

数字集成电路课程设计报告



上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

课程设计名称: Motion Estimation Circuit Design

姓 名: 汪嘉杭

学 号: 519021911336

小 组 成 员: 吴非, 贾鑫鹏, 王玉麟, 汪嘉杭

提 交 日 期: 2022/5/4

目录

1. 设计规范简介
2. 电路性能分析与电路结构设计
3. 电路 RTL 模型与仿真验证
4. 电路逻辑综合策略与综合结果
5. 电路物理实现与结果分析
6. 任务分工与设计总结

1.课程设计规范简介

对所要实现电路的设计规范进行描述，包括功能、性能等要求，

可参考课程设计的规范文件。

- 1) 每秒 60 帧 4K 视频 ($3840 \times 2160@60\text{fps}$) 的实时处理能力
- 2) 采用全搜索 ME 算法、支持 8×8 块大小的 SAD 计算、搜索区间为 $[-7,8]$
- 3) 芯片设计工艺：华力 55nm 工艺
- 4) 评价指标：电路的实时处理能力、芯片的 PPA (Performance or frequency, Power, Area)、输入/输出数据的带宽及其利用效率

理论压缩率：

原图片 bit 数： $3840 \times 2160 \times 8\text{bits}$

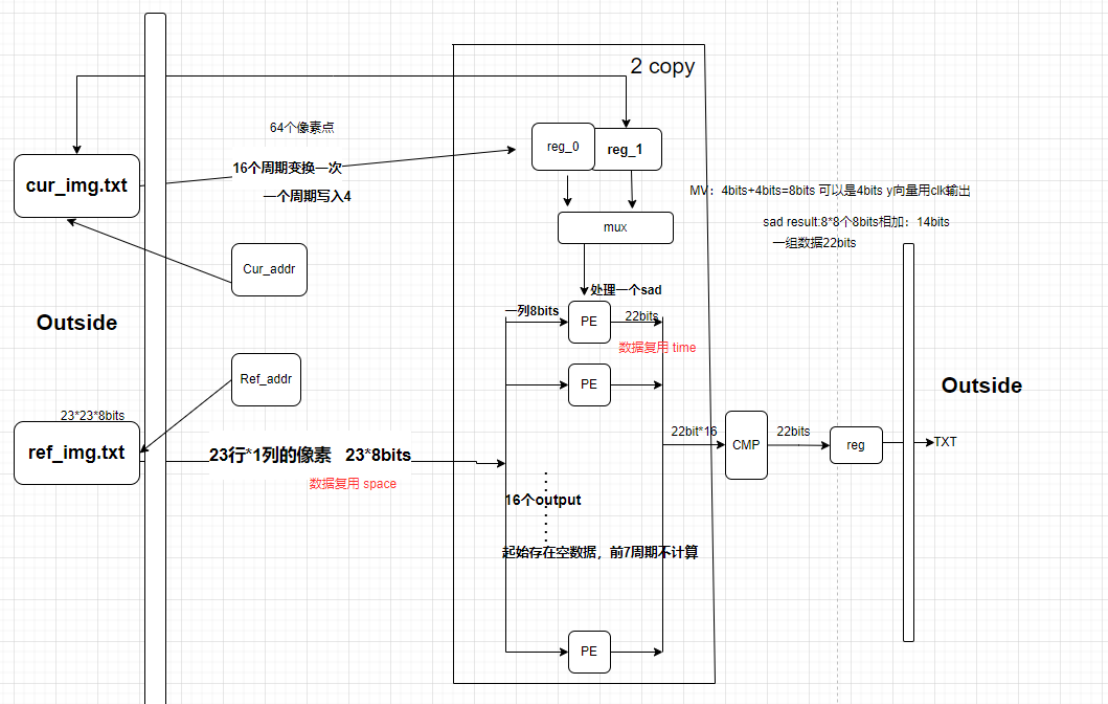
压缩后 $3840 \times 2160/8/8 \times 8\text{bits}$

显然压缩比率为 $1/64$

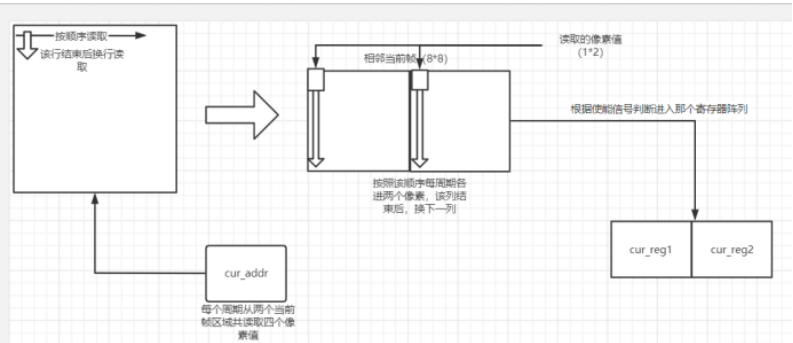
2.电路性能分析与电路结构设计

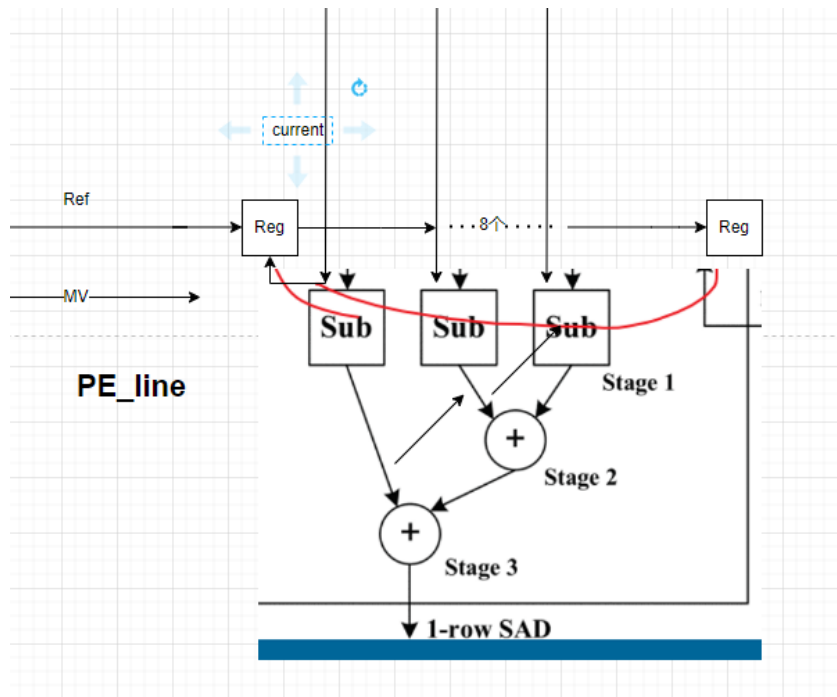
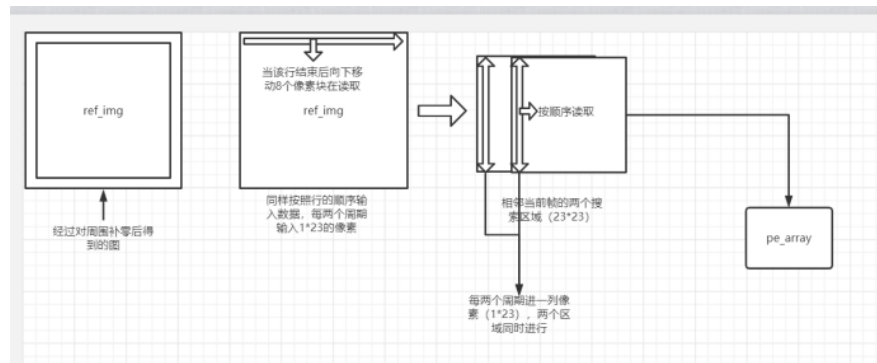
(1) 对于 ME、SHA256 和 NoC、RSIC-V 等电路，根据电路设计规范，分析电路的性能、吞吐率、（峰值）带宽、平均带宽和（实时）工作频率等指标，给出必要的计算公式和估算依据。在此基础上，给出基于模块化划分的电路硬件结构图，并进行说明。

整体架构：



如图，本架构的主要思想是每个周期计算一行 16 个 sad 块，然后更新参考帧数据，具体在 pe_line 图中的 8 个 reg。主要数据流动如下





虽然是参考帧的数据在不断流动，但是我们可以想像，这就好比当前块在搜索区域内滑动，这和卷积操作很相似，当前块为卷积核，不过不同之处在于我们一次计算一列内的所有值。

由于使用一套计算模块，当我们计算下一个当前块的时候会出现参考帧的数据接不上的情况，但是隔一个当前块刚好接上，但是由于需要，不能间隔输出，因此我们通过并行技术复制一份一样的模块，但是这样又会让性能过剩，为了减小发热，以及提高寿命，在一个周期停顿不计算，一个周期计算，这样性能不会改变。由于上述分析它和初始模块差了 16 个周期，因此也需要另一个 `cur_reg` 存储下一个当前块。

具体每个小模块：

`cur_addr` 和 `ref_addr` 产生需要的数据地址。

`cur_reg`: 存储当前块数据，有相对于当前 `sad` 二倍大小的空间（为了切换当前块的时候不停顿）。

`pe, pe_line`: 核心计算模块（同时也存储了参考帧数据）。

`cmp`: 比较一个周期来的 16 个 `sad` 值，选出最小的，并和全局的最小值比较，16 个周期后输出当前 `sad` 的最小值。

电路指标：

性能:

1/60 秒内需要计算 $3840 \times 2160 / 8 / 8 = 129600$ 个当前块的数据, 一个当前块需要 16 个周期, 200Mhz 带入计算出需要 0.010368s 计算一张图片, 计算 60 张需要 0.622s 小于 1s, 1s 内能计算约 96 张图片。

带宽:

```
module me[  
    input wire clk,  
    input wire rst,  
  
    input wire [15:0]cur_input0,  
    input wire [15:0]cur_input1,  
  
    input wire[3:0]ref0,input wire[3:0]ref1,input wire[3:0]ref2,input wire[3:0]ref3,input wire[3:0]ref4,  
    input wire[3:0]ref5,input wire[3:0]ref6,input wire[3:0]ref7,input wire[3:0]ref8,  
    input wire[3:0]ref9,input wire[3:0]ref10,input wire[3:0]ref11,input wire[3:0]ref12,  
    input wire[3:0]ref13,input wire[3:0]ref14,input wire[3:0]ref15,input wire[3:0]ref16,  
    input wire[3:0]ref17,input wire[3:0]ref18,input wire[3:0]ref19,input wire[3:0]ref20,  
    input wire[3:0]ref21,input wire[3:0]ref22,  
    output reg finish_flag,  
]
```

由图可见, 输入带宽: $2 + 32 + 4 \times 23 = 126\text{bits}$

输出带宽:

```
output reg finish_flag,  
output reg ber,  
output reg [22:0]cur_ad,  
output reg [22:0]ref_ad,  
output reg [13:0]min_sad0,  
output reg [13:0]min_sad1, //这个只能是reg?  
  
output reg finish_a_cur0,output reg finish_a_cur1,  
output reg [3:0]mv_x0,  
output reg [3:0]mv_y0,  
output reg [3:0]mv_x1,  
output reg [3:0]mv_y1
```

$2 + 46 + 28 + 2 + 16 = 94\text{bits}$

利用效率:

输入: 参考帧数据一直输入, 当前帧的数据也只是在换行的时候停顿 32 个周期, 而一行内一共有 3840 个周期, 这很微小, 因此可以近似为 100%

输出:

每 16 个周期输出一次, 因此效率为 1/16, 当然也可以进行改进为近 100%, 每个周期输出所有数据的 16 份中的一份, 让后续模块整合 16 个周期的数据为完整的数据即可。

PPA:

功耗: 51.2346mW

面积: 芯片面积: 1646578um³

(2) 对于加法器电路, 根据电路设计规范, 分析电路的性能瓶

颈, 给出必要的估算依据, 给出提升性能的方法和途径。

在此基础上, 给出基于模块化划分的电路硬件结构图, 并

进行说明。

理论分析：

1s 内一共要处理的数据量：1s * 60 * 3840 * 2160 * 8bit
= 3981312000bit，一次 sad 计算处理 8*8*8bit=512bit，因此 1s 内一共要处理 7776000 个块

$$\frac{1}{2 \times 10^8 \text{Hz}} = T \text{ (周期)} \\ \frac{1s}{T} \cdot x = 7776000 \div 16 \times 16 \\ x = 9.95 \approx 10 \text{ (每周处理 10 个 sad)}$$

由此得到最少一个周期需要计算 10 个 sad 值，为了避免电路过于复杂，我们直接采取一个周期计算一系列的 sad 值，也就是 16 个 sad 值，这也就意味着性能会超出大约 60%。

同时根据全加器的延迟为 0.17ns，频率为 200mhz

$1 \times 10^9 / (2 \times 10^8 \times 0.17) = 29.5$ 级，因此全加器一个周期可以有 29.5 级

而算一个 8bit 的加法，需要 8,4,2,1 共 4 级，一行 8 个点，加起来需要 3 级，一个 sad8 行，需要三级，一共 16 个 sad 相互比较，需要 4 级，一共需要 14 级远小于 29.5，满足要求。同时为了能够让频率更快，以及更好的区分计算 sad 值和比较 sad 值的功能，将其分成 pe 和 cmp 两个功能模块，两模块之间用 reg 分隔做流水线。

性能瓶颈：

根据上述的全加器级数，显然应该是数据从 pe 进到出 pe 的路径为关键路径，这是电路的性能瓶颈。

提升性能方法：

- 增加流水线级数
- 并行
- 采用更先进的架构（改进电路一个周期停顿一个周期计算的缺点）。

(3) 介绍自己在上述部分中的主要工作；

查找论文等相关资料，参与讨论设计架构。

3.电路 RTL 模型与仿真验证

(1) 简要描述电路的 RTL 模型设计，重点介绍电路的验证方法、验证平台的搭建、和验证结果；

数据预处理：

电路整体以当前帧和搜索区域的一列为整体进行输入，所以在选择输入之前，需要对图像进行预处理，对于先前帧而言，需要对其周围[-7,8]范围内进行补零操作，在补零完成后以列的顺序存进寄存器中，而对于当前帧，为了方便对所需块的读取，按照 1*8 的大小从左向右进行读入，即每次存储 8 个像素点，向右读取，直至本行读完，然后进行换行操作。（该部分操作由脚本完成）

代码：

1. `cur_addr.v` 和 `ref_addr.v`：

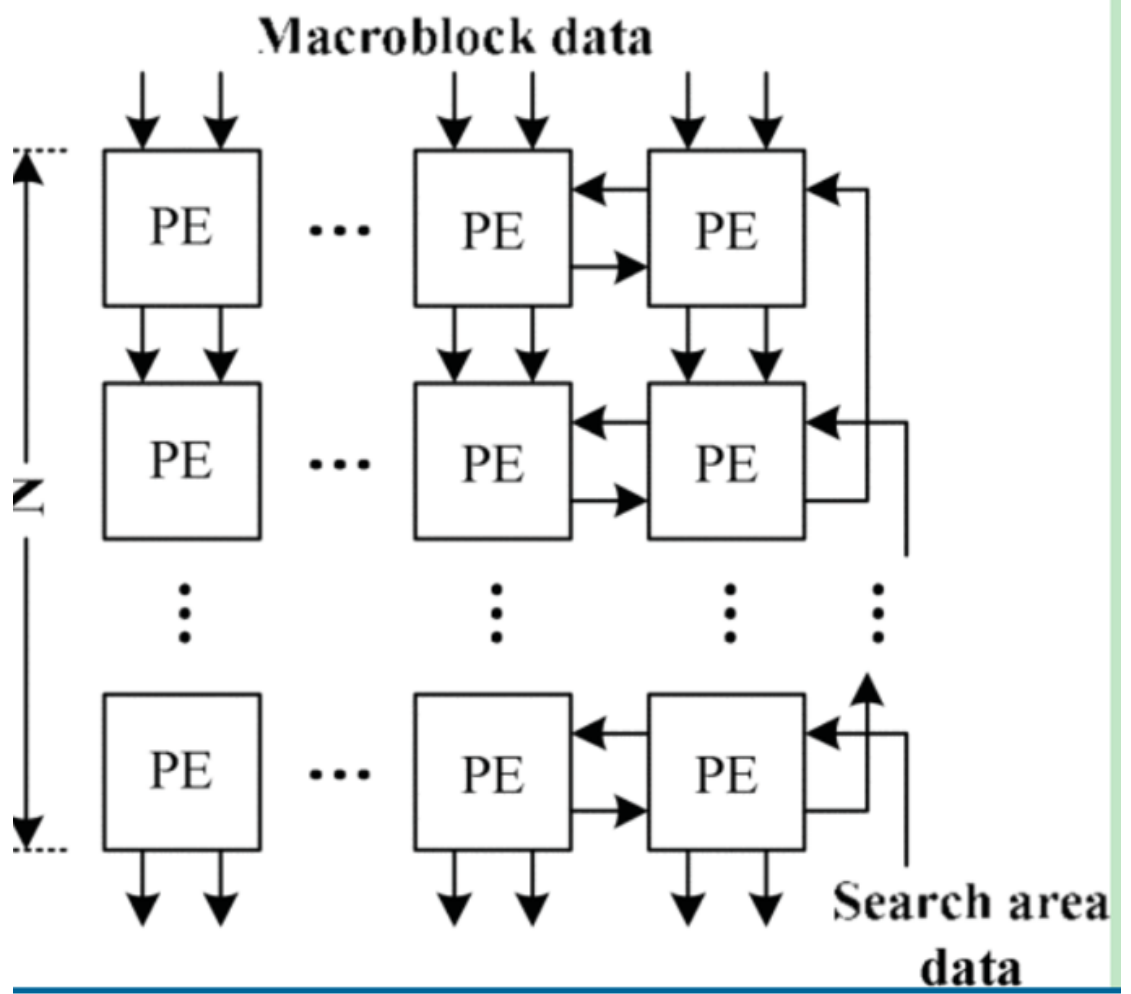
根据电路架构功能输出地址给 `me_tb.v`，`me_tb.v` 返回数据给芯片。

2. `cur_reg.v`：

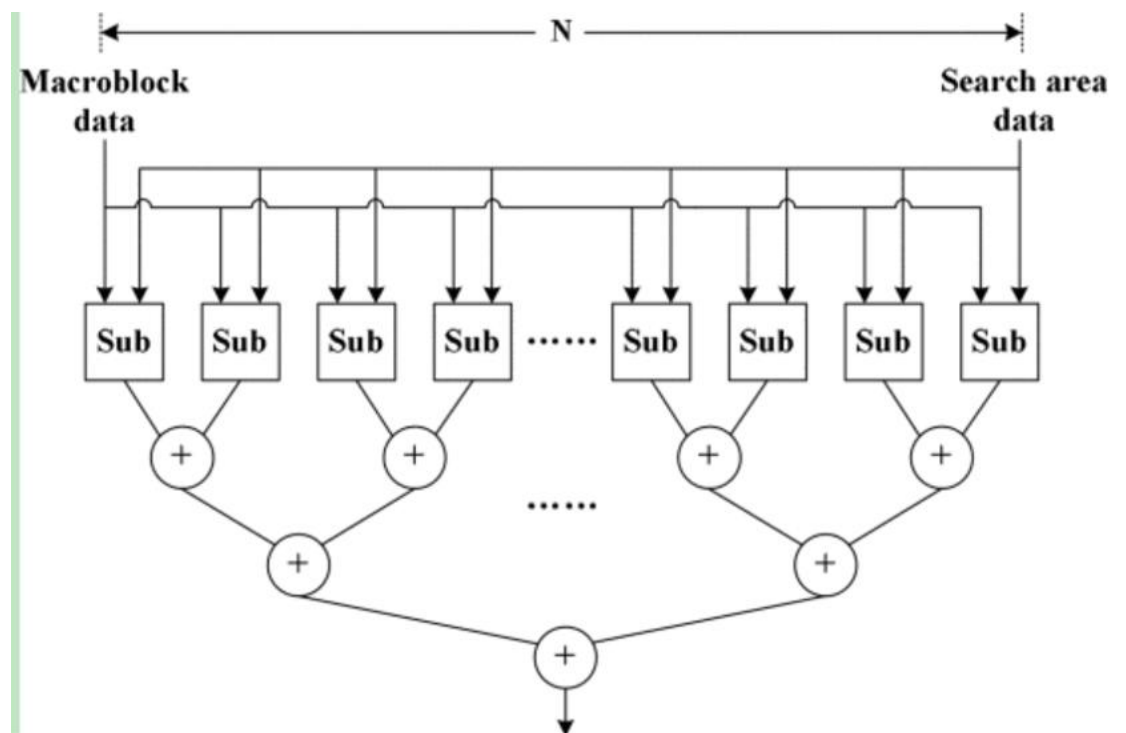
由于当前块数据需要保存 32 个周期不变，为了减小 io 数量，提高性能，使用了两个 `cur_reg` 模块来存储当前块数据，之所以用两个是因为一个供给计算模块计算，另一个同时更新下一个当前块的数据，避免停顿问题。

3. `pe_array.v,pe.v,pe_line.v` 如二板块所描述，电路核心计算模块，采用混合架构（收缩阵列架构和树结构混合）。

收缩阵列架构：



4. 树架构:



5. **cmp.v:**
在计算 16 个 sad 值的同时产生移动向量 mv_x 和 mv_y
6. **delay.v:**
当前块数据延迟，使能信号延迟
7. **output_tb.v:**
输出结果到文件

验证结果：

结果和参考结果对比，正确。

仿真性能：

由之前的性能分析，计算 60 张需要 0.622s 小于 1s，符合要求。

(2) 验证结果包括：功能点的验证、测试向量的多少、仿真性能
能的统计；

(3) 介绍自己在上述部分中的主要工作；

我负责前期 RTL 代码中的 mux 和 decoder，以及 testbench 的编写，包括 pe 模块例化文件 `pe_chip` 和顶层文件的例化 `me_chip` 的编写

4. 电路逻辑综合策略与综合结果

- (1) 给出电路简要的逻辑综合流程、参数设置以及综合结果；
- (2) 综合结果包括：时序结果、面积结果、功耗结果、资源使用结果和设计违例，附上相关的结果；
- (3) 介绍自己在上述部分中的主要工作；

1. 逻辑综合流程

第一步：设置 lib 以及 search path

```
set search_path "$search_path ../rtl/idct ../scripts /home/student/Desktop/Work  
space/55nm ../work ../mem/asdrispkb1p64x16cm2sw0/tt1p2v25c"  
set target_library "hu55npkldut_tt1p0v25c.db HL55LPGP3VDS_SL_A01_P2_TT1D8V1D2  
V25C.db asdrispkb1p64x16cm2sw0_lib.db"  
set symbol_library "hu55npkldut.sdb"  
set link_library "* $target_library $symbol_library"
```

图 1

第二步：顶层文件的书写与例化，读取 RTL

```
analyze -format verilog ../rtl/idct/cmp.v  
analyze -format verilog ../rtl/idct/cur_addr.v  
analyze -format verilog ../rtl/idct/cur_reg.v  
analyze -format verilog ../rtl/idct/delay.v  
analyze -format verilog ../rtl/idct/ref_addr.v  
analyze -format verilog ../rtl/idct/sub.v  
analyze -format verilog ../rtl/idct/pe_line.v  
analyze -format verilog ../rtl/idct/pe.v  
analyze -format verilog ../rtl/idct/pe_array.v  
analyze -format verilog ../rtl/idct/me.v  
analyze -format verilog ../rtl/idct/me_chip.v  
elaborate me_chip
```

图 2

```

module me_chip (
    input wire clk,
    input wire rst,

    input wire [15:0]cur_input0,
    input wire [15:0]cur_input1,

    output reg finish_flag,

    input wire[3:0]ref0,input wire[3:0]ref1,input wire[3:0]ref2,input wire[3:0]ref3,input wire[3:0]ref4,
    input wire[3:0]ref5,input wire[3:0]ref6,input wire[3:0]ref7,input wire[3:0]ref8,
    input wire[3:0]ref9,input wire[3:0]ref10,input wire[3:0]ref11,input wire[3:0]ref12,
    input wire[3:0]ref13,input wire[3:0]ref14,input wire[3:0]ref15,input wire[3:0]ref16,
    input wire[3:0]ref17,input wire[3:0]ref18,input wire[3:0]ref19,input wire[3:0]ref20,
    input wire[3:0]ref21,input wire[3:0]ref22,

    output reg ber,
    output reg [22:0]cur_ad,
    output reg [22:0]ref_ad,
    > output reg [13:0]min_sad0,
    > output reg [13:0]min_sad1, // 这个只能是reg?

    output reg finish_a_cur0,output reg finish_a_cur1,
    output reg [3:0]mv_x0,
    output reg [3:0]mv_y0,
    output reg [3:0]mv_x1,
    output reg [3:0]mv_y1
);

```

图 3.me_chip 顶层文件

共包括 230 个 pad，其中 136 个为输入 pad，94 个为输出 pad。

第三步：设置约束

时钟周期为 5ns，其他参数按照周期等比例缩放。

```

create_clock -period 5 [get_ports clk]
set_clock_uncertainty 0.25 [get_clocks clk]
set_clock_transition 0.1 [get_clocks clk]

##### YBR #####
set_clock_latency -source 4 [get_clocks clk]
set_clock_latency 0.3 [get_clocks clk]
##### YBR #####

# 2.2
set_input_delay -max 2.5 -clock clk [remove_from_collection [all_inputs] [get_ports clk]]
# 2.3
set_output_delay -max 2.5 -clock clk [all_outputs]

```

第四步：compile

2. 逻辑综合结果

1)时序结果

| Point | Incr | Path |
|---------------------------------------|--------|--------|
| ----- | | |
| clock clk (rise edge) | 0.00 | 0.00 |
| clock network delay (ideal) | 4.30 | 4.30 |
| inst_me/ber_reg/CK (SVL_FDPRBQ_1) | 0.00 # | 4.30 r |
| inst_me/ber_reg/Q (SVL_FDPRBQ_1) | 0.23 | 4.53 r |
| inst_me/ber (me) | 0.00 | 4.53 r |
| HPDWUW1416DGP_ber/PAD (HPDWUW1416DGP) | 1.90 | 6.43 r |
| ber (out) | 0.00 | 6.43 r |
| data arrival time | | 6.43 |
| | | |
| clock clk (rise edge) | 5.00 | 5.00 |
| clock network delay (ideal) | 4.30 | 9.30 |
| clock uncertainty | -0.25 | 9.05 |
| output external delay | -2.50 | 6.55 |
| data required time | | 6.55 |
| ----- | | |
| data required time | | 6.55 |
| data arrival time | | -6.43 |
| ----- | | |
| slack (MET) | | 0.12 |

1

2) 面积及资源使用结果

总面积为 1692353 平方微米，详细资源使用结果见下图。

```
Report : area
Design : me_chip
Version: 0-2018.06-SP1
Date   : Mon Apr 18 13:38:23 2022
*****

Library(s) Used:

    hu55npkldut tt1p0v25c (File: /home/student/Desktop/Workspace/55nm/hu55npkldut tt1p0v25c.db)
    HL55LPGP3VDS_SL_A01_TT1D8V1D2V25C (File: /home/student/Desktop/Workspace/55nm/
    HL55LPGP3VDS_SL_A01_P2_TT1D8V1D2V25C.db)

Number of ports:          229712
Number of nets:           505172
Number of cells:          244949
Number of combinational cells: 198285
Number of sequential cells:  38747
Number of macros/black boxes:  220
Number of buf/inv:        91004
Number of references:      4

Combinational area:       543527.865505
Buf/Inv area:             88170.317663
Noncombinational area:    301825.717290
Macro/Black Box area:     847000.000000
Net Interconnect area:    undefined (Wire load has zero net area)

Total cell area:          1692353.582795
Total area:               undefined
1
```

3) 功耗结果

除去 io_pad 的功耗，芯片的功耗为 50.7025mW

| Power Group (%) Attrs | Internal Power | Switching Power | Leakage Power | Total Power |
|----------------------------|-------------------|--------------------|------------------|----------------|
| io_pad (81.32%) | 18.4651 | 202.2242 | 2.5926e+03 | 220.6925 |
| memory (0.00%) | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| black_box (0.00%) | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| clock_network (0.00%) | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| register (15.30%) | 40.1817 | 1.2585 | 6.9610e+04 | 41.5111 |
| sequential (0.00%) | 4.6645e-05 | 5.5282e-06 | 39.3468 | 9.1519e-05 |
| combinational (3.39%) | 4.5880 | 4.4391 | 1.6418e+05 | 9.1914 |
| Total | 63.2349 mW | 207.9219 mW | 2.3642e+05 nW | 271.3950 |

4) 设计违例

最小电容违例

| min_capacitance | | | | |
|-----------------|-------------------------|-----------------------|-------|------------|
| Net | Required Capacitance | Actual Capacitance | Slack | |
| clk | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[0] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[1] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[2] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[3] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[4] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[5] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[6] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[7] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[8] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[9] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[10] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[11] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[12] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[13] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[14] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input0[15] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input1[0] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input1[1] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input1[2] | 6.43 | 1.43 | -5.00 | (VIOLATED) |
| cur_input1[3] | 6.43 | 1.43 | -5.00 | (VIOLATED) |

(3)

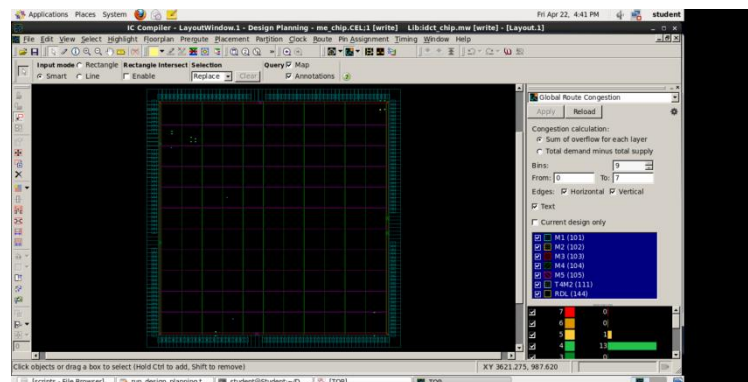
这部分全部由我负责。

5. 电路物理实现与结果分析

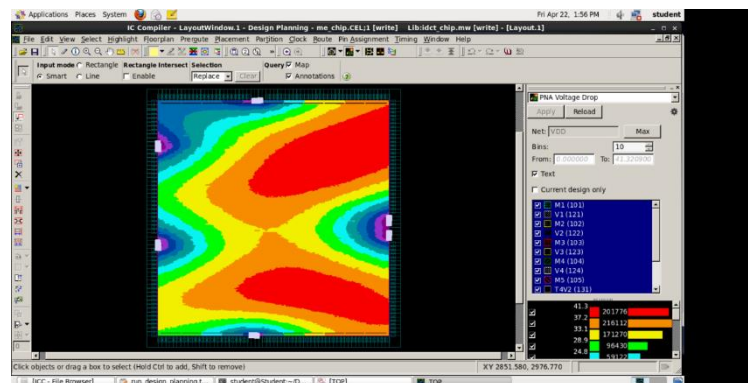
- (1) 给出利用 ICC 设计电路的各步骤结果；
- (2) 步骤包括：局部规划、布局、时钟树综合和布线；
- (3) 结果包括：完成布局规划的芯片和完成布局布线的芯片概貌图、时序结果、面积结果、功耗结果、芯片的压降、芯片的拥塞分析、资源使用结果和设计违例；
- (4) 介绍自己在上述部分中的主要工作；

ICC 设计电路包括，版图规划，布局，时钟树综合，布线
一下是各步骤结果；

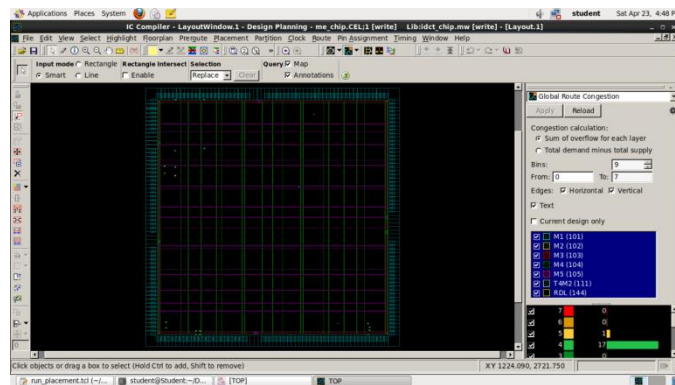
版图规划：芯片概貌图



压降

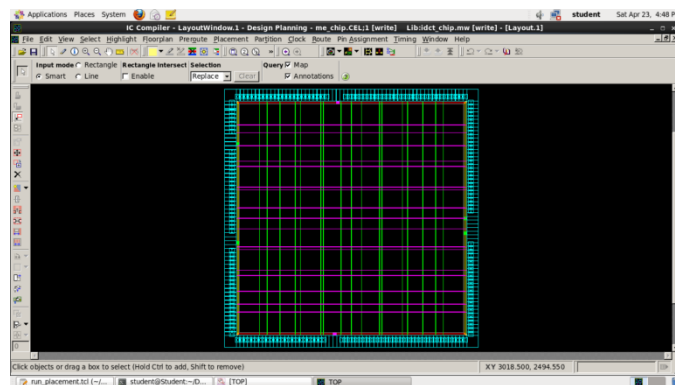


拥塞



Placement

芯片概貌图



时序结果

```
student@Student:~/Desktop/Workspace/IDCT_55nm_mem/ICC/work
File Edit View Search Terminal Help
inst_me/ref_addr/mult_48/FS_1/SUM_17_ (ref_addr_DW01_add 1)
0.00 7.48 r
inst_me/ref_addr/mult_48/PRODUCT_19_ (ref_addr_DW02_mult 0)
0.00 7.48 r
inst_me/ref_addr/U51/X (SVL_E02_S_2)
0.00 * 8.28 f
inst_me/ref_addr/ad1[19] (ref_addr)
0.00 8.28 f
inst_me/ref_ad_reg_19/_D (SVL_FDPRBQ_F_2)
0.16 * 8.44 f
data arrival time
8.44

clock clk (rise edge)
5.00 5.00
clock network delay (ideal)
4.30 9.30
clock uncertainty
-0.25 9.05
inst_me/ref_ad_reg_19/_CK (SVL_FDPRBQ_F_2)
0.00 9.05 r
library setup time
-0.25 8.80
data required time
8.80
data arrival time
-8.44

slack (MET)
0.36

1
icc_shell>
```


时钟树综合

时序结果

```
student@Student:~/Desktop/Workspace/IDCT_55nm_mem/ICC/work
File Edit View Search Terminal Help
inst me/ref_addr/U58/X (SVL_AN2_0P5) 0.15 * 7.14 r
inst me/ref_addr/U56/X (SVL_AN2_0P5) 0.12 * 7.26 r
inst me/ref_addr/U54/X (SVL_AN2_0P5) 0.11 * 7.36 r
inst me/ref_addr/U52/X (SVL_AN2_0P5) 0.12 * 7.48 r
inst me/ref_addr/U51/X (SVL_E02_5_2) 0.72 * 8.20 f
inst me/ref_addr/ad1[19] (ref_addr) 0.00 8.20 f
inst me/ref_ad_reg_19_D (SVL_F0PRBQ_F_2) 0.16 * 8.37 f
data arrival time 8.37

clock clk (rise edge) 5.00 5.00
clock network delay (ideal) 4.30 9.30
clock uncertainty -0.20 9.10
inst me/ref_ad_reg_19_D /CK (SVL_F0PRBQ_F_2) 0.00 9.10 r
library setup time -0.24 8.86
data required time 8.86
-----
data required time 8.86
data arrival time -8.37
-----
slack (MET) 0.50

l
icc_shell>
```

Skew

```
icc_shell> report_clock_tree -summary

*****
Report : clock tree
Design : me_chip
Version: 0-2018.06-SP1
Date : Tue May 3 22:13:46 2022
*****

Information: Float pin scale factor for the 'max' operating condition of scenario 'default' is set to 1.000 (CTS-375)

===== Clock Tree Summary =====
Clock Sinks CTBuffers ClkCells Skew LongestPath TotalDR
C BufferArea
-----
clk 22200 594 595 0.9478 4.1307 281
3120.3157
l
icc_shell>
```

布线

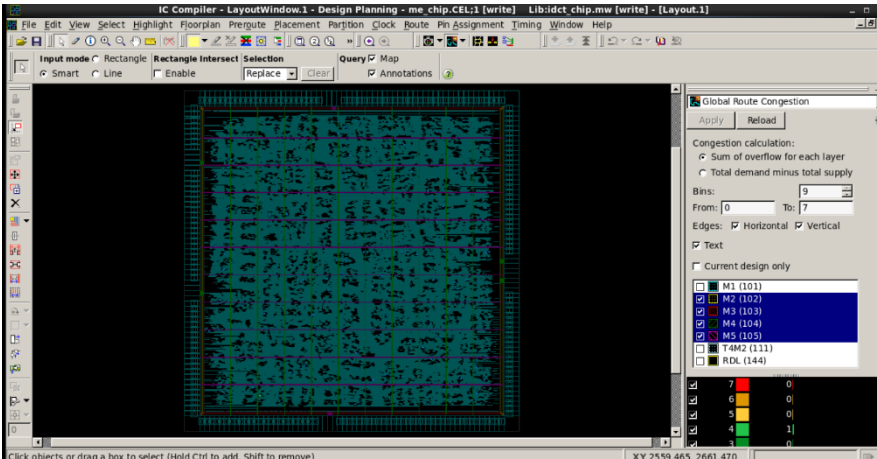
时序结果

```
File Edit View Search Terminal Help
inst me/ref_addr/mult_48/U26/X (SVL_E02_1) 0.10 & 6.07 r
inst me/ref_addr/mult_48/U18/X (SVL_AN2_0P5) 0.12 & 6.20 r
inst me/ref_addr/mult_48/S4_3/S (SVL_A0DF_1) 0.18 & 6.37 f
inst me/ref_addr/mult_48/U05/X (SVL_E02_1) 0.07 & 6.45 f
inst me/ref_addr/mult_48/FS_1/A_12 (ref_addr_DW01_add_1) 0.00 6.45 f
inst me/ref_addr/mult_48/FS_1/U59/X (SVL_NR2_0P5) 0.11 & 6.56 r
inst me/ref_addr/mult_48/FS_1/U2/X (SVL_INV_1) 0.04 & 6.60 f
inst me/ref_addr/mult_48/FS_1/U53/X (SVL_A0I21_0P5) 0.15 & 6.74 r
inst me/ref_addr/mult_48/FS_1/U51/X (SVL_A0I21_0P5) 0.10 & 6.85 f
inst me/ref_addr/mult_48/FS_1/U43/X (SVL_A0I21_0P5) 0.17 & 7.02 r
inst me/ref_addr/mult_48/FS_1/U41/X (SVL_A0I21_0P5) 0.10 & 7.12 f
inst me/ref_addr/mult_48/FS_1/U33/X (SVL_A0I21_0P5) 0.16 & 7.28 r
inst me/ref_addr/mult_48/FS_1/U32/X (SVL_E02_0P5) 0.14 & 7.41 r
inst me/ref_addr/mult_48/FS_1/SUM_17 (ref_addr_DW01_add_1) 0.00 7.41 r
inst me/ref_addr/mult_48/PRODUCT_19 (ref_addr_DW02_mult_0) 0.00 7.41 r
inst me/ref_addr/U51/X (SVL_E02_5_2) 0.01 c 8.03 f
inst me/ref_addr/ad1[19] (ref_addr) 0.00 8.03 f
inst me/ref_ad_reg_19_D (SVL_F0PRBQ_F_2) 0.15 c 8.18 f
data arrival time 8.18

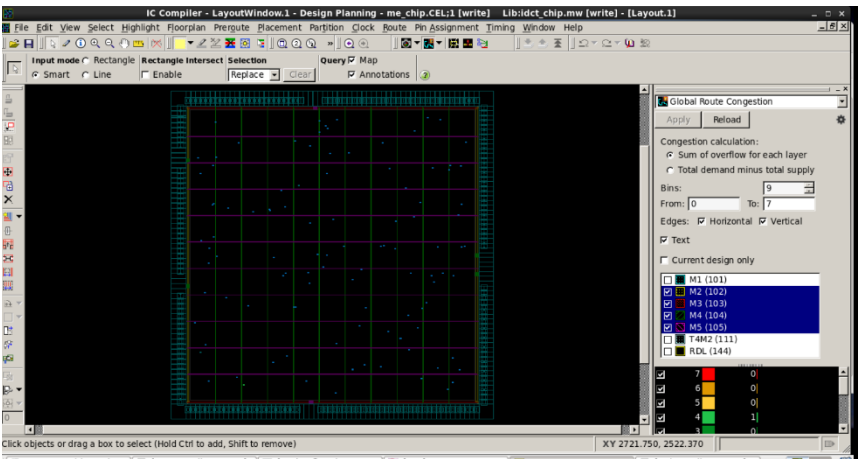
clock clk (rise edge) 5.00 5.00
clock network delay (ideal) 4.30 9.30
clock reconvergence pessimism 0.00 9.30
clock uncertainty -0.20 9.10
inst me/ref_ad_reg_19_D /CK (SVL_F0PRBQ_F_2) 0.00 9.10 r
library setup time -0.23 8.87
data required time 8.87
-----
data required time 8.87
data arrival time -8.18
-----
slack (MET) 0.69

l
icc_shell>
```

Finish



拥塞



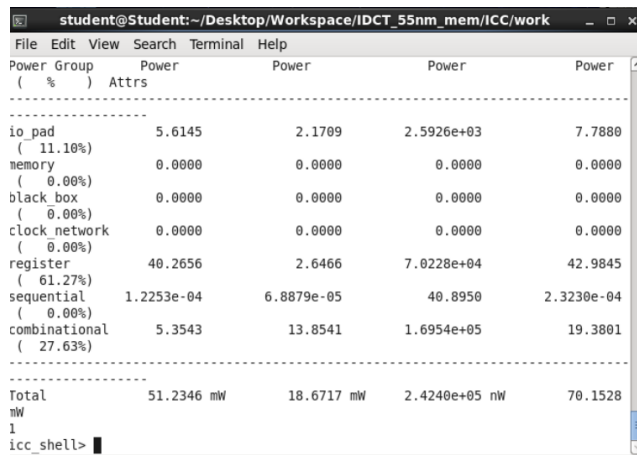
面积结果

```
Area
-----
Combinational Area:  497640.067265
Noncombinational Area:
301937.997286
Buf/Inv Area:        44641.519311
Total Buffer Area:    15376.76
Total Inverter Area:  29264.76
Macro/Black Box Area: 847000.000000
Net Area:             0.000000
Net XLength           : 3409100.50
Net YLength           : 5150305.00
-----
Cell Area:           1646578.064551
Design Area:         1646578.064551
Net Length           : 8559406.00
```

面积利用率

```
icc_shell> get_utilization
Top level utilization: 0.161928
0.161928
icc_shell>
```

功耗结果



| Power Group | (%) | Attrs | Power | Power | Power | Power |
|---------------|-----------|-------|------------|------------|---------------|------------|
| io_pad | (11.10%) | | 5.6145 | 2.1709 | 2.5926e+03 | 7.7880 |
| memory | (0.00%) | | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| black box | (0.00%) | | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| clock network | (0.00%) | | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| register | (61.27%) | | 40.2656 | 2.6466 | 7.0228e+04 | 42.9845 |
| sequential | (0.00%) | | 1.2253e-04 | 6.8879e-05 | 40.8950 | 2.3230e-04 |
| combinational | (27.63%) | | 5.3543 | 13.8541 | 1.6954e+05 | 19.3801 |
| Total | | | 51.2346 mW | 18.6717 mW | 2.4240e+05 nW | 70.1528 |
| 1 | | | | | | |

资源使用结果

```
Design Rules
-----
Total Number of Nets:      212172
Nets With Violations:      359
Max Trans Violations:      0
Max Cap Violations:        3
Max Fanout Violations:     356
-----

Hostname: Student

Compile CPU Statistics
-----
Resource Sharing:           0.00
Logic Optimization:         0.00
Mapping Optimization:       182.91
-----

Overall Compile Time:       197.09
Overall Compile Wall Clock Time: 199.59
-----
```

从上述结果可以看到，我们的 ICC 设计，总体来看效率很高，使用了 230 个 pad，面积、资源使用结果较多，导致面积利用率较低，在最初阶段存在一定的拥塞，但在经过优化后，拥塞得到了完美解决。压降最大为 40mV<50mV，时序也均满足，符合要求。

(4)

这部分由王玉麟同学负责，结果不符时，与负责 RTL 的同学沟通后，我再进行逻辑综合，让后将生成的脚本给王同学进行后续工作。

6.任务分工与设计总结

简要说明课程设计中各小组成员的任务分工，课程设计过程中遇到的主要问题，解决思路及其它问题。

吴非：

电路架构图绘制和架构的设计

RTL 代码中 `pe,pe_line` 基本计算模块，`cmp` 比较器，输出到文件以及延迟模块的编写(包括相关模块的 `testbench`)

代码最后的整合和测试，以及后续逻辑，物理综合不符合后的修改。

问题：

Verilog 代码基本语法的熟悉：

问助教，上网查阅，同学讨论，编写 `testbench` 验证想法是否正确。

电路架构有功能 bug：

查阅资料，小组讨论修改架构和 RTL 代码。

代码调试：

换行后 32 个周期的停顿需要很准确的控制各个使能信号，通过查看波形调试，这部分所占用的时间甚至超过代码编写的时间。

前期讨论和进展不顺利：

定期督促和开会讨论。

贾鑫鹏：

参与架构设计的讨论，对数据的预处理。

RTL 代码中，地址产生模块 `cur_addr,ref_addr`，与当前帧 存储模块 `cur_reg` 的编写与调试。

后期代码运行中结果存在 bug，参与调试与解决。

问题：

Verilog 代码熟悉程度不高。

查资料熟悉代码内容，并对代码进行修改。

数据预处理

通过 `cpp` 脚本进行处理，在过程中出现了多次错误的处理

最终通过与原数据的多次对比，得到正确结果。

小模块中时序问题。

通过观察输出波形进行调试，改变使能信号与判断信号达到要求。

汪嘉杭：

参与电路架构的讨论

前期 RTL 代码中的 mux 和 decoder，以及 testbench

pe 模块例化文件 pe_chip 和顶层文件的例化 me_chip 的编写

逻辑综合

问题及解决思路

时序不满足要求

通过分析时序报告来发现优化策略

王玉麟：

参与电路架构的讨论

RTL 代码中，cmp 比较器初版，输出到文件及 testbench

物理综合

问题及解决思路

电路架构有功能 bug:

查阅资料，小组讨论修改架构和 RTL 代码

输入违例

输入时先经过 reg 再输入

电压压降，

先定义全部 pad 位置，调整电源 pad 位置

时序

调整 pad 位置，使布线距离更短

电路 skew 较大

优化时，设置调整参数

绕线偏差

实际 placement 存在误差，每次布局优化后，都需 route_zrt_global 重新评

估