Microelectronics Journal 116 (2021) 105198

Contents lists available at ScienceDirect

Microelectronics Journal

journal homepage: www.elsevier.com/locate/mejo

Enhanced pin-access prediction and design optimization with machine learning integration

Suren Abazyan [a,*], Vazgen Melikyan [b]

[a] Yerevan State University, Yerevan, Armenia
[b] National Polytechnic University of Armenia, Yerevan, Armenia

ARTICLE INFO

Keywords:
Pin access
Machine learning
Prediction and optimization

ABSTRACT

In daily increasing integrated circuit complexity, standard cell pin accessibility is becoming more significant part of design process. As pin density is increasing, fast and efficient algorithms for pin-access prediction and optimization are needed. However current methods do not ensure the best trade-off between DRV count decrease and tool runtime optimization.

In this paper pin accessibility checking and optimization method is proposed, which is using machine learning algorithms to increase accuracy of pin-access predictions and decrease DRV count. Results show that with using proposed method, DRV count can be decreased by 47%, while having increase in runtime by 23%.

1. Introduction

Accessibility of standard cells' pins is becoming challenging issue for integrated circuits (IC), which is using nanotechnology processes [1,2]. With the technology scaling, transistors sizes are becoming smaller, hence density of standard cell I/O pins is becoming higher. This is rising the need of having more complex layout rules, along with harder constraints.

Based on this standard cell pin accessibility prediction and optimization is becoming more challenging, and more algorithms are being created for solving accessibility issue from early cell design to IC design stages.

From the other hand fast developing machine learning methods are opening new opportunities for more optimal algorithm creation for pin accessibility prediction/optimization and more. Algorithms, which are powered with machine learning methods [3–6], are achieving good results and ensuring more reliability.

Another methods, which is using machine learning algorithms [4] is aiming on fault prediction at the transistor level. This method is using Fast Fourier Transform and Principal Component Analysis for correspondingly getting the fault frequency and getting the most important data with lower dimensions. Also the method is using Convolutional Neural Network for fault classifications. The method has been validated on two circuits like comparator and amplifier. Failure dataset has been extracted for temperature, voltage, noise, delay and current and then the dataset has been feed into training model using TensorFlow. What is more, the mentioned model considers shorts, aging and open circuits. The prediction accuracy for comparator was 98.93% and for amplifier −98.91%

In [5] a method is proposed, which is reducing the number of hidden layers from N to $N/2$ but at the same time keeps the accuracy and has small increase in time complexity. The main idea behind the mentioned method is multiplexing the input and output layers. It includes memories to save weights. Based on multiplexing signals the memory return either first weight or second weight. This way, one node will function as two normal nodes, but in different time sequences, hence there is increase in time complexity. Based on described experiments, the method is saving 42% area with the time increase of $(0.1/N) * T$, where N is the number of layers for normal neural network and T is the total computation time of that network.

For proper DRV prediction and placement optimization, the proposed method is using the power of machine learning algorithms aligned with special placement spacing rule creation tool. Main steps, that are included in proposed method are:

- Pin location extraction
- Pin accessibility prediction
- Spacing rule generation

With this steps, DRV count can be optimized by approximately 47%, but because of some additional post-processing, tool runtime will be increased by about 23%.

* Corresponding author.
E-mail addresses: su.abazyan@gmail.com (S. Abazyan), vazgenm@seua.am (V. Melikyan).

https://doi.org/10.1016/j.mejo.2021.105198
Received 17 February 2021; Received in revised form 12 July 2021; Accepted 4 August 2021
Available online 17 August 2021

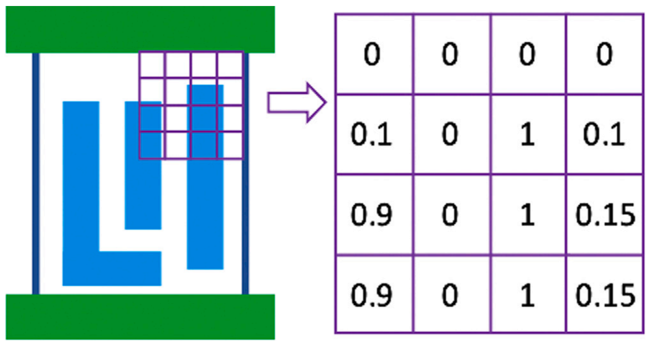

Fig. 1. Design derivation into pixels.

2. Pin access check and optimization

For proper prediction and optimization of design pin accessibility, some methods are focused on pin accessibility of standard library cells during layout design process [1,2,7,8]. Some other methods are using special placement constraints for cells to ensure better accessibility during routing stage.

One of good pin accessibility prediction and optimization methods is Pin Accessibility Prediction and Optimization with Deep Learning-based Pin Pattern Recognition [1]. This method is aiming to solve two main problems:

- Develop deep learning model from number of routed designs with DRC reports using pin pattern as the major feature to predict DRV.
- Use pre-trained model to predict if given legalized placement has DRVs. Avoid pin pattern with bad accessibility during detailed placement.

For reaching to solutions of described problems, the method is using two base deep learning models:

- Pin Pattern Recognition (PPR) — only uses pin pattern as a future
- Design Feature-aware PPR (DFPPR) , which adopts more design features to further enhance the prediction accuracy.

Main feature of the described method is to predict the possibility of second metal layer (M2) shorts. For proper work of the described algorithm, it is using different designs, which are at least in route stage, and their DRC reports. For proper predictions, DRC report needs to be with and without DRVs. After getting number of designs, method is training a model of Pin pattern generation algorithm, which later will be used in Design feature-aware pin pattern prediction algorithm.

After inputting designs into the described method environment, the utility is using the design as graphic and derives the design into pixels. Width and height of each pixel are selected to be as the minimum width of the first metal layer (M1), to prevent pixel from being occupied by two different M2 pins. (Fig. 1)

The second model of described method — DFPPR, is using PPR models, which is being generated from PPR method run, and add design specific options:

Pin access point information: This feature is computed by the ratio of the number of pins to the number of access points in the input pin pattern.

Self-crossing net information: A self-crossing net is caused by the fly lines of two local nets that intersect to each other [9]. A fly line is the straight line connecting the left-bottom access points of two pins. This feature is computed by the ratio of the number of self-crossing nets to the number of local nets in the input pin pattern [10,11].

Unfriendly cell information: With routed designs, we first compute the frequency that each library cell induces M2 shorts [1]. Then, this feature is obtained by the ratio of the sum of the frequencies of the

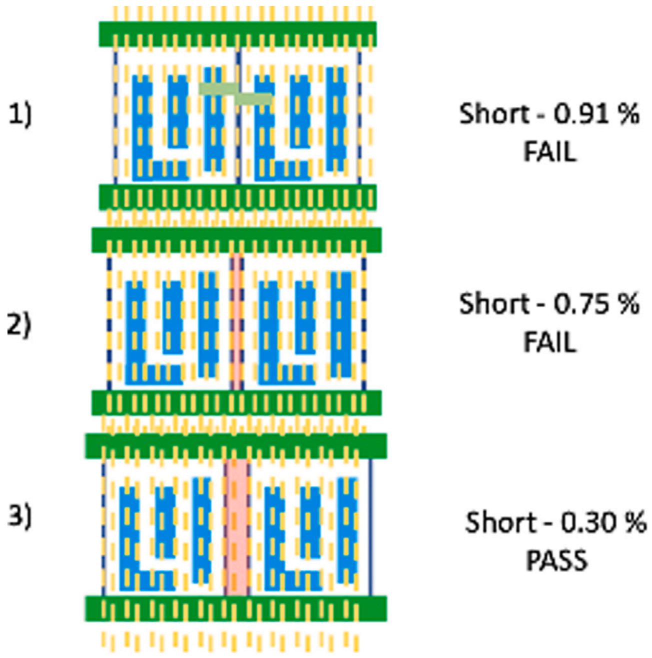

Fig. 2. Spacing rule generation and usage.

cells covered by the pin pattern to the total frequency of all cells. After DFPPR generation, described method is also creating design rules for better reputability design improvement. Particularly, it is running pin accessibility check on new design, and if there are any possible M2 shorts, it is creating spacing rules for under test cells with one unit placement shift. After which process is being replied, unless placement and pin access prediction are satisfying given criteria. (Fig. 2)

However, being power by deep learning algorithm and design feature aware pin pattern generation, described method also has some disadvantages, which need to be taken care.

- Many designs are used for prediction training model generation as deep learning model will have direct connection with training data amount.
- Need to train model for each pin/cell update as after pin/cell updates, locations of them will be changed, hence using previous model will be not accurate.
- Only M2 shorts are being considered, while after having pattern of pins in design, more design rules can be considered.
- Design is being used and inputted for training as graphic, which is making the need of having one more computational iteration while deep learning model generation.

3. Pin-access prediction and design optimization with machine learning integration

For addressing some issues of described method, Enhanced Pin-access prediction and design optimization with machine learning integration method is described. Main idea behind the proposed method is to use machine learning for design rule violation prediction and optimization, while having more predicted and checked design rules [12]. Described algorithm consists of three main steps: pin location extraction from designs, machine learning model training, standard cell vertical and horizontal spacing rule generation for better routability [13]. (Fig. 3)

3.1. Pin location extraction from designs

As pin locations are one of the main criteria for learning model training, extraction of pins must be done with high accuracy and

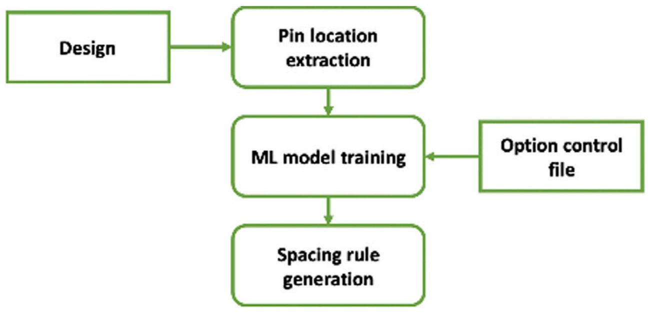

Fig. 3. Main steps for proposed algorithm.

Table 1
Design reports.

Line	Data			
	Reference cell name	Pin name	Pin layer	Coordinates
1	HVT_ANDX1	B	M2	{33.7321.46} {33.7322.27} {34.6322.27} {34.6321.46}
2	LVT_INVX8	A	M1	{13.9330.37} {13.9331.18} {14.5331.18} {14.5330.37}
3	LVT_FDPSQX2	Q	M2	{34.9337.66} {34.9338.47} {35.8338.47} {35.8337.66}

at the same time keep records simple for later processing [14]. For pin location extraction, the method is using placement and routing tool's internal commands aligned with Tool Command Language (TCL) scripting language commands.

The general steps for pin location extraction can be divided into steps:

- Taking all cells from design.
- Finding all pins of each cell.
- Getting the pin metal layers. In case there are multiple metal layers for particular pin, the top layer is being selected.
- Taking the top metal layer coordinated from pin.

Later the method is reporting reference cell name, pin name, pin layer name and coordinates. Reports are being sorted and stored with exact format (Table 1). Sort is being done based on pin location, where first coordinate is pin's low left corner, second coordinate is up left coordinate, third and fourth coordinates are correspondingly low right and up right coordinates.

As design is being used not as graphic, but as a placement and routing tool format design, and, besides from that, outputted file is in text format, pin location extraction process does not require much time and machine resources to be done.

3.2. Machine learning model training

After pin information generation from all the under-test designs, proposed method tool is running machine learning model generation for problematic standard cell prediction.

There are 7 types of design rule violations that are being considered by the time of model training:

- M1 end-of-line spacing rule
- M1 minimum spacing rule
- M1 short rule
- M2 short
- M2 same net minimum spacing rule
- M3 minimum area rule

For method runtime optimization, some of mentioned rules can be switched off while model training.

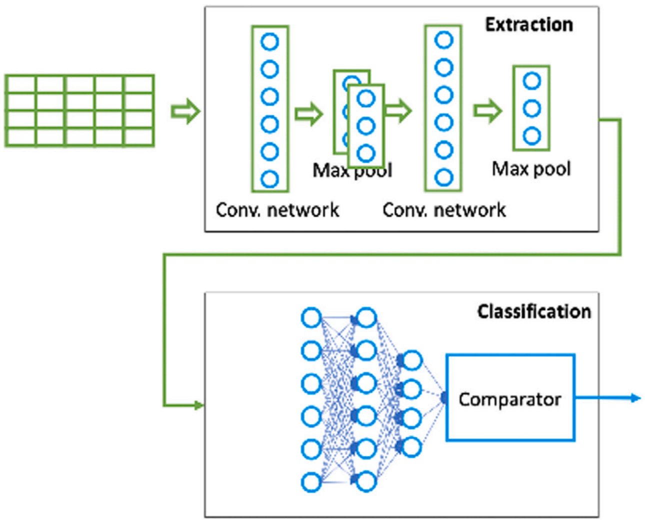

Fig. 4. Machine learning training flow.

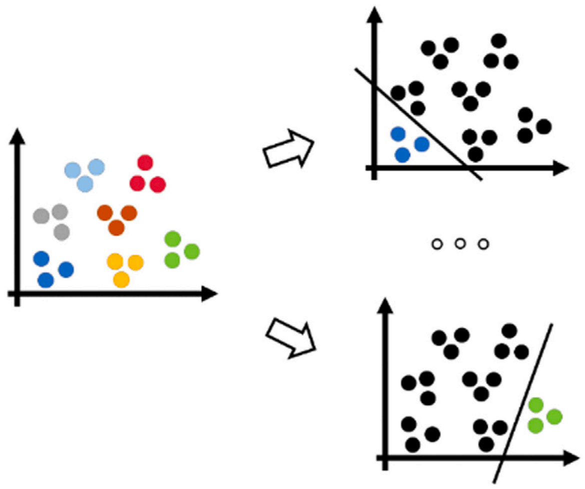

Fig. 5. The Multi-class classification method.

Machine Learning model training flow is shown in (Fig. 4).

The architecture of model training consists of two main blocks: feature extraction and classification

Pin feature extraction process is being done at first stage of machine learning model training. In this process, tool is first checking which rules have been turned on and makes decision which pin information must be kept for later processing.

For pin feature extraction, 4 neural layers are used. First is convolutional layer with $39 \times 39 \times 3$ neurons and 10×10 filters, which is for extracting critical parameters of pins. After that max pooling layer is used for deducting noises, which is $37 \times 37 \times 10$. Next to these layers there are additional convolutional layer with $17 \times 17 \times 20$ neurons and 40×40 filters and max pooling layer with $7 \times 7 \times 40$ filters for more proper extraction of pin patterns. Two convolutional layers are with trainable weight connections, which are being changed also based on selected options of DRV check count. When the feature extraction is complete, extracted features are being inputted into classification process.

In the classification block, multi-class classification technique is used, which allows to categorize the test data into multiple labels. There are mainly two type of this classification technique: one-vs-one and one-vs-all, from which one-vs-all type is selected for solving this problem. (Fig. 5)

Table 2
Comparison results for two methods.

Design	Standard design results			[1] proposed method				Proposed method			
	#DRV	#M2 sh	WL	#DRV	#M2 sh	WL	CPU time (s)	#DRV	#M2 sh	WL	CPU time (s)
D1	1971	61	5 532 549	1007	14	5 606 690	391	934	23	5 259 218	485
D2	572	7	5 518 484	232	0	5 586 467	385	267	4	5 241 467	472
D3	663	10	5 510 817	295	4	5 578 543	387	253	7	5 233 484	480
D4	796	19	5 503 626	288	5	5 573 211	388	312	18	5 230 246	475

For described problem, each class is selected to be design rule violation, which is under consideration. Based on that, model can classify up to 8 labels, from which 7 are DRVs and one is "Clean" label, which is being generated, if the given data line does not contain one of 7 predictable violations.

While using all-vs-one multi-class classification method, one label is "1", and all other labels are considered as "0". This way, training dataset is being divided into up to 8 datasets: one dataset for one class. After that "+1" is used for that particular class from which the dataset is used, and "−1" for other classes.

After training, when data is being inputted to classification algorithm, it is considering as input for all binary classification algorithms, and the class which is the predicted one, will have positive class while all others will have negative class value. (Formula (1))

$$A_X > 0 \tag{1}$$

In Formula (1), X is the predicted class as it has possible class value.

3.3. Spacing rule generation

After model training, the algorithm is generating spacing rules for directly using into design and avoiding the DRVs, that are coming from cell placement. This approach can be used with any design that is using same cell set. In order to prevent DRVs, model is checking the distance between problematic pins and, based on the predicted output, generated appropriate count of spacing grids between cells. For example, if for U_1 and U_2 cells method is predicting M1 min spacing (Fig. 6), it then checks the spacing value required for avoiding the DRV, then writes spacing rule for U_1 and U_2. After that, in design the rule will be included, to prevent DRVs.

4. Experimental results

For demonstrating the proposed method, four types of designs are used with about 35…40% utilizations. Cell sets for all designs are the same, but design logic is different, hence different combination of cells are presented.

Overall dataset for model training here consists of nearly ~ 6000 samples. Then the dataset is being divided into training data and testing data by 70:30 ratio. Later, the second data split is being done after first training process with the same training/testing datasets' dividing ratio. As this process is manual, the tool will wait for the input from the user to specify ratio and to start the second run.

The technology node used for experiments is 14 nm FinFET with 14 nm gate length and 0.074 nm cut poly pitch [15]. In this technology libraries there are about ~ 1000 cells for each threshold voltage type, hence tool can flexibly select cells to use. The proposed method has not been tested on other tech-nodes, however, there is an in inbuild tech map, which can be updated accordingly to support other technology nodes.

Machine used for the runs is with 2.4 GHz CPU, 2 Cores, Linux operating system. For comparison, machine runtimes are captured for standard run, with using [1] presented method and with proposed method. DRVs are checked with signoff tool.

Comparison results are demonstrated in Table 2.

For demonstrating proposed method influence on metal layers different that M2, Table 3 is presented.

In Tables 2 and 3 used abbreviations and explanations are listed below:

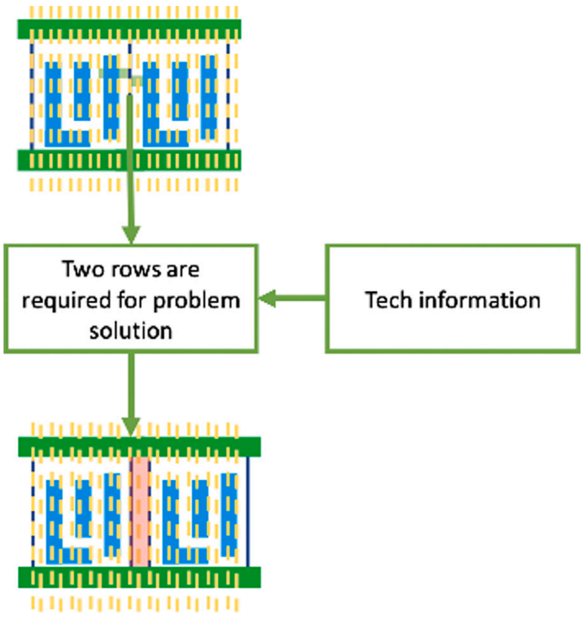

Fig. 6. Generation of spacing rules for proposed method.

Table 3
Method usage on other layers.

Design	Standard design results			Proposed method			
	#DRV	#M1-M3	WL	#DRV	#M1-M3	WL	CPU time (s)
D1	1971	61	5 532 549	934	568	5 259 218	485
D2	572	7	5 518 484	267	103	5 241 467	472
D3	663	10	5 510 817	253	176	5 233 484	480
D4	796	19	5 503 626	457	446	5 230 246	475

- M2 sh — shorts of M2 layer. Same as in [1].
- WL — total wire length
- D1–D4 — 4 different designs
- M1–M3 — All DRV issues that are related to M1, M2, M3 as described in proposed method

Aside from mentioned comparisons, machine learning model accuracy calculation has also been done. For this calculation the (Formula (2)) has been used.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \tag{2}$$

where:

TP — Number of correctly predicted error cases
TN — Number of correctly predicted error-free cases
FP — Cases, when error is predicted, but it was not there
FN — Cases, when error is not predicted, but it is there

Based on the calculation, the proposed model accuracy for each type of design rule is:

- M1 end-of-line spacing rule 51%
- M1 minimum spacing rule 42%
- M1 short rule 61%

- M2 short 64%
- M2 same net minimum spacing rule
- M3 minimum area rule 57% 49%
- Error-free case 68%

The overall accuracy is selected as the average of upper described values and it is approximately 56%.

5. Conclusion

In this paper method for pin accessibility optimization is proposed. Proposed method is using machine learning algorithms for 7 types of DRV predictions and generates appropriate spacing rules for avoiding them. Results show, by using the proposed method, DRV count can be decreased by approximately 47%, however, runtime will be increased by about 23%.

CRediT authorship contribution statement

Suren Abazyan: Data curation, Software, Writing – original draft. **Vazgen Melikyan:** Conceptualization, Validation, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] T.-C. Yu, S.-Y. Fang, H.-S. Chiu, K. Hu, P.H. Tai, C. Shen, H. Sheng, Pin accessibility prediction and optimization with deep learning-based pin pattern recognition*, in: 2019 56th ACM/IEEE Design Automation Conference (DAC), 2019, pp. 1–6.

[2] M. Danigno, P. Butzen, J. Ferreira, A. Oliveira, E. Monteiro, M. Fogaça, R. Reis, Proposal and evaluation of pin access algorithms for detailed routing, in: 2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS), 2019, pp. 602–605.

[3] K. Khalil, O. Eldash, A. Kumar, M. Bayoumi, Economic lstm approach for recurrent neural networks, IEEE Trans. Circuits Syst. II: Express Briefs 66 (2019) 1885–1889.

[4] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, J.S. Rellermeyer, A survey on distributed machine learning, ACM Comput. Surv. 53 (2020) 1–33.

[5] K. Khalil, O. Eldash, A. Kumar, M. Bayoumi, Machine learning-based approach for hardware faults prediction, IEEE Trans. Circuits Syst. I. Regul. Pap. 67 (2020) 3880–3892.

[6] K. Khalil, O. Eldash, A. Kumar, M. Bayoumi, An efficient approach for neural network architecture, in: 2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS), 2018, pp. 745–748.

[7] S.-R. Fang, C.-W. Tai, R.-B. Lin, On benchmarking pin access for nanotechnology standard cells, in: 2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2017, pp. 237–242.

[8] S.-S. Abazyan, N.-E. Mamikonyan, V.-A. Jampoladov, Standard cell pin access checking integration into test design verification, in: Proceedings of NAS RA and NPUA, in: Series of Tech. sci., vol. 73, 2020, pp. 74–81, N. 1.

[9] W. Chan, Y. Du, A. Kahng, S. Nath, K. Samadi, Beol stack-aware routability prediction from placement using data mining techniques, in: 2016 IEEE 34th International Conference on Computer Design (ICCD), 2016, pp. 41–48.

[10] X. Xu, B. Cline, G. Yeric, B. Yu, D. Pan, Self-aligned double patterning aware pin access and standard cell layout co-optimization, IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 34 (2015) 699–712.

[11] M. Danigno, P. Butzen, J. Ferreira, A. Oliveira, E. Monteiro, M. Fogaça, R. Reis, Proposal and evaluation of pin access algorithms for detailed routing, in: 2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS), 2019, pp. 602–605.

[12] B. Yu, J.-R. Gao, D. Ding, X. Zeng, D. Pan, Accurate lithography hotspot detection based on principal component analysis-support vector machine classifier with hierarchical data clustering, J. Micro/Nanolithogr. MEMS MOEMS 14.

[13] W. Chan, P.-H. Ho, A. Kahng, P. Saxena, Routability optimization for industrial designs at sub-14 nm process nodes using machine learning, in: Proceedings of the 2017 ACM on International Symposium on Physical Design.

[14] X. Xu, B. Yu, J.-R. Gao, C.-L. Hsu, D. Pan, Parr: Pin access planning and regular routing for self-aligned double patterning, in: 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), 2015, pp. 1–6.

[15] A.M.G.P. Vazgen Melikyan, Meruzhan. Martirosyan, Meruzhan Martirosyan 14 nm educational design kit: Capabilities, deployment and future, in: Proceedings of the 7th Small Systems Simulation Symposium, 2018, pp. 37–41.