 mhsa_acc_wrapper 中原设计使用了自定义行为级建模的存储模块 mem。该模块在仿真阶段便于调试与验证，但由于为行为级建模，在逻辑综合阶段无法映射至具体工艺单元，因此必须用课程提供的 SRAM 宏模块替代，才能生成符合标准单元库的门级网表。

根据课程提供的 SRAM.lib 手动编写了与之接口完全一致的 SRAM.v，如下：

```
module SRAMMACRO (
    input CLK, // 时钟
    input CEB, // 芯片使能（低有效）
    input WEB, // 写使能（低有效）
    input SD, // 电源控制（高有效）
    input SLP, // 睡眠模式（高有效）

    input [12:0] A, // 地址
    input [63:0] D, // 写数据
    input [63:0] BWEB, // 写掩码
    output [63:0] Q, // 读数据

    input VDD, // 电源
    input VSS, // 地
    input VDD_i // 内部电源（由 SD / SLP 控制）
```

```
);
    /* synthesis syn_black_box */
endmodule
```

该模块与.lib 文件中的 SRAMMACRO cell 精确匹配，Design Compiler 可在综合过程中自动识别并将其映射到预定义的物理实现。

为了避免子模块综合失败，删掉.VDD、.VSS、.VDD_i 这些 pg_pin 类型的接口连接。在综合阶段，DC 会根据.lib 中 pg_pin 定义自己处理电源。

## 3.3 约束配置

为了保证综合质量与后续物理实现的一致性，需要合理设置时序、面积与功耗相关的约束和脚本控制策略。

### 3.3.1 时序约束设置

在本设计中，MHSA 加速器的整体工作频率目标为 150.4 MHz，对应的时钟周期为 6.65ns。为实现准确的时序收敛和有效的路径优化，需在综合前为 Design Compiler 设置清晰的时钟约束及输入输出路径约束。

首先需要为所有输入输出端口设置合理的时序裕量。设定的输入/输出延迟为 0.2ns，用于模型与实际系统中输入输出驱动器之间存在的接口延时，便于后续物理综合或布图阶段收敛。

时钟的不确定性与转换时间也需显式约束。对最坏路径增加 0.1ns 的 setup margin，对最短路径添加 0.2ns 的 hold margin，限制输入时钟上升/下降时间的最大值为 0.2ns，以引导使用较快缓冲链并减小负载分布问题。

### 3.3.2 综合优化策略与指令配置

为了在逻辑综合阶段充分优化 MHSA 加速器的面积、时序与功耗表现，本设计在 Design Compiler 综合脚本中启用了多项针对数据通路与时序驱动的高级优化选项。以下为主要综合指令及其对应策略：

- **set compile_enable_datapath_opt true**：启用数据通路级别优化，减少关键路径延迟，提高乘加等运算模块性能，对结构中有乘法器/加法器模块尤为有效。
- **set compile_seqmap_merge_identical_modules true**：合并结构上等价的顺序模块，减少冗余逻辑，节省面积与功耗，适用于 MHSA 多头结构中重复控制路径的场景。
- **set compile_ultra_optimization true**：启用 Ultra 优化模式，在保证功能正确的前提下，最大化优化综合结果，综合时间会显著增加，适用于面积/功耗敏感设计。

- **set compile_use_timing_driven true**：启用基于时序驱动的综合，时序收敛更快，适合高频工作目标，搭配时钟约束效果更佳。
- **set power_collapse true**：打开功耗折叠优化策略，降低静态功耗，提升能效比，尤其适用于大规模乘加模块空闲期间的功耗控制。

此外，在进行资源共享（如乘法器复用）和控制逻辑压缩设计后，配合上述指令设置可有效防止综合器错误复展结构或在门级自动重复实例化原本已复用的模块，从而避免额外的面积开销。

## 3.4 综合执行

在确认顶层设置与存储器接口替换无误后，进入 project 对应的文件夹，在 run 子文件夹下打开终端，输入 dc 进入 DC 综合软件，然后输入以下命令进行综合：

```
source ../script/dc_main.tcl
```

该命令将依次执行脚本中的综合命令，包括读取文件、设置约束、执行综合等，执行过程中可实时查看控制台输出以确认模块是否成功识别、实例化是否正确连接、是否存在未解析信号或端口未连接等异常。

## 3.5 形式验证

在完成综合后，为确保综合前后的 RTL 设计与门级网表在功能上保持一致，本实验使用 Synopsys Formality 工具进行了形式验证。通过形式验证，可以确认综合过程中未引入功能性错误，保证设计逻辑的正确性与可靠性。

在进行 formal 验证前，首先需要根据实验环境对验证脚本中的路径变量进行调整。具体而言，需要修改脚本中几个关键路径，包括 svf_path、sverilog_path、implement_path，使其指向本地实际使用的文件位置。

完成路径配置后，启动 Formality 环境，运行验证脚本。脚本会自动完成原始 RTL 代码与综合后网表的比对工作。验证过程中，Formality 工具依据综合过程中生成的 SVF 文件，指导比对过程，确保能够正确映射综合优化后的网表结构。

```
****************************** Verification Results ******************************
Verification SUCCEEDED
   ATTENTION: synopsys_auto_setup mode was enabled.
              See Synopsys Auto Setup Summary for details.
----------------------------------------------------------
 Reference design: r:/WORK/mhsa_acc_top
 Implementation design: i:/WORK/mhsa_acc_top
 12643 Passing compare points
------------------------------------------------------------------------------------
Matched Compare Points     BBPin    Loop   BBNet     Cut    Port     DFF     LAT   TOTAL
------------------------------------------------------------------------------------
Passing (equivalent)           0       0       0       0     389   12254       0   12643
Failing (not equivalent)       0       0       0       0       0       0       0       0
Not Compared
  Unread                       0       0       0       0       0   10100       0   10100
************************************************************************************
```

图 2: 形式验证结果

脚本执行后，Formality 工具自动完成了设计匹配与等价性比对。终端输出结果如图 2所示，从结果上看，最终显示 “SUCCEEDED”，表示形式验证顺利通过。这一结果表明，mhsa 设计与原始 RTL 设计在功能上完全一致，综合过程中未引入任何逻辑错误或功能偏差。

# 4 综合结果分析

本节将基于综合工具 Synopsys Design Compiler 对 MHSA 加速器在 45nm 工艺下的综合结果进行性能分析，主要从时序收敛情况、逻辑面积分布以及静态功耗和动态功耗等方面进行评估。所使用的工艺库为课程提供的 gscl45nm.lib 和 SRAM.lib。

## 4.1 时序分析

在本次综合过程中，采用默认的 top-level wire load model。时钟周期设定为 6.65 ns（约 150 MHz）。以下为 MHSA 加速器关键路径和整体时序性能的详细分析。

### 4.1.1 时钟域分析

本设计采用统一的时钟域，主时钟信号为 clk，所有触发器均由该时钟控制。在此基础上，工具将时序路径划分为如下几类：

- **reg2reg**：寄存器到寄存器路径
- **in2reg**：输入口到寄存器路径
- **reg2out**：寄存器到输出口路径
- **in2out**：输入口到输出口路径

### 4.1.2 最差路径分析

根据综合工具的静态时序分析（STA）结果，典型工况下，MHSA 加速器关键路径满足时序要求，所有路径均无负 slack，设计可收敛。关键路径分别从寄存器到寄存器（reg2reg）、输入到寄存器（in2reg）、输入到输出（in2out）与寄存器到输出（reg2out）四类时序路径进行分析汇总，如表 1所示。

表 1: 路径时序分析表

| 路径类型 | 起点 | 终点 | 数据到达时间 (ns) | 所需数据时间 (ns) | Slack (ns) | 是否满足时序 |
|---|---|---|---|---|---|---|
| reg2reg | col_o_reg[3] | res_reg[31] | 2.14 | 2.14 | 0.00 | 是 |
| in2reg | start（输入） | res_reg[31] | 2.14 | 2.14 | 0.00 | 是 |
| in2out | soc_addr[17]（输入） | soc_data_out[4]（输出） | 2.19 | 6.35 | 4.16 | 是 |
| reg2out | state_reg[1] | done（输出） | 0.26 | 6.35 | 6.09 | 是 |

关键路径细节说明：

- **Reg2Reg 路径（0 slack）**：路径位于 mm_systolic 内部两个处理单元之间，总延迟为 2.14ns，为目前路径中最长，主要包括 DW02_mult 和 DW01_add 单元，核心计算逻辑为乘加阵列（Systolic Array）内数据传播。
- **In2Reg 路径（0 slack）**：由控制信号 start 到 res_reg[31]，经过多个转换通路进入处理阵列，路径与上面的 Reg2Reg 相似，存在较多组合逻辑（INV, AND, OR, XNOR）级联。
- **In2Out 路径（Slack = 4.16ns）**：来自地址输入 soc_addr[17]，最终输出为 soc_data_out[4]，中间逻辑包含大量地址译码与控制路径，但由于路径短、延迟小，Slack 裕量充足，未形成关键路径。
- **Reg2Out 路径（Slack = 6.09ns）**：控制状态寄存器 state_reg[1] 到最终控制输出 done，路径逻辑简单，仅为控制信号直接传出，延迟仅 0.26ns，为所有路径中最短。

整体而言，MHSA 加速器综合后在目标时钟频率（150 MHz）下可满足时序闭合，但存在两条临界路径（Reg2Reg 与 In2Reg），其 Slack 为 0.00 ns，表明电路在当前频率下工作在边界状态，后续可能受工艺波动影响导致时序违约。在后续优化中 k 可能需要重点考虑路径削减和组合逻辑压缩策略，以增强时序裕度。

## 4.2 面积分析

在不指定线网负载模型的前提下，Design Compiler 面积报告如表 2所示。

表 2: 面积组成分析表

| 面积组成类别 | 元素数量（个） | 面积（$um^2$） | 占总面积比例（%） |
|---|---|---|---|
| 组合逻辑（Combinational） | 305,918 | 803,535.92 | 56.99 |
| 缓冲器/反相器（Buf/Inv） | 125,724 | 192,466.03 | 13.65 |
| 时序单元（Sequential） | 24,861 | 180,238.42 | 12.79 |
| 宏单元（Macro/Black Box） | 4（含 SRAM） | 425,846.53 | 30.21 |
| 线网面积（Net） | - | 未定义 | - |
| **总面积（Total Cell）** | **332,410** | **1,409,620.87** | **100** |

面积分析说明：

- **组合逻辑占比最高（57%）**：体现出 MHSA 加速器中乘加矩阵计算、Softmax 模块等核心功能的逻辑密集度较高，符合其典型特征。

- **缓冲器与反相器占比（约 13.65%）**：说明设计中用于驱动大扇出、提升时序性能的缓冲结构较多，数量达到 12 万，可能为乘法器阵列中的时钟及数据驱动电路贡献较大面积。
- **宏单元（SRAM）面积显著（30.21%）**：这与设计中多个头（head）共享查询/键/值矩阵存储有关，属于典型的数据本地化策略所致。
- **寄存器面积适中（12.79%）**：体现设计中状态与结果寄存需求基本满足同步控制要求。
- **线网面积未统计**：由于本次综合未启用后端物理布局流程，也未指定 wireload model，因此线网（routing/interconnect）面积处于未定义状态，预计后续布局布线阶段会略微抬高总面积。

整体而言，MHSA 加速器的总单元面积约为 1.41 $mm^2$，其中组合逻辑与宏单元占比显著，表明设计以算力密集型模块与片上存储为核心。该面积水平在 45nm 工艺下属于中等复杂度，适合集成于 E203 这类轻量级 RISC-V SoC 体系中。

## 4.3 功耗分析

为进一步评估其在目标工艺和典型工作条件下的能效表现，对功耗进行了静态与动态功耗估算。功耗报告总览如表 3所示，功耗分布细分如表 4所示。

表 3: 功耗分析表

| 功耗类型 | 数值（mW） | 占比（%） | 说明 |
|---|---|---|---|
| Cell Internal Power | 90.4028 | 84.93 | 逻辑门切换内部短路和电容充放电过程功耗 |
| Net Switching Power | 11.7859 | 11.07 | 网络导线由于电容驱动而产生的切换功耗 |
| Cell Leakage Power | 5.2461 | 4.94 | 漏电流产生的静态功耗 |
| **总功耗** | **107.4324** | **100** | **动态 + 静态合计** |

功耗分析说明：

- **动态功耗为主导因素**：其中 Cell Internal Power 占比高达 84.9%，表明在典型输入激励下 MHSA 模块存在大量内部节点切换，这与其多头乘法阵列并发计算高度相关。
- **Net Switching Power 占比（约 11.1%）**：推测总线连接与中间节点的布线长度仍在可接受范围内，后续布局布线阶段可进一步优化缓冲与插值平衡。

表 4: 功耗分组分析表

| 功耗分组 | Internal (mW) | Switching (mW) | Leakage (mW) | Total (mW) | 占总功耗比例（%） |
|---|---|---|---|---|---|
| IO Pad | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.00 |
| Memory (SRAM) | 0.0000 | 0.0119 | 0.0705 | 0.0824 | 0.08 |
| Register Group | 77.6614 | 0.5968 | 1.2723 | 79.5307 | 74.03 |
| Combinational Logic | 12.7387 | 11.1774 | 3.9034 | 27.8193 | 25.89 |
| Clock Network | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.00 |
| Sequential Logic | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.00 |
| Black Box | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.00 |
| **总计** | **90.4001** | **11.7861** | **5.2461** | **107.4324** | **100.00** |

- **静态功耗（Leakage）占比不到（5%）**：反映出本次使用的工艺库中静态功耗控制较好，亦符合 45nm CMOS 工艺特性。
- **寄存器相关功耗显著**：合计达 79.5 mW，占总功耗超 74%，这可能由于多头输入/输出缓存和数据暂存机制（如 Q/K/V 缓存寄存）带来较多寄存资源消耗。
- **SRAM 功耗极低**：因被视为宏单元（Macro/Black Box），仅参与接口建模，未仿真内部结构，真实功耗需后仿验证。

## 4.4 优化前后对比分析

为了提升 MHSA 加速器在面积与功耗上的综合表现，并在保证关键路径时序不显著恶化的前提下进行优化，本设计在初步综合完成后启用了多项高级优化指令，相关优化指令如 set compile_enable_datapath_opt true 等。

优化前后 MHSA 加速器在面积与功耗方面取得了明显改善，尽管时钟周期略有增加（从 6.43ns 增至 6.65ns），但整体能效大幅提升，对比如**??**所示。

表 5: 优化前后对比表

| 项目 | 优化前 | 优化后 | 相对变化 |
|---|---|---|---|
| 时钟周期（ns） | 6.43 | 6.65 | ↑ 3.42% |
| 面积（$um^2$） | 1,512,818.53 | 1,409,620.87 | ↓ 6.83% |
| 功耗（mW） | 144.93 | 107.43 | ↓ 25.86% |

分析总结：

- **时序方面**：由于超优化过程可能引入更复杂的逻辑路径，导致关键路径延迟略微增加，但仍处于系统可接受范围内。

- **面积方面**：在保证功能等价的前提下，优化后设计面积减少超过 10 万 $m^2$，表明大量冗余逻辑成功被结构重构和模块合并所消除。
- **功耗方面**：通过启用门控技术与数据通路优化，总功耗降低约 25%，尤其是组合逻辑与寄存器组的动态功耗收敛显著。

优化结果显示，通过合理配置综合参数，可以在不显著牺牲时序性能的前提下，获得更优的面积与功耗表现，为后续布局布线与低功耗设计奠定基础。

# 5 物理设计

物理设计作为连接前端逻辑综合与流片制造的中间环节，其核心任务是将门级网表中定义的逻辑结构映射为可制造的物理布局，并满足面积、时序、功耗和信号完整性等设计约束。整个物理设计过程主要包括五个阶段：设计初始化（design setup）、设计规划（design planning）、单元放置与优化（place_opt）、时钟树综合（clock_opt）以及布线优化（route_opt）。

将综合得到的 mhsa_acc_wrapper.mapped.v 门级网表文件放入指定文件夹，并在终端输入脚本里的物理设计命令，包括初始化、布局规划、单元放置、时钟数综合、布线。得到的版图如图 3所示。

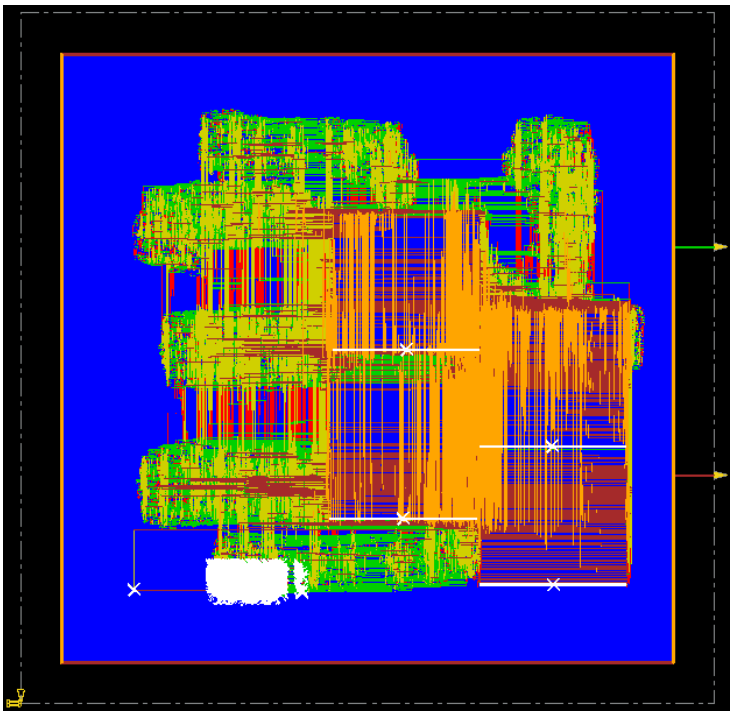

图 3: 物理设计版图

最终生成了布局布线完成后的后端门级网表。该网表在功能上保持与综合网表一致，同时针对布线后可能出现的时序问题进行了多项结构级优化。

与最初的综合网表相比，物理设计后的网表在单元数量上有所增加。造成这种变化的主要原因，是工具在布局过程中自动插入了大量缓冲器和反相器，用于驱动长距离布线或较大负载，以保证信号传播的可靠性和时序性能。此外，在一些关键路径上，工具还对逻辑门进行了尺寸调整或结构替换，以提升时序裕量。这些优化虽然会改变网表的结构，但不会影响电路的功能。

目前虽然还没有生成后仿时序文件（如.sdf）或寄生参数文件（如.spef），因此暂时无法进行精确的静态时序分析，但这个物理设计后的网表已经为下一阶段的仿真、功耗评估和版图导出打下了基础。接下来可以进一步提取布线延迟、进行时序签核，并最终导出标准版图文件（如.def 或.gds），完成整个设计流程。

# 6 总结

本次 MHSA 加速器综合设计在保证功能正确性的基础上，通过启用结构优化、模块合并、时序驱动等多项指令，有效降低了电路面积和动态功耗。在优化后，时钟周期为 6.65ns，面积为 1.41 $mm^2$，功耗为 107.4 mW，系统整体资源使用更为高效。

尽管当前结果较为理想，后续仍有以下优化空间：

- **时序方面**：可结合综合报告，针对关键路径插入缓冲或管线寄存器。
- **面积方面**：可进一步采用模块复用、数据路径共享等策略。
- **功耗方面**：加入时钟门控与更精细的功耗域划分。

物理设计中，通过脚本完成了各阶段操作。总体而言，本次设计完成了 mhsa 加速器从前端 RTL 综合到后端物理实现的完整流程，后续可继续进行后期分析、验证、封装、时序、功耗、面积、DRC/LVS 检查等用途。