Library characterizer for open-source VLSI design

Shinichi Nishizawa* and Toru Nakura†
*Graduate School of Information, Production and System, Waseda University *nishizawa@aoni.waseda.jp*
†Department of Electronics Engineering and Computer Science, Fukuoka University, Fukuoka, Japan

***Abstract*—This paper introduce an open source cell library characterizer. Free and open-sourced silicon design communities are attracted by hobby designers, academies and industries, and these open-sourced silicon designs are supported by free and open sourced EDAs. However, in our knowledge, tool-chain lacks cell library characterizer. This paper introduce our on-going open source cell library characterizer which can generate timing models and power models of standard cell library.**

I. Introduction

Recently, several open-sourced VLSI design projects have been proposed [1], [2] to encouledge many peoples to join open-source VLSI design. Many open EDAs are widely proposed and EDAs enable these open-source VLSI design projects. Many open-sourced EDAs are proposed to support system design to logic design, logic synthesis, place-and-route, test insertion, verification, and summarized as several RTL-to-GDS flows [3]. However, to our knowledge, tool-chain lacks library characterizer to extract timing and power of cell library to enable precise timing and power estimation using Static Timing Analysis (STA). Without this timing information, designed circuit cannot ensure its feasibility of operation speed and operation correctness. To make designers to add own combinational or sequential cells into his/her circuit, it is very useful to use the characterizer to extract the timing and power information.

This paper introduce an open and free timing characterizer, named **libretto** [4], to support accurate timing analysis in digital circuit design flow. **libretto** uses free spice simulator ngspice [5] for timing and power simulation, and create timing and power library as Synopsys Liberty format [6]. Key extension of this paper is introduce recovery/removal support for Flip-Flops with asyncronous set and reset.

The rest of this paper is organized as follows. Section II describes related works on this field. Section III describes the overview of our characterizing flow. Section IV describes the details of timing and power characterization. Section V describes the experimental results. Section VI concludes this paper.

II. Related Works

Several works are available for non-commercial timing and power characterizing of standard cell library.

pharsoc [7] is a characterizer and standard cell libraries provided by *The Art of Standard Cell Library Design*. **pharsoc** can characterize both of combinational and sequential cells, and generate timing information as Liberty format. Disadvantage is that only the binary of the program is distributed. It looks that the individually different binaries are used for different types of logic, and only one type of Flip-Flop without set/reset is supported for characterization. Also, **pharsoc** uses Winspice3 as simulation engine but it is a commercial simulator.

AutoLibGen [8] is an open sourced characterizer which extract information as Liberty format. Advantage is that it can express timing information as Composite Current Source (CCS) model to enable accurate timing estimation compared to conventional Non-Linear Delay Model (NLDM). Disadvantage is that it has no support for sequential cells and gate with multiple outputs, and simulator is Synopsys Hspice.

LiChEn [9] is the another open sourced characterizer. **LiChEn** has been proposed to characterize asynchronous standard cells but it also characterizes basic combinational cells for VLSI design. Program code written in C/C++ is opened by author so anyone can use and extend this work. Disadvantage is that it uses Cadence Spector as a simulation engine, and lacks the ability of the sequential cell characterization.

ASCLIC [10] [11] is a characterizer, which is developed to use characterizer in free. **ASCLIC** target to generate NLDM model and Non-Linear Power Model (NPTM) to enable both timing simulation and power simulation, and authors claim **ASCLIC** support multi-core system efficiently. Disadvantage is that it does not support sequential cell characterization.

This work, named **libretto**, is a free and open-sourced characterizer. **libretto** is implemented using Python3, and ngspice is selected for simulation engine, assumes to running on Linux system. It supports to characterize both combinational and sequential cells. Advantage is the wide support of the functions for characterization, especially for sequential cells with positive edge or negative edge clock, set and reset support.

III. Overview of characterization

Figure 1 shows the internal flow of proposed characterizer. It needs spice netlist include transistors model and command file which contains characterize settings for library and individual logic cells.

In first stage, library common settings are defined. These common settings are common parameters for whole library, such as library name, cell name prefix suffix (if needed), units and definition of voltage, current, capacitance, resistance, power.

In second stage, characterized conditions are defined. These are also common settings for whole library, such as PVT (process, voltage, temperature) conditions, logic threshold for delay definitions.

Third stage branches depend on the target cell function (combinational or sequential). In this stage, individual settings for each logic cell are defined. Each cell needs spice subcircuit name, logic function type, input/output pin name, input slopes, output loadings, simulation time step, function written in Verilog-HDL. Additionally, sequential cell requires following information to characterize, such that clock pin name, set/reset pin name (if it has), name of storage elements.

In fourth stage, spice files of target cell are generated and invoke spice simulation. This simulation stage composed of two stage. Firstly, timing simulation stage is performed to extract propagation delay, transition delay, start point of input waveform and end point of output waveform. Secondly, power simulation stage is performed to extract leakage power and dynamic power. Combinational cell needs this simulation for all of the input patterns with different input slew and output loading. Sequential cell needs much larger simulation runs to find the setup time and hold time (detail is described in the next section).

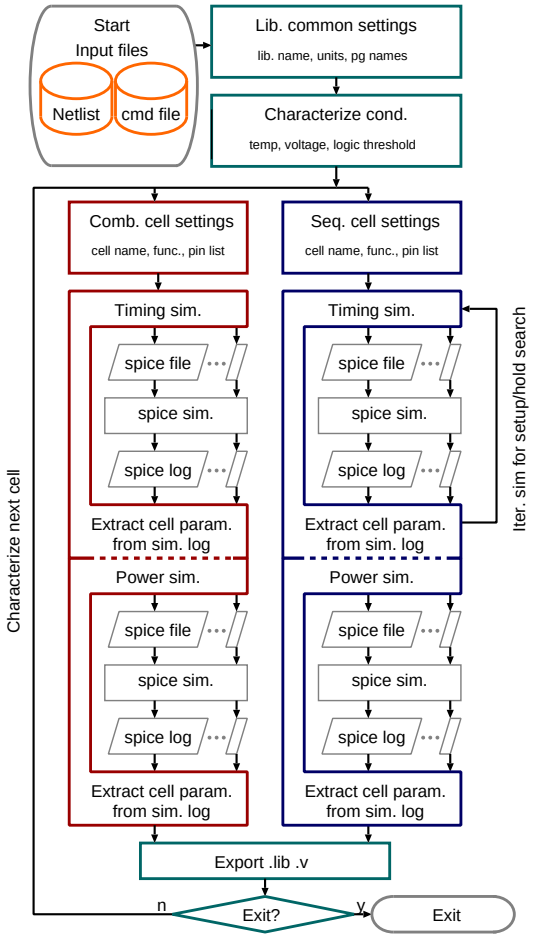


Fig. 1: Internal flow of **libretto**.

IV. Detail of Characterization

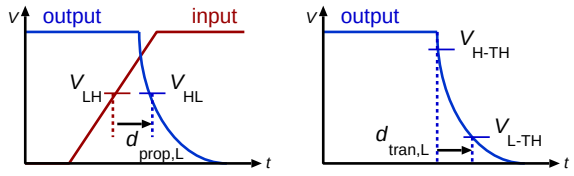
This section describes the detail of the timing characterization and power characterization. **libretto** uses straight forward implementation of cell characterization, based on Liberty Users Guide [6].

A. Timing characterization

1) Common setting for timing simulation: In timing characterization, propagation delay and transition delay are extracted from simulation. Propagation delay ($d_{\mathrm{prop,L(H)}}$) is the time from half-VDD of input ($V_{\mathrm{LH(HL)}}$) to output ($V_{\mathrm{HL(LH)}}$), and transition delay ($d_{\mathrm{tran,L(H)}}$) is the time from output logic-high(low) threshold ($V_{\mathrm{H(L)-TH}}$) to output logic-low(high) threshold ($V_{\mathrm{L(H)-TH}}$), and these definitions are illustrated as figure 2 [1]. **libretto** also measure the start time of input swing (t_{start}) and end time of output swing (t_{end}) for dynamic power calculation. Figure 3(a) shows the ".measure" parameters in the timing simulation.

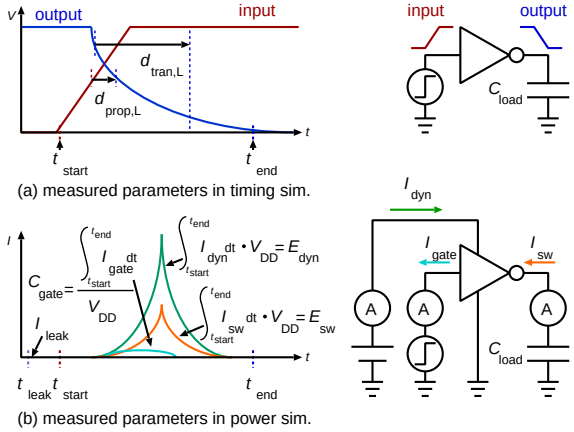
2) Setup/Hold search for sequential cells: Sequential cell has several metrics of delay. Let us consider a Flip-Flop with positive edge clock, which has data input (D), clock input (C), and data output (Q). C-to-Q (C2Q) is the delay from clock edge to output change. The data input must be stable around some time window of clock edge. D-to-C (D2C) is the delay between data edge before the clock and clock edge, and it is called setup time which is the minimum required time that data input must be stable before the rising edge of clock input. C-to-D (C2D) is the delay between clock edge and data edge after the clock, and it is called hold time which is the minimum required time that data input must

[1]User of **libretto** can redefine these thresholds by command.



(a) Propagation delay (b) Transient delay

Fig. 2: Definition of propagation delay and transition delay.



(a) measured parameters in timing sim.

(b) measured parameters in power sim.

Fig. 3: ".measure" parameters in timing and energy evaluation.

be stable after the rising edge of clock input. D-to-Q (D2Q) is the minimum delay from data edge to the output change, and it is the sum of D2C and C2Q delay.

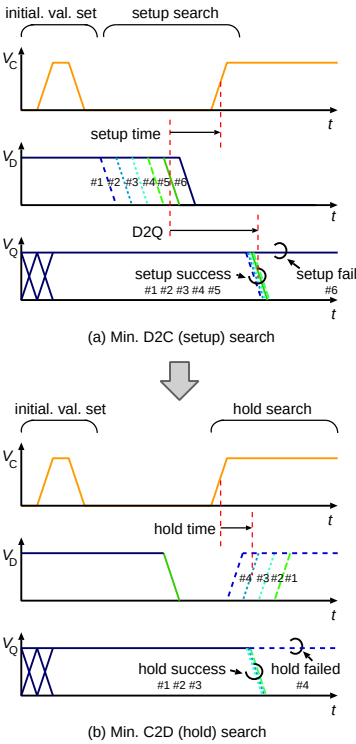
Since the definition of setup and hold time is the minimum value of sequential cell time, we need to search this timing with multiple simulation runs. In this paper, we use minimum D2Q to search setup time [13]. Figure 4 show the setup, D2Q, and hold search in simulation. Firstly, **libretto** generates multiple spice files from larger D2C to smaller D2C and find the minimum D2Q and define D2C as setup. After the setup time is fixed, secondly, **libretto** generates multiple spice files from larger C2D to smaller C2D, and find the minimum C2D with correct Flip-Flop operation as hold time.

3) Recovery/Removal search for sequential cells: Recovery/removal are important metric to guarantee the operation correctness of asynchronous set and reset operation. Recovery is the minimum allowable time between the edge of set/reset signal to inactive state to synchronous clock signal. Recovery time is used to ensure the operation correctness of triger the correct data input after set/reset is inactivated. Removal is the minimum allowable time between the synchronous clock signal at the active set/reset to the edge of inactive set/reset signal. Removal time is used to ensure the operation correctness of set/reset operation at the edge of synchronous clock. Figure 5 show the recovery/removal search in simulation. Similar to the setup/hold search, **libretto** search recovery constraint for set/reset pin. After recovery is fixed, **libretto** search removal constraint using the recovery constraint.

It is clear that this simulation requires large simulation time, and parallel execution will help to speed up the characterization. However, this version of **libretto** uses sequential run of spice simulation and parallel execution is our future work.

B. Power characterization

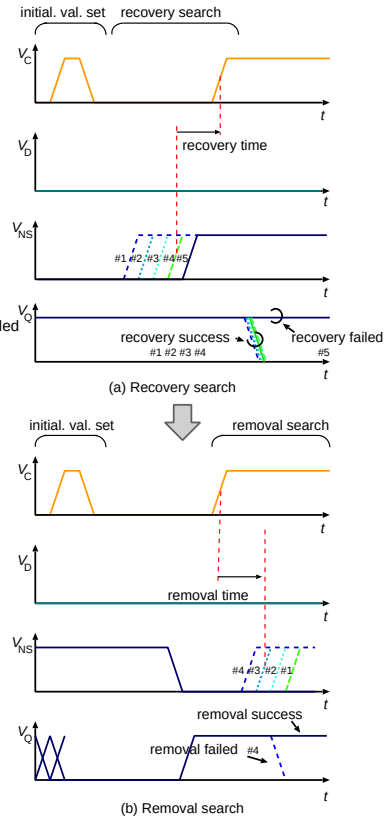
In power characterization, we need to measure energy consumption per operation and leakage power. Current flow from power source (I_{dyn}) and output loading (I_{sw}) are integrated from t_{start} to t_{end}, then converted as energy (E_{dyn}, E_{sw}). Energy consumption is calculated subtracting E_{sw} from E_{dyn}. For leakage power, **libretto** measure the average of current consumption



(a) Min. D2C (setup) search

(b) Min. C2D (hold) search

Fig. 4: Setup and hold search.



(a) Recovery search

(b) Removal search

Fig. 5: Recovery and removal search.

TABLE I: Characterization setting.

Characterizer	**libretto**		SiliconSmart
Simulator	ngspice	hspice	hspice
Simulator option	default	default	default
# of threads	1	1	32
Process tech.	Commercial 180-nm w/ small modification	Commercial 180-nm	
Input waveform	ramp		
PVT cond.	Typical process, nominal voltage (1.8 V), 25°C		
Input slope	0.01 ns, 0.1 ns, 1.0 ns		
Output load	0.1 pF, 0.7 pF, 4.9 pF		
DUT for comb. cell	Inverter		
DUT for seq. cell	Flip-Flop w/ pos. edge clock		

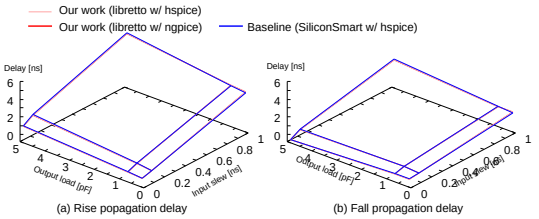
(I_{leak}) of each input patterns at the start time t_{leak} of the simulation then calculate cell leakage power. Gate capacitance is calculated from the charge of gate (integral of gate current I_{gate}) and divided it by supply voltage. This definition is common for both combinational cells and sequential cells. Figure 3(b) shows the ".measure" parameters for power simulation.

V. Experimental Results

We implement **libretto** using Python3, and ngspice is used as simulation engine. We prepare several netlists of the cell from PDK which targets 180-nm process as an evaluation. We use Synopsys SiliconSmart as a baseline of the characterizer and evaluate the performance of **libretto**. Also, we add support of hspice for **libretto** to check the algorithm difference in characterizer. Table I shows the common settings for both characterizers. Inverter is selected as a representative of combinational cell, and positive edge Flip-Flop is selected as a representative of sequential cell. Results does not added any timing and power mergins from simulation result, explicitly.

A. Combinational logic characterization

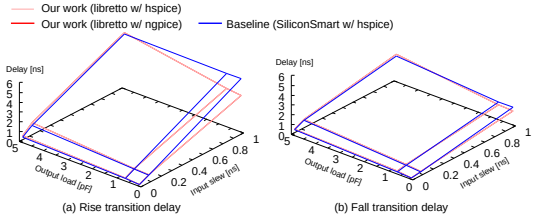
Inverter cell is characterized to compare its delay, energy consumption, leakage power and capacitance. Figure 6,7,8 show



(a) Rise popagation delay

(b) Fall propagation delay

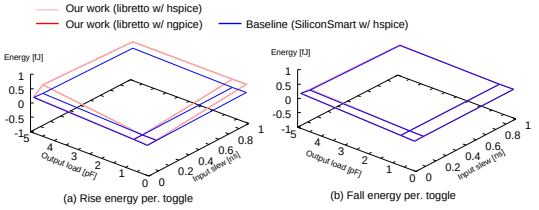
Fig. 6: Propagation delay of Inverter.



(a) Rise transition delay

(b) Fall transition delay

Fig. 7: Transition delay of Inverter.



(a) Rise energy per. toggle

(b) Fall energy per. toggle

Fig. 8: Internal energy of Inverter.

TABLE II: Capacitance and leakage power of Inverter.

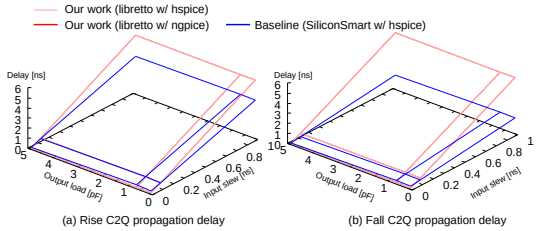
Characterizer	**libretto**		SiliconSmart
Simulator	ngspice	hspice	
Leakage power [pW]	9.862	9.862	9.862
Input capacitance [fF]	4.120	4.063	4.599

the comparison result of propagation delay, transition delay, and energy consumption. Characterization results in **libretto** with two different simulator ngspice (in red color) and hspice (in pink color) show almost same result, and two results are overlapped in these figures. Main difference in characterized result comes from characterizion algorithms in two characterizer. Three results of propagation delay match well. Transition delay has mismatch at the higher input slope. Energy consumption has large mismatch at the input rise case, it will result the power estimation in pessimistic. Maximum errors in propagation delay and transition delay are 0.50% in rise, 1.44% in fall. Internal energy shows 418% pesimistic estimation in rise condition. Table II shows the comparison result of leakage power and input capacitance. Leakage power looks matched well. Input capacitances has almost 10% mismatch.

Flip-Flop is characterized to compare its C2Q propagation delay, transition delay, setup, hold and energy consumption. Comparison results are plotted at figure 9, 10,11,12,13. Propagation delay and transition delay at the fall input pattern have some mismatch at large input slope and large output load conditions. Maximum errors in propagation delay are 1125% and 2407% at rise and fall conditions, respectively [2] .

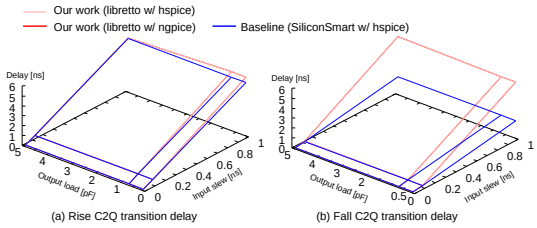
From figure 11 and 12, **libretto** reports smaller setup time but larger hold time. One main reason is inter-dependence of the setup and hold time: tighter setup(hold) constraint requires longer hold(setup) time for correct operation of Flip-Flop [14]. **libretto** first tries to search minimum setup time of Flip-Flop thus it results larger hold time. The characterizing flow used in **libretto** is straightforward and reduce the accuracy of STA, it is

[2]Maximum errors are observed at the small output load condition thus ratio value becomes large.



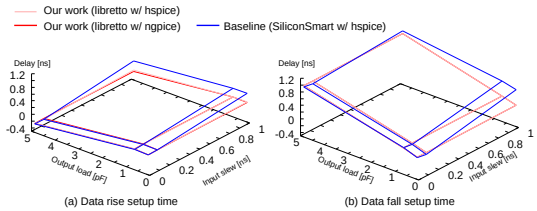
(a) Rise C2Q propagation delay (b) Fall C2Q propagation delay

Fig. 9: C2Q propagation delay of Flip-Flop.



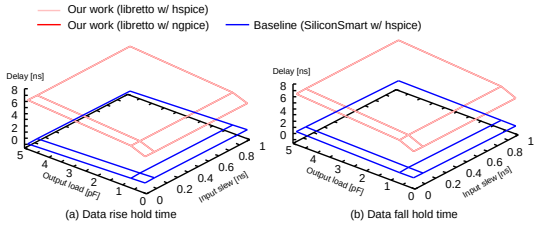
(a) Rise C2Q transition delay (b) Fall C2Q transition delay

Fig. 10: C2Q transition delay of Flip-Flop.



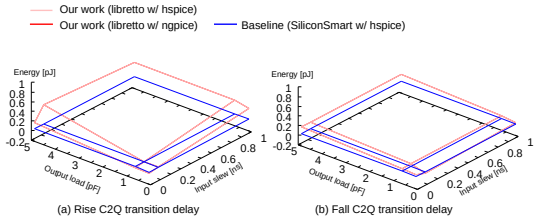
(a) Data rise setup time (b) Data fall setup time

Fig. 11: Setup time of Flip-Flop.



(a) Data rise hold time (b) Data fall hold time

Fig. 12: Hold time of Flip-Flop.



(a) Rise C2Q transition delay (b) Fall C2Q transition delay

Fig. 13: Energy consumption of Flip-Flop.

required to handle this inter-dependence of setup and hold time [15] in future work. SiliconSmart can handle this issue to search setup and hold in the same time to find much more balanced constraint. Figure 13 shows the energy consumption of Flip-Flop, and **libretto** reports pesimistic result of energy consumption. Maximum errors in energy consumption of Flip-Flop are 889% and 1915% at rise and fall conditions, respectively. Different setup constraints may affect this difference, since tighter setup constrain affect robust operation of and energy consumption of Flip-Flop.

B. Runtime comparison

We use Linux machine with AMD Ryzen 2990wx 32-core CPU with 96GB memory for characterization. To characterize Inverter and Flip-Flop, **libretto** with ngspice in single thread simulation requires 2.5 hour in total. **libretto** with hspice in single thread simulationachieves 2.26× faster characterization. SiliconSmart with hspice in 32-thread simulation achieves 215× faster characterizatin, compared to **libretto** with ngspice. This is because the sequential operation of Flip-Flop setup and hold search, since this search requires multiple spice simulation to find the minimum. Current version of **libretto** uses simple gradient serarch with single-thread simulation, thus it requires huge runtime compare to commersial characterizer. Utilize the parallelism is our future work.

VI. Conclusion

In this paper, we introduce our free and open-sourced characterizer. It supports the characterization of both combinational cells and sequential cells. It supports both timing model and power model to enable timing estimation and power estimation in STA. Free and open-sourced characterizer will help open-sourced VLSI design project to add designers own cell into VLSI design. Program code and environment is uploaded in authors GitHub page.

https://github.com/snishizawa/libretto

Our future work is performance improvement and accuracy improvement in characterization. Characterization time can be reduced drastically when parallelism is enabled in spice simulation. Input waveform model should be improved from current simple ramp waveform to realistic driver model. Interdependency of setup time and hold time should be consider in the characterization to improve the accuracy of STA. Need more support and verification for several transistor models are also important issue.

Acknowledgment This work has been supported by Logic Research (190402, 210104), and funding from Fukuoka University (197105, 217301). This work is also partly supported by VDEC, the University of Tokyo in collaboration with Synopsys, Inc..

References

[1] "MakeLSI Project." [Online]. Available: https://github.com/MakeLSI
[2] R. T. Edwards, "Google/SkyWater and the Promise of the Open PDK," in *Workshop on Open-Source EDA Technology*, 2020.
[3] R. T. Edwards, et al., "Real Silicon Using Open-Source EDA," *IEEE Design and Test*, vol. 38, no. 2, pp. 38–44, 2021.
[4] S. Nishizawa and T. Nakura, "libretto : An Open Cell Timing Characterizer for Open Source VLSI Design," *IEICE Transactions on Fund. (to appear)*, 2022.
[5] Ngspice, *ngspice - open source spice simulator*.
[6] Synopsys, *Liberty User Guide Volume 1*.
[7] G. Bronstein, et al.,, "Asic standard cell library design by graham petley." 1991. Available: http://www.vlsitechnology.org
[8] I. K. Rachit and M. S. Bhat, "AutoLibGen: An open source tool for standard cell library characterization at 65nm technology," in *Intl. Conf. on Electronic Design*, 2008.
[9] M. T. Moreira, et al., "LiChEn: Automated electrical characterization of asynchronous standard cell libraries," *Euromicro Conf. on Digital System Design*, pp. 933–940, 2013.
[10] M. S. I. et al., "Development of automated standard cell library characterization (ASCLIC) for nanometer system-on-chip design," *Student Conference on Research and Development: Inspiring Technology for Humanity*, vol. 2018, pp. 93–97, 2018.
[11] C. H. Oliveira, et al., "ASCEnD-FreePDK45: An open source standard cell library for asynchronous design," in *Intl. Conf. on Electronics, Circuits and Systems*, 2017, pp. 652–655.
[12] A. Beg, et al., "A collaborative platform for facilitating standard cell characterization," *Intl. Conf. on Computer Supported Cooperative Work in Design*, pp. 202–206, 2013.
[13] N. H. E. Weste and D. M. Harris, *CMOS VLSI Design*, 4th ed., Addison Wesley, 2010.
[14] E. Salman, et al., "Pessimism reduction in static timing analysis using interdependent setup and hold times," in *Intl. Symposium on Quality Electronic Design*, 2006, pp. 159–164.
[15] H. Seo, J. Heo, and T. Kim, "Clock Skew Optimization for Maximizing Time Margin by Utilizing Flexible Flip-Flop Timing," in *Intl. Symposium on Quality Electronic Design*, 2015, pp. 35–39.