A Review on ASIC Synthesis Flow Employing Two Industry Standard Tools

Neha Deshpande
Department of Electronics and Communication
R.V. College of Engineering
Bengaluru, Karnataka, India – 560059

Sowmya K B
Department of Electronics and Communication
R.V. College of Engineering
Bengaluru, Karnataka, India – 560059

***Abstract* – The advancement in technologies and electronic design automation tools has made it possible to design ASIC chips and analyze its parameters extensively. Design Compiler by Synopsys and Genus Synthesis Solution by Cadence are some of the EDA tools which are used for running synthesis flow for various blocks of hardware SoC design. This paper provides an insight to brief survey of the various techniques involved in synthesis flow using Design Compiler tool and Genus Synthesis Solution tool. The techniques used in running the synthesis flow are designed such that parameters - power, area and timing are optimized and results in enhancing the overall performance. Therefore, the brief literature survey provides a comparison among the various proposed work in terms of optimized values of power, area and performance (timing) and the results provide an overview of the parameters evaluation using both the tools.**

Keywords – Application Specific Integrated Circuit, Register Transfer Logic, Electronic Design Automation, Design Compiler, Synopsys, Genus, Cadence, Synthesis, Optimization, Power, Area.

I. INTRODUCTION

With the advent of technologies, it has become possible to design hardware for any block/module with the aid of ASIC design flow. Application Specific Integrated Circuit is a chip designed mainly for a specific application like for a protocol, satellite, voice recorder etc. ASIC design flow includes a series of steps - chip design specification, architectural design, behavioral and functional modelling, logical implementation, synthesis and testing, floor planning and design layout. In order to meet the user requirements regarding design of chip, changes are done accordingly in design tools, procedures and software/hardware abilities. All the stages of ASIC design flow make use of electronic design automation tools that enables to implement the design and the necessary changes in the design through an efficient manner.

Logical Synthesis is one of the stages of ASIC flow. Synthesis is defined as a process that involves three steps – translation, mapping and optimization. With the help of these three steps, synthesis converts RTL code (Verilog or VHDL) into gate level netlist. The conversion from RTL code into design with logic gates is done with the help of synthesis tools. The synthesis tools available are Design Compiler by Synopsys and Genus Synthesis Solution by Cadence. The main aim of employing these tools is to enhance the productivity during conversion of RTL code and to provide highest quality of reports at the end of implementation. Power, performance and area are three important parameters to be taken into consideration at every stage of ASIC flow. These parameters play a key role in defining the way tools can be used to obtain the best results. Obtaining optimal PPA along with trade-offs has led to improving the functionality of EDA tools. Therefore, EDA tools – Design Compiler and Genus Synthesis Solution play a crucial role in optimizing power, performance and area during synthesis process.

II. SYNTHESIS FLOW

ASIC design flow is used for designing and implementation of architectural designs with the help of RTL code. Fig. 1 shows the various steps involved in ASIC flow, thus the implementation of any design starts by providing its specifications and is completed with netlist handoff. Among all the steps, logic synthesis is a methodology which transforms RTL (HDL or VHDL) code into a netlist describing the hardware (logic gates and the wires connecting them). Standard cell library consisting of basic logic gates like AND, OR, and NOR, or macro cells like adder, multiplexers, memory, and flip-flops are used in logic synthesis. Standard cells libraries put together are called technology library which are used for optimizing the design. Main objectives of Synthesis are – to minimize area, to minimize power and maximize performance. Synopsys's Design Compiler tool performs synthesis of a design by carrying out the following steps - analysing & elaborating the RTL code, applying design constraints to the design, compiling and optimizing the design followed by inspection of results (output files).

Fig. 2 and Fig. 3 show the steps involved in performing synthesis using Synopsys Design Compiler tool and Genus Synthesis Solution tool respectively. Firstly, libraries and designs are loaded, design constraints are applied, and the design is synthesized. The generated output files are analyzed and mainly parameters like power, area, timing paths and quality of reports are extracted.

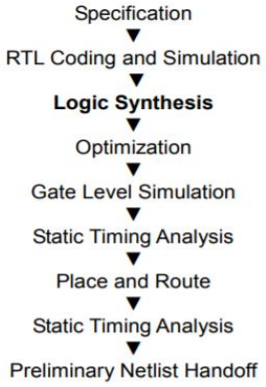

Fig. 1. ASIC Design Flow [6]

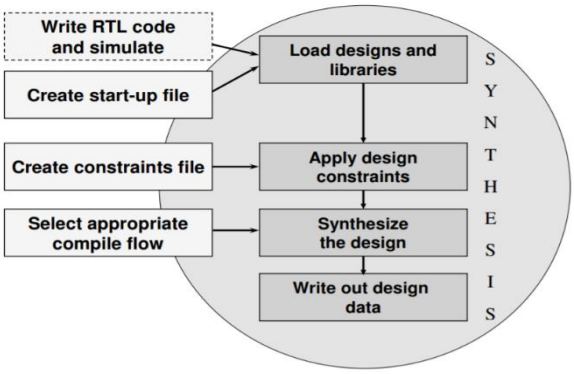

Fig. 2. Steps involved in synthesis flow using Design Compiler tool by Synopsys [1]

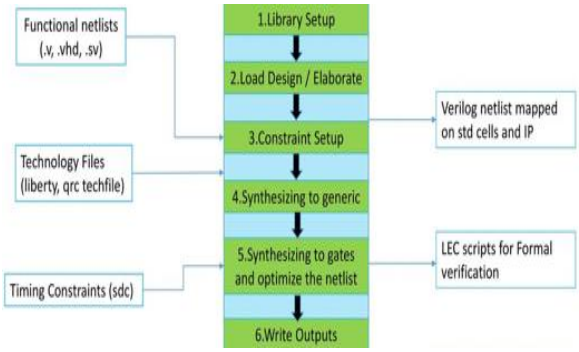

Fig. 3. Steps involved in synthesis flow using Genus Synthesis Solution tool by Cadence [3]

III. LITERATURE SURVEY

The different types of methodologies carried out in synthesis flow with the help of Design Compiler tool by Synopsys and Genus Synthesis Solution tool by Cadence is available in the literature. The main aim of all the proposed work is to improve the efficiency of synthesis flow and to reduce the three important parameters – power, timing and area significantly.

A comparative study of area, power and timing results of the case study which performs scan insertion synthesis and synthesis with and without constraints is presented. With suitable inbuilt optimization and mapping techniques, Design Compiler performs synthesis and optimization and Conformal EC tool performs Logic Equivalence Check (LEC) on the case study [1].

A technique for enhancing area, power and performance by applying an ungrouping method in synthesis stage of Design Compiler tool and an optimization method in floorplan, placement and clock tree synthesis of IC Compiler tool is proposed. The comparative analysis of results of concurrent clock and data optimization and conventional clock tree synthesis implies that with concurrent clock and data optimization technique area, power (static and dynamic), worst negative slack and total negative slack is improved. Therefore, the Synopsys tools work efficiently in improving the overall behavior of the design [2].

In paper [3] Genus Synthesis tool by Cadence in synthesis process, Synopsys IC Compiler tool in placement and routing and Tempus tool in sign-off static timing analysis is employed. VSDFLOW is tested on designs like picorv32 - a RISC-V CPU core which makes use of 180nm PDK's and 45nm PDK's from OSU and Nangate respectively and comparison drawn based on the results of different combination of tools by running VSDFLOW multiple times shows that with each design, the area is optimized.

A synthesis technique using Synopsys Design Compiler involving a combination of PTL and CMOS logic cells, which can be inserted into standard cell-based design flow is proposed . The experiments based on UMC 90-nm technology conveyed that PTL surpasses CMOS in terms of parameters like power consumption, area, and area–delay–power product while hybrid CMOS/PTL yields better results in area-optimization and delay optimization-based synthesis flow. Therefore, PTL and hybrid CMOS/PTL can be employed in applications having delay, power and area as critical parameters in the design [4].

A VHDL code-based synthesis flow for FPGA design employing Xilinx PlanAhead tool is proposed. With this tool, it is easier to perform synthesis of RTL code and helps in obtaining better results. High performance, good flexibility and code processing

speed and the dynamic resources placement of modules are few advantages offered by PlanAhead tools which makes them convenient to be used for carrying out synthesis flow for any application [5].

Paper [6] focuses on power reduction in scan synthesis stage of automatic test pattern generator. The aim of reducing cell internal power, net switching power and total dynamic power by 50% is achieved by applying low power techniques to phase locked loop circuit with the aid of Synopsys Design Compiler tool. Another important inference that can be drawn is that scan clock frequency reduction has impact only on power and not on IC's functionality and performance. Thus, Synopsys DC tool plays a crucial in design for testability stage.

In paper [7] two techniques towards efficient VLSI circuits are presented. Firstly, a group of graph-based algorithms were proposed to come up with good QoR with the use of simple cells like inverters and either 2- input NAND/NOR or 2-input XOR/XNOR. The results showed that parameters like power, delay and transistor count was optimized. Secondly, a pattern-based algorithm was proposed to design xor-and-inverter graphs using C+11 coding language. The proposed approaches resulted in 8-10% reduction in area, 5-13% reduction in power and 16-19% reduction in performance in comparison to reference tools.

A null convention logic based on Quasi-delay-insensitive designs is proposed. The gates required to implement NCL design are 35% - 43% less as compared to conventional design. By changing the gate behaviour, null convention logic can be varied to generate NCL+, INCL, INCL+ and INV. The results of all the logics are compared in terms of total gates, total area, total leakage power and total dynamic power. Along with optimization techniques being included in these logics, it becomes possible to improve the parameters such as area and power (static and dynamic) [8].

A synthesis methodology keeping in consideration the outcomes obtained post place and route and static timing analysis is presented. Physical synthesis is introduced in the flow where in adequate physical information is available in earlier stages which allows to improve timing, power post signoff process and provides less violations in the initial iteration. Thus, physical synthesis is an approach to mitigate the errors, provide optimal timings results pre and post placement and routing and help in reducing total runtime from synthesis to signoff as violations were reduced in the beginning iterations [9].

Paper [10] puts forward the method of Boolean matrix factorization involving approximate logic. The main of the proposed technique is to break down large circuits to smaller blocks which makes it easier to perform Boolean matrix factorization on them. The results conveyed that with significant trade-off, it is possible to accommodate all the QoR metrics in factorization. Therefore, with the introduction of this technique, new ways to improve the efficiency of synthesis has been obtained.

Vedic mathematics in the design of 24-bit floating point multiplier has been introduced in this proposed work. The multiplier design is simulated using Xilinx ISE 12.1 software and synthesized using CADENCE Genus synthesis solution tool. Comparison is drawn between critical parameters - area, power and timing using Xilinx software and Cadence tool and it was observed that better results were gained with the aid of Cadence tool. So, these multipliers are better compared to conventional ones as the main benefit is less area and power consumption and enhanced performance [11].

A functional-flow parallel programming language Pythagoras based VLSI synthesis procedure is proposed. This supports in conversion of architectural description into RTL code and enables to execute verification, testing debugging and optimization with the aim of achieving reliability and quality with reduced timing. As a result, this concept is recommended for carrying out synthesis on system components for control systems and spacecrafts [12].

A HEVC based 8 point 2-D Discrete Cosine Transform stage which is fabricated of adders that are efficient in terms of power and energy is designed in this proposed work. The architecture was designed using Verilog HDL and synthesized using Cadence Genus Synthesis Solution tool. Two important observations derived from the synthesized results of proposed design were reduction of 14.76% and 13.99% in power consumption and area respectively when compared to design made up of standard adders. Therefore, these results imply that discrete cosine transform designed with proposed method can be used effectively in image and signal processing applications [13].

In paper [14], synthesized designed elliptic curve point multiplication method using FPGA and ASIC technology is presented. Comparative study is carried out by synthesizing the VHDL code on Virtex 6, Virtex 7 and using 65 nm United Microelectronics standard cell library of Synopsys DC tool. It was observed that delay is less in ASIC technology than that obtained using Virtex 6 and Virtex 7. Hence, ASIC based flow is a more appropriate method for implementation than FPGA as parameters can be easily optimized to the required optimal values.

A low power design flow based on UPF as UPF is a format which provides all the details regarding different low power techniques being implemented in the flow is proposed. The proposed flow implements clock gating, multi voltage, power gating methods

using Synopsys tool involving two libraries – 90nm and 32/28nm on a multiplier design. Thus, UPF plays an important role in the synthesis flow as it determines power which is vital parameter for any design [15].

IV. METHODOLOGY

In all the real time applications, synthesis flow plays a vital step in providing various details about a block/module such as – clocking paths being used, timing related information like hold time, setup time, worst negative slack, total negative slack, static and dynamic power consumption with respect to combinational and sequential instances in the design, information regarding number of ports, nets, cells(combinational and sequential), macros present in the design and area occupied by each of them and most importantly it generates the optimized RTL netlist which is used in chip fabrication.

The outline of the steps carried out in synthesis flow is as shown in Fig. 4. It starts with collecting all the necessary input files – verilog file consisting of RTL code, library list(technology library, symbol library, DesignWare library), constraints file and script file for the tool(.tcl file). The further steps involved are compiling library files to generate db files(binary format), performing analyze to read RTL code and to check for syntax errors. Elaborate step is mapping the RTL code to GTECH (technology independent) library cells. In compile and optimization step, optimization is performed at architectural, logic-level and gate level. Gate level optimization marks the end of the synthesis flow. Lastly, output files like log files, check files, reports (area, power, port, quality of report) and optimized netlist (.v format) is generated.

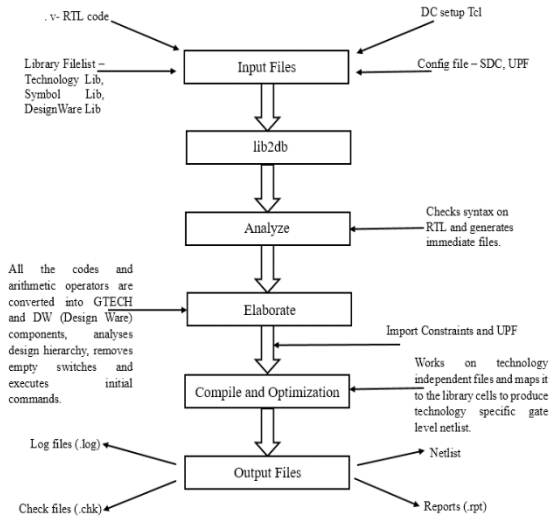

Fig. 4. Detailed description of the steps followed in synthesis flow

V. RESULTS

In the proposed work, focus is on performing synthesis flow on a block which can be modelled as hardware accelerator using Design Compiler tool and Genus tool and to compare the Quality of Results as shown in Fig. 5 and Fig. 6 which are templates of QoR obtained using Design Compiler and Genus respectively. In general, on the basis of comparative analysis of Quality of Results of both the tools, it is observed that Design Compiler tool is efficient in terms of area and power consumption. So, if Design Compiler tool can achieve good timing performance then it is preferred over Genus as it yields better results for area and power parameters when compared to Genus.

```
****************************************
Report : qor
Design : TEST_TOP
Version: E-2010.12
Date   : Sun Oct 24 02:02:17 2010
****************************************

Timing Path Group 'clk'
-----------------------------------
Levels of Logic:                1.00
Critical Path Length:           0.02
Critical Path Slack:           -0.68
Critical Path Clk Period:       8.00
Total Negative Slack:        -393.61
No. of Violating Paths:       1066.00
Worst Hold Violation:           0.00
Total Hold Violation:           0.00
No. of Hold Violations:         0.00

----------------------------------------
Cell Count
----------------------------------------
Hierarchical Cell Count:          7
Hierarchical Port Count:       2360
Leaf Cell Count:              43978
Buf/Inv Cell Count:            6764
CT Buf/Inv Cell Count:            4
Combinational Cell Count:     37212
Sequential Cell Count:         6773
Macro Count:                      0
----------------------------------------
Area
----------------------------------------
Combinational Area:  562404.432061
Noncombinational Area:
                    6720840.351067
Net Area:             0.000000
Net XLength        :  2836623.25
Net YLength        :  2555007.75
----------------------------------------
Cell Area:           7283244.783128
Design Area:         7283244.783128
Net Length         :  5391631.00

Design Rules
----------------------------------------
Total Number of Nets:          47207
Nets With Violations:              5
Max Trans Violations:              3
----------------------------------------

Hostname: machine
Compile CPU Statistics
----------------------------------------
Resource Sharing:              21.54
Logic Optimization:           182.63
Mapping Optimization:         230.79
----------------------------------------
Overall Compile Time:         631.32
Overall Compile Wall Clock Time:  288.11
```

Fig. 5. Quality of Result Report for Design Compiler Tool [16]

```
Timing
--------
        Clock          Period
-----------------------------------
CLK_GROUP_1            7000.0
CLK_GROUP_2            7000.0
CLK_GROUP_3            1839.0
CLK_GROUP_4            3696.0
CLK_GROUP_5            7000.0
CLK_GROUP_6           20000.0
CLK_GROUP_7            7000.0
CLK_GROUP_8            7000.0
CLK_GROUP_9            7000.0
CLK_GROUP_10           6000.0

Instance Count
--------------
Leaf Instance Count                1084119
Physical Instance count                  0
Sequential Instance Count           132339
Combinational Instance Count        928506
Hierarchical Instance Count           9881
Area
----
Cell Area                        172042.059
Physical Cell Area                    0.000
Total Cell Area (Cell+Physical)  172042.059
Net Area                              0.000
Total Area (Cell+Physical+Net)   172042.059

Power
-----
Leakage Power                    37903.649 nW
Dynamic Power                 44824046.315 nW
Total Power                   44861949.96 nW

Number of Clock Gating Logic          6929

Cell Count
----------
Hierarchical Cell Count:          10167
Hierarchical Port Count:        1416293
Leaf Cell Count:                1084191
Buf/Inv Cell Count:              128798
Buf Cell Count:                   11405
Inv Cell Count:                  114476
CT Buf/Inv Cell Count:                0
Combinational Cell Count:       9639315
Sequential Cell Count:           132619
Macro Count:                        798
Physical-only Cell Count:             0

Area
----
Combinational Area:          58230.499
Noncombinational Area:       43463.357
Buf/Inv Area:                 3767.352
Total Buffer Area:             669.293
Total Inverter Area:          4436.645
Macro/Black Box Area:        70445.601
Physical Cell Area:              0.000
Net Area:                        0.000
Cell Area:                  172244.069
Design Area:                172244.069

Net Count
---------
Total Number of Nets:         2048976
```

Fig. 6. Quality of Result Report for Design Compiler Tool [17]

VI. CONCLUSION

The outcome of the proposed work has diverse applications in the field of VLSI and SoC design and the main objective is to inculcate various techniques in the synthesis flow methodology so that there is a wide scope of improvement in the results obtained which in turn plays a crucial role while performing activities such as design for testability, static timing analysis, formal verification etc. From the literature survey it is evident that in all the proposed work, the stages of synthesis flow are modified in such a way that designed techniques are being utilized to the maximum extent to produce efficient results. Therefore, from Quality of Results of the both the tools it is conveyed that the flow aims at PPA exploration, which means minimizing power, maximizing performance and minimizing area which are of upmost importance in synthesis flow and ASIC design.

REFERENCES

[1] S. Gayathri and T. C. Taranath, "RTL synthesis of case study using design compiler," *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, Mysuru, 2017, pp. 1-7.

[2] S. Iyengar and L. Shrinivasan, "Power, Performance and Area Optimization of I/O Design," *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, 2018, pp. 415-420.

[3] K. P. Ghosh and A. K. Ghosh, "Technology mediated tutorial on RISC-V CPU core implementation and sign-off using revolutionary EDA management system (EMS) — VSDFLOW," *2018 China Semiconductor Technology International Conference (CSTIC)*, Shanghai, 2018, pp. 1-3.

[4] S. Hsiao, M. Tsai and C. Wen, "Low Area/Power Synthesis Using Hybrid Pass Transistor/CMOS Logic Cells in Standard Cell-Based Design Environment," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, no. 1, pp. 21-25, Jan. 2010.

[5] M. A. L. Sarker and M. H. Lee, "Synthesis of VHDL code for FPGA design flow using Xilinx PlanAhead tool," *International Conference on Education and e-Learning Innovations*, Sousse, 2012, pp. 1-5.

[6] A. Yadlapati and H.K. Kakarla, "Low-power design-for-test implementation on phase-locked loop design," Measurement and Control, June 2019.

[7] J. M. Matos, J. Carrabina and A. Reis, "Efficiently Mapping VLSI Circuits With Simple Cells," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 4, pp. 692-704, April 2019.

[8] M. T. Moreira, P. A. Beerel, M. L. L. Sartori and N. L. V. Calazans, "NCL Synthesis With Conventional EDA Tools: Technology Mapping and Optimization," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 6, pp. 1981-1993, June 2018.

[9] L.E.Geralla, M.J.D. Guzman, and J.A. Hora, "Optimization of Physically Aware Synthesis for Digital Implementation Flow," International Journal of Engineering and Technology, pp. 31-35, April 2018.

[10] S. Hashemi, H. Tann and S. Reda, "BLASYS: Approximate Logic Synthesis Using Boolean Matrix Factorization," *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, San Francisco, CA, 2018, pp. 1-6.

[11] S. Javeed and S.S. Patil, "Low Power High Speed 24 Bit Floating Point Vedic Multiplier using Cadence," International Research Journal of Engineering and Technology (IRJET), vol. 05, no. 07, July 2018.

[12] O.V. Nepomnyashchiy, I.V. Ryjenko, V.V. Shaydurov, N.Y. Sirotinina, A.I. Postnikov, "The VLSI High-Level Synthesis for Building Onboard Spacecraft Control Systems," in Proceedings of the Scientific-Practical Conference "Research and Development - 2016".

[13] L. F. Sequeira *et al*., "Low-Power HEVC 8-point 2-D Discrete Cosine Transform Hardware Using Adder Compressors," *2018 16th IEEE International New Circuits and Systems Conference (NEWCAS)*, Montreal, QC, 2018, pp. 309-312.

[14] M.S. Hossain, E. Saeedi and Y. Kong, "Parallel point-multiplication architecture using combined group operations for high-speed cryptographic applications" PLoS ONE 12(5), May 2017.

[15] V. Gourisetty *et al*., "Low power design flow based on Unified Power Format and Synopsys tool chain," *2013 3rd Interdisciplinary Engineering Design Education Conference*, Santa Clara, CA, 2013, pp. 28-31.

[16] Synopsys.(2011)., Design compiler user guide.

[17] Cadence.(2019)., Genus user guide.