2021 11th IEEE International Conference on Control System, Computing and Engineering (ICCSCE) | 978-1-6654-1281-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCSCE52189.2021.9530746

VLSI Design Course with Commercial EDA Tools to Meet Industry Demand – From Logic Synthesis to Physical Design

Siu Hong Loh*, Ing Ming Tan, Jia Jia Sim
Universiti Tunku Abdul Rahman
Kampar, Malaysia
*lohsh@utar.edu.my

***Abstract*—With the growing demand on IC design engineers globally, this requires engineering graduates, particularly in the area of electronic engineering, to be sufficiently trained and equipped with the knowledge in IC design. Institution of higher learning plays an important role to ensure that these graduates acquire not only solid fundamentals, but also adequate practical skills. This paper presents the approach that has been utilized in the VLSI design course which is implemented in Universiti Tunku Abdul Rahman (UTAR). Using commercial electronic design automation (EDA) tools, the course helps to equip students with the knowledge in industry standard VLSI design flow, with the emphasis on the process flow from logic synthesis to physical design.**

***Index Terms*—VLSI, EDA, IC design, Logic synthesis, Physical design.**

I. Introduction

In 2020, the worldwide semiconductor industry had achieved sales of $439.0 billion as reported by the Semiconductor Industry Association (SIA). In fact, the sales rose by 6.5 percent throughout the year [1]. John Neuffer, president and CEO of SIA claimed that the success in the sales is due to the escalating amount of product which strongly require the embedding of semiconductors and the developing technologies such as the Internet of Things (IoT), artificial intelligence (AI), as well as virtual reality (VR) [1]. An electronic circuit can be shrank into a single chip which is known as an integrated circuit. The construction of the essential components such as resistors, capacitors, inductors, diodes, as well as transistors on a semiconductor material like silicon makes a chip and enables it to carry out specific functions. The functions can be as simple as a logical operation or it can be complex like a general-purpose Central Processing Unit (CPU) [2]. The increasing demand of more sophisticated functions in data processing and telecommunication requires the integration of these functions to be more compact as well [3]. In Malaysia, Electrical and Electronics Productivity Nexus (EEPN) has recently initiated the Structured Industry Apprenticeship Programme (SIAP), which is an industry-initiated curriculum to train IC design talents. More than 5000 IC design engineers are required by the nation in the next five years [4].

Most of the engineering institutions are using educational and non-commercial electronic design automation (EDA) tools for their VLSI design course. The main reasons include non-commercial tools are more readily available, having user-friendly interface and relatively cost-effective. Some of these tools may provide a decent introductory insight of IC design flow. However, non-commercial EDA tools usually could not offer the practical experience with tools that the students will utilize in the real industry world. Commercial EDA tools such as those from Synopsys, Cadence and Mentor Graphics incorporate very stringent design flow to guarantee the first-time success of the fabricated chip. Additionally, graduates without any prior experience of the commercial design flow are often expected to undertake training with substantial amount of period before they can embark on the core IC design projects. Conversely, graduates who have been trained to utilize commercial EDA tools are likely to increase their employability.

To create a chip, its physical layout is required. The process to acquire that layout is known as VLSI design. The designers may fully or partially utilize the EDA software, also known as computer-aided design (CAD) tools throughout the VLSI design flow. Basically, the idea of a system is first recorded along with its specifications such as speed, power and area in a formal language, then the register-transfer level (RTL) function is written using Hardware Description Language (HDL). Furthermore, the abstract RTL description is interpreted into gate-level or logic level through logic design. Then, the optimized gate-level netlist can be implemented at transistor or circuit level through circuit design, normally the process is fully automated by the EDA. At this point, the implementation is at the lowest level of abstraction. Finally, the physical layout which concerning the chip's geometry aspect can be generated through physical design. The optimized gate-level netlist can also be converted into layout in physical domain which is known as module layout through physical design [3].The file format for the layout can be recorded as Graphical Database System (GDS) II Format. It is a binary file consisting of references of cells and geometry.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on July 01,2024 at 10:14:13 UTC from IEEE Xplore. Restrictions apply.

The organization of the paper is as follows: Section 2 discusses the Methodology of the VLSI design course. The results and discussion of using commercial EDA tools is presented in Section 3. The paper ends with a conclusion in Section 4.

II. Methodology

The course, titled VLSI Design Lab and coded UGEA4793, is a fourth-year core course in the curriculum of undergraduate program. There are several prerequisite modules for this course such as basic electronics and digital system design where students learn the fundamental of electronics and introductory digital electronics. Students have also learned some basic HDL programming before they can enroll in this course. The main objectives of this course are as follows:

- To facilitate the exposure to industry-based commercial IC design tools.
- To provide the opportunity to the students to be familiar with the IC design flow adopted in the industry.
- To foster the appreciation of teamwork in IC design project.

The course is conducted during short semester which last for only seven weeks. There are 14 hours of lectures and 56 hours of practical session. Assessment of this course include assessments for report and presentation. Since it is a full coursework-based module, no examination is available for this course. Students work in groups for this course. The topics are outlined in a weekly manner as shown in Table 1.

TABLE I. Seven Weeks Course Outline

Week	Topic
1	Logic Synthesis
2	Floorplanning
3	Placement
4	Clock Tree Synthesis
5	Routing
6	Chip Finishing
7	Physical Verification

A. Logic Synthesis

Synthesis is the implementation of hardware to produce desired output from the given input. Logic synthesis aims to meet the constraints on power, area, and speed [5]. Therefore, optimization challenge is to minimize logic gates' power dissipation, the overall design chip area, as well as delay on the critical path. Additionally, logic synthesis also aimed to produce designs with a good degree of testability for verification of the fabricated chip [5].

B. Physical Design

The design process to transform the gate-level netlist obtained from logic synthesis into geometrical layout is known as physical design. The physical layout includes information such as standard cells, clock trees, power nets, as well as placement and routing. Naturally, the files used in logic synthesis such as constraint file and standard cell library files are required for the physical design implementation. The flow of physical design can be seen in Fig. 1, adapted from a VLSI blog [6]. In physical design, the design components such as macro cells, standard cells, logic gates and transistors are instantiated with the corresponding geometrical representations. For instance, the components with fixed shapes and sizes per fabrication layer are assigned with spatial locations and their connections are completely routed in the metal layers. It is known that physical design has direct impacts on circuit's performance, reliability, power, area, as well as manufacturing yield.

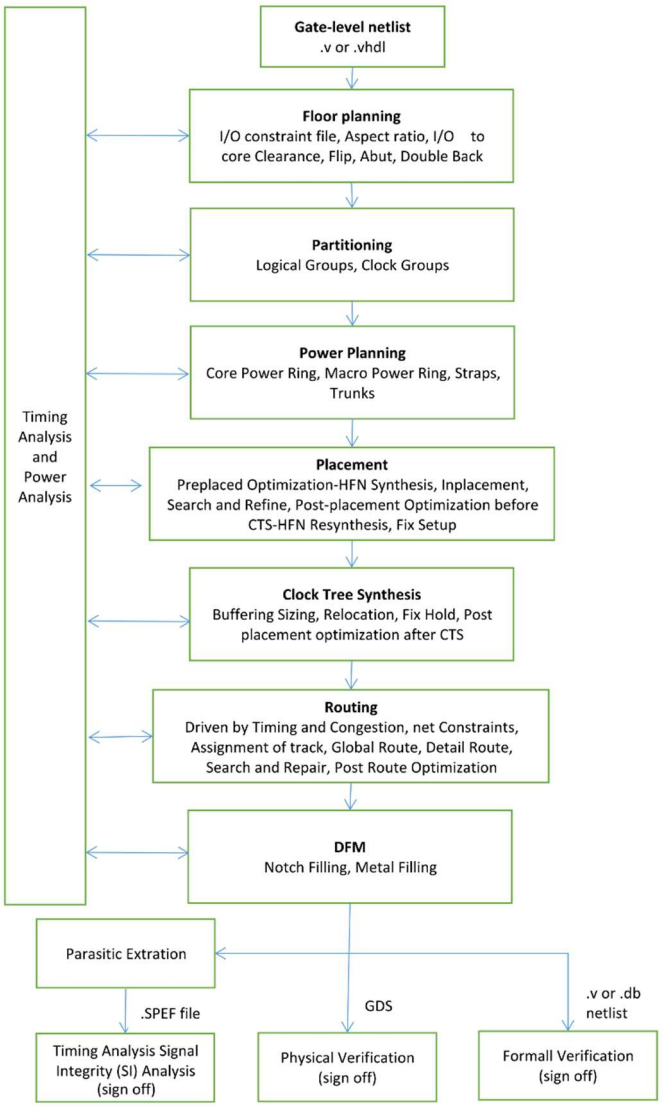

Fig 1: Physical design flow, adapted from [6].

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on July 01,2024 at 10:14:13 UTC from IEEE Xplore. Restrictions apply.

C. Floorplanning

The input for floorplanning are a hierarchy of netlist from the design containing information regarding the blocks such as the area and the interconnection, the location of pins of each of the blocks, and flexible or fixed blocks. The netlist mentioned is a description of the design in the form of HDL, generally in VHDL or Verilog format. It contains design components such as functional blocks and logic cells, macros, memory units, and registers. Furthermore, it also describes the input and output ports of the blocks as well as the relationship between the blocks. In floorplanning, every block will be assigned to a particular shape and location which will be useful during the placement stage. Also, every external connection pin will be assigned a location which allows the internal and external nets to be routed during the routing stage.

D. Placement

Generally, the placement techniques for VLSI circuits are global placement, legalization, as well as detailed placement. During global placement, objects that are movable will be linked to a general location. The objects' shapes and sizes, as well as the alignment of their assigned location will be overlooked in the process. In fact, the focus of global placement is on the distribution of overall density and the global positioning. When placement is done, the standard cells and macro cells will be assigned to a specific location on the specified core area and is ready for next stage.

E. Clock Tree Synthesis

The generation of clock, known as Clock Tree Synthesis (CTS) will be implemented after the placement stage, mainly because the exact locations of the cells are needed for the clock traversal. The traverse of clock will affect the accuracy of the delay calculation. The important constraints required for CTS are clock skew, latency, maximum time of transition, capacitance, and the fan-out of the cells [7]. CTS reduces the clock skew and delay insertion. Also, there should be improvement on the hold slack after CTS.

F. Routing

Based on the netlist obtained from previous stages, the components on the VLSI circuit is interconnected through routing. Routing aims to finish the interconnect of the instances as specified in the netlist without causing violation on the design constraints. Therefore, the routing step must minimize the chip area, the total chip wirelength, the changes in layer for interconnection, as well as the critical paths' net delay [5].

G. Chip Finishing

Before VLSI chip tapes out, there is an important preparation step known as chip finishing. One of the tasks in chip finishing is filler cell insertion. In reality, it is impossible to fill up the standard cell rows with regular cells. For the purpose of fabrication, filler cell insertion is needed to add in filler cells to fill in the empty rows between the standard cell. This process ensures the N-well and P-well continuity as well as the power buses continuity for the fabrication process. For the reason to meet the density requirements specified by the foundry, extra metals and polys fill layers need to be presented to the areas that is not routed.

H. Physical Verification

The IC design has to go through physical verification to verify its manufacturability and electrical connectivity [8]. Physical verification can be separated into three several steps, namely electrical rule check (ERC), design rule check (DRC), antenna check, layout versus schematic (LVS) check, and parasitic extraction. Before the final layout is sent for fabrication, it must be checked properly through these verification steps [5].

III. Results and Discussion

In this course, the VLSI design flow from RTL to GDSII are broken down into front-end design and back-end design. For RTL simulation, ModelSim is used. The EDA tools used for logic synthesis is Synopsys Design Compiler (DC). The EDA tools used for physical design is Synopsys IC Compiler (ICC). The details of the practical implementation in this course will be discussed in this section.

A. RTL Simulation with ModelSim

In this course, student will be working on a moderately complex design such as a 16-bit single cycle MIPS (Microprocessor without Interlocked Pipelined Stages) processor, as shown in Fig. 2 below.

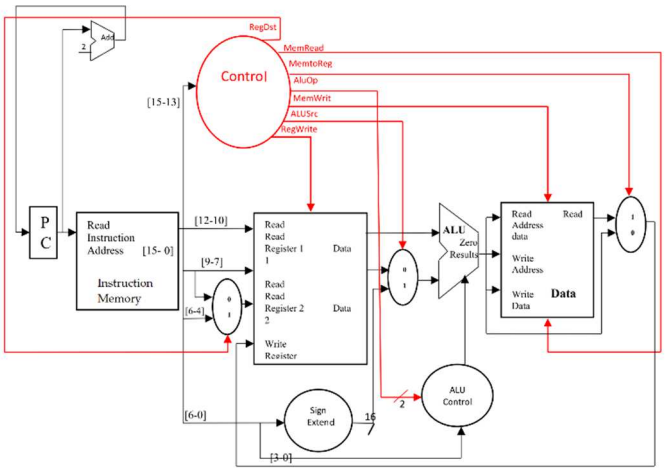

Fig 2: Block diagram of a 16-bit single cycle MIPS

In the design, it includes the program counter, instruction memory, registers, ALU, data memory, ALU control, sign extend, and a control unit. The instruction set architecture format is shown in Table II below. Two types of instruction format are used, namely R-format and I-format. Example of instructions used for R-format include addition, subtraction, operation OR and operation AND. On the other hand, instructions for I-format are load and store operations.

ModelSim is used to perform simulation of the RTL codes and generate waveforms. The simulation result of addition is shown in Fig. 3. In this example, a 16-bit instruction (0000

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on July 01,2024 at 10:14:13 UTC from IEEE Xplore. Restrictions apply.

1100 1010 0000) is read from the instruction memory. The first group of 3 bits (000) is the opcode which specifies the type of instruction. The following two groups (011 = $3) and (001 = $1) specify the source registers and the fourth group (010 = $2) specify the destination register. The last group (0000) specifies the operation of addition to be performed. The current values in the source registers of $3 and $1 are 0000 0000 0000 0011 and 0000 0000 0000 0001 respectively. After performing addition operation, the result 0000 0000 0000 0100 is stored in the destination register $2.

TABLE II. INSTRUCTION SET ARCHITECTURE

Name	Fields				
Field Size	3 bits	3 bits	3bits	3bits	4 bits
R-format	Op	Rs	Rt	Rd	Funct
I-format	Op	Rs	Rt	Address/ immediate	

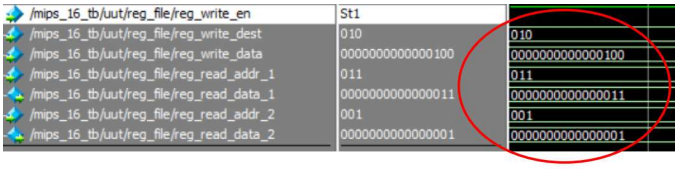

Fig 3: Simulation result of addition

The simulation results of operation subtraction, operation AND and operation OR are shown in Fig. 4, Fig. 5 and Fig. 6, respectively.

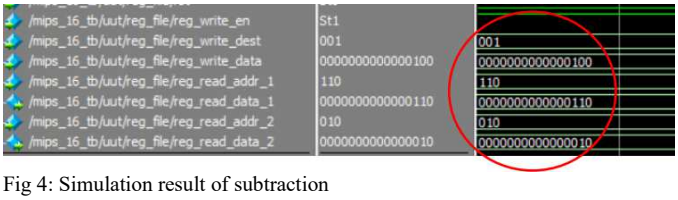

Fig 4: Simulation result of subtraction

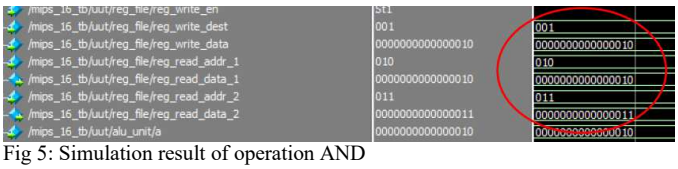

Fig 5: Simulation result of operation AND

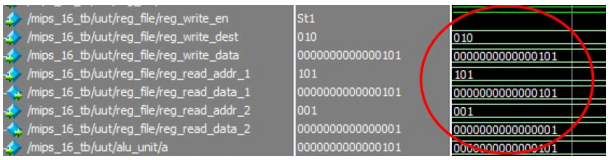

Fig 6: Simulation result of operation OR

When the functional correctness of the design is verified, student will proceed to the next step, which is the logic synthesis step.

A. Logic Synthesis with Synopsys Design Compiler

In the DC environment, the command 'read_file -format -rtl {$file_list}' is used to read multiple design files into the memory of DC. The RTL design files are written in verilog HDL. Then, 'analyze -format verilog {$file_list}' is used to check the design for errors. Next, 'elaborate' translates the design into a technology-independent design (GTECH) from the intermediate files produced during analysis. Also, the design references are resolved by performing 'link' command. If there's any unresolved references in the netlist, it means the design data for the module is not detected and cannot be used in processing. To resolve the reference, it has to be made sure that the design name can be found in the DC memory, or that the library cell used is from the link library.

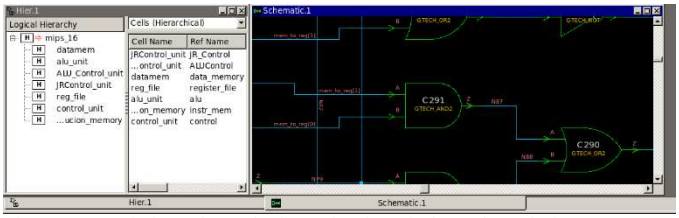

Fig 7: Part of the schematic in technology-independent design (GTECH) format.

The next step is to set up design constraints. Design constraints can be separated into design rule check (DRC) and design optimization constraints. DRCs are included in the technology libraries used. On the other hand, design optimization constraints are the user-specified target timing and area for DC to achieve. In facts, DRCs have higher priority and DC will never try to violate them in order to achieve timing or area goals. Next, the mapping and optimization of the design is performed using *compile* command. The output of this command is a gate-level implementation or netlist. During mapping process, DC identify the logic of the design and assign the suitable logic cells available from the technology library to those logic. After mapping, there might be some design violations and therefore optimization was required. The optimization process honoured the DRCs and optimization constraints provided by the designer and it attempted to create an optimized gate-level netlist in many iterations. By default, DC attempted to create the smallest possible design while meeting the timing specification.

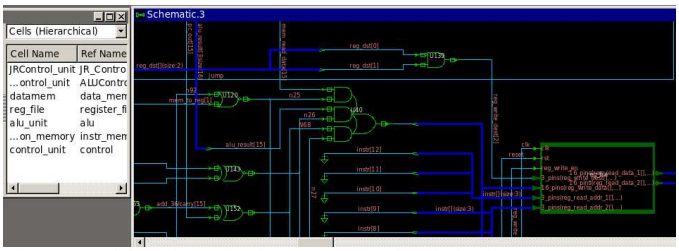

Fig 8: Part of the schematic after mapping and optimization.

Fig. 9 depicts the portion of synthesized gate-level netlist after the design is constrained, mapped, and optimized.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on July 01,2024 at 10:14:13 UTC from IEEE Xplore. Restrictions apply.

```
module control ( opcode, reset, reg_dst, mem_to_reg, alu_op, jump, branch,
       mem_read, mem_write, alu_src, reg_write, sign_or_zero );
  input [2:0] opcode;
  output [1:0] reg_dst;
  output [1:0] mem_to_reg;
  output [1:0] alu_op;
  input reset;
  output jump, branch, mem_read, mem_write, alu_src, reg_write, sign_or_zero;
  wire   \mem_to_reg[0] , \alu_op[1] , n7, n8, n9, n10, n11, n12, n13, n1, n2,
         \reg_dst[1] , n4, n6;
  assign mem_read = \mem_to_reg[0] ;
  assign mem_to_reg[0] = \mem_to_reg[0] ;
  assign alu_src = \alu_op[1] ;
  assign alu_op[1] = \alu_op[1] ;
  assign mem_to_reg[1] = \reg_dst[1] ;
  assign reg_dst[1] = \reg_dst[1] ;

  or03d1 U4 ( .A1(opcode[2]), .A2(reset), .A3(opcode[1]), .Z(n8) );
  an02d1 U5 ( .A1(n2), .A2(n7), .Z(reg_dst[0]) );
  an02d1 U6 ( .A1(n6), .A2(n11), .Z(jump) );
  nd04d0 U14 ( .A1(n8), .A2(n9), .A3(n1), .A4(n10), .ZN(reg_write) );
  nd03d0 U15 ( .A1(opcode[0]), .A2(opcode[1]), .A3(alu_op[0]), .ZN(n9) );
  nr03d0 U16 ( .A1(opcode[2]), .A2(reset), .A3(n11), .ZN(n7) );
  nr03d0 U17 ( .A1(n12), .A2(opcode[1]), .A3(n2), .ZN(mem_write) );
```

Fig 9: Portion of synthesized gate-level netlist after the design was constrained, mapped, and optimized.

DC will refer to the technology library to calculate the delay through each of the points as shown in the 'Incr' column in the timing report. The Static Timing Analysis (STA) algorithm is used to generate timing report, as shown in Fig. 10.

```
Startpoint: pc_current_reg[1]
            (rising edge-triggered flip-flop clocked by clk)
Endpoint: pc_current_reg[15]
          (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: max

Des/Clust/Port     Wire Load Model       Library
------------------------------------------------
mips_16            70000                 cb13fs120_tsmc_max

Point                                    Incr       Path
-----------------------------------------------------------
clock clk (rise edge)                    0.00       0.00
clock network delay (ideal)              0.00       0.00
pc_current_reg[1]/CP (dfcrq1)            0.00 #     0.00 r
pc_current_reg[1]/Q (dfcrq1)             0.36       0.36 r
U177/Z (an02d0)                          0.18       0.53 r
U175/Z (an02d0)                          0.19       0.72 r
U173/Z (an02d0)                          0.19       0.90 r
U171/Z (an02d0)                          0.19       1.09 r
U169/Z (an02d0)                          0.19       1.28 r
U167/Z (an02d0)                          0.19       1.46 r
U165/Z (an02d0)                          0.19       1.65 r
U163/Z (an02d0)                          0.19       1.83 r
U161/Z (an02d0)                          0.19       2.02 r
U159/Z (an02d0)                          0.19       2.20 r
U157/Z (an02d0)                          0.19       2.39 r
U155/Z (an02d0)                          0.19       2.57 r
U153/Z (an02d0)                          0.15       2.72 r
U152/Z (xr02d1)                          0.28       3.00 f
U103/ZN (aoi22d1)                        0.29       3.29 r
U125/ZN (nd02d1)                         0.06       3.35 f
pc_current_reg[15]/D (dfcrq1)            0.00       3.35 f
data arrival time                                   3.35

clock clk (rise edge)                    5.00       5.00
clock network delay (ideal)              0.00       5.00
pc_current_reg[15]/CP (dfcrq1)           0.00       5.00 r
library setup time                      -0.07       4.93
data required time                                  4.93
-----------------------------------------------------------
data required time                                  4.93
data arrival time                                  -3.35
-----------------------------------------------------------
slack (MET)                                         1.58
```

Fig 10: Timing report generated with STA algorithm in DC

With all the delays, the data arrival time can be calculated and used for setup timing analysis. DC will also calculate the data required time for the signal to safely travel to the endpoint. In this case, the data required time is the clock period minus the clock network delay, clock uncertainty period, and the library setup time. Since the data arrival time did not exceed the data required time, it is said to have a positive slack. In case of negative slack, there will be a setup timing violation as the arriving signal might cause unexpected and undesired behaviours. Generally, hold time analysis is not the concern during synthesis process as it will normally appear only after CTS.

B. Physical Design with Synopsys IC Compiler

The verilog netlist generated after logic synthesis is used in physical design using ICC. After the initial floor plan was generated, virtual flat placement was quickly done to check the congestion and timing. It is good to make sure there is no congestion and timing issue during floor planning so that at the later stage the design can be cleaner. Generally, the maximum IR drop across the core area should not be exceeding 10 % of the supply voltage. Before proceeding to the placement stage, the design was checked again to make sure there is no violation caused by the design planning stage. An actual placement of standard cells is then performed after checking. The placement process can also fix congestion issues.

Prior to performing CTS, the clock target skew must be set to certain realistic value and the clock uncertainty has to be set as well to account for clock jitter. Basically, the hold-time violation after CTS is fixed by adding buffers which also slightly increased the design area. To minimize the design area, an area optimization was specified during an incremental clock optimization. This feature will perform down-sizing of the buffers which impacts the timing paths negatively. To prevent timing degradation during routing stage causing setup violations, a timing margin of 5 % of the clock period is normally adopted. Fig. 11 shows a sample of the layout at the core area after placement, clock tree synthesis and routing.

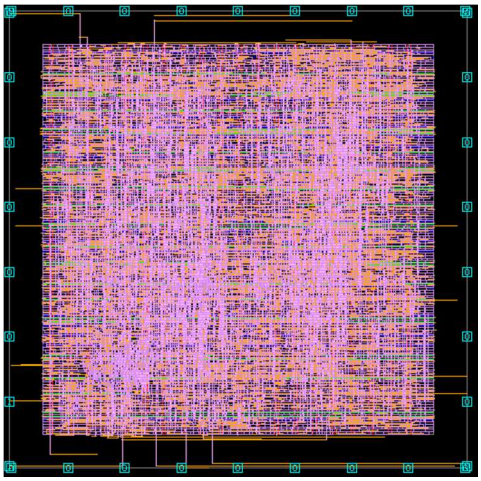

Fig 11: Layout generated after placement, CTS and routing

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on July 01,2024 at 10:14:13 UTC from IEEE Xplore. Restrictions apply.

After performing routing, the chip finishing task involves standard filler cells insertion in the core area to ensure P/G continuity and continuity on n-wells and p-wells in each row of standard cells. Lastly, metal fill is inserted so that the design meet metal density rules. This process is done last because it used up most of the remaining routing resources. In the final stage, the quality of the finished design was checked to make sure there is no degradation. Specifically, the design is checked so that it does not violate any design constraints (area, timing and power). Finally, the finished design has to pass ERC, DRC, antenna check, LVS before streaming out to GDSII file format. Fig. 12 depicts the report generated for QoR (Quality of Results). As can be seen in the report, the total design area is 49259.8 μm^2. Combinational cell count and sequential cell count are 2901 and 4240, respectively. Total dynamic power is about 12 mW.

```
Cell Count
----------------------------------
Hierarchical Cell Count:          8
Hierarchical Port Count:        339
Leaf Cell Count:               7141
Buf/Inv Cell Count:             373
Buf Cell Count:                 301
Inv Cell Count:                  72
CT Buf/Inv Cell Count:           71
Combinational Cell Count:      2901
Sequential Cell Count:         4240
Macro Count:                      0
----------------------------------


Area
----------------------------------
Combinational Area:     7749.500000
Noncombinational Area: 30837.000000
Buf/Inv Area:            532.000000
Total Buffer Area:           477.25
Total Inverter Area:          54.75
Macro/Black Box Area:      0.000000
Net Area:              10673.329438
Net XLength        :      270495.47
Net YLength        :      440865.44
----------------------------------
Cell Area:             38586.500000
Design Area:           49259.829438
Net Length         :      711360.88
```

Fig 12: QoR report

IV. Conclusion

In summary, students who have enrolled in this course were given the opportunity to experience the VLSI design flow adopted in the IC design industry by using commercial EDA tools. Front-end design included the technology mapping and optimization of the RTL to produce a gate-level netlist in verilog format. The verilog netlist was then used for back-end design to produce the physical layout in GDSII format. Back-end design included floor planning, placement, clock tree synthesis, routing, and chip finishing. The design and verification were performed by the EDA tools. The design automation focused on optimizing the area, power, and timing of the design while honouring the design rules constraints. At every stage of design, the quality of the result, including timing, power consumption, and area were monitored. In the future, this course will be conducted in the long semester so that more contents could be incorporated into the syllabus.

References

[1] Semiconductor Industry Association (SIA). "Global Semiconductor Sales Increase 6.5% to $439 billion in 2020." semiconductors.org. https://www.semiconductors.org/global-semiconductor-sales-increase-6-5-to-439-billion-in-2020/ (accessed April 19, 2021).

[2] L. Xiu, *VLSI Circuit Design Methodology Demystified: A Conceptual Taxonomy.* Hoboken, New Jersey: John Wiley & Sons, 2007.

[3] S. M. Kang, Y. Leblebici, C. Kim, *CMOS Digital Integrated Circuits: Analysis & Design.* 4th Ed., Singapore: McGraw-Hill, 2015.

[4] Malaysian Investment Development Authority (MIDA). "Readying the E&E Industry for Growth." mida.gov.my https://www.mida.gov.my/mida-news/readying-the-ee-industry-for-growth/ (accessed April 20, 2021).

[5] D. Das, *VLSI Design.* New Delhi: Oxford University Press, 2010.

[6] VLSI Physical Design. "Physical Design Flow." vlsijunction.com http://www.vlsijunction.com/2015/08/physical-design-flow.html (accessed April 21, 2021).

[7] P. V. Vishnu, N. Kotha and A. R. Priyarenjini, "Clock Tree Synthesis Techniques for Optimal Power and Timing Convergence in SoC Partitions," 2019 4th International Conference on Recent Trends on Electronics Information Communication & Technology (RTEICT), Bangalore, India, May 2019, pp. 276-280.

[8] R. M. Anand, S. Ravula, M. A. Kalpashree and S. Satavisa, "Automated Physical Verification of I/O Pads in Full-Custom Environment," Fifth International Symposium on Electronic System Design, Surathkal, India, Dec 2014, pp. 203-205.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on July 01,2024 at 10:14:13 UTC from IEEE Xplore. Restrictions apply.