



Indian Institute of Technology Bombay

EE 705 VLSI Design Lab

Project RISC-V - ASIC

ASIC IMPLEMENTATION OF RISC-V

Course Instructor:

Prof. Laxmeesha Somappa

Team Members:

Algorithm Avengers	Silicon Syndicate	RAF
Srinivas Nagireddy (24M1159)	Bala Murugan S (24M1173)	Sumit Pal (24M1192)
Shanmukhi Ganesh Sai (24M1158)	LakshmidEEP Chowdary M (24M1150)	
Harisingh Nenavath (24M1133)	Juhi Bharti (24M1144)	
Ritesh Kumar (24M1160)		

Contents

1	TASK-1 ASIC Implementation for final	
	Integration of RISC-V	4
1.1	Contents of config.json file	4
1.2	Contents of macro_placement.cfg File	6
1.3	LVS Matching Screenshot	6
1.4	Magic Layout	7
1.5	Table of Summary	8
2	TASK-2 – ASIC Implementation of UART	
	(Team: Silicon Syndicate)	9
2.1	Contents of config.json file	9
2.2	Post Synthesis Simulation	10
2.3	LVS Matching Screenshot	11
2.4	Magic Layout	12
2.5	Table of Summary	13
3	TASK-2 – ASIC Implementation of ALU	
	(Team: Algorithm Avengers)	14
3.1	Contents of config.json file	14
3.2	Post Synthesis Simulation	15
3.3	LVS Matching Screenshot	16
3.4	Magic Layout	16
3.5	Table of Summary	17
4	TASK-2 – ASIC Implementation of DECODE	
	(Team: Algorithm Avengers)	18
4.1	Contents of config.json file	18
4.2	Post Synthesis Simulation	19
4.3	LVS Matching Screenshot	20
4.4	Magic Layout	20
4.5	Table of Summary	21
5	TASK-2 – ASIC Implementation of SPI	
	(Team: RAF)	22
5.1	Contents of config.json file	22
5.2	Post Synthesis Simulation	23
5.3	LVS Matching Screenshot	24
5.4	Magic Layout	24
5.5	Table of Summary	25
6	Conclusion	25

List of Figures

1	Screenshot of <code>macro_placement.cfg</code> file	6
2	Screenshot of the <code>39-riscv_soc.lvs.rpt</code> file	7
3	Layout in Magic	7
4	Number of cells used	8
5	Post-Synthesis simulation showing write operation	10
6	Post-Synthesis simulation showing read operation	11
7	Screenshot of the <code>uart_lite.lvs.rpt</code> file	11
8	Magic Layout of <code>uart_lite</code>	12
9	Number of cells used	13
10	Screenshot of <code>config.json</code> file	14
11	Screenshot of <code>pin_order.cfg</code> file	15
12	Post Synthesis Simulation Waveforms	16
13	Screenshot of the <code>39-riscv_alu.lvs.rpt</code> file	16
14	Magic Layout of ALU	17
15	Screenshot of <code>config.json</code> file	18
16	Post Synthesis Simulation Waveforms - I	19
17	Post Synthesis Simulation Waveforms - II	19
18	Screenshot of the <code>39-riscv_decode.lvs.rpt</code> file	20
19	Magic Layout of DECODE	20
20	Screenshot of <code>config.json</code> file	22
21	Screenshot of <code>pin_order.cfg</code> file	22
22	Post Synthesis Simulation Waveforms - I	23
23	Post Synthesis Simulation Waveforms - II	23
24	Screenshot of the <code>39-spi_lite.lvs.rpt</code> file	24
25	Magic Layout of <code>spi_lite</code>	24
26	Number of cells used	25

1 TASK-1 ASIC Implementation for final Integration of RISCv

1.1 Contents of config.json file

We started with uploading all our `riscv_soc` modules to the Openlane installed server. There we wrote a `config.json` where the location of the modules present are mentioned in a hierarchical fashion. The target clock frequency is 100MHz. We included the Instruction cache and data cache macros appropriately using `FP_PDN_MACROHOOKS` command. There also hierarchy should be followed and instance names are used.

```
{
  "DESIGN_NAME": "riscv_soc",
  "VERILOG_FILES": [
    "dir::src/riscv_top/def.v",
    "dir::src/soc/timer_defs.v",
    "dir::src/soc/spi_lite_defs.v",
    "dir::src/soc/irq_def.v",
    "dir::src/soc/timer.v",
    "dir::src/soc/uart.v",
    "dir::src/soc/spi_lite.v",
    "dir::src/soc/gpio.v",
    "dir::src/soc/irq.v",
    "dir::src/soc/axi4_retime.v",
    "dir::src/soc/tap.v",
    "dir::src/soc/arbiter.v",
    "dir::src/riscv_top/alu/ADD_using_BK.v",
    "dir::src/riscv_top/alu/Arith_shift_right.v",
    "dir::src/riscv_top/alu/bit_AND_op.v",
    "dir::src/riscv_top/alu/bit_OR_op.v",
    "dir::src/riscv_top/alu/BK_Adder_32.v",
    "dir::src/riscv_top/alu/Dadda.v",
    "dir::src/riscv_top/alu/Left_Right_shifter.v",
    "dir::src/riscv_top/alu/MUL_using_Dadda.v",
    "dir::src/riscv_top/alu/Signed_compar.v",
    "dir::src/riscv_top/alu/Unsigned_compar.v",
    "dir::src/axi4lite_axi4_conv.v",
    "dir::src/riscv_top/icache_data_ram.v",
    "dir::src/riscv_top/icache_tag_ram.v",
    "dir::src/riscv_top/riscv_regfile.v",
    "dir::src/riscv_top/riscv_csr.v",
    "dir::src/riscv_top/riscv_decode.v",
    "dir::src/riscv_top/riscv_exec.v",
    "dir::src/riscv_top/riscv_fetch.v",
    "dir::src/riscv_top/riscv_alu.v",
    "dir::src/riscv_top/riscv_lsu.v",
    "dir::src/riscv_top/icache.v",
    "dir::src/riscv_top/dport_bridge.v",
    "dir::src/riscv_top/riscv_core.v",
    "dir::src/riscv_top/riscv_top.v",
    "dir::src/soc/soc.v",
  ]
}
```

```

"dir::src/riscv_soc.v"],
  "VERILOG_INCLUDE_DIRS": ["dir::src/riscv_top","dir::src/soc","dir
    ::src/riscv_top/alu"],
  "SYNTH_FLAT_TOP": 0,
  "CLOCK_PORT": "clk_i",
  "CLOCK_PERIOD": 10.0,
  "VDD_NETS": "vccd1",
  "GND_NETS": "vssd1",
  "CLOCK_BUFFER_FANOUT": 16,
  "CLOCK_TREE_SYNTH_MAX_FANOUT": 24,
  "PL_RESIZER_BUFFER_INPUT_PORTS": 1,
  "PL_TARGET_DENSITY": 0.45,
  "PL_MAX_DISPLACEMENT_X": 500,
  "PL_MAX_DISPLACEMENT_Y": 500,
  "GRT_ALLOW_CONGESTION": 1,
  "GRT_OVERFLOW_ITERS": 100,
  "GPL_ROUTABILITY_DRIVEN_OVERFLOW": "0.05",
  "PL_ROUTABILITY_DRIVEN": true,
  "PL_TIME_DRIVEN": true,
  "FP_ASPECT_RATIO": 1,
  "FP_CORE_MARGIN": 30,
  "FP_PDN_VPITCH": 50,
  "FP_PDN_HPITCH": 50,
  "FP_PDN_MACRO_HOOKS": "u_core.u_icache.u_tag0.submodule.sram1
    vccd1 vssd1 vccd1 vssd1, u_core.u_icache.u_tag1.submodule.sram1
    vccd1 vssd1 vccd1 vssd1, u_core.u_icache.u_data0.submodule.
    sram2 vccd1 vssd1 vccd1 vssd1, u_core.u_icache.u_data1.
    submodule.sram2 vccd1 vssd1 vccd1 vssd1",
  "MACRO_PLACEMENT_CFG": "dir::macro_placement.cfg",
  "EXTRA_LEFS": ["dir::src/riscv_top/
    sky130_sram_1kbytes_1rwir_20x256_20/
    sky130_sram_1kbytes_1rwir_20x256_20.lef",
"dir::src/riscv_top/sky130_sram_8kbytes_1rwir_32x2048_32/
    sky130_sram_8kbytes_1rwir_32x2048_32.lef"],
  "EXTRA_GDS_FILES": ["dir::src/riscv_top/
    sky130_sram_1kbytes_1rwir_20x256_20/
    sky130_sram_1kbytes_1rwir_20x256_20.gds",
"dir::src/riscv_top/sky130_sram_8kbytes_1rwir_32x2048_32/
    sky130_sram_8kbytes_1rwir_32x2048_32.gds"],
  "VERILOG_FILES_BLACKBOX": ["dir::src/riscv_top/
    sky130_sram_1kbytes_1rwir_20x256_20/
    sky130_sram_1kbytes_1rwir_20x256_20.v",
"dir::src/riscv_top/sky130_sram_8kbytes_1rwir_32x2048_32/
    sky130_sram_8kbytes_1rwir_32x2048_32.v"],
  "RUN_KLAYOUT_XOR": false,
  "MAGIC_DRC_USE_GDS": false,
  "QUIT_ON_MAGIC_DRC": false,
  "FP_SIZING": "relative",
  "FP_CORE_UTIL": 50,
  "FP_PDN_MULTILAYER": true,
  "pdk::sky130*": {
    "FP_CORE_UTIL": 45,

```

```

    "CLOCK_PERIOD": 15,
    "scl::sky130_fd_sc_hs": {
        "CLOCK_PERIOD": 8
    },
    "scl::sky130_fd_sc_ls": {
        "MAX_FANOUT_CONSTRAINT": 5
    }
},
"pdk::gf180mcu*": {
    "CLOCK_PERIOD": 24.0,
    "FP_CORE_UTIL": 40,
    "MAX_FANOUT_CONSTRAINT": 4,
    "PL_TARGET_DENSITY": 0.5
}
}

```

1.2 Contents of macro_placement.cfg File

The file `macro_placement.cfg` consists of the locations the four macros should be placed. As said before the OpenRoad compiler expects instance names and the proper hierarchical flow to identify the macro. The following image shows the contents of this file.



```

u_core.u_icache.u_tag0.submodule.sram1 75 75 N
u_core.u_icache.u_tag1.submodule.sram1 75 575 N
u_core.u_icache.u_data0.submodule.sram2 575 75 N
u_core.u_icache.u_data1.submodule.sram2 575 1400 N

```

Figure 1: Screenshot of `macro_placement.cfg` file

1.3 LVS Matching Screenshot

After completing the Openlane flow successfully, we navigated to the file `39-riscv_soc.lvs.rpt` which is present inside the `reports/signoff` folder. The following image shows the content of that file.



Figure 2: Screenshot of the 39-riscv_soc.lvs.rpt file

1.4 Magic Layout

The following figure shows the magic layout generated after the complete Openlane RTL to GDS Flow, then the number of sky_130 cells used in this design is also shown in the consecutive figure.

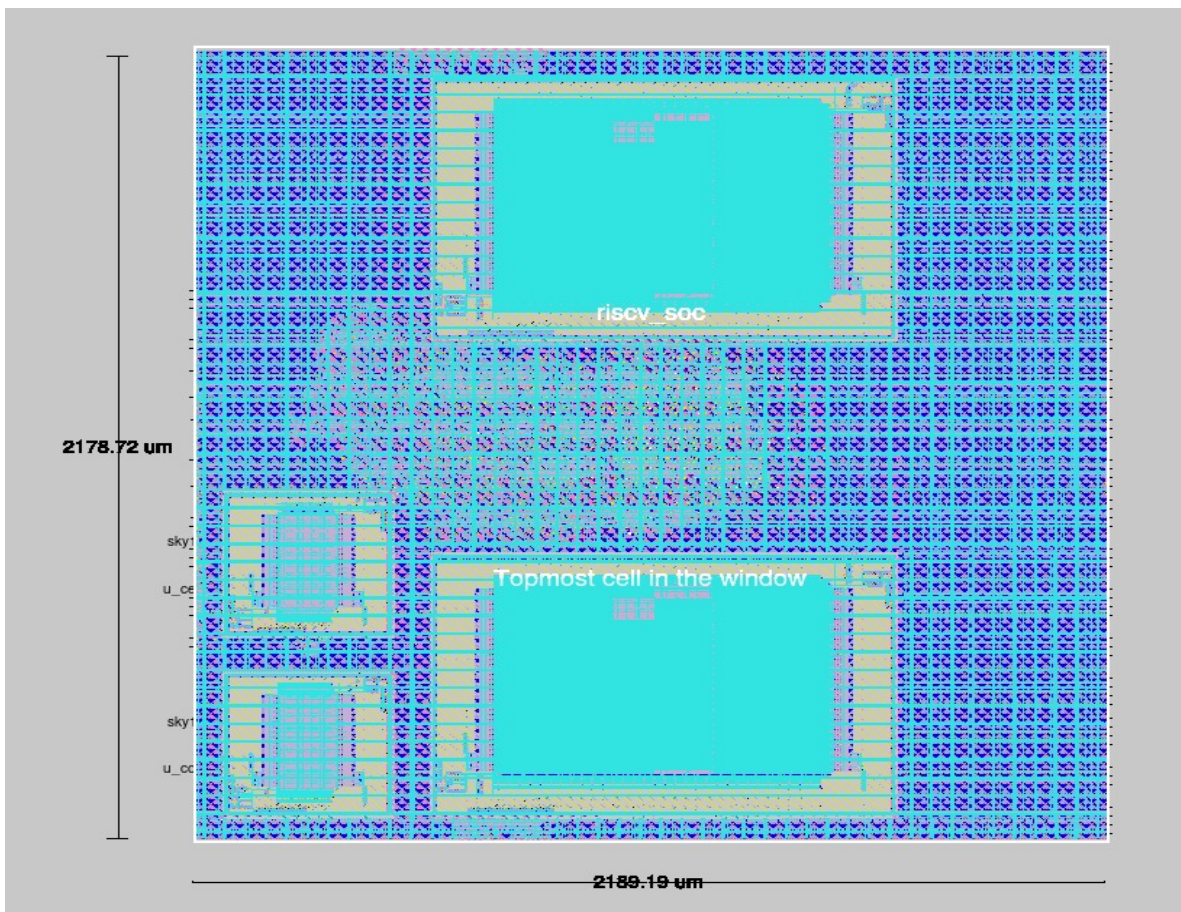


Figure 3: Layout in Magic

```

[EE705_22@vlsi73 results]$ cd synthesis
[EE705_22@vlsi73 synthesis]$ ls
riscv_soc.sdf  riscv_soc.v
[EE705_22@vlsi73 synthesis]$ cat riscv_soc.v | grep "sky130_fd_sc_hd__" | wc -l
23754
[EE705_22@vlsi73 synthesis]$ █

```

Figure 4: Number of cells used

1.5 Table of Summary

The following summary table shows the timing, area and power parameters obtained from the 24-grt_sta.log present inside logs/routing folder.

Clock Frequency (MHz)	100
Worst case setup slack (ns)	2.71
Worst case hold slack (ns)	0.14
Design Area (μm^2)	2264137
Total Power Consumption (μW)	27800

2 TASK-2 – ASIC Implementation of UART (Team: Silicon Syndicate)

2.1 Contents of config.json file

Here, in the config.json, our target clock frequency is 125MHz, then we made a uart_lite.sdc file where there also we specified the target clock frequency as 125MHz. This allows the tool to route efficiently for the given clock frequency, we later tested the uart_lite module with a clock frequency 50MHz.

```
{
  "//": "Basics",
  "DESIGN_NAME": "uart_lite",
  "VERILOG_FILES": "dir::src/*.v",
  "CLOCK_PERIOD": 8,
  "CLOCK_PORT": "clk_i",
  "PNR_SDC_FILE": "dir::src/uart_lite.sdc",
  "SIGNOFF_SDC_FILE": "dir::src/uart_lite.sdc",
  "//": "PDN",
  "FP_PDN_VOFFSET": 5,
  "FP_PDN_HOFFSET": 5,
  "FP_PDN_VWIDTH": 2,
  "FP_PDN_HWIDTH": 2,
  "FP_PDN_VPITCH": 30,
  "FP_PDN_HPITCH": 30,
  "DIE_AREA": "0 0 125 125",
  "FP_CORE_UTIL": 60,
  "PL_TARGET_DENSITY": 0.42,
  "FP_PDN_SKIPTRIM": true,
  "FP_PDN_MULTILAYER": true,
  "//": "Pin Order",
  "//": "Technology-Specific Configs",
  "pdk::sky130*": {
    "FP_CORE_UTIL": 34,
    "CLOCK_PERIOD": 8,
    "scl::sky130_fd_sc_hs": {
      "CLOCK_PERIOD": 6
    },
    "scl::sky130_fd_sc_ls": {
      "MAX_FANOUT_CONSTRAINT": 5
    }
  },
  "pdk::gf180mcu*": {
    "CLOCK_PERIOD": 15.0,
    "FP_CORE_UTIL": 35,
    "MAX_FANOUT_CONSTRAINT": 4,
    "PL_TARGET_DENSITY": 0.5
  }
}
```

2.2 Post Synthesis Simulation

After entering into the openlane container we executed the command `run_synthesis`, which mapped the behavioural model verilog file written by us to the standard cells provided by the `sky130_pdk`. The following screenshots are obtained after running the testbench by using the verilog module obtained after synthesis. The testbench gives a clock signal of frequency of 50 MHz and the baud rate is set as 115200 bits per second. In the testbench, first we sent the hexadecimal data E9 through the pin `rx_i`. Then the transmitter buffer is written a hexadecimal data 55, when the `baud_tick` approaches it starts sending the data stored in TX buffer through the pin `tx_o`. The above mentioned operation is shown in the following images. The second image shows the value of status register as `cfg_rdata_o` and the value E9 stored in RX Buffer.

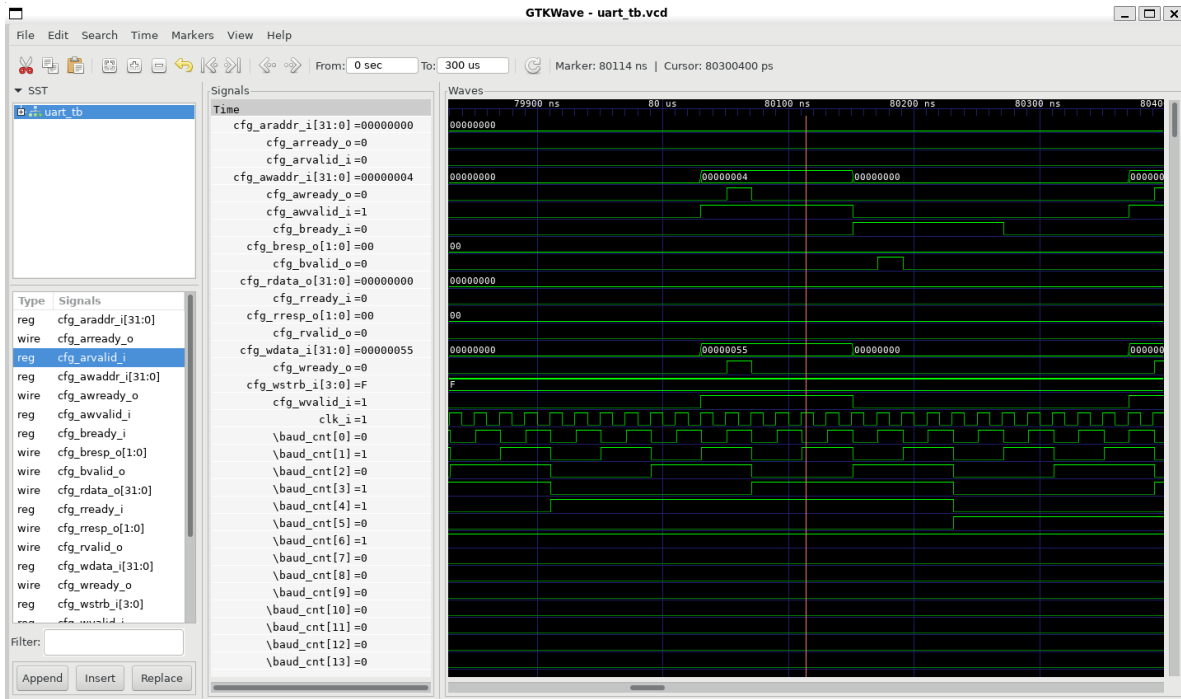


Figure 5: Post-Synthesis simulation showing write operation

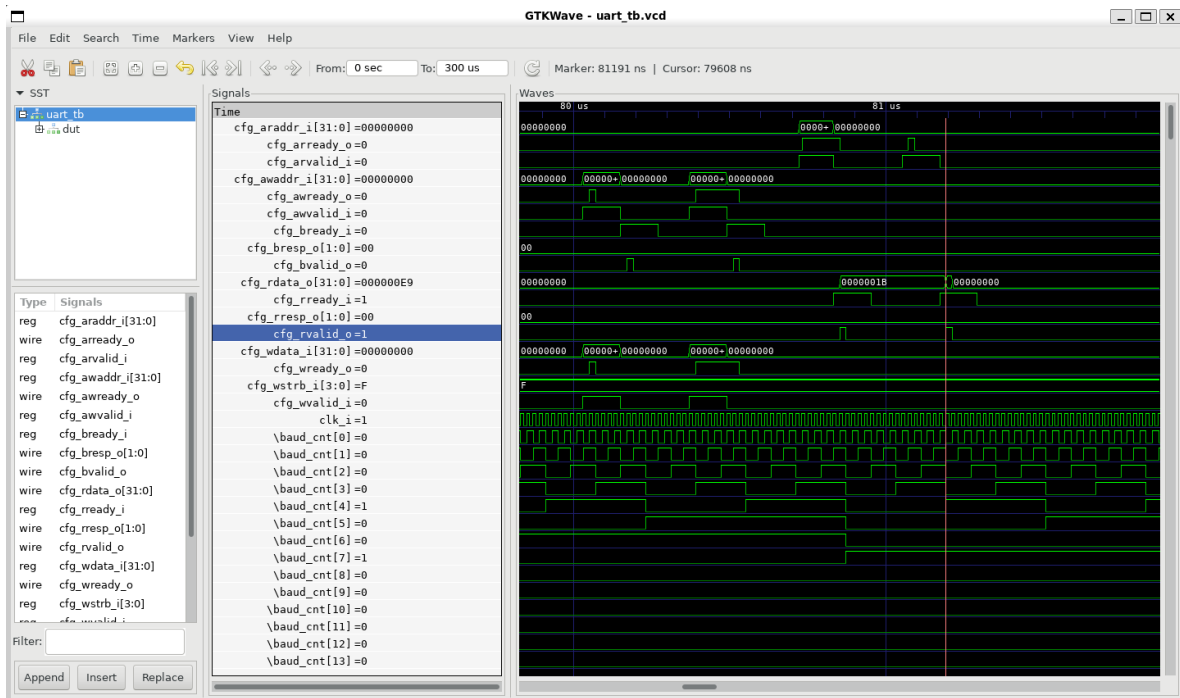


Figure 6: Post-Synthesis simulation showing read operation

2.3 LVS Matching Screenshot

After completing the Openlane flow successfully, we navigated to the file `39-uart_lite.lvs.rpt` which is present inside the `reports/signoff` folder. The following image shows the content of that file.

```

41-uart_lite.lvs.rpt
1 LVS reports no net, device, pin, or property mismatches.
2
3 Total errors = 0
4

```

Figure 7: Screenshot of the `uart_lite.lvs.rpt` file

2.4 Magic Layout

The following figure shows the magic layout generated after the complete Openlane RTL to GDS Flow, then the number of sky_130 cells used in this design is also shown in the consecutive figure.

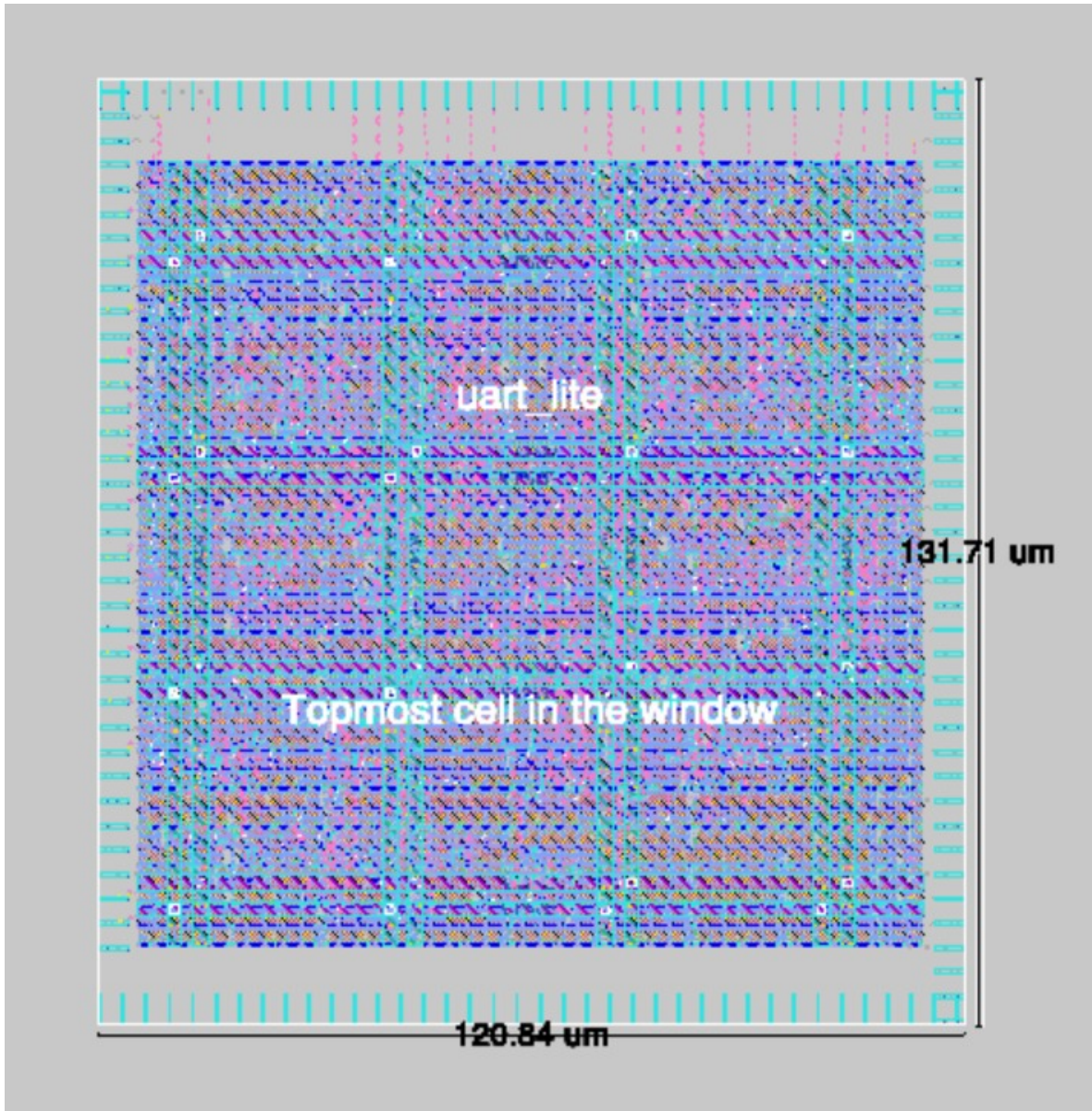


Figure 8: Magic Layout of `uart_lite`

```

[EE705_22@vlsi73 results]$ cd synthesis/
[EE705_22@vlsi73 synthesis]$ ls
uart_lite.sdf  uart_lite.v
[EE705_22@vlsi73 synthesis]$ cat uart_lite.v | grep "sky130_fd_sc_hd__" | wc -l
386
[EE705_22@vlsi73 synthesis]$ █

```

Figure 9: Number of cells used

2.5 Table of Summary

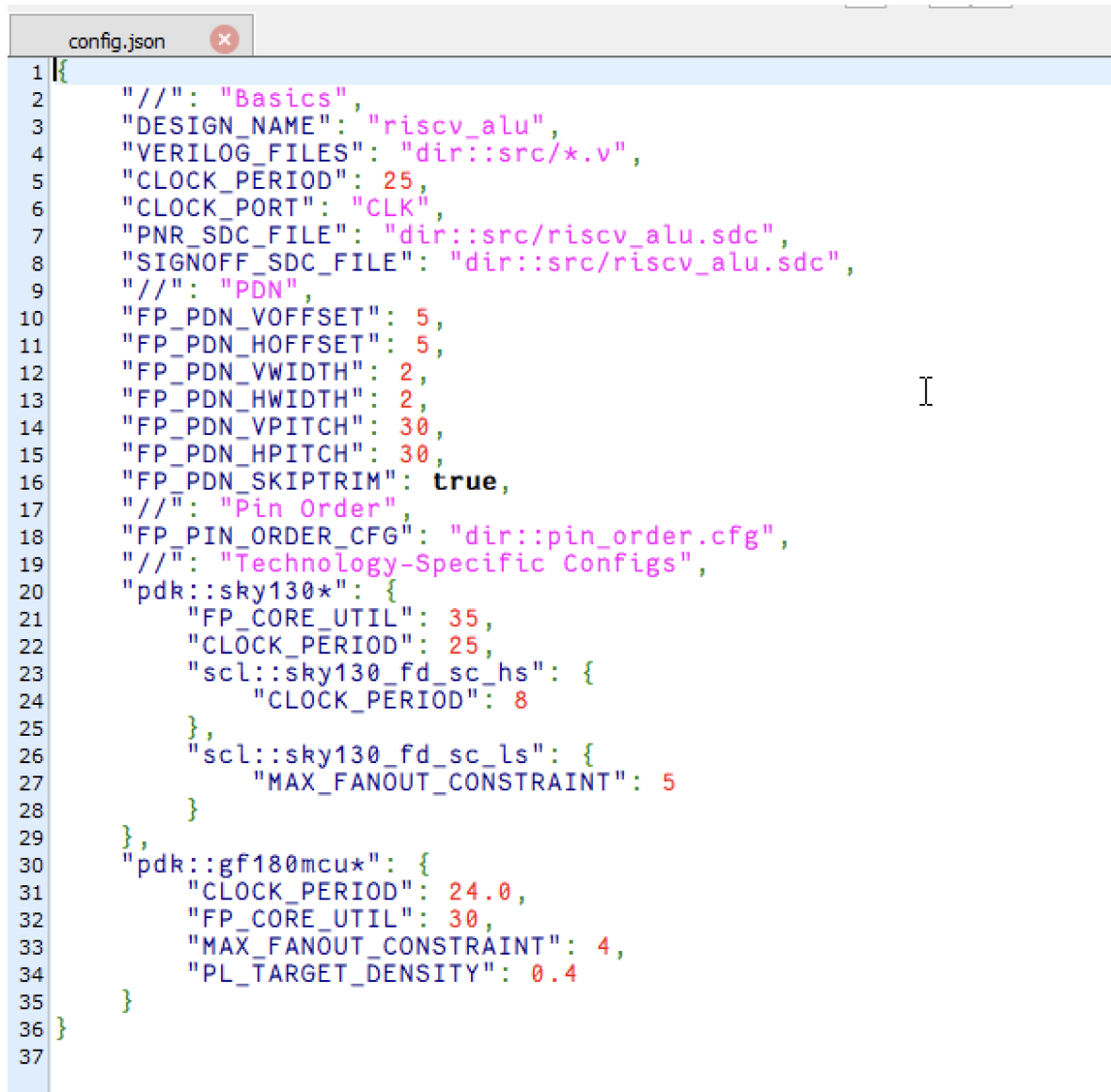
The following summary table shows the timing, area and power parameters obtained from the 24-grt_sta.log present inside logs/routing folder.

Clock Frequency (MHz)	125
Worst case setup slack (ns)	5.95
Worst case hold slack (ns)	0.47
Design Area (μm^2)	4383
Total Power Consumption (μW)	793

3 TASK-2 – ASIC Implementation of ALU (Team: Algorithm Avengers)

3.1 Contents of config.json file

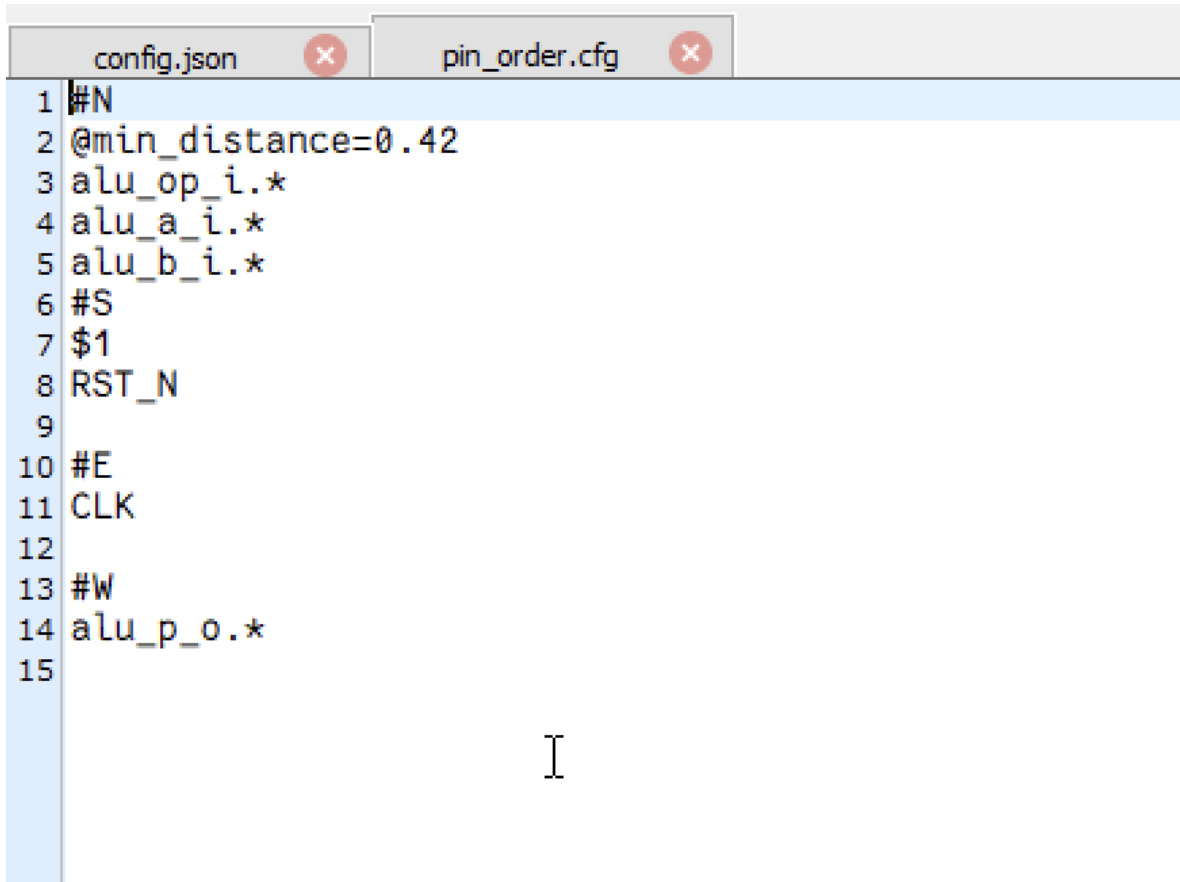
The following image shows the config.json file for the module riscv_alu.v.



```
1 {
2   "//": "Basics",
3   "DESIGN_NAME": "riscv_alu",
4   "VERILOG_FILES": "dir::src/*.v",
5   "CLOCK_PERIOD": 25,
6   "CLOCK_PORT": "CLK",
7   "PNR_SDC_FILE": "dir::src/riscv_alu.sdc",
8   "SIGNOFF_SDC_FILE": "dir::src/riscv_alu.sdc",
9   "//": "PDN",
10  "FP_PDN_VOFFSET": 5,
11  "FP_PDN_HOFFSET": 5,
12  "FP_PDN_VWIDTH": 2,
13  "FP_PDN_HWIDTH": 2,
14  "FP_PDN_VPITCH": 30,
15  "FP_PDN_HPITCH": 30,
16  "FP_PDN_SKIPTRIM": true,
17  "//": "Pin Order",
18  "FP_PIN_ORDER_CFG": "dir::pin_order.cfg",
19  "//": "Technology-Specific Configs",
20  "pdk::sky130*": {
21    "FP_CORE_UTIL": 35,
22    "CLOCK_PERIOD": 25,
23    "scl::sky130_fd_sc_hs": {
24      "CLOCK_PERIOD": 8
25    },
26    "scl::sky130_fd_sc_ls": {
27      "MAX_FANOUT_CONSTRAINT": 5
28    }
29  },
30  "pdk::gf180mcu*": {
31    "CLOCK_PERIOD": 24.0,
32    "FP_CORE_UTIL": 30,
33    "MAX_FANOUT_CONSTRAINT": 4,
34    "PL_TARGET_DENSITY": 0.4
35  }
36 }
37
```

Figure 10: Screenshot of config.json file

The following image shows the contents of `pin_order.cfg` file.



```
1 #N
2 @min_distance=0.42
3 alu_op_i.*
4 alu_a_i.*
5 alu_b_i.*
6 #S
7 $1
8 RST_N
9
10 #E
11 CLK
12
13 #W
14 alu_p_o.*
15
```

Figure 11: Screenshot of `pin_order.cfg` file

3.2 Post Synthesis Simulation

We have successfully simulated the ALU for all the specified operations. Upon analyzing the waveform results, we observe that each operation is functioning correctly and producing the expected outputs. For instance, when the opcode for multiplication (MUL), which is 0110, is provided along with `input_a = 30` and `input_b = 10`, the output generated is 300, which confirms the correct execution of the multiplication logic. Similar validations have been done for other operations, all of which have produced accurate results as per the expected ALU behavior. The following images shows the output waveform of the `riscv_alu.v` module after synthesis. Now the file will be mapped to sky_130 cells. The verilog file obtained after `run_synthesis` is tested and the following waveforms are obtained.

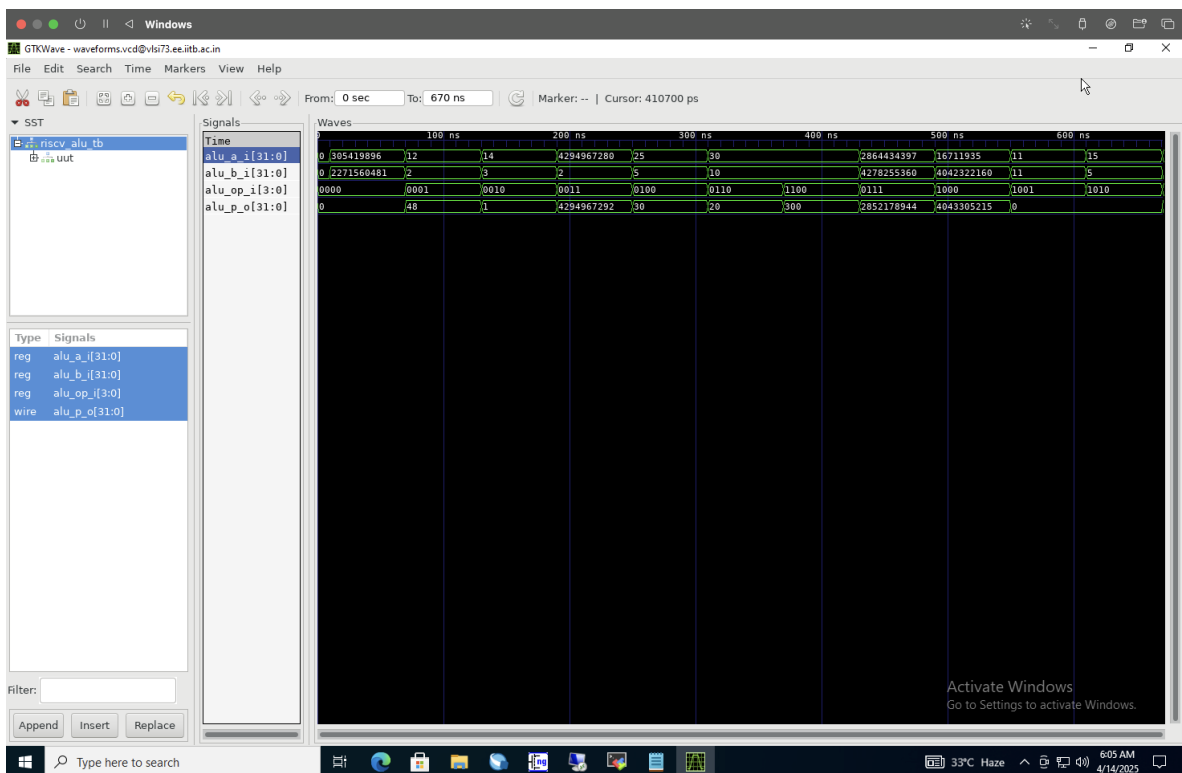


Figure 12: Post Synthesis Simulation Waveforms

3.3 LVS Matching Screenshot

After completing the Openlane flow successfully, we navigated to the file `39-riscv_alu.lvs.rpt` which is present inside the `reports/signoff` folder. The following image shows the content of that file.

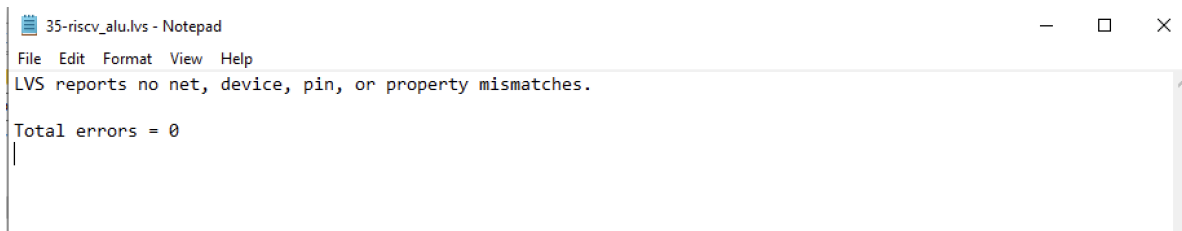


Figure 13: Screenshot of the `39-riscv_alu.lvs.rpt` file

3.4 Magic Layout

The following figure shows the magic layout generated after the complete Openlane RTL to GDS Flow for the ALU module.

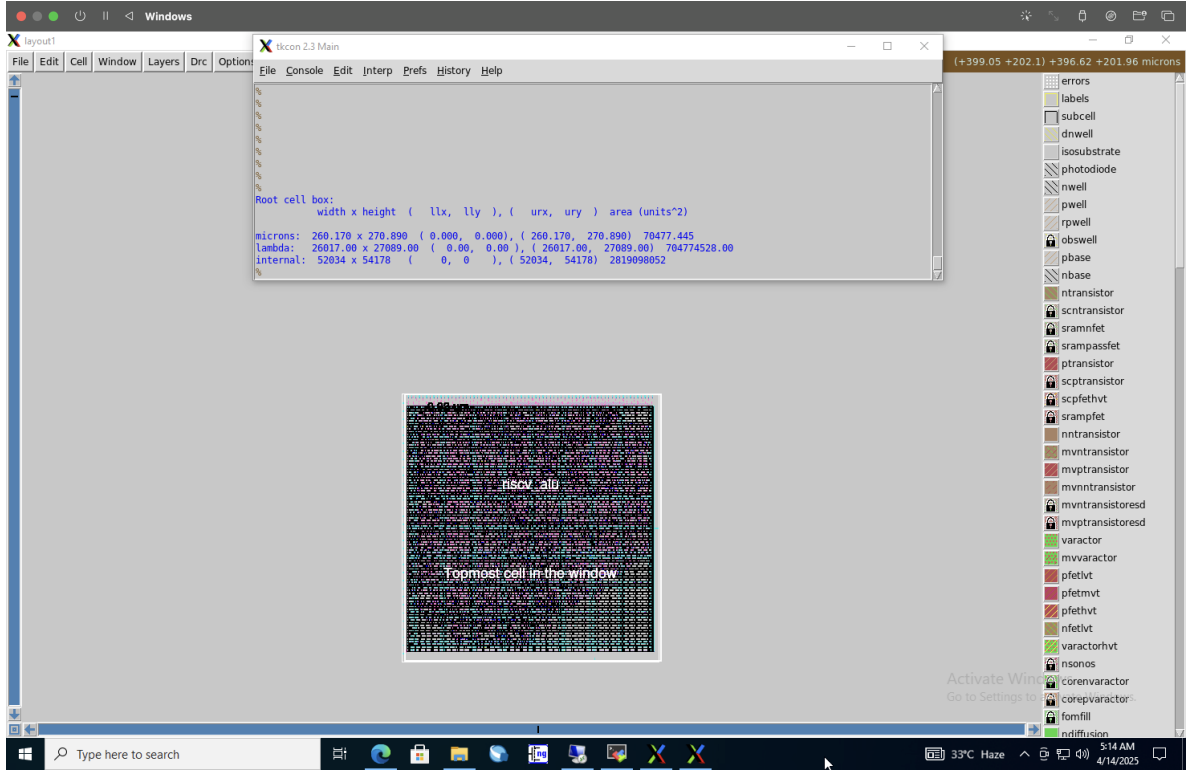


Figure 14: Magic Layout of ALU

3.5 Table of Summary

The following summary table shows the timing, area and power parameters obtained from the 24-grt_sta.log present inside logs/routing folder.

Clock Frequency (MHz)	100
Worst case setup slack (ns)	inf
Worst case hold slack (ns)	inf
Design Area (μm^2)	19950
Total Power Consumption (nW)	9.87
Number of sky_130 cells used	2397

4 TASK-2 – ASIC Implementation of DECODE (Team: Algorithm Avengers)

4.1 Contents of config.json file

The following image shows the config.json file for the module riscv_decode.v.



```
config.json - Notepad
File Edit Format View Help
k
{
  "///": "Basic Design Configuration",
  "DESIGN_NAME": "riscv_decode",
  "VERILOG_FILES": "dir::src/riscv_decode.srscs/source/new/*.v",
  "CLOCK_PERIOD": 25,
  "CLOCK_PORT": "CLK",
  "PNR_SDC_FILE": "dir::src/riscv_decode.srscs/source/new/riscv_decode.sdc",
  "SIGNOFF_SDC_FILE": "dir::src/riscv_decode.srscs/source/new/riscv_decode.sdc",

  "///": "Power Delivery Network",
  "FP_PDN_VOFFSET": 8,
  "FP_PDN_HOFFSET": 8,
  "FP_PDN_VWIDTH": 3.2,
  "FP_PDN_HWIDTH": 3.2,
  "FP_PDN_VPITCH": 80,
  "FP_PDN_HPITCH": 80,
  "FP_PDN_SKIPTRIM": false,
  "FP_PDN_CORE_RING": true,
  "FP_PDN_ENABLE_RAILS": true,

  "///": "Placement Strategy",
  "FP_CORE_UTIL": 25,
  "PL_TARGET_DENSITY": 0.26,
  "PL_TIME_DRIVEN": true,
  "PL_ROUTABILITY_DRIVEN": true,
  "PL_RESIZER_OVERBUFFER": true,
  "PL_RESIZER_TIMING_OPTIMIZATIONS": true,
  "PL_RESIZER_BUFFER_INPUT_PORTS": true,
  "PL_RESIZER_BUFFER_OUTPUT_PORTS": true,
  "PL_MAX_DISPLACEMENT_X": 500,
  "PL_MAX_DISPLACEMENT_Y": 500,

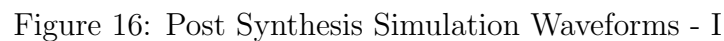
  "///": "Routing Configuration",
  "RT_LAYER_ADJUSTMENT": ["met1 0.3", "met2 0.3", "met3 0.7", "met4 0.8"],
  "GRT_OVERFLOW_ITERS": 200,
  "GRT_ALLOW_CONGESTION": true,
  "GRT_ANTENNA_ITERS": 10,
  "GRT_REPAIR_ANTENNAS": true,
  "GRT_MACRO_EXTENSION": 30,

  "///": "Timing Constraints",
  "CLOCK_UNCERTAINTY": 0.5,
  "IO_DELAY_CONSTRAINT": 2.0,
  "SYNTH_MAX_FANOUT": 8,
  "MAX_TRANSITION_CONSTRAINT": 1.5,

  "///": "Sky130-Specific Optimizations",
  "pdk::sky130*": {
    "FP_IO_MODE": 1,
    "RT_MIN_LAYER": "met1",
    "RT_MAX_LAYER": "met4",
    "DIODE_INSERTION_STRATEGY": 3,
    "FP_WELLTAP_CELL": "sky130_fd_sc_hd__tapvpwrvngnd_1",
    "FP_ENDCAP_CELL": "sky130_fd_sc_hd__decap_4",
    "CTS_CLK_BUFFER_LIST": [
      "sky130_fd_sc_hd__clkbuf_8",
      "sky130_fd_sc_hd__clkbuf_16"
    ],
    "PL_OPTIMIZE_MIRRORING": true,
    "PL_SKIP_INITIAL_PLACEMENT": false
  },
}
```

Figure 15: Screenshot of config.json file

We designed the testbench to cover all major RISC-V instruction types in a controlled environment, ensuring correct decoding and execution. The ‘fetch_valid_i’ and ‘fetch_pc_i’ signals indicate valid instructions and their PC values. On decoding, the ‘exec_opcode_valid_o’ signal is asserted to pass instructions to the execution stage. Results are written back to the register file unless suppressed by the squash signal. Stall conditions deassert ‘fetch_accept_o’ to pause instruction fetch. Branch instructions redirect control flow appropriately. Only one bit in ‘opcode_instr_o[57:0]’ is asserted per instruction, indicating its type. The following images shows the output waveform of the `riscv_decode.v` module after synthesis. Now the file will be mapped to sky_130 cells. The verilog file obtained after `run_synthesis` is tested and the following waveforms are obtained.



4.3 LVS Matching Screenshot

After completing the Openlane flow successfully, we navigated to the file `39-riscv_decode.lvs.rpt` which is present inside the `reports/signoff` folder. The following image shows the content of that file.

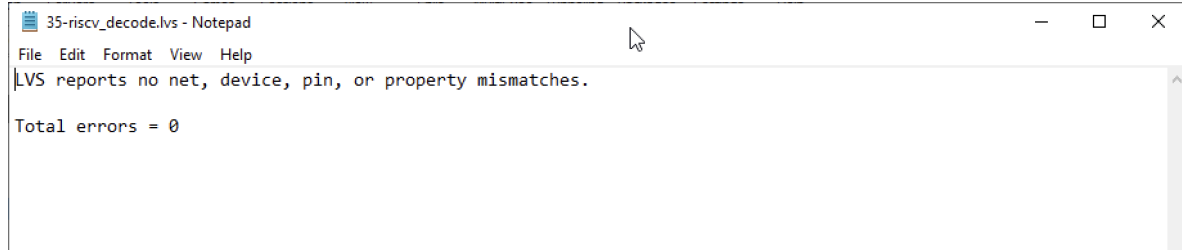


Figure 18: Screenshot of the `39-riscv_decode.lvs.rpt` file

4.4 Magic Layout

The following figure shows the magic layout generated after the complete Openlane RTL to GDS Flow for the ALU module.

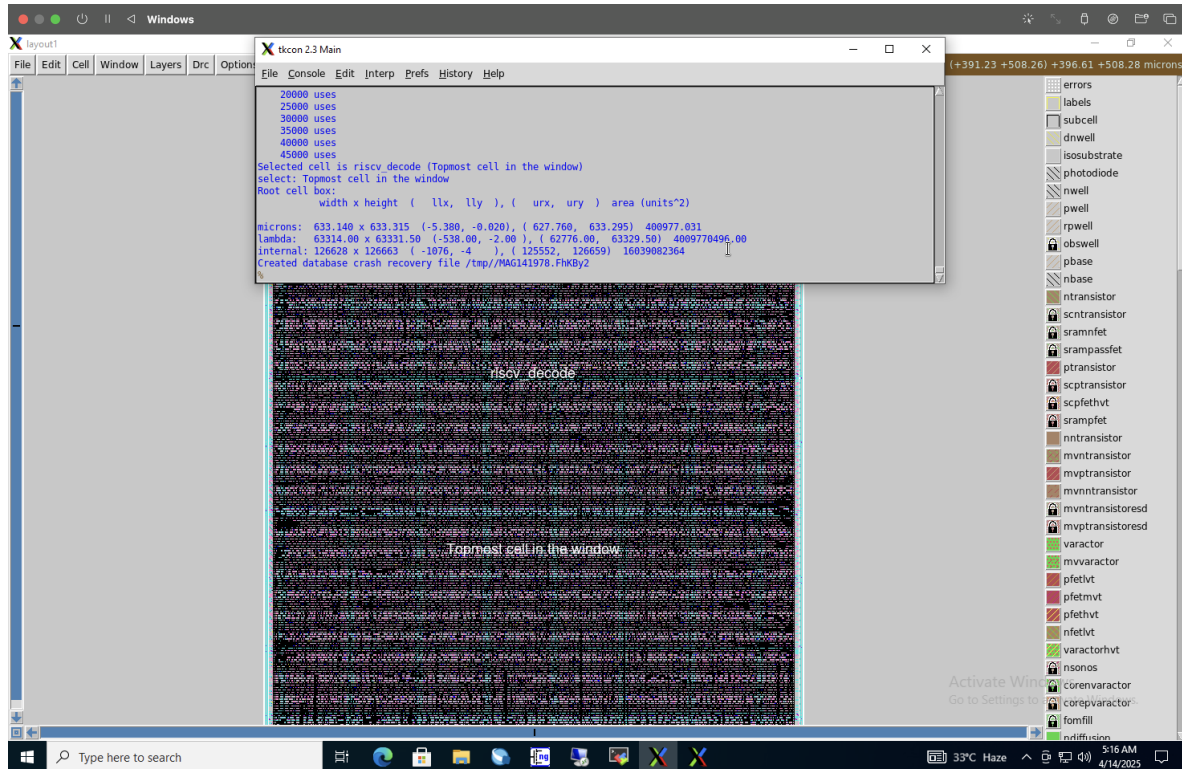


Figure 19: Magic Layout of DECODE

4.5 Table of Summary

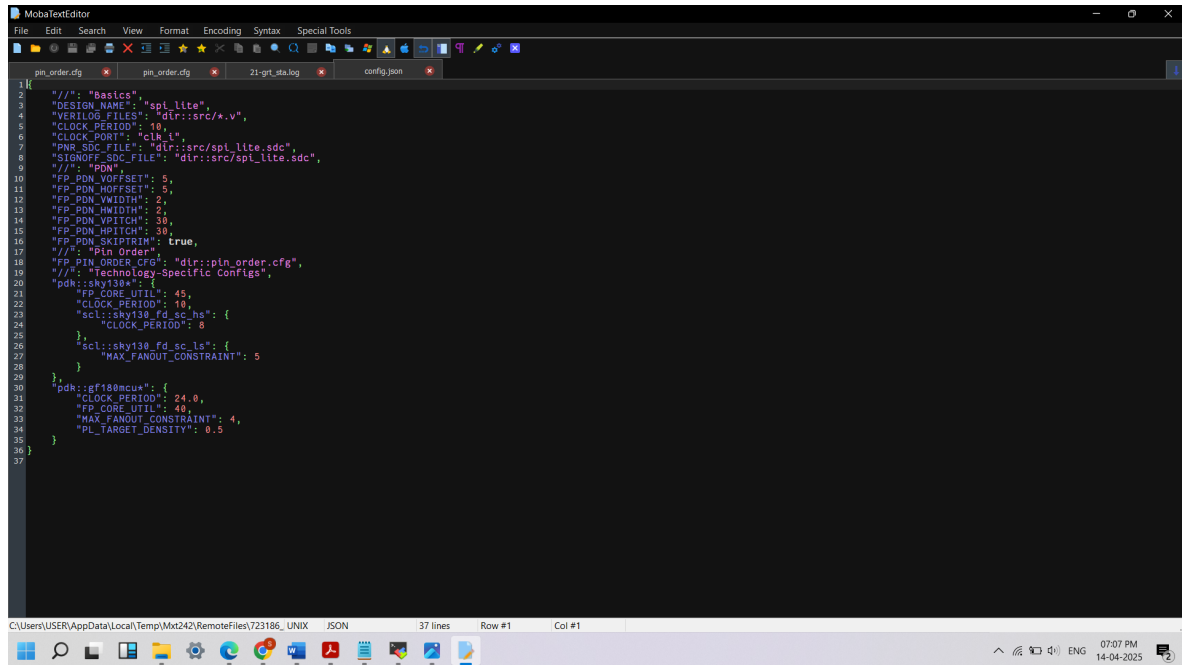
The following summary table shows the timing, area and power parameters obtained from the `24-grt_sta.log` present inside `logs/routing` folder.

Clock Frequency (MHz)	100
Worst case setup slack (ns)	6.45
Worst case hold slack (ns)	0.40
Design Area (μm^2)	115963
Total Power Consumption (nW)	1500
Number of sky_130 cells used	8742

5 TASK-2 – ASIC Implementation of SPI (Team: RAF)

5.1 Contents of config.json file

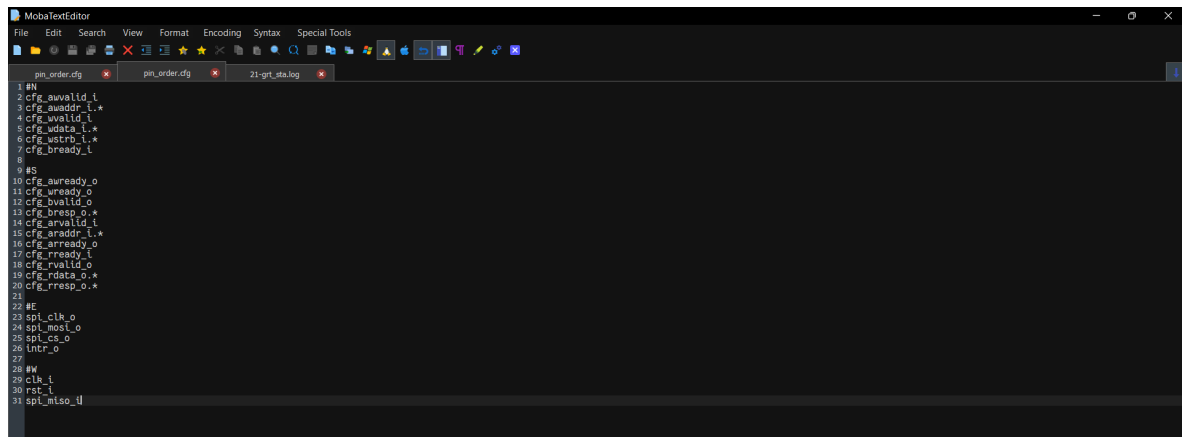
The following image shows the config.json file for the module spi_lite.v.



```
1 {
2   "///": "Basics",
3   "DESIGN_NAME": "spi_lite",
4   "VERILOG_FILES": "dir::src/*.v",
5   "CLOCK_PERIOD": 10,
6   "CLOCK_PORT": "clk_i",
7   "PWR_SDC_FILE": "dir::src/spi_lite.sdc",
8   "SIGNOFF_SDC_FILE": "dir::src/spi_lite.sdc",
9   "///": "PDN",
10  "FP_PDN_VOFFSET": 5,
11  "FP_PDN_VOFFSET": 5,
12  "FP_PDN_VWIDTH": 2,
13  "FP_PDN_WIDTH": 2,
14  "FP_PDN_WIDTH": 30,
15  "FP_PDN_WIDTH": 30,
16  "FP_PDN_SKIPTRIM": true,
17  "///": "Pin Order",
18  "FP_PIN_ORDER_CFG": "dir::pin_order.cfg",
19  "///": "Technology-Specific Configs",
20  "pdk::sky130": {
21    "FP_CORE_UTIL": 40,
22    "CLOCK_PERIOD": 10,
23    "scl::sky130_fd_sc_hs": {
24      "CLOCK_PERIOD": 8
25    },
26    "scl::sky130_fd_sc_ls": {
27      "MAX_FANOUT_CONSTRAINT": 5
28    }
29  },
30  "pdk::gf180mcu": {
31    "CLOCK_PERIOD": 24.0,
32    "FP_CORE_UTIL": 40,
33    "MAX_FANOUT_CONSTRAINT": 4,
34    "PL_TARGET_DENSITY": 0.5
35  }
36 }
37
```

Figure 20: Screenshot of config.json file

The following image shows the contents of pin_order.cfg file.



```
1 #N
2 cfg_avvalid_i
3 cfg_avaddr_i.*
4 cfg_avdata_i.*
5 cfg_wdata_i.*
6 cfg_wstrb_i.*
7 cfg_bready_i
8
9 #S
10 cfg_uready_o
11 cfg_bvalid_o
12 cfg_bready_o.*
13 cfg_avvalid_i
14 cfg_avaddr_i.*
15 cfg_avdata_i.*
16 cfg_wdata_i.*
17 cfg_wstrb_i.*
18 cfg_rready_i
19 cfg_rvalid_o
20 cfg_rdata_o.*
21 cfg_rresp_o.*
22
23 #E
24 spi_clk_o
25 spi_mst_o
26 spi_cs_o
27 intr_o
28
29 #W
30 clk_i
31 rst_i
32 spi_mst_o
33
```

Figure 21: Screenshot of pin_order.cfg file

5.2 Post Synthesis Simulation

The SPI clock (spi_clk_o) and chip select (spi_cs_o) signals are toggling correctly, indicating proper SPI protocol initiation and transaction periods. The pin spi_mosi_o is actively transmitting data when spi_cs_o is high, confirming valid SPI data transfer. The data on cfg_wdata_i and cfg_rdata_o reflects correct write and read operations to/from the SPI slave. Handshake signals like cfg_ready_i and cfg_valid_o are synchronized properly, showing correct communication with the SPI interface logic.

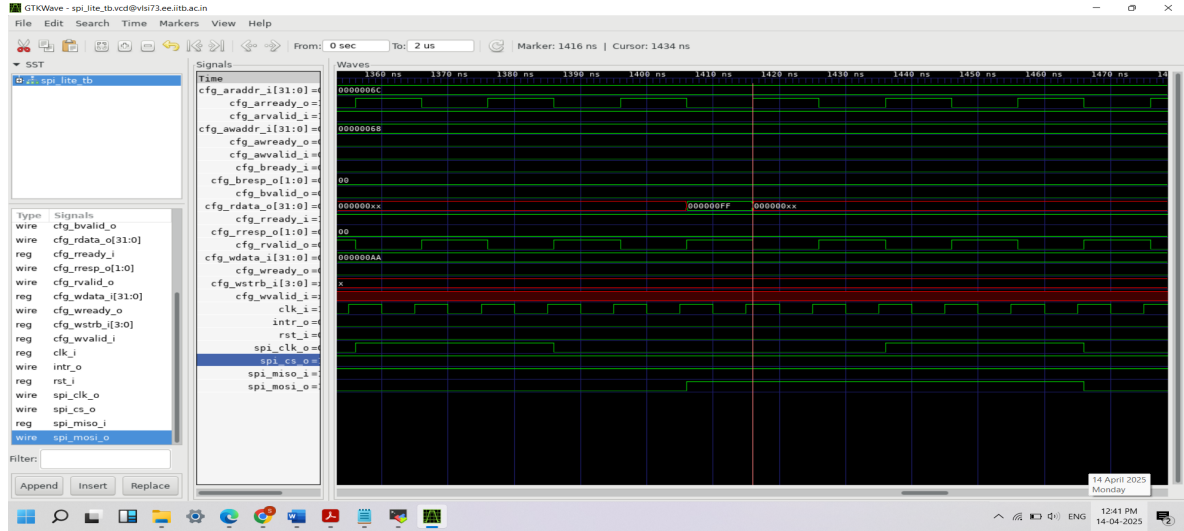


Figure 22: Post Synthesis Simulation Waveforms - I

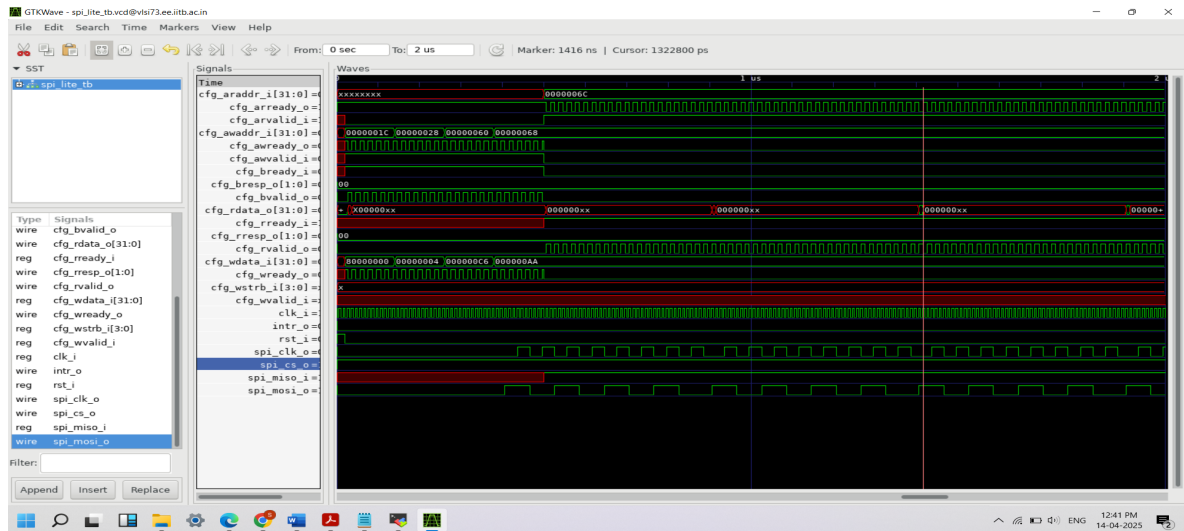


Figure 23: Post Synthesis Simulation Waveforms - II

5.3 LVS Matching Screenshot

After completing the Openlane flow successfully, we navigated to the file `39-spi_lite.lvs.rpt` which is present inside the `reports/signoff` folder. The following image shows the content of that file.

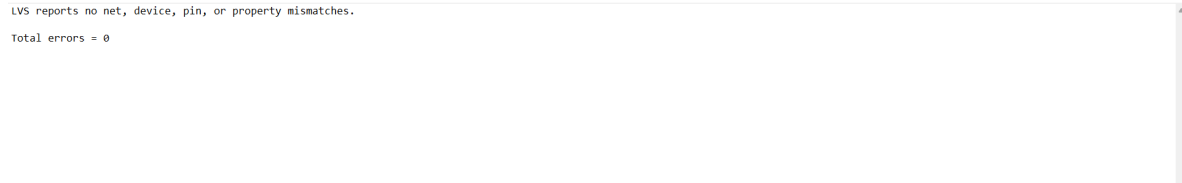


Figure 24: Screenshot of the `39-spi_lite.lvs.rpt` file

5.4 Magic Layout

The following figure shows the magic layout generated after the complete Openlane RTL to GDS Flow for the SPI module, then the number of sky_130 cells used in this design is also shown in the consecutive figure.

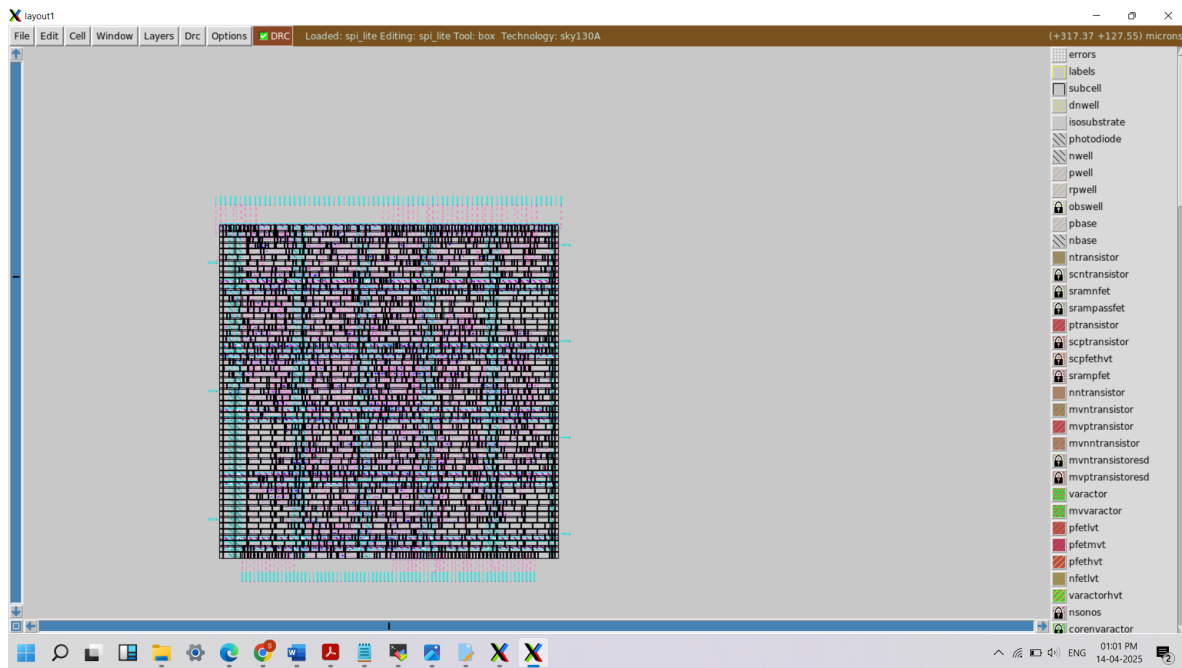


Figure 25: Magic Layout of `spi_lite`

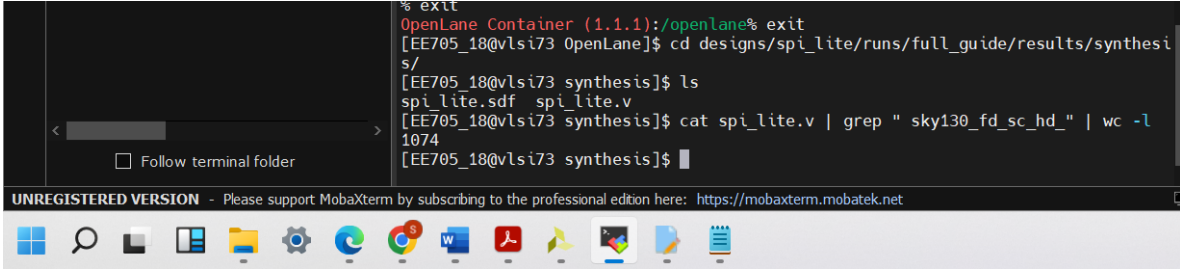


Figure 26: Number of cells used

5.5 Table of Summary

The following summary table shows the timing, area and power parameters obtained from the 24-grt_sta.log present inside logs/routing folder.

Clock Frequency (MHz)	100
Worst case setup slack (ns)	6.75
Worst case hold slack (ns)	0.46
Design Area (μm^2)	11480
Total Power Consumption (nW)	1246

6 Conclusion

In this project, we successfully carried out ASIC synthesis and physical design for the complete `riscv_soc` as well as its individual modules `uart_lite`, `spi_lite`, `alu`, and `decode` using the OpenLane flow. The individual modules are tested after synthesis, and the post synthesis waveforms are obtained and verified successfully. Each design underwent RTL-to-GDSII implementation, including synthesis, floorplanning, placement, clock tree synthesis, routing, and layout generation. All blocks achieved timing closure with successful Static Timing Analysis (STA), indicating that setup and hold constraints were satisfied throughout the designs.