



数字集成电路设计课程设计报告

基于 jpegencode 设计的 ASIC 流程

学 院 电子信息工程学院
班 级 电子信息科学与技术2021-2
学 号
姓 名
指导老师 张德学

2024 年 9 月 2 日

目录

1 设计概述	4
1.1 工程的建立	4
1.1.1 工程文件结构	4
1.2 工程的管理	5
1.2.1 版本控制	5
1.2.2 EDA 环境	5
2 RTL 仿真	6
2.1 仿真环境	6
2.2 仿真结果	8
3 综合与形式验证	10
3.1 原设计 RTL 代码的修改	10
3.2 DC 综合	11
3.2.1 环境配置	12
3.2.2 读入设计	12
3.2.3 设置约束	12
3.2.4 查看报告	14
3.3 形式验证	16
3.3.1 形式验证步骤	16
4 自动布局布线	18
4.1 设计输入	18
4.2 布局规划	19
4.2.1 电源引脚规划	19
4.2.2 芯片大小估算	22
4.2.3 添加 PAD filler 、电源环、电源条等	23
4.2.4 规划 stdcell 摆放区域	24
4.3 其余步骤	25
4.3.1 时序约束	25
4.3.2 Place	26
4.3.3 时钟树综合CTS、Post-CTS优化	26
4.3.4 Route	26
4.3.5 DFM	26
4.3.6 数据导出	26
4.3.7 总结	27

5 物理验证	28
5.1 准备 IC5141 环境	28
5.1.1 导入库文件	28
5.2 为电源 PAD 加 Label	28
5.3 进行 LVS 检查	29
5.4 ANT 检查与修正	30
5.5 DRC检查与修正	30
5.6 设计数据导出	30
6 PT 时序分析	32
6.1 环境配置	32
6.2 查看报告	34
7 设计心得	36

1 设计概述

本课设基于已有 jpegencode 设计¹，以及数字集成电路设计课程提供的 aes_asic 设计的相关脚本等，完成了以下流程：

- 使用 Modelsim 进行仿真验证
- 使用 Synopsys DC 进行综合，Synopsys Formality 进行形式验证
- 使用 Astro 进行布局布线
- 使用 IC5141 和 Calibre 进行后仿真验证
- 使用 Synopsys PT 进行时序分析

本课设项目已上传至 GitHub 仓库²。

1.1 工程的建立

设计采用 Git 进行版本控制。首先将原设计 Fork 至自己的仓库，然后克隆至本地，如下所示：

```
git clone https://github.com/Sh4peshifting/jpegencode
```

1.1.1 工程文件结构

参照 aes_asic 的工程文件结构，整理课设工程文件结构。执行 tree -L 1 -AC 命令，目录结构如下所示：

```
.  
├── Astro  
├── DC  
├── document  
├── FM_RTL_PostDC  
├── jpeg_encode_asic  
├── jpeg_encode_core  
├── layout  
├── lib  
├── original  
├── PT  
└── README.md  
└── run_dc.sh  
└── run_fm.sh  
└── run_sim.sh
```

图 1：工程文件结构

¹<https://github.com/freecores/jpegencode>

²<https://github.com/Sh4peshifting/jpegencode>

1.2 工程的管理

1.2.1 版本控制

编写 `.gitignore` 文件，忽略不需要版本控制的文件。然后配置好 Git 的用户名和邮箱，如下所示：

```
1 git config --global user.name "Your Name"  
2 git config --global user.email "Your Email"
```

配置SSH Key。将本地仓库与远程仓库关联，之后执行以下命令，将本地仓库更改推送至远程仓库：

```
1 git add .  
2 git commit -m "Your commit message"  
3 git push
```

1.2.2 EDA 环境

在数字集成电路设计课程所提供的虚拟机上执行如下命令，配置 EDA 环境：

```
1 source /opt/eda_env.sh
```

如果要使用 Astro 进行布局布线，IC5141 和 Calibre 进行后仿真验证，则需要使用以下命令配置相应环境：

```
1 source /opt/eda_env_old.sh
```

2 RTL 仿真

2.1 仿真环境

本设计的仿真测试文件为 `jpeg_top_tb.v`, 仿真测试文件的主要内容如下:

```

1   `timescale 1ps / 1ps

2

3   module jpeg_top_tb;
4

5     reg end_of_file_signal;
6     reg [23:0]data_in;
7     reg clk;
8     reg rst;
9     reg enable;
10    wire [31:0]JPEG_bitstream;
11    wire data_ready;
12    wire [4:0]end_of_file_bitstream_count;
13    wire eof_data_partial_ready;
14
15 // Unit Under Test
16   jpeg_top UUT (
17     .end_of_file_signal(end_of_file_signal),
18     .data_in(data_in),
19     .clk(clk),
20     .rst(rst),
21     .enable(enable),
22     .JPEG_bitstream(JPEG_bitstream),
23     .data_ready(data_ready),
24     .end_of_file_bitstream_count(end_of_file_bitstream_count),
25     .eof_data_partial_ready(eof_data_partial_ready));
26
27 initial
28 begin : STIMUL
29   #0
30   rst = 1'b1;
31   enable = 1'b0;
32   end_of_file_signal = 1'b0;
33   #10000;

```

```

34      rst = 1'b0;
35      enable = 1'b1;
36      // data_in holds the red, green, and blue pixel values
37      // obtained from the .tif image file
38      data_in <= 24'b0001101100101001101101110;
39      #10000;
40      data_in <= 24'b00011011101010001101111;
41      #10000;
42      /* ... other pixel values */
43      data_in <= 24'b0001100110011110001010111;
44      #10000;
45      #130000;
46      enable <= 1'b0;
47
48      #2000000;
49
50      $finish;
51  end // end of stimulus process
52
53  always
54  begin : CLOCK_clk
55      //this process was generated based on formula: 0 0 ns, 1 5 ns -r 10 ns
56      //#<time to next event>; // <current time>
57      clk = 1'b0;
58      #5000; //0
59      clk = 1'b1;
60      #5000; //5000
61  end
62
63  always @ (JPEG_bitstream or data_ready)
64  begin : JPEG
65      if (data_ready==1'b1)
66          $display("%h", JPEG_bitstream);
67  end
68  endmodule

```

原设计未提供 Modelsim 的仿真脚本，故需要自行编写仿真脚本，仿真脚本如下：

```

1 vlib work
2 vmap work work
3 vlog -work work ../../rtl/*.v jpeg_top_tb.v
4 vsim -voptargs=+acc work.jpeg_top_tb
5 add wave /*
6 run 200000000.00 ns

```

然后执行 `vsim -do jpeg_sim.do` 即可进行仿真。

2.2 仿真结果

仿真结果如图 2 所示，可以看到仿真结果中包含了 JPEG 编码后的比特流。



图 2: 仿真结果

该编码器的输入是 24 位数据总线，红色像素、绿色像素和蓝色像素各占 8 位。信号“`data_in`”包含红色像素值（`[7:0]` 位）、绿色像素值（`[15:8]` 位）和蓝色像素值（`[23:16]` 位）。每个时钟周期应有 24 位数据用于新像素。输入应从图像左上角的 8×8 像素块开始，从左上角像素开始，向右，然后向下到第二行，依此类推。在图像的第一个 8×8 像素块之后，需要至少延迟 13 个时钟周期才能输入下一个块。然后在此延迟之后，“`enable`” 信号应在一个时钟周期内变低，然后在下一个时钟周期输入下一个 8×8 像素块时，“`enable`” 应变高。图像的第二个 8×8 像素块应以与第一个块相同的格式输入。第二个 8×8 像素块将位于第一个块的右侧。应该从图像的左上角向右移动，直到到达图像的右边缘。然后向下移动到第二行并继续，直到图像的最后一行。确保在每个块之间插入 13 个或更多时钟周期的延迟。这是必需的，否则前一个块的计算将尚未完成。

图像数据需要以完整的 8×8 像素块输入。例如，如果图像是 90×90 像素，则需要将像素数据扩展为 96×96 像素。这通常是通过将边缘像素值重复到每行或每列的额外像素上来实现的。

输入最后一个 8×8 像素数据块时，在块开头置位信号“`end_of_file_signal`”。这让其知道没有更多块需要输入，剩余的位将被输出，即使它们没有填满完整的 32 位。如果最后的 JPEG 位不是完整的 32 位，则信号“`eof_data_partial_ready`”将在一个时钟周期内保持高电平，并且信号“`JPEG_bitstream`”中的位将在一个时钟周期内保持有效。信号“`end_of_file_bitstream_count`”中的值表示在图像文件末尾信号“`JPEG_bitstream`”中有多少位有效。

输出是信号“`JPEG_bitstream`”，它是一个 32 位宽度的总线。位是按大端对齐的，因此占据位置 31-24 的位是输出的前 8 位，后面是 23-16，依此类推。当信号

“data_ready” 变为高电平时，信号 “JPEG_bitstream” 中的数据有效。位流数据将在第一个块输入后不久输出，并在像素数据输入到核心时继续输出。

原始项目中包含一个示例测试文件 “jpeg_top_tb.v”。这给出了运行核心所需时间的示例。来自图像文件 “ja.tif” 的数据用作测试台中的像素数据输入。核心的输出位于文件 “ja_bits_out.v” 中。这些位用于创建 jpeg 图像 “ja.jpg”。需要单独创建文件的标头。JPEG 编码器核心不会创建 jpeg 标头文件。此外，需要在 JPEG 位后添加 “FFD9”（扫描结束标记）才能完成 JPEG 文件。此核心仅输出由 Huffman 码创建的 JPEG 位。可以使用 MATLAB 程序从单独的标头和从此 JPEG 编码器创建的 jpeg 位创建 “ja.jpg” 图像。

3 综合与形式验证

3.1 原设计 RTL 代码的修改

为了进行后续自动布局布线步骤，在原设计顶层模块 jpeg_top 上添加了 PAD，主要代码如下：

```

1 module jpeg_asic(
2     PAD_clk_i,
3     PAD_RST_i,
4     PAD_eof_signal_i,
5     PAD_enable_i,
6     PAD_dat_i,
7     PAD_jpeg_bitstr_o,
8     PAD_dat_rdy_o,
9     PAD_eof_bitstr_cnt_o,
10    PAD_eof_dat_partial_rdy_o
11 );
12
13    input PAD_clk_i;
14    input PAD_RST_i;
15    input PAD_eof_signal_i;
16    input PAD_enable_i;
17    input [23:0] PAD_dat_i;
18    output [31:0] PAD_jpeg_bitstr_o;
19    output PAD_dat_rdy_o;
20    output [4:0] PAD_eof_bitstr_cnt_o;
21    output PAD_eof_dat_partial_rdy_o;
22
23    wire      clk;
24    wire      rst;
25    wire      end_of_file_signal;
26    wire      enable;
27    wire      [23:0] data_in;
28    wire      [31:0] JPEG_bitstream;
29    wire      data_ready;
30    wire      [4:0] end_of_file_bitstream_count;
31    wire      eof_data_partial_ready;

```

```

32    //PAD
33
34    PLBI8F U_clk_i      (.D( clk ), .P( PAD_clk_i ), .A( 1'b0 ), .CONOF( 1'b1 )
35      ), .NEN( 1'b0 ), .PD( 1'b0 ), .PEN( 1'b0 ), .PU( 1'b1 ), .SONOF( 1'b0 ) );
36    PLBI8F U_rst_i      (.D( rst ), .P( PAD_rst_i ), .A( 1'b0 ), .CONOF( 1'b1 )
37      ), .NEN( 1'b0 ), .PD( 1'b0 ), .PEN( 1'b0 ), .PU( 1'b1 ), .SONOF( 1'b0 ) );
38
39    PLBI8F U_eof_signal_i   (.D( end_of_file_signal ), .P( PAD_eof_signal_i
40      ), .A( 1'b0 ), .CONOF( 1'b1 ), .NEN( 1'b0 ), .PD( 1'b0 ), .PEN( 1'b0 ), .
41      PU( 1'b1 ), .SONOF( 1'b0 ) );
42
43    /* ... other input ports */
44
45    //out
46
47    PLBI8F U_dat_o_0 (.D( ), .P( PAD_jpeg_bitstr_o[0] ), .A( JPEG_bitstream
48      [0] ), .CONOF( 1'b0 ), .NEN( 1'b1 ), .PD( 1'b0 ), .PEN( 1'b1 ), .PU( 1'b1 )
49      , .SONOF( 1'b0 ) );
50    PLBI8F U_dat_o_1 (.D( ), .P( PAD_jpeg_bitstr_o[1] ), .A( JPEG_bitstream
51      [1] ), .CONOF( 1'b0 ), .NEN( 1'b1 ), .PD( 1'b0 ), .PEN( 1'b1 ), .PU( 1'b1 )
52      , .SONOF( 1'b0 ) );
53
54    /* ... other output ports */
55
56    jpeg_top jpeg_top_inst(
57      .clk(clk),
58      .rst(rst),
59      .end_of_file_signal(end_of_file_signal),
60      .enable(enable),
61      .data_in(data_in),
62      .JPEG_bitstream(JPEG_bitstream),
63      .data_ready(data_ready),
64      .end_of_file_bitstream_count(end_of_file_bitstream_count),
65      .eof_data_partial_ready(eof_data_partial_ready)
66    );
67
68  endmodule

```

3.2 DC 综合

综合就是把行为级的RTL代码在工艺、面积、时序等约束下转换成对应的门级网表，是使用软件来设计硬件，然后将门级电路实现与优化的工作留给综合工具的一种设

计方法。

DC综合分为三阶段：转换、优化、映射。主要流程为读入设计、设置约束、执行综合、查看报告、保存结果。综合的目标是生成一个满足约束条件的电路，同时尽可能满足电路的面积、功耗和时延。

3.2.1 环境配置

在综合之前，需要修改 `.synopsys_dc.setup` 文件，如下所示：

```
1 set topDir      ".../jpeg_encode_asic/rtl"
```

其他内容与 aes_asic 的配置相同，完整内容见仓库 DC/`.synopsys_dc.setup` 文件。

3.2.2 读入设计

在 aes_asic 所提供 DC/scripts 目录下的 `read_rtl.tcl` 文件基础上修改，如下所示：

```
1 set TOP_MODULE jpeg_asic
```

完整内容见仓库 DC/scripts/read_rtl.tcl 文件。然后在 DC 目录下执行 aes_asic 所提供 shell 脚本 `run_dc_read_rtl.sh` 即可读入设计。

```
Writing ddc file '/home/class/U65/Desktop/jpegencode/DC/netlist/jpeg_asic_unmapped.ddc'.
1
date
Sat May 18 15:29:59 2024
*****
# Finish and Quit
*****
if {$exit_switch == "true"} {
exit
}

Memory usage for main task 400 Mbytes.
Memory usage for this session 400 Mbytes.
CPU usage for this session 29 seconds ( 0.01 hours ).
Elapsed time for this session 31 seconds ( 0.01 hours ).

Thank you...
```

图 3: 读入设计脚本执行结果

3.2.3 设置约束

在 aes_asic 所提供 DC/scripts 目录下的 `constraint_compile.tcl` 文件基础上修改，如下所示：

```
1 set TOP_MODULE      jpeg_asic
2 set Rst_list        [list PAD_RST_I]
```

```

3      set Clk_list          [list PAD_clk_i]
4
5      # other unchanged    content
6
7      create_clock -name wb_clk -period $SYS_CLK_PERIOD -waveform [list 0 [expr
$SYS_CLK_PERIOD /2]]  [get_ports PAD_clk_i]
8
9      # other unchanged    content
10
11     set_ideal_network [get_pins "U_clk_i/D"]
12
13     # other unchanged    content
14
15     set wb_in_ports [remove_from_collection [all_inputs]  [get_ports [list
PAD_clk_i PAD_rst_i]]]
16     set wb_out_ports [get_ports [list PAD_jpeg_bitstr_o PAD_dat_rdy_o
PAD_eof_bitstr_cnt_o PAD_eof_dat_partial_rdy_o]]
17
18     set_case_analysis 0 [get_pins "U_RST_i/D"]

```

另外，根据设计要求：

- 时钟频率为 50MHz, uncertainty 为 0.1ns
- 驱动 cell 设为 INVHD2X , 负载设为 3 x INVHD4X
- input_delay 和 output_delay 按 50% 时钟预算

脚本修改内容如下：

```

1      set SYS_CLK_PERIOD 20.0
2      set_clock_uncertainty 0.1      [all_clocks]
3
4      set MAX_LOAD [load_of smic18_ss/INVHD4X/A]
5      set_driving_cell -lib_cell INVHD2X [remove_from_collection [all_inputs] \
6          [get_ports [list PAD_clk_i PAD_rst_i]]]
7      set_load [expr $MAX_LOAD*3] [all_outputs]
8
9      set_input_delay -max 10 -clock wb_clk $wb_in_ports
10     set_input_delay -min 0.1 -clock wb_clk $wb_in_ports
11     set_output_delay -max 10 -clock wb_clk $wb_out_ports
12     set_output_delay -min -1 -clock wb_clk $wb_out_ports

```

完整内容见仓库 `DC/scripts/constraint_compile.tcl` 文件。然后在 DC 目录下执行 `aes_asic` 所提供 shell 脚本 `run_dc_constraint_compile.sh` 即可设置约束并完成综合。

```

set hdout_internal_busses true
true
change_names -hierarchy -rule verilog
Warning: The specified replacement character (_) is conflicting with the specified allowed or restricted character. (UCN-4)
1
define_name_rules name_rule -allowed {a-z A-Z 0-9 _} -max_length 255 -type cell
1
define_name_rules name_rule -allowed {a-z A-Z 0-9 []} -max_length 255 -type net
1
define_name_rules name_rule -map {"\\*cell\\*" "cell"}
1
define_name_rules name_rule -case_insensitive -remove_internal_net_bus -equal_ports_nets
1
change_names -hierarchy -rules name_rule
1
*****
# Output Results
*****
write -format verilog -hierarchy -output ${netlistDir}/${TOP_MODULE}.vg
Writing verilog file '/home/class/U65/Desktop/jpegencode/DC/netlist/jpeg_asic.vg'.
1
write -format ddc -hierarchy -output ${netlistDir}/${TOP_MODULE}.ddc
Writing ddc file '/home/class/U65/Desktop/jpegencode/DC/netlist/jpeg_asic.ddc'.
1
write_sdf ${netlistDir}/${TOP_MODULE}_post_dc.sdf
Information: Annotated 'cell' delays are assumed to include load delay. (UID-282)
Information: Writing timing information to file '/home/class/U65/Desktop/jpegencode/DC/netlist/jpeg_asic_post_dc.sdf'. (WT-3)
Information: Updating design information... (UID-85)
Warning: Design 'jpeg_asic' contains 25 high-fanout nets. A fanout number of 60 will be used for delay calculations involving these nets. (TIM-134)
1
write_sdc -nosplit ${netlistDir}/${TOP_MODULE}.sdc
1
date
Sat May 18 15:51:46 2024
*****
# Finish and Quit
*****
if {$exit_switch == "true"} {
exit
}

Memory usage for main task 1427 Mbytes.
Memory usage for this session 1427 Mbytes.
CPU usage for this session 1301 seconds ( 0.36 hours ).
Elapsed time for this session 1307 seconds ( 0.36 hours ).

Thank you...

```

图 4: 设置约束及综合脚本执行结果

3.2.4 查看报告

综合完成后，可以查看综合报告，报告中包含了综合后的面积、时序等信息。采用 SMIC18 工艺库综合后，整个设计的面积为 9559257，如图 5 所示。

```

*****
Report : area
Design : jpeg_asic
Version: P-2019.03-SP3
Date   : Sat May 18 15:50:32 2024
*****

Library(s) Used:
smic18_ss (File: /home/class/U65/Desktop/jpegencode/lib/SMIC18_Ver2.5/FEView_STDIO/STD/Synopsys/smic18_ss.db)
smic18IO_line_ss (File: /home/class/U65/Desktop/jpegencode/lib/SMIC18_Ver2.5/FEView_STDIO/I/O/Synopsys/smic18IO_line_ss.db)

Number of ports:          64670
Number of nets:           392087
Number of cells:          276499
Number of combinational cells: 235944
Number of sequential cells: 39583
Number of macros/black boxes: 67
Number of buf/inv:         45017
Number of references:      4

Combinational area:       6297258.919346
Buf/Inv area:             309111.589378
Noncombinational area:    2337398.366447
Macro/Black Box area:     924600.000000
Net Interconnect area:    undefined (Wire load has zero net area)

Total cell area:          9559257.285793
Total area:                undefined
1

```

图 5: 综合面积报告

当前时序约束下，时序报告如图 6 所示。

Point	Incr	Path
clock wb_clk (rise edge)	0.00	0.00
clock network delay (ideal)	0.00	0.00
input external delay	10.00	10.00 r
PAD_dat_i[13] (in)	4.76	14.76 r
U_dat_i_13/P (PLB18F)	0.03	14.79 r
U_dat_i_13/P (PLB18F)	1.07	15.86 r
jpeg_top_inst/data_in[13] (jpeg_top)	0.00	15.86 r
jpeg_top_inst/u19/data_in[13] (fifo_out)	0.00	15.86 r
jpeg_top_inst/u19/u14/data_in[13] (pre_fifo)	0.00	15.86 r
jpeg_top_inst/u19/u14/u4/data_in[13] (RGB2YCBCR)	0.00	15.86 r
jpeg_top_inst/u19/u14/u4/mult_92/b[5] (RGB2YCBCR_DW_mult_uns_3)	0.00	15.86 r
jpeg_top_inst/u19/u14/u4/mult_92/U36/S (FAHHD1X)	0.53	16.39 f
jpeg_top_inst/u19/u14/u4/mult_92/U126/C0 (FAHHD2X)	0.33	16.72 f
jpeg_top_inst/u19/u14/u4/mult_92/U125/C0 (FAHHD2X)	0.22	16.94 f
jpeg_top_inst/u19/u14/u4/mult_92/U93/C0 (FAHHD4X)	0.30	17.24 f
jpeg_top_inst/u19/u14/u4/mult_92/U92/Z (NAND2HD2X)	0.12	17.36 r
jpeg_top_inst/u19/u14/u4/mult_92/U97/Z (NAND3HD4X)	0.14	17.51 f
jpeg_top_inst/u19/u14/u4/mult_92/U75/Z (NAND2B1H4X)	0.11	17.62 r
jpeg_top_inst/u19/u14/u4/mult_92/U100/Z (NAND3HD4X)	0.14	17.76 f
jpeg_top_inst/u19/u14/u4/mult_92/U66/Z (NAND2B1H4X)	0.09	17.85 r
jpeg_top_inst/u19/u14/u4/mult_92/U74/Z (NAND3HD4X)	0.11	17.96 f
jpeg_top_inst/u19/u14/u4/mult_92/U72/C0 (FAHHD2X)	0.26	18.21 f
jpeg_top_inst/u19/u14/u4/mult_92/U107/Z (NAND2HD2X)	0.09	18.30 r
jpeg_top_inst/u19/u14/u4/mult_92/U82/Z (NAND3B1H2X)	0.10	18.40 f
jpeg_top_inst/u19/u14/u4/mult_92/U122/C0 (FAHHD2X)	0.23	18.63 f
jpeg_top_inst/u19/u14/u4/mult_92/U124/C0 (FAHHD2X)	0.22	18.85 f
jpeg_top_inst/u19/u14/u4/mult_92/U123/C0 (FAHHD2X)	0.22	19.07 f
jpeg_top_inst/u19/u14/u4/mult_92/U127/C0 (FAHHD2X)	0.26	19.33 f
jpeg_top_inst/u19/u14/u4/mult_92/U61/S (HAHD4X)	0.20	19.53 r
jpeg_top_inst/u19/u14/u4/mult_92/product[19] (RGB2YCBCR_DW_mult_uns_3)	0.00	19.53 r
jpeg_top_inst/u19/u14/u4/U23/Z (A012B2H4X)	0.09	19.63 f
jpeg_top_inst/u19/u14/u4/U24/Z (INVHD4X)	0.06	19.68 r
jpeg_top_inst/u19/u14/u4/CB2_product_reg[19]/RN (FFDCRHDX)	0.00	19.68 r
data arrival time		19.68
clock wb_clk (rise edge)	20.00	20.00
clock network delay (ideal)	0.00	20.00
clock uncertainty	-0.10	19.90
jpeg_top_inst/u19/u14/u4/CK (FFDCRHDX)	0.00	19.90 r
library setup time	-0.22	19.68
data required time		19.68
data required time		19.68
data arrival time		-19.68
slack (MET)		0.00

图 6: 综合时序报告

由图可知， data required time 为 19.68ns， data arrival time 为 -19.68ns， slack 为 0。满足时序约束。

功耗报告如图 7 所示。可知功耗约为 170.2820mW。

Global Operating Voltage = 1.62					
Power-specific unit information :					
Voltage Units = 1V					
Capacitance Units = 1.000000pf					
Time Units = 1ns					
Dynamic Power Units = 1mW (derived from V,C,T units)					
Leakage Power Units = 1nW					
Cell Internal Power	151.7254 mW	(89%)			
Net Switching Power	18.1408 mW	(11%)			
Total Dynamic Power	169.8662 mW	(100%)			
Cell Leakage Power	418.6651 uW				
Power Group	Internal Power	Switching Power	Leakage Power	Total Power	(%) Attrs
io_pad	2.5198	0.4030	29.4599	2.9228	(1.72%)
memory	0.0000	0.0000	0.0000	0.0000	(0.00%)
black_box	0.0000	0.0000	0.0000	0.0000	(0.00%)
clock_network	0.0000	0.0000	0.0000	0.0000	(0.00%)
register	0.0000	0.0000	0.0000	0.0000	(0.00%)
sequential	132.7946	3.4699	7.1580e+04	136.3375	(80.87%)
combinational	16.4077	14.2669	3.4706e+05	31.0218	(18.22%)
Total	151.7221 mW	18.1398 mW	4.1867e+05 nW	170.2820 mW	

图 7: 综合功耗报告

3.3 形式验证

在综合、适配过程后，要确保电路功能不变。后仿真是一种方法，但是后仿真时间精度高，仿真时间长。当设计规模很大时，仿真时间会很长。可以进行分别处理时序分析采用 STA，功能验证采用形式验证。形式验证是一种数学方法，通过数学方法证明电路功能正确。与 RTL 仿真相比，RTL 仿真证明 RTL 功能正确，时间较短，需要保证一定的测试覆盖率。形式验证证明 RTL 与 Gate 代码等价，则说明 Gate 功能也正确。STA 证明 Gate 代码满足时序要求，则证明 Gate 级代码功能、时序均满足。形式验证流程图如图 8 所示。

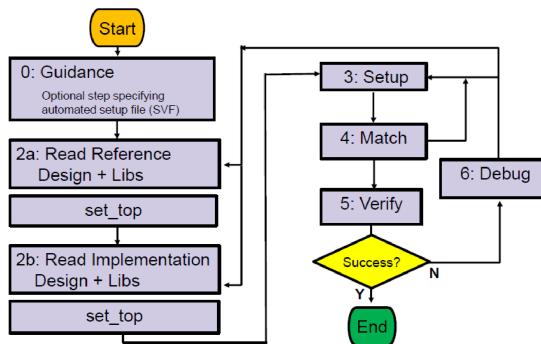


图 8: 形式验证流程图

3.3.1 形式验证步骤

在形式验证之前，据 aes_asic 所提供的环境稍作修改，将 FM_RTL_PostDC/netlist 链接至 DC 综合后生成结果的文件夹。执行如下命令：

```
1 ln -snf ../../DC/netlist/ netlist
```

将 FM_RTL_PostDC/netlist 链接至本设计 RTL 代码所在文件夹。执行如下命令：

```
1 ln -snf ../../jpeg_encode_asic/rtl rtl
```

在 aes_asic 所提供 FM_RTL_PostDC 目录下的 .fms 文件基础上修改，并重命名为 jpeg_encode_asic.fms，完整内容如下所示：

```

1 date
2 set TOP_MODULE jpeg_asic
3 set search_path "$search_path $topDir "
4 set synopsys_auto_setup true
5 set_svf ${svfDir}/${TOP_MODULE}.svf
6 read_db "lib/SMIC18_Ver2.5/FEView_STDIO/STD/Synopsys/smic18_tt.db lib/
SMIC18_Ver2.5/FEView_STDIO/IO/Synopsys/smic18IO_line_tt.db"

```

```

7   read_verilog -r [sh ls $topDir/*.v]
8   set_top ${TOP_MODULE}
9   read_verilog -i ${TOP_MODULE}.vg
10  set_top ${TOP_MODULE}

```

然后对 `run_fm.scr` 修改，如下所示：

```

1  #!/bin/sh
2
3  mkdir -p logs
4  fm_shell -f jpeg_encode_asic.fms | tee -i logs/jpeg_encode_asic.log

```

在 `FM_RTL_PostDC` 目录下执行 `aes_asic` 所提供 shell 脚本 `run_fm.scr` 即可进行形式验证。形式验证结果如图 9 所示。

```

*****
Verification Results *****
Verification SUCCEEDED
ATTENTION: synopsys_auto_setup mode was enabled.
See Synopsys Auto Setup Summary for details.
ATTENTION: RTL interpretation messages were produced during link
of reference design.
Verification results may disagree with a logic simulator.

-----
Reference design: r:/WORK/jpeg_asic
Implementation design: i:/WORK/jpeg_asic
38965 Passing compare points
-----
Matched Compare Points BBPin Loop BBNNet Cut Port DFF LAT TOTAL
-----
Passing (equivalent) 0 0 0 0 39 38926 0 38965
Failing (not equivalent) 0 0 0 0 0 0 0 0
Not Compared
Constant reg 187 0 187
Unread 0 0 0 0 470 0 470
*****
Maximum memory usage for this session: 2517 MB
CPU usage for this session: 2371.63 seconds ( 0.66 hours )
Current time: Sat May 18 16:48:51 2024
Elapsed time: 2374 seconds ( 0.66 hours )

Thank you for using Formality (R)!

```

图 9: 形式验证结果

4 自动布局布线

4.1 设计输入

执行 `prep_fend_data.sh` 脚本，将门级网表、设计约束、工艺库等导入到 Astro 目录下并对约束进行处理使 Astro 可以读入。脚本内容如下：

```

1  #!/bin/bash

2

3  TOP_MODULE=jpeg_asic
4  cp ./DC/netlist/${TOP_MODULE}.vg ./data_fend
5  cp ./DC/netlist/${TOP_MODULE}.sdc ./data_fend
6  cp ./DC/netlist/${TOP_MODULE}.ddc ./data_fend
7
8  (cd data_fend; ./prep_sdc.sh ${TOP_MODULE}.sdc)

```

根据 aes_asic 提供的脚本，修改 `Astro/soc_scripts/1.0_design_setup.cmd` 文件，主要内容如下所示：

```

1  ;# Scheme
2  menuReload "astro_data_prep"
3  auVerilogToCell
4  formDefault "Verilog To Cell"
5  setFormField "Verilog To Cell" "Library Name" "design_lib_jpeg ASIC"
6  setFormField "Verilog To Cell" "Verilog File Name" "./data_fend/
jpeg ASIC.vg"
7  setFormField "Verilog To Cell" "Output Cell Name" "jpeg ASIC"
8  setFormField "Verilog To Cell" "Top Module Name" "jpeg ASIC"
9  setFormField "Verilog To Cell" "Tech File Name" "../lib//SMIC18_Ver2.5/
BEView_STDIO/TECH/smic18_apollo_m6.tf"
10 setFormField "Verilog To Cell" "Net Name for 1'b0" "GND"
11 setFormField "Verilog To Cell" "Open Library and Cell When Done" "1"
12 setFormField "Verilog To Cell" "Initialize Hierarchy Preservation" "1"
13 formButton "Verilog To Cell" "globalNetOptions"
14 setFormField "Verilog To Cell" "Net Name" "VDD"
15 setFormField "Verilog To Cell" "Port Pattern" "VDD"
16 formButton "Verilog To Cell" "apply"
17 setFormField "Verilog To Cell" "Net Name" "GND"
18 setFormField "Verilog To Cell" "Port Pattern" "GND"
19
20 ;# other unchanged content

```

4.2 布局规划

4.2.1 电源引脚规划

首先，根据 aes_asic 的 .tdf 文件，修改 .tdf 文件。要规划电源引脚个数，有两种电源引脚：core 供电 PAD (PLVDDC/PLVSSC, 1.8V)、IO 供电 PAD (PLVDDH/PLVSSH, 3.3V)。根据功耗、电流信息计算，但最好是大于此计算值，一般还会考虑对称性，供电均匀。

本设计中，根据综合报告，如图 7 所示，功耗约为 170.2820 mW。考虑较差情况，电压取值 1.6V。PAD 与 core ring 连接金属为 35μm 时，最大电流应该按 35mA 计算。则需要的 core 供电 PAD 数：

$$\frac{170.2820}{\frac{1.6}{35}} \approx 3.04 \quad (1)$$

因此安排 4 对，4 面各 1 对。

PAD 供电电源数量估算考虑最坏情况，即所有的输出 PAD 与双向 PAD 同时输出。需要的 IO 供电 PAD 数：

$$\frac{39 \times 8}{51} \approx 6.12 \quad (2)$$

安排 8 对，4 面各 2 对。

.tdf 文件主要内容如下：

```

1 ;TDF file for jpeg_asic
2 ;Chip Directions
3 ;;;;;;;;;;;;;;;;;;;
4 ; (Top)          ;
5 ;(Left)          (Right);
6 ; (Bottom)        ;
7 ;;;;;;;;;;;;;;;;;;;
8 define _cell (geGetEditCell)
9
10 ;pad record section
11 tdfPurgePadConstr
12 tdfSetMinIODistance 0 0 0 0
13
14 ;use insertPad or dbCreateCellInst for PLVDD* and PLVSS*
15 dbCreateNet _cell "VDDH"
16 dbCreateNet _cell "VSSH"
```

```

17    insertPad "VDDH" "PLVDDH" "pad_L_VDDH1" "VDDH"
18    insertPad "VDDH" "PLVDDH" "pad_L_VDDH2" "VDDH"
19    insertPad "VDDH" "PLVDDH" "pad_L_VDDH3" "VDDH"
20    insertPad "VDDH" "PLVDDH" "pad_L_VDDH4" "VDDH"
21    insertPad "VDDH" "PLVDDH" "pad_L_VDDH5" "VDDH"
22    insertPad "VDDH" "PLVDDH" "pad_L_VDDH6" "VDDH"
23    insertPad "VDDH" "PLVDDH" "pad_L_VDDH7" "VDDH"
24    insertPad "VDDH" "PLVDDH" "pad_L_VDDH8" "VDDH"

25
26    insertPad "VSSH" "PLVSSH" "pad_L_VSSH1" "VSSH"
27    insertPad "VSSH" "PLVSSH" "pad_L_VSSH2" "VSSH"
28    insertPad "VSSH" "PLVSSH" "pad_L_VSSH3" "VSSH"
29    insertPad "VSSH" "PLVSSH" "pad_L_VSSH4" "VSSH"
30    insertPad "VSSH" "PLVSSH" "pad_L_VSSH5" "VSSH"
31    insertPad "VSSH" "PLVSSH" "pad_L_VSSH6" "VSSH"
32    insertPad "VSSH" "PLVSSH" "pad_L_VSSH7" "VSSH"
33    insertPad "VSSH" "PLVSSH" "pad_L_VSSH8" "VSSH"

34
35    insertPad "VDD" "PLVDDC" "pad_L_VDDC1" "VDD"
36    insertPad "VDD" "PLVDDC" "pad_L_VDDC2" "VDD"
37    insertPad "VDD" "PLVDDC" "pad_L_VDDC3" "VDD"
38    insertPad "VDD" "PLVDDC" "pad_L_VDDC4" "VDD"

39
40    insertPad "GND" "PLVSSC" "pad_L_VSSC1" "GND"
41    insertPad "GND" "PLVSSC" "pad_L_VSSC2" "GND"
42    insertPad "GND" "PLVSSC" "pad_L_VSSC3" "GND"
43    insertPad "GND" "PLVSSC" "pad_L_VSSC4" "GND"

44
45    insertPad "GND" "PLCORNER" "corner_ll" "GND"
46    insertPad "GND" "PLCORNER" "corner_lr" "GND"
47    insertPad "GND" "PLCORNER" "corner_ul" "GND"
48    insertPad "GND" "PLCORNER" "corner_ur" "GND"

49
50    pad "corner_ll" "bottom"
51    pad "corner_ul" "left"
52    pad "corner_lr" "right"
53    pad "corner_ur" "top"

```

```

54      ; Bottom ( --> 23 pads )
55      pad "U_dat_i_0"      "bottom"    1
56      pad "U_dat_i_1"      "bottom"    2
57      pad "U_dat_i_2"      "bottom"    3
58      pad "U_dat_i_3"      "bottom"    4
59      pad "U_dat_i_4"      "bottom"    5
60      pad "pad_L_VSSH1"   "bottom"    6
61      pad "pad_L_VDDH1"   "bottom"    7
62      pad "U_dat_i_5"      "bottom"    8
63      ; ... other pads
64
65      ;Right ( --> 23 pads)
66      pad "U_dat_i_17"     "right"     1
67      pad "U_dat_i_18"     "right"     2
68      pad "U_dat_i_19"     "right"     3
69      pad "U_dat_i_20"     "right"     4
70      pad "U_dat_i_21"     "right"     5
71      pad "pad_L_VSSH3"   "right"     6
72      pad "pad_L_VDDH3"   "right"     7
73      pad "U_dat_i_22"     "right"     8
74      pad "U_dat_i_23"     "right"     9
75      pad "U_dat_o_0"      "right"    10
76      pad "U_dat_o_1"      "right"    11
77      pad "pad_L_VDDC2"   "right"    12
78      pad "pad_L_VSSC2"   "right"    13
79      pad "U_dat_o_2"      "right"    14
80      ; ... other pads
81
82      ; Top ( --> 23 pads )
83      pad "U_dat_o_10"     "top"       1
84      pad "U_dat_o_11"     "top"       2
85      ; ... other pads
86
87      ; Left( --> 22 pads)
88      pad "U_dat_o_27"      "left"      1
89      ; ... other pads
90      ;;Finished

```

完整内容见仓库 Astro/jpeg_asic.tdf 文件。

4.2.2 芯片大小估算

根据公式计算芯片面积：

$$(23 \times 74 + 184 \times 2) - 80 \times 2 - 184 \times 2 = 1542 \quad (3)$$

考虑一定余量，根据该公式分别设置芯片的长宽为 2500，但是在布局规划过程中，Astro 报错，提示芯片面积不够，如下图所示。

```

196  Initializing Data Structure ...
208  Reading netlist information from DB ...
211  830 IO cells/pins
212  265444 cell instances
213  328458 nets
214  Sorting cells, nets, pins ...
215  | net pin threshold = 26
216  Reading misc information ...
217  | array <unit> has 0 vertical and 763 horizontal rows
218  | 32 pre-routes for placement blockage/checking
219  | 15905 pre-routes for map congestion calculation
220  | 9427 nets have positive weight
221  | average positive net weight (>=2) = 2.0
222  | positive net weight range = [2, 3]
223  | standard deviation of net weight = 0.15
224  | 1 net(s) treated as CLOCK for clustering
225  | 1 clock nets, 39583 clocked instances (14.96%)
226  | Auto Set : first cut = vertical
227  Checking information read in ...
228  | design style = Horizontal masters, Horizontal rows
229  Preprocessing design ...
230  | splitting rows by natural obstacles ...
231  | | Unit Tile      Demand      Available
232  | | | unit        2571343     629467 << NOT ENOUGH
233  | calculating design statistics ...
234  | Auto Set : first cut = vertical
235  | split into 763 row segments
236  | RowArea1 = total row area = 2093859.03 micron^2
237  | RowArea2 = RowArea1 - macros = 2093859.03 micron^2
238  | RowArea3 = RowArea2 - fixed standard cells = 2093859.03 micron^2
239  | RowArea4 = RowArea3 - hard blockages and preroutes = 2093859.03 micron^2
240  | RowArea5 = RowArea4 - IO pads and FC drivers = 2093859.03 micron^2
241  | RowArea6 = RowArea5 - soft blockages = 2093859.03 micron^2
242  | RowArea7 = RowArea6 - misc unusable area = 2093859.03 micron^2
243  | ERROR: total cell size (8553315 micron^2) exceeds placeable area (2093859 micron^2)
244  | | cell/row utilization = 408.50%
245  Auto Place: pre-place optimization done (cpuTime = 22 seconds)
246  Fail to execute command

```

图 10: Astro 报错

该错误是 core limited，根据报错给出的大小，进行计算。

$$\sqrt{8553315} \approx 2925 \quad (4)$$

因此设置芯片的长宽为 5000，重新进行布局规划。修改 Astro/soc_scripts 目录下的 2.0_floorplan_before_place_macro.cmd 文件，主要内容如下所示：

```

1  ;# Scheme
2  geOpenLib
3  formDefault "Open Library"

```

```

4      setFormField "Open Library" "Library Name" "design_lib_jpeg_asic"
5      formOK "Open Library"
6      geOpenCell
7      formDefault "Open Cell"
8      setFormField "Open Cell" "Cell Name" "10_design_setup"
9      formOK "Open Cell"

10
11     axgLoadTDF
12     formDefault "Load TDF File"
13     setFormField "Load TDF File" "TDF File Name" "jpeg_asic.tdf"
14     setFormField "Load TDF File" "Cell Name" "10_design_setup"
15     formOK "Load TDF File"

16
17     axgPlanner
18     formDefault "Floor Planning"
19     setFormField "Floor Planning" "Control Parameter" "width & height"
20     setFormField "Floor Planning" "Row/Core Ratio" "0.75"

21
22     setFormField "Floor Planning" "Core Width" "5000"
23     setFormField "Floor Planning" "Core Height" "5000"
24     setFormField "Floor Planning" "Core To Left" "80"
25     setFormField "Floor Planning" "Core To Bottom" "80"
26     setFormField "Floor Planning" "Core To Right" "80"
27     setFormField "Floor Planning" "Core To Top" "80"
28     setFormField "Floor Planning" "Double Back" "1"
29     setFormField "Floor Planning" "Start from first row" "1"
30     setFormField "Floor Planning" "Flip first row" "1"
31     formOK "Floor Planning"

32
33     ;# other unchanged content

```

4.2.3 添加 PAD filler 、电源环、电源条等

然后在 PAD 中间填充 PAD filler， Pad 之间的缝隙需要填充， PAD 之间无连线，电源、地在相同位置，用 Filler 连接起来，形成 Pad rings 环。再将电源 PAD 与 core ring 连接起来。完成后结果如图 11 所示。

修改所提供的 `createstraps.cmd` 文件，如下所示：

```

1 axgCreateStraps
2     setFormField "Create Straps" "Direction" "Vertical"
3     setFormField "Create Straps" "Start X" "800"
4     setFormField "Create Straps" "Net Name(s)" "VDD,VDD,GND,GND"
5     setFormField "Create Straps" "Width" "18"
6     setFormField "Create Straps" "Layer" "66"
7     setFormField "Create Straps" "Extend for Multiple Connections" "1"
8     setFormField "Create Straps" "For Gap <" "2"
9     formApply "Create Straps"
10    setFormField "Create Straps" "Start X" "1500"
11    formApply "Create Straps"
12    setFormField "Create Straps" "Start X" "2200"
13    formApply "Create Straps"
14    setFormField "Create Straps" "Start X" "2900"
15    formApply "Create Straps"
16    setFormField "Create Straps" "Start X" "3600"
17    formApply "Create Straps"
18    setFormField "Create Straps" "Start X" "4300"
19    formOK "Create Straps"

```

在适当的位置添加电源条，方便个cell电压均匀，添加了电源条的版图如图 12 所示。

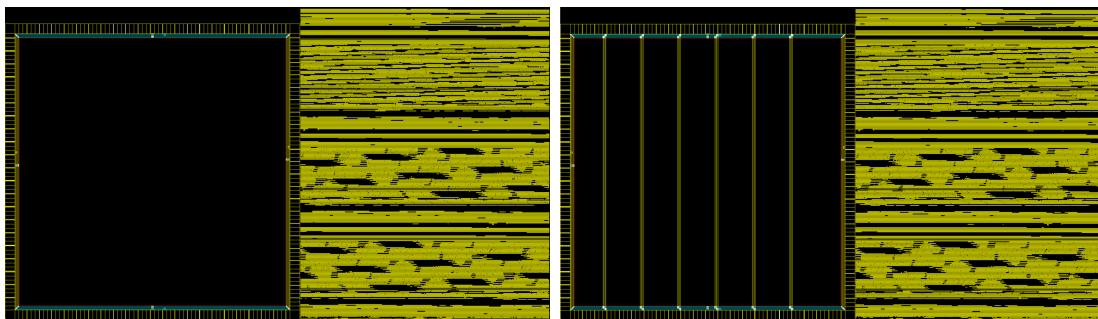


图 11: 放置 PAD 和电源环后版图

图 12: 添加电源条后版图

4.2.4 规划 stdcell 摆放区域

设置 blockage 或者直接 cut row，某些电源 PAD 接入的地方，Metal1 很难连接到电源上，干脆不放 stdcell。另外，电源条下面也不要放上标准单元，通过 cut row 等方式，把电源条位置闪出来。由于初次规划时未注意，导致在物理验证 DRC 检查时出现上千个违例，如图 13 所示。与 aes_asic 设计对比，大概是因为电源条下面放上标准单元了，如图 14 所示。

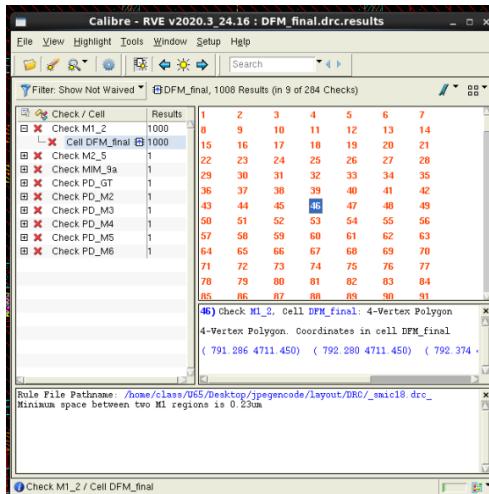


图 13: 大量 DRC 违例

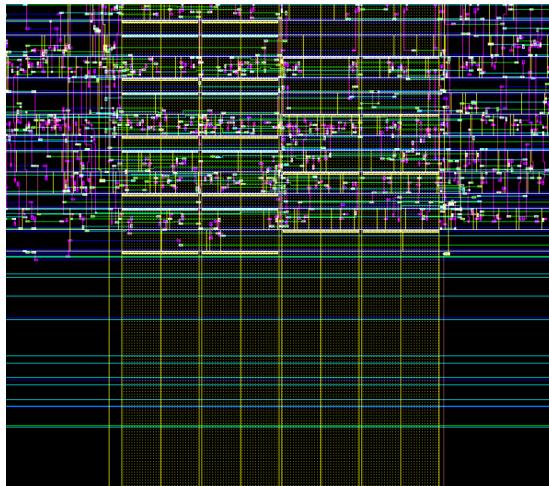


图 14: 设计的电源条下面存在标准单元

完成后保存 floorplan，floorplan 调整比较麻烦，调整好后保存，下次直接调取脚本。再次运行时，可以在脚本中使用 `load "soc_scripts/row.dump"` 方式加载，详细步骤可以参考课件。

4.3 其余步骤

4.3.1 时序约束

Astro 布局布线是 Timing-Driven，要指定时序约束。DC 时做了时序约束，是针对 RTL 级的，有较多的高级特性，其他工具未必理解，需要写出较通用的、门级的 SDC 文件。

步骤如下：

1. 导入 SDC 文件，需要修改 DC 导出的 SDC 文件
2. Timing Setup，Astro时序分析分阶段，各阶段需要单独设置
3. 检查约束是否完备
4. PrePlace Timing Check，此时是比较理想情况的 Timing 检查，一般 violation 不应该很大，否则需要检查约束或者检查 Design 是否能达到要求。

SDC主要修改的语句有去掉 DC 对“时钟理想化”处理的相关语句、去掉 DC 对工作环境的假设、增加 `set_propagated_clock [all_clocks]`。Astro的时序检查分阶段进行，如布局前的、布局后的，时钟树综合前的、时钟树综合后的，布线前的、布线后的。

4.3.2 Place

直接采用Auto place, Astro 的 Place 是 Timing-driven, 需要设置 Timing。另外,要在 Timing Setup 设置工作环境、分析模式等, 优化要留余份, 检查设置是否符合当前分析阶段, 并设置 Placement Common Options。

4.3.3 时钟树综合CTS、Post-CTS优化

进行 CTS, 设定时钟树综合约束, 最大扇出等信息。然后做 Post-CTS 阶段的 Place。Place 确定后, 做 Post-Place 阶段的时钟树优化。之后检查时序, 输出报告。

4.3.4 Route

时钟树综合、布局过程中, 可能引入一些单元, 需要重新连接一次PG网络。Route 步骤如下:

1. 先布时钟线
2. Timing Setup
3. Post-Route Opt 及 CTO
4. Post-Route 时序分析

4.3.5 DFM

DFM 一般包括修正天线效应、加 Filler、过孔优化、Fill Notch and Gap、Add Label、添加 Wire track。

天线效应解决方案有跳线、插入二极管。Astro 所加载的 ANT 规则文件一般应由工艺库中提供, 但若没有提供, 可以根据天线效应检查规则文件反推。Astro按规则文件自动修正天线效应, 若规则得当, 可修掉大部分, 仍可能有部分需要手工处理。添加 Label 是为 LVS 检查做准备, 添加 Wire track 是为了满足后续 DRC 检查中的金属密度规则。

4.3.6 数据导出

在数据导出中会先做 LVS , 确认 LVS 没有问题。然后导出网表, 用于 LVS、后仿真等, 导出 GDSII 数据, 流片数据。导出 SPEF, 用于 PT 时序分析。导出 SDF, 用于后仿真。

4.3.7 总结

其余步骤的脚本内容与 aes_asic 的相同，不再赘述。仅需修改脚本中设计的文件名，然后在 Astro 目录下终端执行 `./soc_scripts/backend.sh` 即可完成布局布线。最终布局布线结果如图 15 所示。

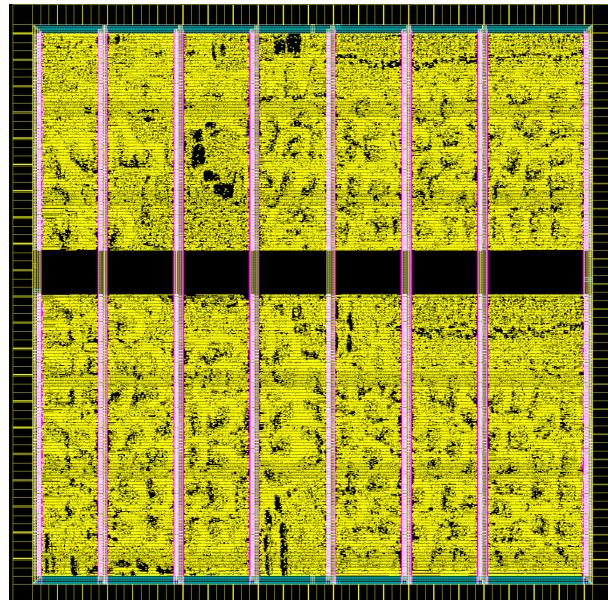


图 15: Astro 布局布线结果

5 物理验证

5.1 准备 IC5141 环境

准备工艺库和显示资源，将 SmicVTTF_L0_SRAM_MR_MM_HV_LC_018.tf display.drf 复制到 layout 目录。准备基本库，新建 cds.lib 文件，内容如下所示：

```

1 INCLUDE $CDSHOME/share/cdssetup/cds.lib
2 DEFINE SMIC18IOLIB_L_M6 SMIC18IOLIB_L_M6
3 DEFINE SMIC18STDLIBM6 SMIC18STDLIBM6
4 DEFINE mytools mytools
5 DEFINE logo logo
6 DEFINE avTech /opt/cadence/assura/tools/assura/etc/avtech/avTech
7 DEFINE design_lib_jpeg_asic design_lib_jpeg_asic

```

进行快捷键、Calibre配置等的设置，新建 .cdsinit 文件，内容如下：

```

1 load( prependInstallPath( "samples/local/schBindKeys.il" ) )
2 load( prependInstallPath( "samples/local/leBindKeys.il" ) )
3 load("/opt/mentor/calibre/aoj_cal_2020.3_24.16/lib/calibre.skl")

```

5.1.1 导入库文件

将 Stdcell、Pad 库以及设计库导入到 IC5141，本设计导入配置已保存为模板文件，可以直接加载。



图 16: IC5141 导入库文件

5.2 为电源 PAD 加 Label

为了进行 LVS，Core 电源 PAD 已加过，只需要为电源 PAD 加 Label。先缩放到某个 VSSH或VDDH附近，添加 Label，名称分别为 VSSH 、 VDDH。只需要设置一组。如图 17所示。

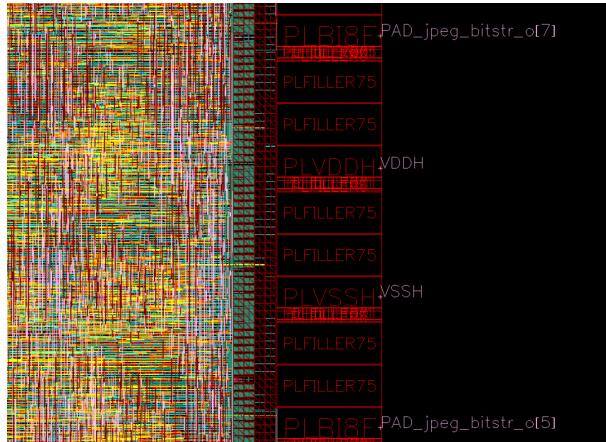


图 17: 为电源 PAD 加 Label

5.3 进行 LVS 检查

先进行 LVS 检查，确认设计正确。进入 layout/LVS 目录，根据情况修改 aes_asic 提供的 gen_spi.sh 脚本。脚本内容如下：

```

1  v2lvs -v jpeg_asic.lvs.vg -l smic18.v -l smic18IO_line.v -s SMIC18IOLIB_L.
2    cdl -s stdcells.cdl -o jpeg_asic.spi -s0 GND -s1 VDD
3    echo '.GLOBAL VDDH' >> jpeg_asic.spi
4    echo '.GLOBAL VSSH' >> jpeg_asic.spi

```

在该目录下终端执行 ./gen_spi.sh，生成.spi文件。然后在 IC5141 中进行 LVS 检查，按照提示，填写参数，设置规则文件、运行目录等。Calibre 自动从 IC5141 中提取 layout 数据，并做 LVS 检查，本设计保存了 runset 文件，可以直接调用。使用中发现，如图 18 所示，netlist 处需要每次重填。LVS 检查结果如图 19 所示，LVS 检查通过。

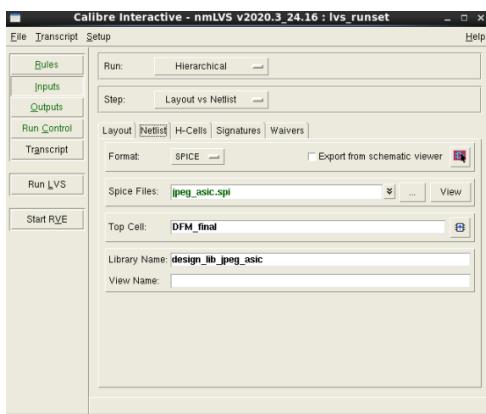


图 18: LVS 检查配置

Overall Comparison Results

```

#      #####
#      #   #
#      #   CORRECT
#      #   #
#      #####
Warning: Unbalanced smashed mosfets were matched
Warning: Ambiguity points were found and resolved arbitrarily.
***** CELL SUMMARY *****
Result      Layout          Source
CORRECT     DFM_final     jpeg_asic

```

图 19: LVS 检查结果

5.4 ANT 检查与修正

ANT 检查要先于 DRC，因为 ANT 修正中可能会引入 DRC 违例，ANT 检查使用 DRC 菜单调用，但规则文件不同。结果如图 20 所示，ANT 检查通过。

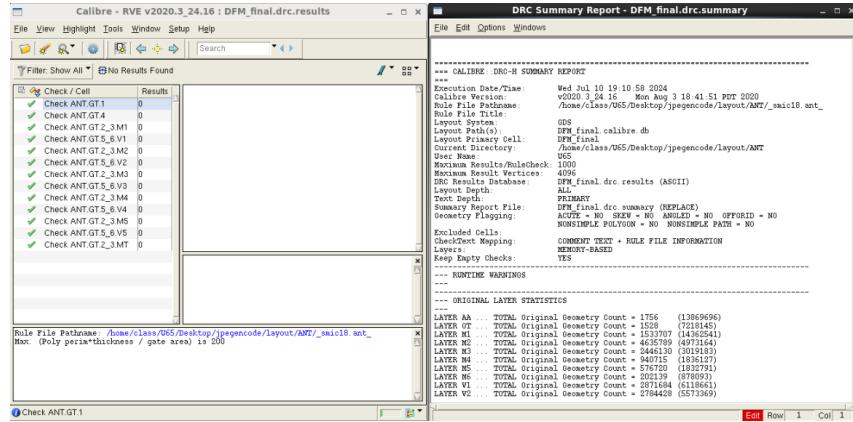


图 20: ANT 检查结果

5.5 DRC检查与修正

DRC 检查也使用 DRC 菜单调用，但规则文件不同。结果如图 21 所示。DRC 大部分通过，出现了较少个数的 DRC 为例，需要进行人工修正。

如图 21a，该错误为无用连线导致，删除即可。同时进行 LVS 检查，确保正确。如图 21b，该错误用 M1 补上缺角即可解决，一般钝角就可以。图 21c 的错误修正前检查违例的金属条有没有上下层连接，没有的话就切短到符合规则，为是 0.6μm。图 21d 的错误检查对角长度是否够 0.28μm，不够就延迟一段 M2。

修正后再次进行 LVS 检查，ANT 检查，DRC 检查。确保均通过。DRC 最终结果如图 22 所示。其中剩余问题均与密度相关。这类 DRC 错误暂时可以忽略。

5.6 设计数据导出

如图 23 所示配置，导出 GDSII。

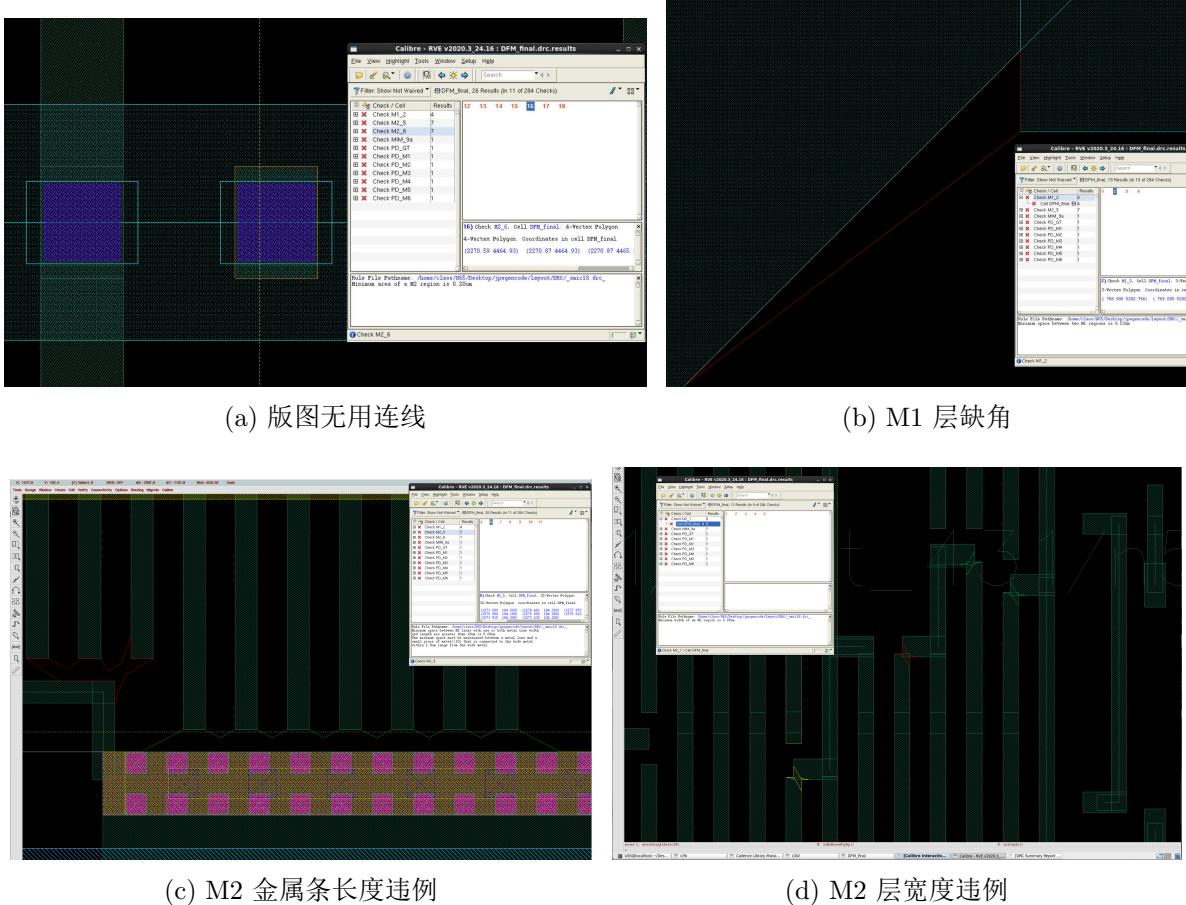


图 21: DRC 检查违例

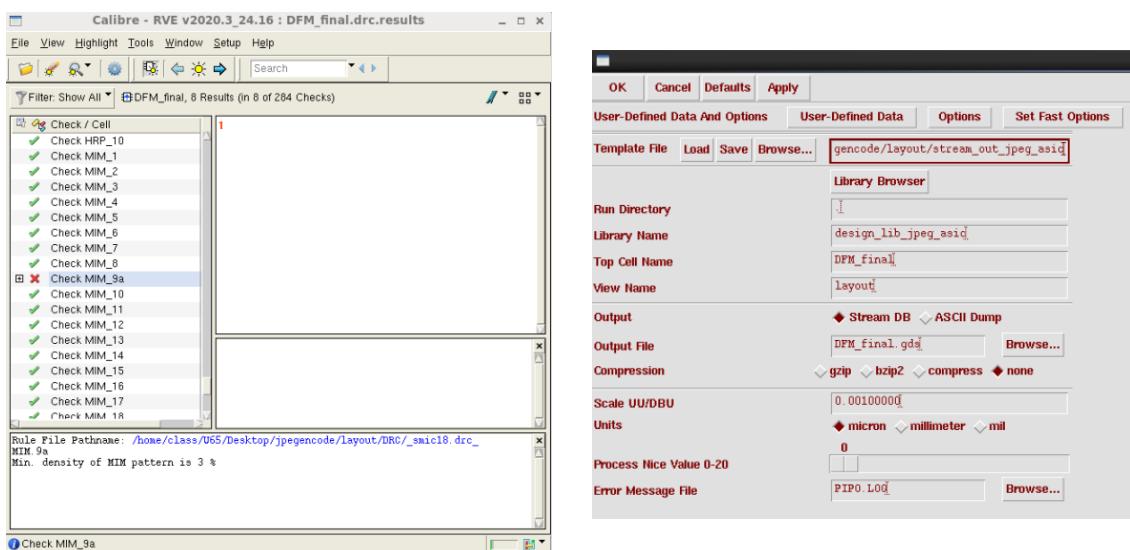


图 22: DRC 最终检查结果

图 23: 设计数据导出配置

6 PT 时序分析

STA，无需输入激励，电路并不动作（静态含义），分析每一个触发器的 setup 时间与 hold 时间，即保证在时钟沿采样数据时，数据是有效的。动态仿真也是确保这一点，下一时钟能得到正确值。动态与静态时序分析比较 STA 无需输入测试向量，覆盖率大。动态仿真只针对特定测试向量，无法证明结果对所有测试向量都成立；大规模电路，穷举测试向量很困难。STA 缺点是异步电路分析困难动态时序分析对同步、异步风格电路没有限制。STA 能处理更大设计，所需时间更短动态仿真缺点是随着设计规模增大，要求时间迅速增长。

PT 使用步骤如下：

1. 读入设计
2. 添加约束
3. 添加例外
4. 检查
5. 分析

6.1 环境配置

参照 aes_asic 设计，确保 PT/data_bend 指向 `../Astro/data_bend/`。

修改 `PT/.synopsys_pt.setup` 文件，内容如下所示：

```
1 lappend search_path      "/opt/synopsys/pts/P-2019.03/libraries/syn"
```

修改 aes_asic 设计提供的 `.pt` 文件，并重命名为 `jpeg_asic.pt`，内容如下所示：

```
1 set SYS_CLK_PERIOD 20.0
2 date
3 # Define some variables for design
4 set TOP_MODULE    jpeg_asic
5 set Rst_list      [list PAD_rst_i ]
6 set Clk_list      [list PAD_clk_i ]
7
8 #Read netlist
9 read_verilog data_bend/jpeg_asic.lvs.vg
10 current_design $TOP_MODULE
11 link_design
```

```

12 # Define The Design Enviroment
13 set_min_library smic18_ss.db -min_version smic18_ff.db
14 set_min_library smic18IO_line_ss.db -min_version smic18IO_line_ff.db
15
16 #Read SPEF
17 reset_design
18 read_parasitics data_bend/jpeg_asic.spef
19
20 set_wire_load_mode "segmented"
21 set_wire_load_model -name reference_area_10000000 -library smic18_ss
22
23 # clock definatoin and reset
24 create_clock -name wb_clk -period $SYS_CLK_PERIOD -waveform [list 0 [expr
$SYS_CLK_PERIOD /2]] [get_ports PAD_clk_i]
25 set_propagated_clock [all_clocks]
26
27 report_clocks -nosplit > ${reportsDir}/${TOP_MODULE}.clocks.txt
28
29 # drive and load, max_fanout,max_capacitance
30 set MAX_LOAD [load_of smic18_ss/INVHD4X/A]
31 set_drive 0 [get_ports "$Rst_list"]
32 set_drive 0 [get_ports "$Clk_list"]
33 set_driving_cell -lib_cell INVHD2X [remove_from_collection [all_inputs] \
34 [get_ports [list PAD_clk_i PAD_rst_i ]]]
35 set_load [expr $MAX_LOAD*3] [all_outputs]
36 set_max_fanout 10 [all_inputs]
37
38 # input delay and output delay
39 set wb_in_ports [remove_from_collection [all_inputs] [get_ports [list
PAD_clk_i PAD_rst_i]]]
40 set wb_out_ports [get_ports [list PAD_jpeg_bitstr_o PAD_dat_rdy_o
PAD_eof_bitstr_cnt_o PAD_eof_dat_partial_rdy_o]]
41
42 set_input_delay -max 10 -clock wb_clk $wb_in_ports
43 set_input_delay -min 0.1 -clock wb_clk $wb_in_ports
44
45 set_output_delay -max 10 -clock wb_clk $wb_out_ports

```

```

47
48     set_output_delay -min -1 -clock wb_clk $wb_out_ports
49
50     # false path
51     set_false_path -from [get_ports "$Rst_list"]
52
53     # case_analysis
54     set_case_analysis 0 [get_pins "U_rst_i/D"]
55
56     # Output Reports
57     report_design -nosplit > ${reportsDir}/${TOP_MODULE}.design.txt
58     report_port -nosplit > ${reportsDir}/${TOP_MODULE}.port.txt
59     report_net -nosplit > ${reportsDir}/${TOP_MODULE}.net.txt
60     report_constraint -nosplit -all_violators > ${reportsDir}/${TOP_MODULE}.constraint.txt
61     report_timing -nosplit > ${reportsDir}/${TOP_MODULE}.timing.txt
62
63     # Output Results
64     write_sdf data_bend/${TOP_MODULE}_post(APR.PT).sdf
65
66     date
67     exit

```

然后对 run_pt.sh 修改，如下所示：

```

4     #!/bin/bash
5     mkdir -p logs
6     mkdir -p reports
7     pt_shell -f jpeg_asic.pt | tee -i logs/jpeg_asic.pt.log

```

在 PT 目录下终端执行 ./run_pt.sh，执行结果如图 24 所示。

6.2 查看报告

PT 时序分析报告输出到 PT/reports 目录下，其中时序报告如图 25 所示。

由图可知，data required time 为 21.45ns，data arrival time 为 -20.06ns，slack 为 1.39ns。满足时序约束。

```

Note - message 'SDF-036' default limit (100) exceeded. Remainder will be suppressed.
Note - message 'SDF-036' default limit (100) exceeded. Remainder will be suppressed.
1
date
Sat Jun 22 23:53:59 2024
*****
# Finish and Quit
*****
exit
Information: Defining new variable 'SYS_CLK_PERIOD'. (CMD-041)
Information: Defining new variable 'TOP_MODULE'. (CMD-041)
Information: Defining new variable 'wb_in_ports'. (CMD-041)
Information: Defining new variable 'Rst_list'. (CMD-041)
Information: Defining new variable 'wb_out_ports'. (CMD-041)
Information: Defining new variable 'MAX_LOAD'. (CMD-041)
Information: Defining new variable 'Clk_list'. (CMD-041)
Suppressed Messages Summary:
Id      Severity    Occurrences   Suppressed
-----
SDF-036    Warning        95947       95858
Total 1 type of message is suppressed

Timing updates: 2 (2 implicit, 0 explicit) (0 incremental, 1 full, 1 logical)
Noise updates: 0 (0 implicit, 0 explicit) (0 incremental, 0 full)
Maximum memory usage for this session: 1524.92 MB
CPU usage for this session: 168 seconds
Elapsed time for this session: 59 seconds
Elapsed time spent in memory sampling: 0 seconds
Diagnostics summary: 150 warnings, 44 informations

Thank you for using pt_shell!

```

图 24: PT 脚本执行输出

Point	Incr	Path
clock wb_clk (rise edge)	0.00	0.00
clock network delay (propagated)	0.00	0.00
input external delay	10.00	10.00 r
PAD_dat_i[21] (in)	4.74 &	14.74 r
U_dat_i_21/P (PLBI8F)	0.00 &	14.74 r
U_dat_i_21/D (PLBI8F)	1.03 &	15.76 r
jpeg_top_inst/u19/u14/u4/U_dat_i_21ASTzrInst32345/Z (BUFHD20X)	0.16 &	15.92 r
jpeg_top_inst/u19/u14/u4/U_dat_i_21ASTzrInst32346/Z (BUFHD20X)	0.21 &	16.13 r
jpeg_top_inst/u19/u14/u4/U_dat_i_21ASTzrInst32347/Z (BUFHD20X)	0.22 &	16.35 r
jpeg_top_inst/u19/u14/u4/U_dat_i_21ASTttcInst30855/Z (BUFHD8X)	0.18 &	16.53 r
jpeg_top_inst/u19/u14/u4/mult_90/U102/S (FAHHD2X)	0.46 &	16.99 f
jpeg_top_inst/u19/u14/u4/mult_90/U108/C0 (FAHHD2X)	0.31 &	17.30 f
jpeg_top_inst/u19/u14/u4/mult_90/U66/C0 (FAHHD2X)	0.23 &	17.53 f
jpeg_top_inst/u19/u14/u4/mult_90/U107/C0 (FAHHD2X)	0.22 &	17.74 f
jpeg_top_inst/u19/u14/u4/mult_90/U103/C0 (FAHHD2X)	0.22 &	17.96 f
jpeg_top_inst/u19/u14/u4/mult_90/U120/C0 (FAHHD2X)	0.21 &	18.17 f
jpeg_top_inst/u19/u14/u4/mult_90/U76/C0 (FAHHD4X)	0.29 &	18.46 f
jpeg_top_inst/u19/u14/u4/mult_90/U68/Z (NAND2HD4X)	0.11 &	18.57 r
jpeg_top_inst/u19/u14/u4/mult_90/ASTtlrInst31475/Z (NAND3B1HD4X)	0.13 &	18.69 f
jpeg_top_inst/u19/u14/u4/mult_90/ASTtlrInst31473/Z (NAND2HD4X)	0.08 &	18.78 r
jpeg_top_inst/u19/u14/u4/mult_90/ASTtlrInst31472/Z (NAND3B1HD4X)	0.10 &	18.88 f
jpeg_top_inst/u19/u14/u4/mult_90/U85/Z (NAND2HD2X)	0.08 &	18.96 r
jpeg_top_inst/u19/u14/u4/mult_90/U78/Z (NAND3B1HD2X)	0.10 &	19.06 f
jpeg_top_inst/u19/u14/u4/mult_90/U118/C0 (FAHHD2X)	0.22 &	19.29 f
jpeg_top_inst/u19/u14/u4/mult_90/U117/C0 (FAHHD2X)	0.21 &	19.50 f
jpeg_top_inst/u19/u14/u4/mult_90/ASTtlrInst31441/Z (MUXI2HD4X)	0.22 &	19.71 r
jpeg_top_inst/u19/u14/u4/mult_90/U90/Z (NAND2HD1X)	0.09 &	19.81 f
jpeg_top_inst/u19/u14/u4/mult_90/U110/Z (NAND2HD2X)	0.08 &	19.89 r
jpeg_top_inst/u19/u14/u4/ASTtlrInst31436/Z (NAND2HDLX)	0.08 &	19.97 f
jpeg_top_inst/u19/u14/u4/ASTtlrInst31438/Z (NAND2HD1X)	0.09 &	20.06 r
jpeg_top_inst/u19/u14/u4/Y3_product_reg_18./RN (FFDCRHD1X)	0.00 &	20.06 r
data arrival time	20.00	20.00
	1.67	21.67
	0.00	21.67
	-0.22	21.45
		21.45
data required time	21.45	
data arrival time	-20.06	
slack (MET)	1.39	

图 25: PT 时序分析报告

7 设计心得

在本次数字集成电路设计课程设计中，我选择了已有设计 jpegencode，该设计可以将 TIF 格式图像转换为 JPEG 格式。整个设计过程中，我完成了以下任务：RTL 级仿真、DC 综合、FM 形式验证，以及课程要求的可选部分：Astro 自动布局布线、后端版图验证、PT 时序分析。

在 RTL 级仿真阶段，我使用了自写的仿真脚本并借助 Questasim 工具完成了仿真工作。仿真脚本的编写使我深入理解了设计的各个模块和其交互方式。DC 综合使用了 SMIC18 工艺库进行。并按照要求设置了约束。综合完成后，我详细检查了时序报告，分析了最差时序路径。通过这些分析，我了解到设计在实际运行时可能遇到的瓶颈。

Astro 自动布局布线阶段，我遇到了一些困难。在老师的帮助下，我逐步解决了这些问题，成功完成了后端物理验证。这个过程让我认识到，后端设计不仅需要扎实的理论基础，还需要耐心和细致的实践操作。PT 时序分析阶段，我重点关注了设计的时序约束，通过详细的时序分析，确保了设计的时序稳定性。

在整个项目进行过程中，我使用了 Git 进行版本控制。这不仅帮助我有效地管理了代码版本，还让我在遇到问题时能够快速回溯到之前的稳定版本。在此过程中，我也学到了许多关于版本控制的知识和技巧。

这次课程设计不仅让我掌握了数字集成电路设计的基本流程和工具使用方法，还让我深刻体会到实践操作的重要性。每一个环节的设计和验证，都需要仔细的思考和不断的尝试。在这个过程中，我也学会了如何有效地利用工具和资源，解决实际设计中的问题。这次课程设计让我在理论知识和实践技能方面都有了很大的提升，也让我对数字集成电路设计有了更深入的理解和更强的兴趣。未来，我将继续深入学习和实践，争取在这一领域取得更大的进步。