Lightweighted Shallow CTS Techniques for Checking Clock Tree Synthesizable Paths and Optimizing Clock Tree in RTL Design Time

Nayoung Kwon[1] and Daejin Park[1*]
[1]School of Electronic and Electrical Engineering, Kyungpook National University
[*]Correspondence to: boltanut@knu.ac.kr

***Abstract*—The microcontroller unit (MCU) are mainly used in low power devices, which use limited energy sources such as batteries, energy harvesting, and wireless communications. Therefore, reducing the operating power of the MCU is important to improve energy efficiency by extending battery life or minimizing energy consumption. The MCU is one of the chip designs composed of digital integrated circuits. The clock signal is important element to the MCU. The clock tree, which consists of the clock signal, is directly related to MCU low power operation and performance improvement. Also, chip verification process is important role to improve performance of overall system and reliability to the MCU. However, as the degree of integation of chips, the chip verification process increases complexity and time-comsumption to process many data. Currently, many users dependent on licensed electronic design automation (EDA) tools to ensure high accuracy, minimizing errors in circuit design and improving reliability. The use of licensed EDA tool puts a burden on users including high costs, limited license, difficulty in customization, slow speed, etc. An effective approach to avoid problems by using licensed EDA tools proceeds verification that is unrestricted license and customization for possible using only a register transfer level (RTL) source. In this paper, we propose to predict roughly pre-estimated CTS results using an RTL source in which temporary logic using random buffer insertion is placed before the route process. This paper contributes to reducing MCU operating power and hardware area by performing optimized CTS and minimizing resources according to the RTL structure to be designed.**

***Keywords*—micro controller unit (MCU), low power, clock tree synthesis (CTS), placement and route (P&R), shallow CTS, register transfer level (RTL), licensed EDA tool, synthesizable**

I. Introduction

The microcontroller unit (MCU) is used to low power devices, which uses restricted resources, for example, battery, energy harvesting, wireless communication, etc [1], [2]. It is important to reduce operating power consumption to improve energy efficiency and system stability on the MCU. The MCU, which is one of integrated circuits (IC) goes through the process of the chip verification process. Chip verification process is more important to application specific integrated circuit (ASIC). If the verification process inappropriately performs, it increases error probability and cost and time-consumption to iteratively verify and revise chip defect. As the degree of integration of chips, there has been a demand for an increase in verification speed and verification accuracy. Many people depend on various electronic design automation (EDA) tools such as Synopsys and Cadence, but we need affordable license limitations or license server costs. Also, the processing speed of EDA tools can be slow because licensed EDA tools are very complex and have to deal with large circuits. Using a licensed EDA tool to verify chips accurately requires a lot of expense, circuit design time, and verification time for the user.

We conduct research focusing on clock tree synthesis (CTS) of placement and route (P&R) process, which is time-consuming process during the verification process. The CTS is process of optimizing the clock tree so that the clock signal is passed correctly to all circuits. Also, the CTS can optimize power consumption of chips. It can optimize switching speed of the circuit and minimize unnecessary power consumption to the transmission path of the clock signal by optimizing the transmission path of the clock signal. The CTS is optimizing operation of digital circuit, increasing trasmission stability of the clock signal and minimizing power consumption. If it constructs efficient clock tree through CTS, the overall system of MCU can improve performance and reliablity, distributing the clock signal efficiently on the MCU [3], [4]. The MCU have various clock operation modes and timing constraints, and clock signals are accurately distributed through CTS to ensure stable MCU operation. The CTS can also help optimize the MCU's power consumption. In addition, CTS can also contribute to improving the performance of MCUs. When the clock signal arrives exactly at the desired timing, the operating performance of the MCU improves, enabling stable operation even at high clock frequencies.

The previous works on CTS have tried to preconstruct a lightweight clock tree based on RTL source for minimizing clock skew. It has some restrictions, which does not apply on general Verilog netlist sources on previous works. And, it performed some of CTS to solve hold violation from input to D flip-flop data path because it only considered data path from input to D flip-flop data path. Also, it does not verify accurately overall CTS result using previous works.

We research CTS pre-estimation accesible general larger Verilog netlist case using overall clock tree, which includes all clock path from clock port to D flip-flop clock sink. Fig.

This study was supported by the BK21 FOUR project (4199990113966), the Basic Science Research Program (NRF-2018R1A6A1A03025109), (NRF-2022R1I1A3069260) through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, and (2020M3H2A1078119) by Ministry of Science and ICT. This work was partly supported by an Institute of Information and communications Technology Planning and Evaluation (IITP) grant funded by the Korean government (MSIT) (No. 2021-0-00944, Metamorphic approach of unstructured validation/verification for analyzing binary code, 10%) and (No. 2022-0-01170, PIM Semiconductor Design Research Center, 20%) and (No. RS-2023-00228970, Development of Flexible SW-HW Conjunctive Solution for On-edge Self-supervised Learning, 20%) and (No. RS-2022-00156389, Innovative Human Resource Development for Local Intellectualization support program, 50%). The EDA tool was supported by the IC Design Education Center (IDEC), Korea.

979-8-3503-1327-7/23/$31.00 ©2023 IEEE

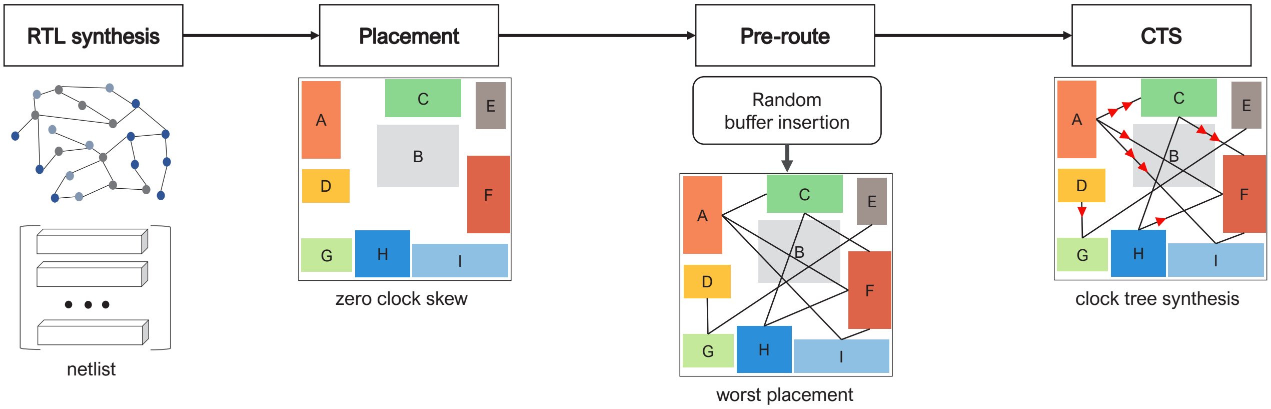

Fig. 1. Research approaches

1 is described that it conducts CTS pre-estimation using an RTL synthesis Verilog netlist without timing and parasitic information on the P&R process. Currently, it cannot require the CTS result with the EDA tool by using only RTL synthesis Verilog netlsit. To approximate the CTS results of the EDA tool, we assume in this study that the logic is arbitrarily placed after going through the placement process before the route process described as pre-route. Also, it assumes that the result of placement is zero clock skew state and worst clock skew occurs by using random buffer insertion algorithm, which is placing temporary logic on pre-route. We propose lightweighted CTS optimizing worst placement result, which has worst clock skew in this research.

We introduce prior studies relatied this study in Section II. We explain a method for constructing the CTS process between pre-route and route processes in Section III. Our approach involves using an open source Parser-Verilog tool to parse the RTL source and construct a general tree structure from the parsed components. We discuss how to route the clock path and set the margin for clock path delay, as well as how to perform random buffer insertion in the pre-route process and preconstruct clock tree prediction algorithms. In Section IV, we provide detailed analysis based on a specific RTL source using a TSMC 180 nm standard cell library. We evaluate the effectiveness of our heuristics algorithm for closing clock tree preconstruction, and observe a reduction in clock skew and standard deviation before and after CTS pre-estimation. We compare the maximum clock frequency achieved by our shallow CTS approach with that of an open-source EDA tool after CTS. Finally, in Section V, we summarize this research and suggest directions for future research.

II. Previous work

This research aims to the highly expected, CTS process of place and route (P&R), which is a time-consuming process in chip verification. The CTS involves optimizing the performance, power, and area (PPA) of a chip by efficiently placing and wiring its elements, particularly focusing on fast and accurate clock signal transmission across the chip [5]. To reduce the overall power consumption of the chip, a well-designed clock tree implementation is crucial, often achieved through the utilization of clock distribution algorithms. However, the CTS process is known to be time-consuming because of the need for enhancing circuit stability and performance.

Prior research has primarily focused on CTS of the data path, specifically from input to D flip-flop's data path, with limitation given to overall CTS considerations [6]–[8]. Moreover, prior studies have been limited to specific RTL sources, typically involving Verilog sources with one output and multiple inputs. In contrast, the current research aims to overcome these limitations by addressing Verilog sources with multiple outputs and inputs, which represent general cases. Furthermore, while previous studies have successfully eliminated hold violations from input to D flip-flop's data path, the hold violations specifically pertaining to the clock path have not yet been resolved.

Summarizingly, this research aims to develop an efficient and lightweight clock tree synthesis methodology based on an RTL source, eliminating the need for a licensed EDA tool. The primary objectives are to reduce both cost and time associated with the CTS process, considering general cases with multiple outputs and inputs and addressing hold violations in clock path.

III. Proposed Method

The proposed lightweight clock tree implementation process is depicted in Fig. 2. We provide a detailed explanation of how to pre-estimate the results of clock tree synthesis from a Verilog RTL file. The entire process can be broadly divided into three stages: RTL synthesis, pre-route, and shallow CTS. To accomplish this, we utilize the open RTL synthesis tool Yosys to determine if the RTL source is synthesizable. For the pre-route and shallow CTS processes, we use Parser-Verilog [9]–[11].

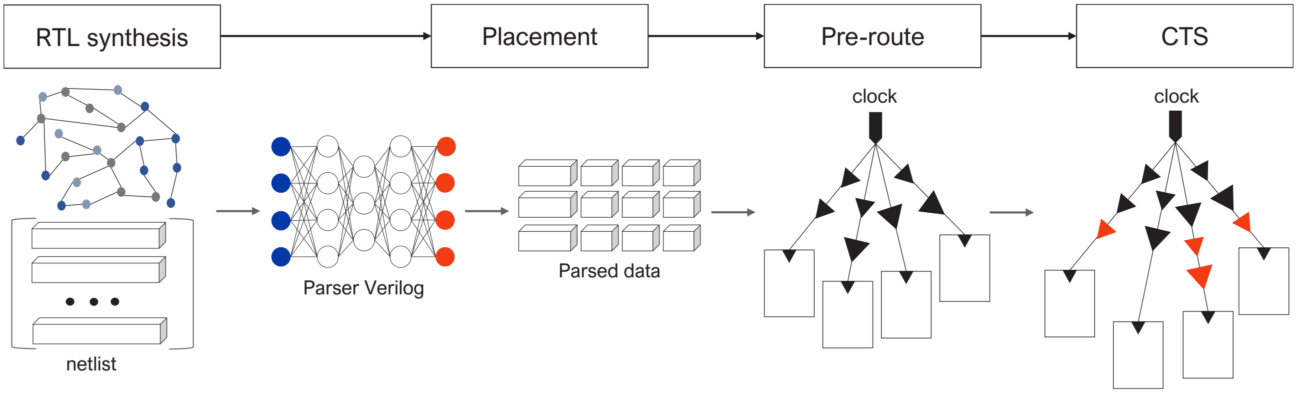

Fig. 2. proposed CTS flow

The input to the process is an RTL synthesizable Verilog file that describes the design at a behavioral level. Typically, the CTS process in EDA tools proceeds with an optimized gate-level representation, incorporating parasitic and timing information. However, in this study, we perform the shallow CTS process at the primitive gate level to decide whether to synthesize a clock tree during RTL time, without considering data such as timing and parasitic information.

To achieve this, the original Verilog source is transformed into an RTL synthesizable form using Yosys. It is then matched with the appropriate standard cell library for the desired fabrication process and modified to meet various constraints, such as safety, timing, and load.

Initially, the clock signal exists in an ideal mode that is not present in the physical clock distribution during RTL design, synthesis, and placement [12]. In this ideal mode, the clock signal is assumed to reach the clock pins of all D flip-flops simultaneously. The original Verilog source is converted to an RTL synthesizable source using Yosys. It matches the standard cell library for a suitable fabrication process and modifies it to meet various constraints such as safety, timing problems, and load. The pre-route process involves arbitrary logic placement before the actual routing process. After pre-route process, the clock is physically connected, resulting in clock skew due to individual logic placements [13]. It is important to note that this study does not consider timing constraints or parasitic components such as R, C, cell strength. Furthermore, it assumes that the clock path experiences the worst clock skew as a result of random placements after the pre-route process. The random placement process is implemented through random buffer insertion. This research focuses on developing a lightweight clock tree synthesis methodology using an RTL source, and the process involves steps such as RTL synthesis, pre-route, and shallow CTS.

Random buffer insertion is essential to expressing the state that the physical clock is connected, and clock skew occurs in the placement process before the route process. If random placement does not execute, the clock skew of clock paths does not generate because this study assumes that it calculates the path's static delay concerning the standard library cell's delay. Fig. 3 represents the difference between the clock tree after RTL synthesis and the clock tree after the pre-route process. It is simply connected to the clock buffer and clock sink of the D flip-flop in Fig. 3 (a). There is no clock skew because all clock paths have the same clock path delay after RTL synthesis. On the other side, it represents the existence of clock skew by inserting a random buffer after the pre-route process in Fig. 3 (b). Even so, random buffer insertion is not inserted randomly but focused on the number of loads connected to the clock buffer. If there are many loads connected to the clock buffer, the clock skew will be reduced and the overall chip area will be larger by placing clock buffers of greater strength. If placing clock buffers of greater strength, there is a possibility of an enlarged chip area. To prevent this problem, load balancing should be carried out by considering the R and C components of the front and rear stages, but only standard library cell delay is assumed except timing constraints: parasitic components like R, C, and cell strength, etc. Therefore, random buffer insertion is performed by simply determining the number of

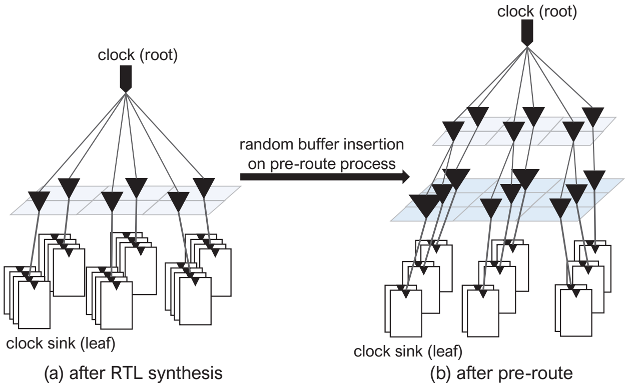

Fig. 3. random buffer insertion

loads.

The shallow CTS process is used to place arbitrary logic and carry out random buffer insertion that generates clock skew through the pre-route process. The pre-route process indicates when the physical clock is connected to all D flip-flops' clock sinks. When the physical clock signal is applied, it arrives at the clock pin at different times. The clock skew created by clock uncertainty is corrected through the CTS process. The CTS process progresses to adjust the clock skew that is generated after physical clock connection. The CTS recognizes the clock signal from the clock source pin and delivers the clock signal to thousands of D flip-flops. The CTS is used to form a buffer tree to match the skew of clock nets and high fanout nets and to meet design rules such as maximum capacitance, maximum transition time, etc. The shallow CTS process is performed in Parser-Verilog using the RTL synthesizable netlist. In this process, the buffer insertion algorithm is applied to reach time closure in the clock path.

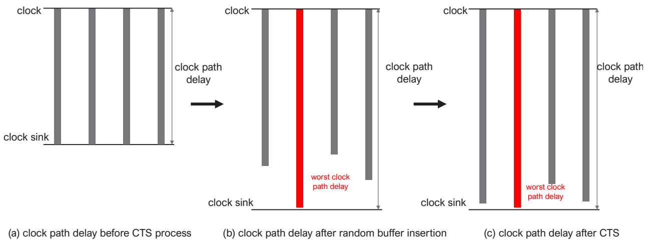

Fig. 4. Comparison CTS

The overall process synthesizes a clock tree through buffer insertion based on the worst clock skew of the clock path using an RTL synthesizable source. The variation of clock skew on the pre-route and CTS processes is illustrated in Fig. 4. Logic placement corrects the worst clock skew through random buffer insertion on the pre-route process in Fig. 4 (b). After the pre-route process, it is difficult to solve PPA and reach the time closure because the worst clock skew occurs in the clock paths. The CTS inserts buffers based on the worst clock path to solve overall chip timing and PPA problems in Fig. 4 (c). It can show reduced clock skew after the CTS process compared to the pre-route process.

IV. Experiments

The experiments conducted in this research aimed to evaluate the effectiveness of the proposed CTS algorithm in reducing clock skew using different RTL sources. The RTL sources used in the experiments included a 4-bit divider, map9v3 with an 8-bit linear feedback shift register, and picoRV32, which implements a CPU core using RISC-V instructions. The experiments involved measuring the number of inserted buffers and standard deviation during the pre-route and CTS processes. The maximum clock frequency was also computed to compare the CTS results with those obtained using Qflow, an open-source EDA tool. The experiments were performed using a TSMC 180 nm standard cell library.

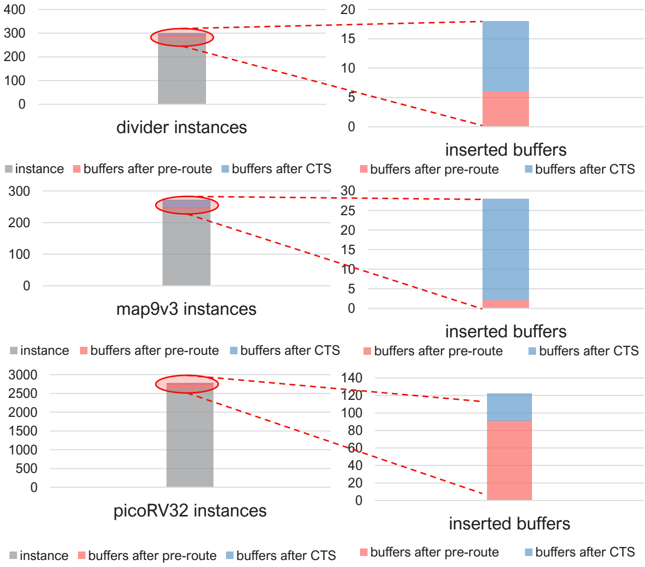

Fig. 5. Comparision to inserted buffers on pre-route and shallow CTS process.

It is important how much inserted buffers account for overall chip area and power consumption. We examined the percentage of inserted buffers and the number of total instances in Fig. 5 and Table. I. The percentage of inserted buffers and the number of total instances were analyzed to assess their impact on the overall chip area and power consumption. The results showed that the divider had approximately 6.36% of inserted buffers, map9v3 had around 11.52%, and picoRV32 had about 1.59%.

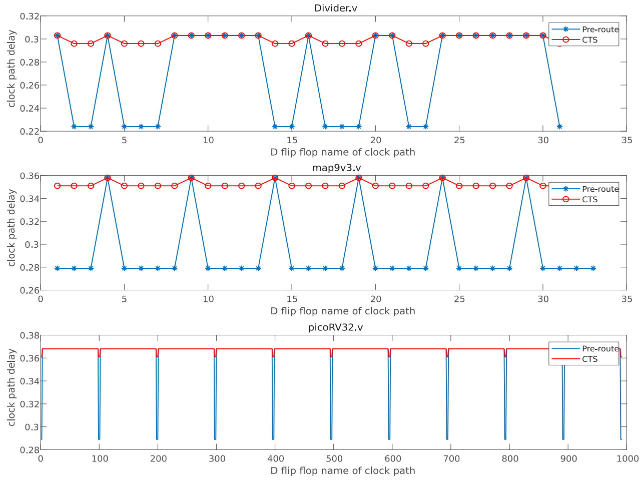

Fig. 6. Comparision to standard deviation on pre-route and shallow CTS process.

We measured clock path delay and standard deviation on pre-route and CTS to evaluate the skew of each clock skew,

RTL source	Instance number on RTL synthesis	inserted buffers after pre-route	inserted buffers after shallow-CTS	standard deviation after pre-route	standard deviation after shallow-CTS	clock frequency after pre-route	clock frequency after CTS
divider	265	91	31	0.0396 ns	0.0035 ns	367.904 MHz	403.154 MHz
map9v3	215	2	26	0.0309 ns	0.0027 ns	645.218 MHz	813.366 MHz
picoRV32	7534	6	12	0.0140 ns	0.0012 ns	974.056 MHz	1153.65 MHz

TABLE I
COMPARISION NUMBERS OF INSERTED BUFFER ON PRE-ROUTE AND SHALLOW-CTS PROCESS.

as in Fig. 6 and Table I. In Fig. 6, the x axis represents the path from the clock to a D flip-flop, such as the clock path from the clock to D flip-flop1, and the y axis represents the standard deviation of the clock path delay. Table I calculates the standard deviation on the pre-route and the shallow CTS process. Clock path delay and standard deviation were measured during the pre-route and CTS processes to evaluate the clock skew. The results demonstrated that after the shallow CTS process, the standard deviation of the clock skew was reduced compared to the pre-route process. Clock path delay and standard deviation were measured during the pre-route and CTS processes to evaluate the clock skew. The results demonstrated that after the shallow CTS process, the standard deviation of the clock skew was reduced compared to the pre-route process. The divider had a standard deviation of approximately 0.0361 ns, map9v3 had about 0.0282 ns, and picoRV32 had around 0.0128 ns.

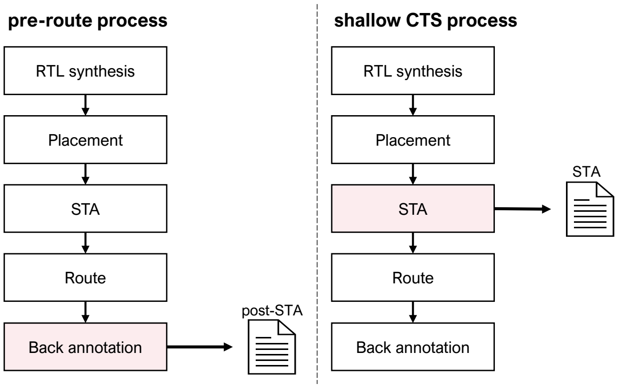

Fig. 7. Comparison STA result of Qflow on pre-route and shallow CTS process.

We analyzed the static timing analysis(STA) result of Qflow in order to proceed with a quantitative comparison. Fig. 7 represents how to obtain the maximum clock frequency on the pre-route and the shallow CTS process using the STA result of Qflow. The open EDA tool Qflow executes the CTS on the route process. We aimed to acquire a collinear comparison based on the STA that has undergone CTS once to evaluate the performance of the proposed CTS algorithm. The netlist of the pre-route process proceeded to the back-annotation process to acquire a post-STA result that reports the final CTS results after Qflow's CTS. It determined the maximum clock frequency of post-STA on the pre-route process. Further, the netlist of the shallow CTS process proceeded to the STA because it already progresses using the proposed shallow CTS algorithm.

We analyzed the maximum clock frequency after pre-route and shallow CTS because the clock frequency is an important factor in chip performance: the results are shown in Table I. In addition, most RTL sources tend to slow clock frequency after the CTS process. The frequency of the pre-route process measured in the post-STA is the result of Qflow's own CTS after the pre-route process. The frequency of the shallow CTS result measured in STA is the result after the shallow CTS process. The clock frequency is the measured fast clock frequency after the shallow CTS process.

V. CONCLUSION

The results of experiments show that the proposed shallow CTS algorithm is efficient for preconstructing a clock tree and checking whether the clock tree is synthesizable. The standard deviation of the shallow CTS not only obviously reduced the standard deviation of the pre-route, but it also measured a higher maximum clock frequency than the maximum clock frequency of Qflow post-STA, which is over after the CTS. It verified that the performance of the results approaches zero clock skew through the shallow CTS algorithm, which focused on clock skew, and the maximum clock frequency of the shallow CTS process is higher than the maximum clock frequency of Qflow. The experiments provided insights into the effectiveness of the proposed CTS algorithm in reducing clock skew, analyzing the impact on chip performance through maximum clock frequency measurements, and comparing the results with an open-source EDA tool. It clearly proves weak points using a robust clock path standard deviation calculation.

It can be concluded that performing CTS pre-estimation using a heuristic algorithm can serve as a cost-effective alternative to using licensed EDA tools. The use of licensed EDA tools often incurs high costs and significant time investments to check the CTS results. Furthermore, with RTL design, it is not possible to determine synthesizability before synthesis, leading to increased design costs and time-consuming error detection. By utilizing shallow CTS before using licensed EDA tools, users can reduce the expected cost and gain economic benefits. Shallow CTS provides valuable information such as approximate timing results and area additions, allowing for iterative modifications of the RTL source to align with chip verification requirements. We conducted in this study focuses on CTS during the chip verification process, specifically targeting the pre-route and CTS processes. The CTS is a crucial step in optimizing the clock tree to ensure correct transmission of the clock signal and minimize power consumption in integrated circuits, particularly in MCUs used in low-power devices. By optimizing the transmission path of the clock signal, the CTS can improve the performance, reliability, and power efficiency of the overall MCU system. The research highlights the potential of using a heuristic algorithm for CTS pre-estimation as a cost-effective and efficient approach. It enables users to obtain crucial information and make iterative design modifications before resorting to licensed EDA tools. The users can reduce expected costs, improve chip verification, and achieve economic benefits by using shallow CTS.

The implementation of the CTS pre-estimation approach in this research has certain limitations that should be addressed for further improvement. One limitation of our implementation is that it is not a precise delay calculation. Qflow, which is an open EDA tool, makes elaborate delay adjustments by using Elmore delay calculation and considers strength, parasitic elements, and network delay of the front and rear ends. To enhance the accuracy of the delay calculation in the CTS pre-estimation, it would be necessary to consider these factors and incorporate more advanced algorithms. Another limitation of this research involves the issue of reduced performance on the shallow CTS process. It takes a long time to execute the route and buffer insertion algorithm because it retrieves all paths. Addressing these limitations would contribute to enhancing the accuracy of the CTS pre-estimation and improving the overall performance and efficiency of the proposed approach.

Future research should be devoted to the development of precise delay calculation and overall performance improvement using graph neural network and machine learning. In addition, future research should apply standard cell libraries of various fabrications.

References

[1] Z. Zhao, Y. Zhou, H. Zhu, Q. Sun, and D. W. Zhang, "Gate oxide and implantation process co-optimization for low-power mcu applications," *IEEE Journal of the Electron Devices Society*, vol. 9, pp. 1055–1059, 2021.

[2] M. Lueders, B. Eversmann, J. Gerber, K. Huber, R. Kuhn, M. Zwerg, D. Schmitt-Landsiedel, and R. Brederlow, "Architectural and circuit design techniques for power management of ultra-low-power mcu systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 11, pp. 2287–2296, 2014.

[3] Y.-C. Lu, J. Lee, A. Agnesina, K. Samadi, and S. K. Lim, "Gan-cts: A generative adversarial framework for clock tree prediction and optimization," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019, pp. 1–8.

[4] A. B. Chong, "Hybrid multisource clock tree synthesis," in *2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, 2021, pp. 1–6.

[5] P. Bhaskara and P. Bharadwaja, "A robust cts algorithm using the h-tree to minimize local skews of higher frequency targets of the soc designs," in *2020 7th International Conference on Smart Structures and Systems (ICSSS)*, 2020, pp. 1–5.

[6] N. Kwon and D. Park, "Lightweight buffer insertion for clock tree synthesis visualization," in *2022 International Conference on Electronics, Information, and Communication (ICEIC)*, 2022, pp. 1–3.

[7] ——, "Lightweighted shallow cts techniques for checking clock tree synthesizable paths in rtl design time," in *2022 19th International SoC Design Conference (ISOCC)*, 2022, pp. 394–395.

[8] ——, "Lightweighted CTS Preconstruction Techniques for Checking Clock Tree Synthesizable Paths in RTL Design Time (KCI)," *Journal of the Korea Institute of Information and Communication Engineering*, vol. 26, no. 10, pp. 1537–1544, 2022.

[9] [Online]. Available: https://github.com/YosysHQ/yosys

[10] [Online]. Available: https://github.com/OpenTimer/Parser-Verilog

[11] S. Nn, Y. Vahvale, N. Praveena, and A. Mamatha, "Design and implementation of 64-bit sram and cam on cadence and open-source environment," *International Journal of Circuits, Systems and Signal Processing*, vol. 15, pp. 586–594, 07 2021.

[12] T. Na, J. H. Ko, and S. Mukhopadhyay, "Clock data compensation aware clock tree synthesis in digital circuits with adaptive clock generation," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, 2017, pp. 1504–1509.

[13] M. Lin, H. Sun, and S. Kimura, "Power-efficient and slew-aware three dimensional gated clock tree synthesis," in *2016 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, 2016, pp. 1–6.