Using Machine Learning Techniques to Determine DDR5 SDRAM I/O Buffer's Slew Rate at Different PVT Variations

1st Lean Karlo S. Tolentino
Department of Electrical Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan
d093010006@student.nsysu.edu.tw

2nd Jhih-Ying Ke
Department of Electrical Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan
ko7569647@vlsi.ee.nsysu.edu.tw

3rd Cheng-Yao Lo
Department of Electrical Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan
loyao0819@vlsi.ee.nsysu.edu.tw

4th Chua-Chin Wang
Department of Electrical Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan
ccwang@ee.nsysu.edu.tw

Abstract—**This study focuses on optimizing specialized input/output (I/O) buffers for DDR5 SDRAMs, specifically examining slew rate variations due to Process, Voltage, and Temperature (PVT) variations. Using advanced techniques like Generalized Regression Neural Network (GRNN) and Genetic Algorithms (GA), the research models slew rate (SR) changes under diverse PVT variations. The dataset taken from simulations has been partitioned into two distinct sections, namely the training set and testing set, with proportions of 80% and 20% respectively. The simulation yields comparative outcomes for the real and anticipated SR. Furthermore, GA exhibits superior performance in forecasting slew rate when compared to GRNN. The regression value yields a significantly closer approximation to unity, thereby achieving a better fit to the dataset. The R2 coefficient of 0.98705 indicates a strong linear relationship between the provided dataset and the line of best fit. Surprisingly, based on the model, a temperature detector circuit was found redundant, resulting in substantial power and area savings. However, voltage variations significantly impacted slew rate. A voltage detector design is therefore recommended. This research advances I/O buffer optimization, offering crucial insights into temperature, voltage, and slew rate dynamics.**

Keywords—I/O buffer, DDR5, GRNN, genetic algorithm, slew rate

I. Introduction

The frequency of the input/output (I/O) driver interface is increasing due to advancements in high-speed technologies, particularly in DDR5 SDRAMs. Additionally, the standards for the quality of output signals have been significantly elevated. For example, the I/O interfaces of DDR5 SDRAMs adhere to DDR5DB01 (DDR5 Data Buffer Specification) established by JEDEC [1], [2]. DDR5 SDRAMs require specific parameters for correct operation, including a VDDIO of 1.1 V, an I/O pad load capacitance (C_L) ranging from 0.4 to 0.9 pF, and the implementation of a duty cycle ratio (DR) of 50±5%.

When designing mixed-voltage I/O buffers, it is imperative to take into account the slew rate (SR) in addition to ensuring voltage level compatibility. The SR variation poses a significant challenge when it comes to the interfacing and communication between legacy process-based systems and advanced technology-based systems like FinFETs. The SR stability is influenced by variations in process, voltage, and temperature (PVT). There have been endeavors undertaken to discern these process, voltage, and temperature (PVT) scenarios inherent in these input/output (I/O) buffers for the purpose of rectifying and calibrating signal integrity [3]-[8], albeit their capability was limited to detecting only three corners, namely, fast-fast (FF), slow-slow (SS), and typical-typical (TT). Furthermore, their methodologies result in an extended settling time and a deteriorated SR owing to the absence of corner points.

Based on the aforementioned reports and statements, no existing models have been developed to characterize the correlation between SR and process, voltage, and temperature (PVT) variations in an input/output (I/O) buffer. In this manuscript, cutting-edge methodologies such as the Generalized Regression Neural Network (GRNN) and Genetic Algorithms (GA) were employed to ascertain and simulate alterations in slew rate amidst a wide range of Process, Voltage, and Temperature (PVT) fluctuations. The dataset acquired from simulations has been segregated into two discrete sections, specifically the training set and testing set, with proportions of 80% and 20% correspondingly. The simulation produces results that can be compared between the actual and expected power. Moreover, the genetic algorithm (GA) demonstrates enhanced efficacy in predicting the rate of change of voltage (slew rate) in comparison to the generalized regression neural network (GRNN). The regression value exhibits a significantly reduced deviation from unity, thereby attaining an enhanced alignment with the dataset. The R^2 coefficient, with a value of 0.98705, signifies a robust linear association between the given dataset and the optimal line of fit.

II. Related Works

A. Generalized Regression Neural Network (GRNN)

Generalized Regression Neural Network also known as kernel regression, is a highly efficient approach for modeling and forecasting [9]. Fig. 1 depicts the architectural framework of the GRNN, incorporating input parameters including the N process corner, P process corner, Voltage (V), and Temperature (T). The output parameter of interest in this context is SR. The input layer employs input variables that are fed into the network and are associated with the neurons sequentially, subsequently transitioning to the subsequent layer. The hidden layer encompasses the completely interconnected layer with a bias, succeeded by the activation function. The output layer obtains the forecasting results [10].

The GRNN is subjected to training by utilizing a collection of training data, which encompasses the input parameters alongside their corresponding output Slew Rate. During the training process, it acquires knowledge regarding the correlation between the input parameters and the output Slew Rate.

Once the GRNN has undergone the training process, it can be effectively utilized for the purpose of predicting the SR based on a novel collection of input parameters. To accomplish this task, it initially computes the Euclidean distance between the newly provided input parameters and every individual data point within the training dataset. It employs a weighted average of the output SR values from the training data points that exhibit the closest proximity to the new input parameters, to forecast the SR for said new input parameters.

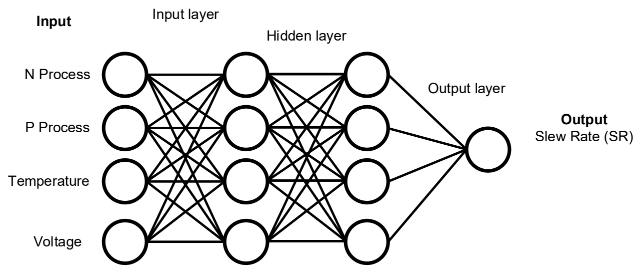

Fig. 1. GRNN architecture

The subsequent citations exhibit a subset of the implementations employed by GRNN. The utilization of GRNN (Generalized Regression Neural Network) is employed for the purpose of forecasting the minimum miscibility pressure of crude oil [11]. It is worth noting that GRNN exhibited superior performance when compared to other pre-existing models utilized for this particular task. Another application of GRNNs is employed in the control of dynamic systems, specifically in the implementation of predictive controllers and estimator controllers [12]. The research work is conducted, wherein the implementation of GRNN is employed for system identification and control of dynamic systems [13]. This approach exhibits reduced training time and improved accuracy in comparison to the conventional backpropagation neural network. Meanwhile, the proposed model employs the GRNN for the seven input variables, specifically the IV characteristic curve, weather condition, and temperature parameters obtained from the testbed solar panel [14]. Lastly, the provided reference presents an empirical dataset derived from meteorological data, wherein solely irradiance and module temperature are utilized as input parameters. The power output of the photovoltaic (PV) panel is then assigned as the designated output [15].

B. Genetic Algorithm (GA)

The Genetic Algorithm (GA) architecture in Fig. 2 is comprised of the subsequent components [16]-[18]:

1) Population: The population can be conceptualized as an assemblage of potential solutions, referred to as candidate solutions, that aim to address the given problem. Every potential solution is encoded as a collection of genetic elements. In this scenario, every gene symbolizes one of the input variables: N Process, P Process, Voltage, or Temperature.

2) Fitness function: The fitness function is responsible for assessing the performance of every potential solution. In this scenario, the fitness function shall assess the Slew Rate of the circuit based on the provided input parameters.

3) Selection operator: The selection operator, being an integral part of the evolutionary algorithm, performs the crucial task of carefully choosing a subset of the population. This carefully chosen subset is then utilized to generate new candidate solutions, ensuring the progression of the algorithm towards optimal outcomes. The selection operator is commonly designed to introduce a bias towards candidate solutions that exhibit higher fitness values.

4) Crossover: The crossover operator effectively merges the genetic material of two carefully chosen candidate solutions, resulting in the generation of a novel candidate solution.

5) Mutation: The mutation operator stochastically alters the genetic elements of a candidate solution.

The GA operates through a series of iterative procedures, encompassing the subsequent actions:

1. Create an ensemble of potential solutions as the initial population.
2. Assess the efficacy of every candidate solution from a fitness standpoint.
3. Elect an assemblage of the populace to serve as the foundation for generating novel potential resolutions.
4. Generate novel candidate solutions by applying the crossover and mutation operators.
5. Substitute the previous populace with the updated populace.
6. Iteratively execute the instructions delineated in steps 2 through 5 until the attainment of a predetermined termination condition.

The termination condition is commonly satisfied when a specific count of iterations has been generated or when a candidate solution with a suitably high fitness value has been discovered. The GA architecture is a robust tool for discovering optimal solutions to intricate problems, such as the task of optimizing the SR of an I/O buffer based on a given set of input parameters.

For example:

1. Suppose we intend to employ a GA for the purpose of identifying the most favorable combination of input parameters (N Process, P Process, Voltage, and Temperature) to achieve the maximum Slew Rate of the I/O buffer.
2. Initially, it is imperative to generate an initial population of candidate solutions to proceed with the subsequent steps. Every potential solution would be encoded as a collection of four genetic elements, with each element symbolizing one of the input parameters.
3. Next, it is imperative to assess the efficacy of every potential solution in terms of its fitness. To accomplish this task, it is imperative that we engage in the

simulation of the circuit utilizing the provided input parameters, subsequently enabling us to compute the SR. The fitness metric of a candidate solution shall be commensurate with the SR exhibited by the circuit.

4. Subsequently, we shall employ the selection operator to meticulously choose a subset from the population, which will serve as the foundation for generating novel candidate solutions. The selection operator is designed to favor candidate solutions that exhibit higher fitness values.

5. Subsequently, the crossover and mutation operators shall be employed to generate novel candidate solutions. The crossover operator shall effectively merge the genetic material of two carefully chosen candidate solutions, thereby engendering a novel candidate solution. The mutation operator shall stochastically modify the genetic elements of a candidate solution.

6. We would subsequently substitute the outdated population with the updated population.

7. The iterative process of steps 2-5 shall be executed until the specified termination condition is satisfied. The termination condition is commonly satisfied when a specific count of iterations has been generated or when a candidate solution with a satisfactorily high fitness value has been discovered.

8. Upon achieving convergence of the genetic algorithm, the optimal configuration of input parameters (N Process, P Process, Voltage, and Temperature) would be determined, thereby maximizing the SR of the I/O buffer.

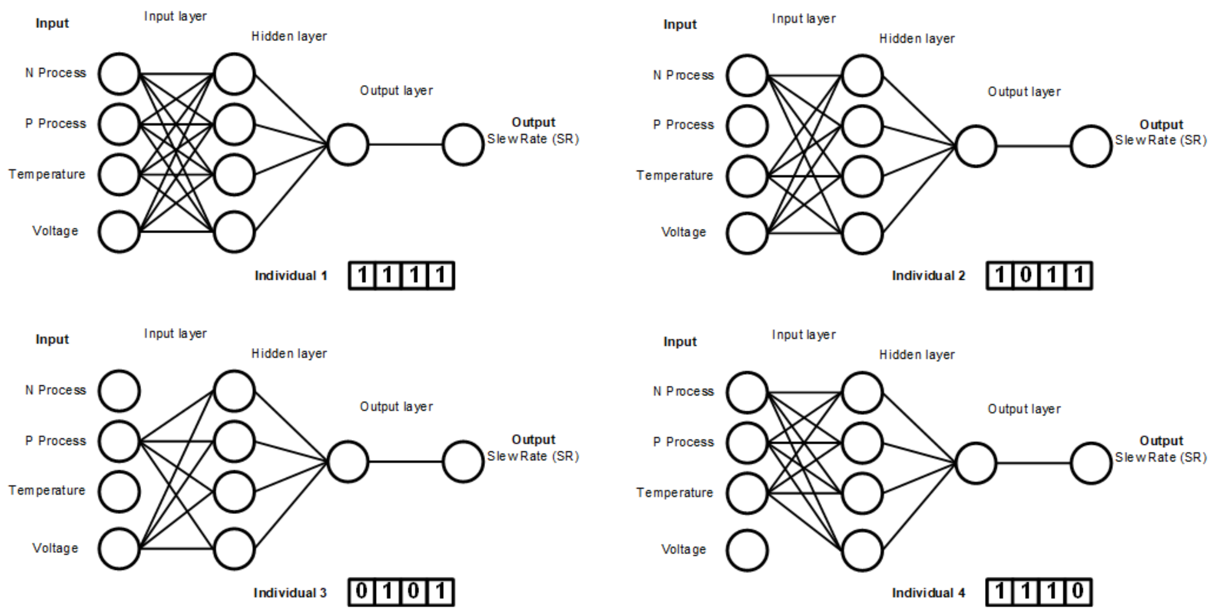

Fig. 2. GA architecture

III. Methods

The depicted framework is illustrated in Fig. 3. The following elucidations delineate each sequential procedure employed in this paper.

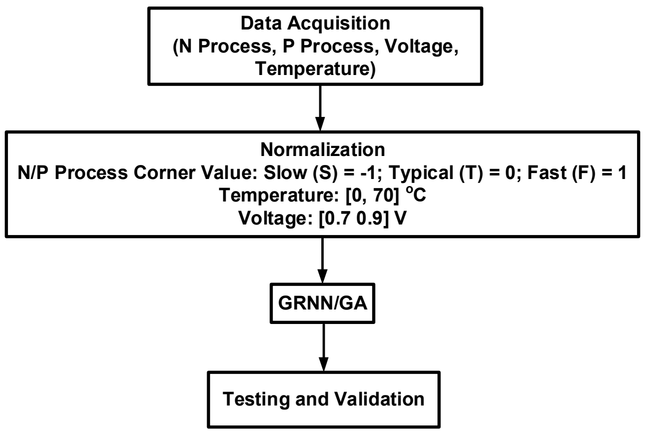
Data Acquisition
(N Process, P Process, Voltage, Temperature)
Normalization
N/P Process Corner Value: Slow (S) = -1; Typical (T) = 0; Fast (F) = 1
Temperature: [0, 70] °C
Voltage: [0.7 0.9] V
GRNN/GA
Testing and Validation

Fig. 3. Proposed methodology

A. Datasets

Fig. 4 shows the output buffer for the DDR4 SDRAM implemented using FinFET process. It has transistors MP1A, MP2, MN2, and MN1A that serves as the buffer at normal mode or no PVT compensation. The other transistors MP1b, MP1c, MN1b, and MN1c are compensating transistors that will be used to regulate the SR under varying PVT conditions. To get the slew rate values at different PVT conditions, MP1A, MP2, MN2, and MN1A are on while MP1b, MP1c, MN1b, and MN1c are off. Using HSPICE, simulations of the output buffer in Fig. 4 consists of N Process, P Process corner values, temperature, voltage, and the corresponding SR of the I/O buffer comprise the dataset. The dataset is partitioned into two distinct sections, specifically the training and testing sets, with proportions of 80% and 20%, correspondingly. Sample dataset in this manuscript is shown in Table I.

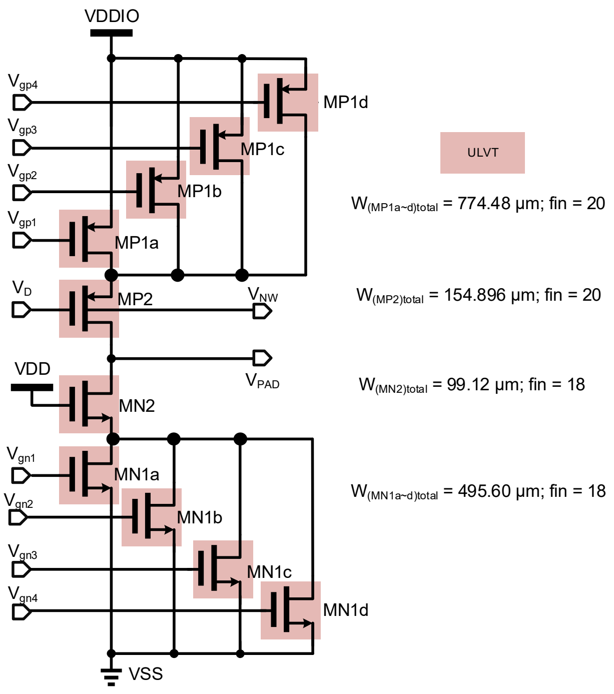
VDDIO
Vgp4
Vgp3
Vgp2
Vgp1
MP1d
MP1c
MP1b
MP1a
ULVT
W(MP1a~d)total = 774.48 µm; fin = 20
VD
MP2
VNW
W(MP2)total = 154.896 µm; fin = 20
VDD
VPAD
MN2
W(MN2)total = 99.12 µm; fin = 18
Vgn1
MN1a
Vgn2
MN1b
W(MN1a~d)total = 495.60 µm; fin = 18
Vgn3
MN1c
Vgn4
MN1d
VSS

Fig. 4. Output Buffer schematic.

TABLE I. SAMPLE DATASET

Process N	Process P	Temperature (oC)	Voltage (V)	Slew Rate (V/ns)
0	0	0	0.70	7.21
0	0	0	0.72	7.61
0	0	0	0.75	7.91
0	0	0	0.78	8.22
-1	-1	20	0.78	8.11
-1	-1	20	0.79	8.42
-1	-1	20	0.80	8.71
-1	-1	20	0.81	9.02
-1	1	70	0.78	10.60
-1	1	70	0.79	11.00
-1	1	70	0.80	11.40
1	-1	40	0.72	10
1	-1	40	0.75	10.40
1	-1	60	0.70	9.49
1	-1	60	0.72	9.88
1	1	20	0.72	10.20
1	1	20	0.75	10.60
1	1	20	0.78	11.00
1	1	20	0.79	11.40

B. *Input Parameters and Data Normalization*

1) N Process Corner (PN): This value affects the NMOS, MN2 and MN1a. It can be slow, fast, or typical. A slow and fast device translates to low and high SR, respectively. Hence, the designation of slow (S), fast (F), and typical (TT) values are -1, 1, and 0, respectively.

2) P Process Corner (PP): This value affects the PMOS, MP2 and MP1a. Same as N Process Corner, it can be slow, fast, or typical. A slow and fast device translates to low and high SR, respectively. Hence, the designation of slow (S), fast (F), and typical (TT) values are -1, 1, and 0, respectively.

3) Temperature (T): This value ranges from 0 to 70 degrees Celsius. In NMOS transistors, a rise in temperature generally results in an augmentation of carrier mobility, thereby causing a corresponding rise in the output current. The augmentation in electrical current frequently exhibits an exponential behavior, adhering to the principles dictated by the fundamental physics governing semiconductor devices. In the context of PMOS transistors, it is worth noting that elevated temperatures have the potential to diminish carrier mobility, thereby resulting in a decline in the output current. The decrease in current typically exhibits a linear relationship with respect to the rise in temperature.

4) Voltage (V): This is the supply or I/O voltage of the output buffer in FinFET process which ranges from 0.7 to 0.9 V. As the voltage increases, the output current for both NMOS and PMOS generally exhibits exponential increase. Nevertheless, the voltage output frequently exhibits a linear growth pattern. In relation to SR voltage variations have the potential to exert an influence. In specific scenarios, an augmentation in voltage could potentially induce a swifter SR in signal, especially if it engenders a proportional increase in the output current.

C. *GRNN Implementation*

The implementation and algorithm of the GRNN is already discussed [14]. The dataset should be appropriately structured, incorporating the following columns: N Process, P Process, Voltage, Temperature, and Slew Rate. The aforementioned dataset shall be employed for the purposes of training and evaluating the GRNN. Partition the data into distinct subsets for training and testing purposes. In this scenario, it is advisable to allocate 80% of the data for training purposes, while reserving the remaining 20% for testing. Utilize the training data to train the GRNN. Perform the evaluation of the trained GRNN by utilizing the designated testing dataset.

D. *GA Implementation*

The prediction of the slew rate (S) based on input parameters such as N Process, P Process, Voltage, and Temperature using genetic programming in MATLAB involves a systematic approach. Firstly, the problem is clearly defined: an equation needs to be found that accurately predicts the slew rate using the specified inputs. A comprehensive dataset containing values for N Process, P Process, Voltage, Temperature, and their corresponding slew rates is gathered; this dataset serves as the foundation for training and validating the genetic programming model.

Using MATLAB as shown in Fig. 5, a fitness function is created to quantify the goodness of fit for candidate equations. Then, the terminal set (variables like N Process, P Process, Voltage, and Temperature) and the function set (mathematical operations like addition, subtraction, multiplication, division) that the genetic programming algorithm can employ to generate candidate equations are defined.

The GA algorithm parameters, including the population size, number of generations, and crossover/mutation operators, are configured. The algorithm is run on the dataset, allowing it to evolve equations over multiple generations, optimizing the fitness function in the process.

The best-evolved equation is evaluated using a validation dataset to ensure its accuracy and reliability. The evolved equation is interpreted to comprehend the relationship between the input parameters and the slew rate. Subsequently, the evolved equation is tested with new data points to validate its performance. Continuous improvement is emphasized: if more data becomes available or enhancements are needed, the algorithm is rerun with the updated dataset, or the terminal and function sets are adjusted for better accuracy and predictive power.

```
function slew_rate_equation = genetic_programming(N_process, P_process, voltage, temperature,
slew_rate, stopping_criterion)

% Initialize the population of random mathematical expressions
population = generate_random_population(input_parameters);

% Evaluate the fitness of each expression
fitness_values = evaluate_fitness(population, training_data);

% Select the best-performing expressions
best_expressions = select_best_expressions(population, fitness_values);

% Mutate and crossover the expressions
new_population = mutate_and_crossover(best_expressions);

% Repeat steps 2-4 until a satisfactory solution is found
while not stopping_criterion_met(fitness_values, stopping_criterion):
    population = new_population;
    fitness_values = evaluate_fitness(population, training_data);
    best_expressions = select_best_expressions(population, fitness_values);
    new_population = mutate_and_crossover(best_expressions);

% Return the best expression
slew_rate_equation = best_expressions(1);

end

% Function to generate a random population of mathematical expressions
function population = generate_random_population(input_parameters)

% Number of expressions in the population
population_size = 100;

% Initialize the population
population = cell(population_size, 1);

% Generate a random mathematical expression for each individual in the population
for i = 1:population_size

    % Generate a random expression tree
    expression_tree = generate_random_expression_tree(input_parameters);

    % Convert the expression tree to a mathematical expression
    expression = expression_to_string(expression_tree);
```

Fig. 5. GAalgorithm.

E. Testing

There is an 80%:20% split between the training set and the test set. At first, we used the training model to simulate the testing datasets. The following additional steps were taken as well: Percentage errors were calculated by comparing the model's predictions to the observed data. To verify the accuracy of the model's projections, new data were collected. Validation of findings via comparison to simulated data, Finally, a simulated GRNN version of the dataset was compared to the GA version.

IV. Results and Discussion

Eqn. (1) shows the SR function in terms of P_P, P_N, V, and T generated using GRNN. However, the percent error of this generated equation is about ±20% which is higher as shown in Fig. 6. Hence, a GRNN model is not suitable for the SR model.

$$SR = 0.7753\,P_N + 0.7751\,P_P - 0.0086T + 13.4423V - 0.0243 \qquad (1)$$

Eqn. (2) shows the SR function generated using GA. As shown in Fig. 7, its resulting R^2 is 0.98705 and RMSE is 0.21455, making the GA model suitable for the SR output parameter.

$$SR = 0.758P_N + 0.758P_P - 0.00758T + 13.9V + 4.23\tanh(V) + 2.07P_P^2\,V^2 + 0.944P_P^2 - 0.944\,P_N P_P V - 5.04 \qquad (2)$$

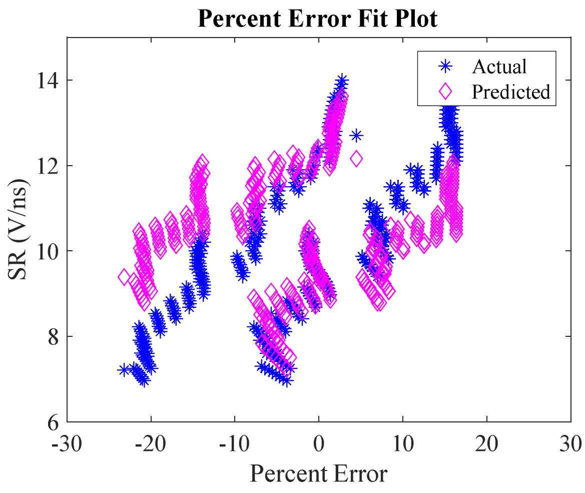

Fig. 6. SR vs. percent error plot. Notice that the percent error ranges from about ±20%

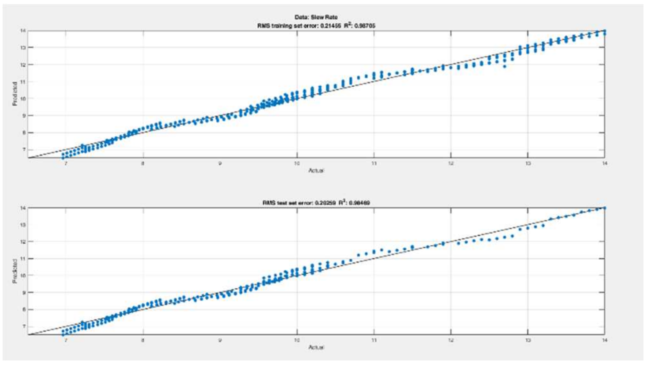

Fig. 7. SR's predicted vs. actual values using GA.

As seen in both Eqn. (1) and (2), the temperature coefficient (0.0086 and 0.00758) is quite low compared to the other three parameters. This indicates that the temperature does not affect the SR greatly. Moreover, it is noted that the voltage coefficient (13.4423 and 13.9) is the highest which means that it greatly affects the SR. Significantly, the GA-generated SR model emphasizes the influence of both N and P process variations on the SR. Furthermore, the GA-generated SR model introduces multi-variable terms that were not initially present in the SR model created by the GRNN. This suggests that the GA has the capacity to identify and incorporate additional factors or variables that contribute to the overall understanding of the system response, enhancing the model's comprehensiveness and predictive capabilities. The use of GA in generating the SR model not only highlights the impact of process variations but also introduces new, previously unidentified multi-variable terms, enriching the overall predictive power of the model.

V. Conclusion

A model for the slew rate (SR) for the DDR5 SDRAM is successfully modeled thanks to GA. The indication of the low temperature coefficient and a high voltage coefficient as seen from the generated equation for SR recommends to carefully design a good voltage sensor/detector for SR regulation and compensation at different PVT variations. It is also good to note that a temperature sensor/detector is not needed which makes the I/O buffer save more power and area.

Acknowledgment

The researchers utilized the EDA tool offered by TSRI (Taiwan Semiconductor Research Institute). Financial support for this project was provided by the NSTC (National Science and Technology Council) of Taiwan under grant numbers NSTC 110-2221-E-110-063-MY2 and 112-2221-E-110-063-MY3.

References

[1] JEDEC, DDR5 Data Buffer Definition (DDR5DB01), December 2021. Accessed on: June 12, 2022. [Online]. Available: https://www.jedec.org/standards-documents/docs/jesd82-521

[2] J.-Y. Ke, L. K. S. Tolentino, C.-Y. Lo, T.-J. Lee, and C.-C. Wang, "A 2.6-GHz I/O buffer for DDR4 & DDR5 SDRAMs in 16-nm FinFET CMOS process," in *Proc. 2023 IEEE Asia Pacific Conference on Circuit and Systems (APCCAS)*, pp. 1-5, Nov. 2023.

[3] T. Qian, L. Chen, X. Li, H. Sun and J. Ni, "A 1.25Gbps Programmable FPGA I/O Buffer with Multi-Standard Support," in *Proc. 2018 IEEE 3rd International Conference on Integrated Circuits and Microsystems (ICICM)*, pp. 362-365, Nov. 2018.

[4] Y. Lin, X. Zou, Z. Zheng, W. Huo, X. Chen and W. Kang, "High-speed, low switching noise and load adaptive output buffer," in *Proc. 2009 12th International Symposium on Integrated Circuits*, pp. 280-282, Dec. 2009.

[5] T.-J. Lee, W.-J. Su, L. K. S. Tolentino and C.-C. Wang, "A 2.5-GHz 2×VDD 16-nm FinFET digital output buffer with slew rate and duty cycle self-adjustment," in *Proc. 2021 IEEE Asia Pacific Conference on Circuit and Systems (APCCAS),* pp. 153-156, Nov. 2021.

[6] C.-C. Wang, L. K. S. Tolentino, T.-J. Lee and W.-J. Su, "Mixed-voltage output buffer," TW Patent I772240B, July 21, 2022.

[7] C.-C. Wang, L. K. S. Tolentino, S.-W. Lu, O. L. J. A. Jose, R. G. B Sangalang, T.-J. Lee, P.-Y. Lou, W.-C. Chang, "A 2xVDD digital output buffer with gate driving stability and non-overlapping signaling control for slew-rate auto-adjustment using 16-nm FinFET CMOS process," *Integration*, vol. 90, pp. 245-260, May 2023.

[8] C.-C. Wang, "Tutorial: design of high-speed nano-scale CMOS mixed-voltage digital I/O buffer with high reliability to PVTL variations," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 2, pp. 562-567, Feb. 2021.

[9] S. Makridakis, E. Spiliotis and V. Assimakopoulus, "Statistical and machine learning forecasting methods: Concerns and ways forward," *Plos one*, vol. 13, no. 3, pp. 1-126, Mar. 2018.

[10] D. Niu, H. Wang, H. Chen and Y. Liang, "The general regression neural network based on the fruit fly optimization algorithm and the data inconsistency rate for transmission line icing prediction," *Energies*, vol. 10, no. 12, pp. 1-20, Dec. 2017.

[11] O. A. Alomair and A. A. Garrouch, "A general regression neural network model offers reliable prediction of CO_2 minimum miscibility pressure," *Journal of Petroleum Exploration and Production Technology*, vol. 6, pp. 351-365, Sep. 2016.

[12] A. J. Al-Mahasneh, S. Anavatti, M. Garatt and M. Pratama, "Applications of general regression neural networks in dynamic systems," *Digital Systems,* pp. 1-21, Nov. 2018.

[13] A. J. Al-Mahasneh, S. G. Anavatti and M. A. Garratt, "Review of applications of generalized regression neural networks in identification and control of dynamic systems," in *Proc. International Conference on Aeronautics, Astronautics and Aviation*, pp. 1-5, May 2018.

[14] R. O. Serfa Juan and J. Kim, "Implementation of generalized regression neural network (GRNN) for solar panel power estimation," in *Proc. 2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 294-299, Oct. 2020.

[15] K. Kerbouche, D. Haddad, A. Rabhi, A. Mellit, M. Hassan Ali and A. El Hajjaji, "A GRNN based algorithm for output power prediction of a PV panel," in *Proc. 2017 International Conference on Artificial Intelligence in Renewable Energetic Systems*, pp. 291-298, 2017.

[16] R. S. Concepcion, L. C. Ilagan, and I. C. Valenzuela, "Optimization of nonlinear temperature gradient on eigenfrequency using genetic algorithm for reinforced concrete bridge structural health, " in *Proc. World Congress on Engineering and Technology; Innovation and its Sustainability*, pp. 141-151, Nov. 2018.

[17] R. Concepcion et al., "Towards the integration of computer vision and applied artificial intelligence in postharvest storage systems: Non-invasive harvested crop monitoring," in *Proc. 2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, pp. 1-6, Nov. 2021.

[18] I.C. Valenzuela, "Application of computational intelligence in plant growth modelling," Doctoral dissertation, 2019.