Fast and Accurate Library Generation Leveraging Deep Learning for OCV Modelling

Eunice Naswali[1], Namhoon Kim[2], Pravin Chandran[3]
[1]Intel Corporation, San Jose CA, [2]Intel Corporation, San Jose CA, [3]Intel Corporation, India
[1]eunice.naswali@intel.com, [2]namhoon.kim@intel.com, [3]pravin.chandran@intel.com

Abstract

Statistical timing characterization for modeling On-Chip Variation (OCV) is critical in current technology nodes to avoid over-design and to improve design convergence and predictability. OCV characterization, however, is resource intensive as it involves running millions of Monte-Carlo spice simulations to cover different timing arcs for multiple cells in standard-cell library. We have developed a neural network model that fully comprehends multiple cell types to model cell propagation delays as well as OCV sigma at target process-voltage-temperature (PVT) corners with a significantly reduced number of simulations. The proposed method generates Liberty Variation Format (LVF) models which are the latest and most accurate representation of OCV margin in the industry's standard tools and flows.

On extensive testing with 7 million OCV delay values in 10nm node, we attained 60% reduction in runtime while maintaining prediction-error less than 5% for 99.98% arcs which can be used for early timing integration.

Keywords

Library cell characterization, PVT, EDA, OCV, LVF, Neural Networks, Spice.

1. Introduction

As the technology node shrinks, the cell performance becomes very sensitive to multiple variability sources arising from device as well as interconnects. A margin or derating value is typically added to account for this variability during Static Timing Analysis (STA). For older technology nodes, a global derating value was used. This, however, resulted in excessive pessimism and over-design leading to severe performance limitations. Advanced on-chip variation (AOCV) was then adopted to account for cancellation of random variation effects with path depth and distance. This methodology reduced pessimism associated with the use of global derating value. The derating methodology was further enhanced to Parametric On-Chip Variation (POCV), which models the local variation as a function of the intrinsic cell delay and load parasitics. POCV can be represented with the liberty variation format (LVF), a widely used industry standard. LVF models capture delay impact of OCV at multiple slew and load combinations in delay tables. Each of these methodologies have been developed to incrementally decrease pessimism associated with variability modeling and improve accuracy of static timing analysis. But these accuracy improvements come at the expense of increase in runtime and resource usage in library characterization stage. A major contributor to the extensive runtime and resource usage mentioned is due to the fact that designs must work across multiple operating conditions and contexts that cover the entire PVT space.

Research on the trade-off between PVT coverage and time-to-market has been a focus on timing analysis for a while. Finding the minimum set of dominant corners has become one of the important research topics for example in [2]-[4]. Reducing runtime while improving accuracy for timing library generation has also been a mainstream of modeling topics as shown in [5]-[8]. Machine learning has been gradually involved in EDA field given the increasing complexity in semiconductor design and analysis. There are many efforts to expand the ML solution to EDA tools and hardware design methodologies for example as depicted in [9]. Predictive algorithm, self-learning mechanism and many other ML concepts have already been explored in timing analysis. [10] proposes a corner-less static timing analysis approach that uses a single run of STA to cover all process corners with propagation of delay and slew models as linear functions of process parameters. As an extension of this approach for better accuracy, [11] proposed a prediction of the timing analysis at unobserved corners. It used trained analysis results at observed corners based on multivariate linear regression and predicted possible timing violations at unobserved corners. [12] proposed a learning-based STA library to predict the negative bias temperature instability (NBTI) induced delay degradation. It utilized smaller designs to learn the degradation on gates and predicted timing of gates in processor-sized designs. Deep Learning was employed to improve the computational efficiency of timing library characterization in [13]. This approach predicted propagation delays at all necessary slew and load values using delays at fewer set of (slew, load) points as reference.

In this paper, we seek to resolve the trade-off between accuracy and characterization runtime, with the use of an enhanced deep neural network to accurately model the variation coefficients and delay values in LVF CCS format across all signoff PVT corners. This would help improve design convergence and reduce significant number of spice simulations during characterization, so that it is no longer necessary to sacrifice accuracy over runtime or vice versa.

The remainder of this paper is organized as follows. In Section 2 we give the reader background on the concepts of variation, characterization and library development. Sections 3 and 4 presents the proposed approach, while Sections 5 and 6 depict the results and conclusions.

2. Background

The core of our work presented in this paper is on PVT variation; an extremely complex phenomenon as it contains systematic and random components. A representation of PVT space is provided in Figure 1. Manufactured devices can fall anywhere in the PVT space and still expected to be functional.

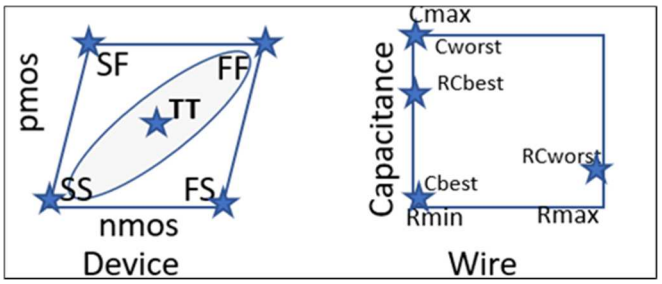

Figure 1: Process Complexity

An engineering approach to address this is to break it down into systematic (global) and random (local) components and address each of them separately as illustrated in Figure 2. To handle global variation multiple corners, representing the device and interconnect at different points in PVT space, are defined for a project and the chip is analyzed and validated at all these corners. There exists a strong correlation between the number of corners and confidence with which a design can be taped-out. To support analysis at multiple corners and manage complexity, cell/gate behavior is abstracted through cell-characterization at each predefined global corner. Local variations are handled by adding their cumulative impact to delays during analysis at each global corner. To characterize the delay impact due to local variations, several Monte-Carlo simulations are run at each (slew, load) point for each arc/condition/cell in the library. The specifics on characterization methodology for local OCV are discussed in the next section.

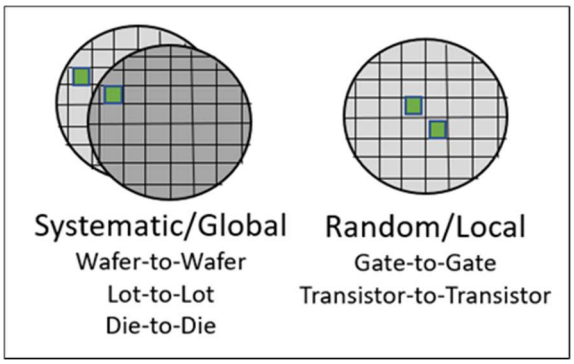

Figure 2: Classification of device variation into global and local variation.

2.1. Characterization Process

As described earlier, timing abstracts/models are generated through a process referred to as timing characterization. This process involves a series of spice analysis runs for measuring several parameters like propagation delay, output transition time, input capacitance, setup time, hold time, etc., and capturing the results in a lookup table format. This measurement occurs over a range of input slews and output loads and at several PVT corners. Timing characterization for local variation captures delay impact due to OCV, which is later used to derate path-delays during STA. Local variation is captured using Monte-Carlo simulations which runs hundreds of spice simulations for variation effects at each (slew, load) characterization points and is represented in industry standard formats like LVF. A conceptual illustration of the characterization process is depicted in Figure 3. Spice simulations would need to be run at each of the multiple simulation contexts (sim context) which include corners, slews, loads, etc., for all the cells in the library. This results in an extensive number of required Monte-Carlo simulations that need to be completed.

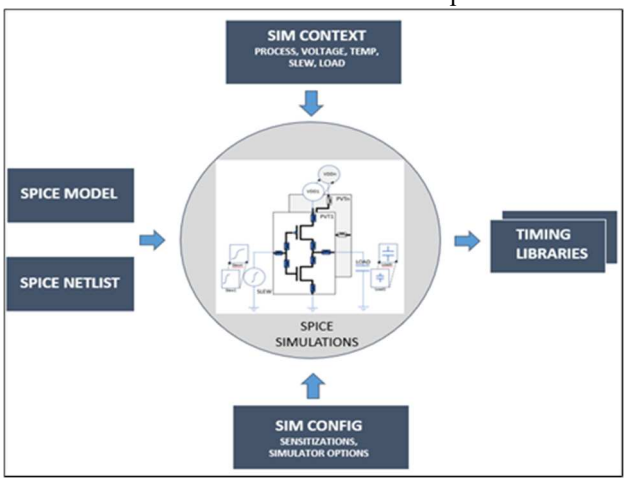

Figure 3: Library Characterization Flow

2.2. Library Development Challenges

Recent EDA characterization tools offer required automation to run spice simulations at all the specified simulation contexts and generate libraries with OCV data. However, their compute resource requirement is very high, due to the reasons stated earlier and the flow cycle can take several months even on reasonably sized compute farms. For observed timing library generation, statistical OCV characterization has been shown to occupy as much as 80% to 90% of the total characterization effort depending on the cell complexity and spice optimization settings [1]. Due to this bottleneck, library characterization and library releases/deliverables typically happens in phases spanning several months. Early availability of complete PVT libraries would benefit all the downstream tools and help design closure with improved predictability. This work aims to bridge the gap in the current library generation and release process by supplementing predicted ocv delays for early integration cycles and switch to simulation-based delays closer to tapeout.

3. Methodology and Models

As introduced in the previous sections, statistical variability in electrical behavior of a gate is a complex phenomenon and variability characterization is an extremely time-consuming process. A model-based estimation can offer significant Time-To-Market benefits compared to brute-force characterization of entire library. Current methodologies for modeling variation effects result in either a) Increase in number of simulations, if full blown Monte-Carlo simulations are performed at all necessary (slew, load) points for all arcs/cells in a library or b) Increase in number of models needed to generate LVF tables, if multiple models are created to model different arcs, functionality, cell-types and V_{th} types. Modelling methodologies that are arc/gate specific, cannot scale to new cells or corners. This work eliminates the need for multiple cell-specific models by using arc/gate specific attributes for an arc as the 'Gate-Reference', when performing delay prediction for any (slew, load) index

corresponding to the arc. So, the global model essentially uses the information in the gate-reference to make a prediction for the specific arc while learning OCV scaling behavior from a variety of arcs and cells. Use of a gate-reference, allows us to characterize and feed a sparse OCV delay table to a trained model and generate a dense OCV delay table as shown in Figure 4. The methodology for creating the gate-reference is outlined in the following section.

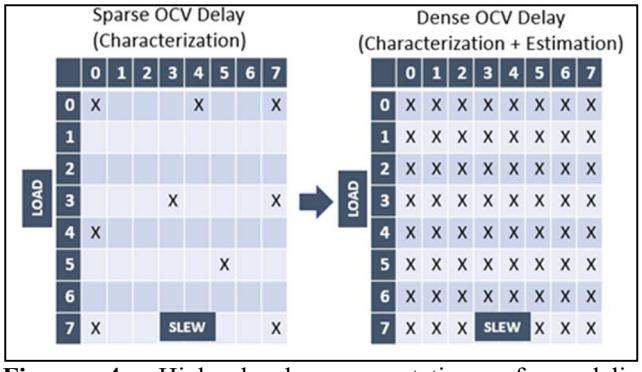

Figure 4: High level representation of modeling methodology: Dense OCV table is generated from a Sparse OCV table using a global neural network model for all cells in library at a target PVT corner.

Gate-Reference: We choose OCV delays at selected (slew, load) indices, along with their corresponding slew and load values as illustrated in Figure 4 and use it as the gate-reference. Delays from same indices are picked for all arcs/cells in the library. The points are strategically handcrafted across the boundary and diagonal (slew, load) indices with the intent to keep the total number of gate-reference points to a minimum. During our trials as shown in Table 1, we found the model to offer similar performance even when indices were slightly varied. If a higher error is slightly tolerable as it would yield further savings, a lower gate reference i.e. 5points could be utilized. For this work, 10 points for the Gate-Reference that corresponds to ~16% of total points for an 8x8 (slew, load) table gave a good trade-off between accuracy and runtime.

Table 1: Trial runs for selecting optimal neural network parameters.

Training Data	Gate Reference	Topology		Arcs with Error > 5%
		Blocks	Layers	
10%	10	4	40	0.16
20%	10	4	40	0.06
30%	10	4	40	0.024
30%	5	4	40	0.06
30%	7	4	40	0.047
30%	10	4	40	0.024
30%	18	4	40	0.017

Neural network modeling has two major phases: Training and Inference. The Training phase is used to build a model using data for which output values are known. Neural network training requires large volumes of data and our methodology to supply attributes of arc as gate-reference allows us to merge data from different arcs/cells in a library to create this training data. When a model is trained, it can be used to predict unknown delays in Inference phase.

The proposed methodology for reduction in statistical local variability characterization is summarized below;

a) For 30% of the library cells, full blown Monte-Carlo simulation is performed at all (slew, load) points and the data is used for training. Cells are chosen using a random selection to allow model to learn OCV behavior across different arcs and cell-types.
b) Sparse characterization is performed for remaining 70% cells in the library. This sparse library (16% of 8x8 table) is used as gate-reference during inference to supply arc/cell specific attributes to the global model.
c) Use the model from step (a) to predict OCV delays for non-characterized 84% of the (slew, load) points for 70% of the cells from step (b).
d) This translates requirement to characterize only ~40% of total points (Training: 30% + Gate-Reference for Inference: 16% of 70% = 11.2%), resulting in 60% reduction in runtime including characterization effort needed for training.

High level representation of the modelling methodology is presented in Figures 4 and 5 where ocv_dly_l for an arc is modelled using gate-reference vector x_b and input vector x_a. X_b matrix is comprised of ocv_dly for 10 (slew, load) reference points along with other cell attributes; threshold voltage V_{th} type, rise/fall and early/late sigma indicators. X_a matrix is comprised of random selection of ocv_dly from all 64 (slew, load) points.

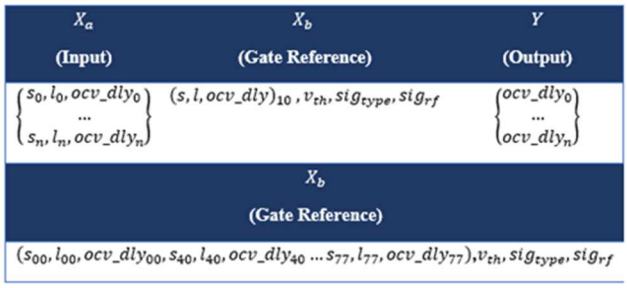

Figure 5: Representation of Modeling Methodology: Gate-Reference X_b from sparse characterization is fed to model along with input X_a for the target arc to predict output Y $f(X_a, X_b)$. $X_b = (s_{ij,} l_{ij,} ocv\_dly_{ij,}, v_{th}, sig_{type}, sig_{rf})$ where $ij \in (00,40,70,33,73,04,55,07,47,77)$ and $v_{th} \in (high, low, nom)$. $X_b = (s_{ij,} l_{ij,} ocv\_dly)$ where $ij \in (8x8)$.

Figure 6 shows the neural network architecture implemented. The network consists of combination of Fully Connected layers and ReLu activation functions similar to the architecture in [12]. Optimization process for neural network training is discussed in section 4.

We have tested our methodology for OCV sigma estimation at three major front-end categories of corners (slow, typical and fast) with consistent results which suggests that it is easy to expand this methodology to model entire set

of library corners. Our methodology combines characterized data with model-based estimates to achieve 60% reduction in characterization effort, which, in turn, enables variation aware timing analysis to be performed much earlier in the design cycle. Our model-based libraries can be swapped with characterization-based libraries at any stage in design cycle in a seamless fashion. The proposed solution can further be extended to other various characterization areas such as dynamic power, leakage power, noise, and other sensitive variability sources by simply updating the library parser to work with each objective function.

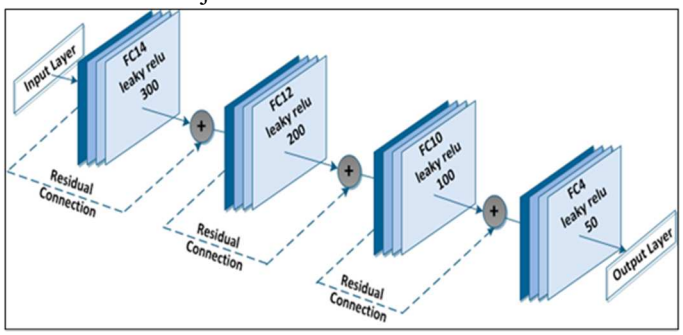

Figure 6: DNN library generator model architecture using deep neural networks with residual connections for accurate delay modeling.

4. Training and Model Optimization

This section focuses on steps followed to build the neural network model in this paper. Neural networks have shown promise in several industry domains and have been an area of thrust for research and development for modeling. There are several open-source frameworks available to build neural network models and most frameworks would support our architecture. A large training dataset is required to achieve high accuracy with neural networks and neural network training is also impacted by choice of hyper-parameters used for training.

Neural network architecture was optimized over several trials using training vs test-loss. Training-loss is the prediction error seen for the same dataset that used for training and test-loss is prediction error seen in case of new data that was not used for training. A good model is expected to have test-loss comparable to training-loss. Also, a model with lower values for these metrics is desirable compared to one with higher values. Explorations using residual connections [14] resulted in reduction in loss and was adopted for our final architecture presented in Table 2. The number of layers and blocks was determined based on experiments carried out on authors' previous related work [13] that showed fewer layers resulted in larger rms loss and less train-vs-validation accuracy. The optimal numbers picked gave a small deviation between train and validation rms loss as well as an acceptable accuracy.

We used 30% of standard cells for training in our current implementation which lead to 2.2million records covering multiple timing arcs across cell-types, Vth types, rise/fall conditions and different sensitizations. As mentioned earlier, these cells were selected based on random selection. Establishing the lower-bound for characterization simulations required, will be explored in future.

Table 2: Neural Network Architecture. Set of Fully Connected Layers (FC) along with a residual connection is referred to as a block.

	Type	Layers	Hidden Units
Block1	FC+ ReLu	14	300
Block2	FC+ ReLu	12	200
Block3	FC+ ReLu	10	100
Block4	FC+ ReLu	4	50

Neural network training typically also involves several hyper-parameters like batch-size, number of epochs for training, learning-rate, etc., which were refined based on several experimental trials. A snapshot of critical parameters of our training setup is presented in Table 3. We trained neural network models on representative libraries from slow, fast and typical regions of PVT spectrum covering four representative Vth-types from High Vth to Ultra-low Vth. The results from our explorations are presented in the next section.

Table 3: Neural Network Training for single corner and single Vth.

Item	Detail
Training Set	2.2 million
Test Set	7.4 million
Unique Cells	260
Batch Size	16
Epochs	100
Learning Rate	0.001
Optimizer	Adam

5. Results

We analyze the modeling error associated with OCV delays in this section. The difference between measurements from actual characterization and model-based estimates is used to quantify the Prediction Error. Prediction Error as a function of actual delay is plotted in Figure 7, for a few representative corners. For our evaluation, we target prediction error within 5% which is the most common targeted accuracy in spice simulation results for timing analysis purposes. A reference error line is drawn in green in Figure 7 to represent our error target. The corner names and Vth types have been anonymized for presentation and publication purposes.

We present results from two different models at a target corner.

a) Single Vth Model: Model built using data from cells corresponding to one Vth at a time [Figure 7].

b) Multi Vth Model: Model built by combining data for all Vth together [Figure 8].

From the error distribution plots in Figure 7, we find that, for the single Vth model (Ex: Vth1-Corner1), out of 7.4 million (slew, load) points tested, less than 1000 points have predicted error greater than 5% which is an accuracy percentage acceptable for spice simulations. This suggests that the model-based prediction is within an acceptable

accuracy for 99.988% of the cells/arcs. A similar trend is observed with multi Vth model, as shown in Figure 8. These results offer credence to our belief that the proposed methodology and architecture can scale well under multiple contexts and corners.

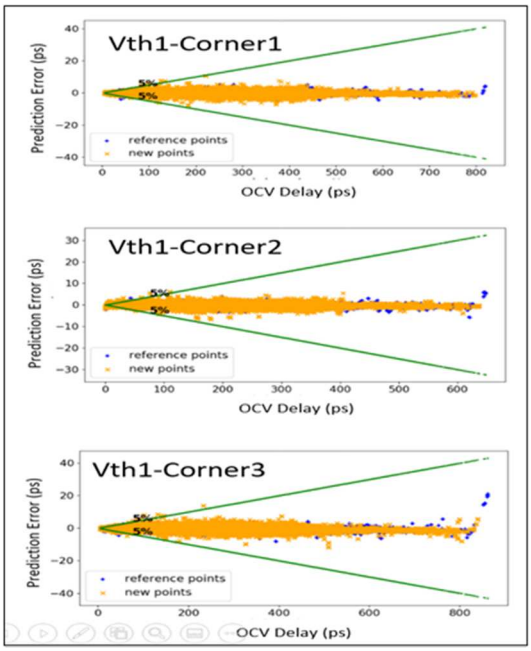

Figure 7: Distribution of Prediction Error for a Vth type at different corners picked by fast, slow and typical regions in PVT space. Observed error is within 5% for 99.988% of the cases.

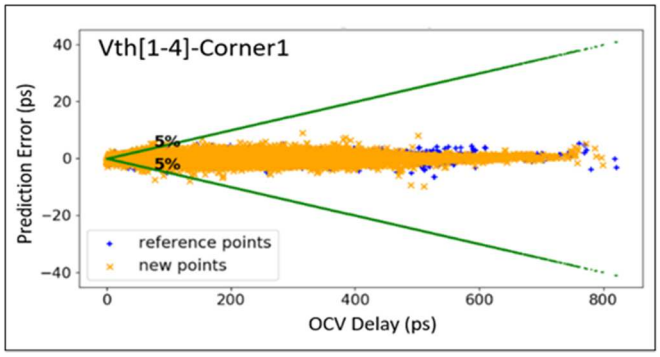

Figure 8: Distribution of Prediction Error for all Vth-Types at single PVT corner. Observed error is within 5% for majority of the cases.

The distribution of cases with 'Prediction Error' greater than 0.012% for Vth1-Corner1 is presented in Table 4. Each cell in the table holds the percentage of arcs that lie outside error tolerance, at the indicated (slew, load) index. By indexing the table by slew, load indices we obtain a good view of regions of error concentration. From the table, it can be observed that majority of the error cases are concentrated at the boundary (slew, load) points. For instance, 80% of points with an error greater than acceptable tolerance corresponds to a slew index '0' or a load index '0', which is the smallest slew/load used for characterizing those cells.

Table 4: Distribution of Error Across Slew and Load Index for the 0.012% of total data points with error greater than 5%. Each cell displays the percentage of arcs with errors at specified (slew, load) point. Majority of cases are limited to the smallest load or sharpest slew.

slew/ load	0	1	2	3	4	5	6	7
0	35%	19%	7%	4%	1%	0%	0%	0%
1	9%	3%	1%	1%	0%	0%	0%	0%
2	2%	1%	0%	0%	0%	0%	0%	0%
3	2%	1%	0%	0%	0%	0%	0%	0%
4	2%	0%	0%	0%	0%	0%	0%	0%
5	2%	0%	0%	0%	0%	0%	0%	0%
6	1%	0%	0%	0%	0%	0%	0%	0%
7	4%	1%	1%	0%	0%	0%	0%	0%

For max-timing, boundary slew and load points in this region are mostly characterized to enable interpolation and gates in an optimized design and the design is not expected to operate here. Moreover, large percentage error in this region would still translate to a small absolute error. For these reasons, we argue that some error in this region is tolerable for early timing analysis for max-timing. For min-timing however, this region is critical and future work to minimize this error can be explored through use of different indices for gate-reference as shown in Figure 9. Alternatively, error can also be minimized by using smaller-than-expected slew, load values for index (0, 0), so that this boundary is far from actual operating region.

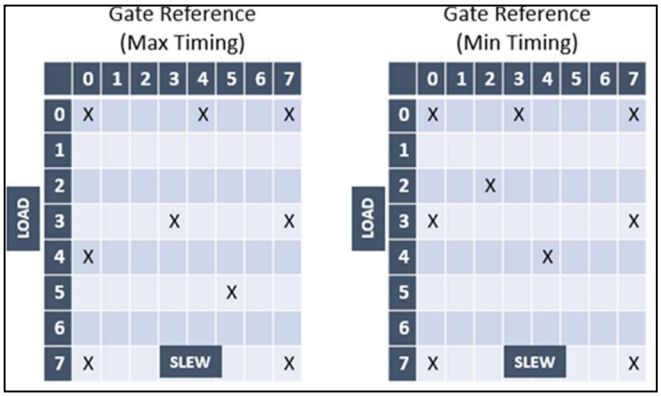

Figure 9: Gate-Reference (slew, load) indices for max and min libraries are chosen strategically to achieve better accuracy.

The results presented validate our claim that our methodology and framework have a good comprehension of local variation and can offer high quality estimates for OCV sigma while significantly reducing the characterization requirements.

6. Conclusion

With increase in significance of on-chip variation effects and an explosion in the number of analysis corners, library characterization for variability has become exceedingly expensive and time consuming. This demands innovations outside of typical EDA tool enhancements to allow faster time to market with high accuracy and predictability. We have presented a deep learning-based modeling framework to comprehend variation effects on timing that can easily scale well for different library characterization tasks. We have also demonstrated that our proposed approach achieves high accuracy (>95% given that 5% which is the most common targeted accuracy in spice results for timing analysis

purposes.) for 99.988% of data points, out of 7 million data points tested. Early availability of the full set (i.e. all corners) of libraries achievable with our model helps identify and address issues early in design cycle and that goes a long way in improving designer productivity by minimizing rework. Data from proposed model can then be replaced with actual characterization data, once it is available, thereby removing library-characterization from the critical path in design integrations, without compromising accuracy. By combining deep learning techniques with data from EDA tools, we are able to significantly reduce time-to-market for hardware products and foster competitive advantage.

7. References

[1] T. Ragheb, S. Chan, N. Jin and R. Trihy, "Advanced Node Random Device Variability Modeling and Margining in Characterization and STA", SNUG Silicon Valley, 2014

[2] J. Nian, S. Tsai and C. Huang, "A Unified Multi-corner Multimode Static Timing Analysis Engine", Asia and South Pacific Design Automation Conference, 2010.

[3] S. Onaissi, F. Taraporevala, J. Liu and F. Najm, "A Fast Approach for Static Timing Analysis Covering All PVT Corners", Design Automation Conference, 2011.

[4] S. Matarrese, A. Shaligram, M. Karna and S. Hirani, "Methodology to Close Timing with Hundreds of Multi-mode/multi-corner Scenarios", SNUG Silicon Valley, 2016.

[5] J. Chang, L. Johnson and C. Liu, "Piecewise Linear Delay Modeling of CMOS VLSI Circuits", International Midwest Symposium on Circuits and Systems, 2009.

[6] Y. Wang and M. Zwolensky, "Analytical Transient Response and Propoagtion Delay Model for Nanoscale CMOS Inverter", International Symposium on Circuits and Systems, 2009.

[7] S. Miryala, B. Kaur, B. Anand and S. Manhas, "Efficient nanoscale VLSI standard cell library characterization using a novel delay model", International Symposium on Quality Electronic Design, 2011.

[8] A. Ciccazzo, G. Pillo and V. Latorre, "Support Vector Machines for Surrogate Modeling of Electronic Circuits", Neural Computing Applications 24, 2014.

[9] L. Wang and M. Luo, "Machine Learning Applications and Opportunities in IC Design Flow", International Symposium on VLSI Design, Automation and Test, 2019.

[10] A. Kahng, "Machine Learning Applications in Physical Design: Recent Results and Directions", International Symposium on Physical Design, 2018.

[11] A. Kahng, U. Mallappa, L. Saul and S. Tong, "Unobserved Corner" Prediction: Reducing Timing Analysis Effort for Faster Design Convergence in Advanced-Node Design", Design Automation and Test in Europe 2019.

[12] S. Bian, M. Sintani, M. Hiromoto and T. Sato, "Learning-Based Static Timing Analysis for High-Dimensional Correlated On-Chip Variations", Design Automation Conference, 2017.

[13] E. Naswali, A. Quiros and P. Chandran, "DNNLibGen: Deep Neural Network Based Fast Library Generator", International Conference on Electronics, Circuits and Systems, 2019

[14] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016