Fast Full-Chip Parametric Thermal Analysis Based on Enhanced Physics Enforced Neural Networks

Liang Chen[1,2], Jincong Lu[1], Wentian Jin[1] and Sheldon X.-D. Tan[1]
[1]Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521
[2]School of Microelectronics, Shanghai University, Shanghai 201800
cl5555cl@outlook.com, jincong.lu@email.ucr.edu, wjin018@ucr.edu, stan@ece.ucr.edu

2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD) | 979-8-3503-2225-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICCAD57390.2023.10323696

***Abstract*—In this work, we propose a fast full-chip thermal numerical analysis approach based on an enhanced physics-informed neural networks (PINN) framework. The new method, called *ThermPINN*, leverages both PINN-based DNN optimization framework and analytic solutions of simplified thermal problems for solving thermal partial differential equations (PDE). The resulting *ThermPINN* leads to more efficient training speed of DNN networks and more scalability for solving large PDE problems. Specifically, we propose to partially enforce physics laws based on closely related analytic solutions to simpler problems. As a result, we are able to significantly reduce the number of variables in the loss function and easily meet boundary conditions. To consider the impact of various ambient temperatures and effective convection coefficients, which are influenced by different design parameters and run-time conditions, we develop a parameterized thermal analysis technique. This technique enables design space exploration and uncertainty quantification (UQ), which are critical for ensuring the reliability of integrated circuits under various operating conditions. The numerical results on alpha21264 processor show that the proposed *ThermPINN* has 2× speedup and 3× better accuracy over the state-of-the-art thermal simulator, VarSim. The experimental results for 2-D full-chip thermal analysis of 3171 cases show that the proposed parameterized *ThermPINN* considering both training and inference time can achieve a 6× speedup over commercial COMSOL with an average mean absolute error (AE) of 0.47 K. In terms of training time, the proposed parameterized *ThermPINN* is 11× faster than the parameterized plain PINN with similar accuracy. The UQ analysis with 5000 samples for maximum temperature propagated from ambient temperature shows that the parameterized *ThermPINN* and parameterized plain PINN are 113× and 22× faster than COMSOL, respectively.**

I. Introduction

As technology advances, today's high-performance multi/many core microprocessors are becoming more thermally constrained due to steadily increasing power densities [1]. To maintain reliability, many system-level thermal/power regulation techniques such as clock gating, power gating, dynamic voltage and frequency scaling (DVFS) and task migration have been proposed in the past [2]–[4]. One critical aspect of those methods mentioned above is correctly estimating the full-chip temperature profile to properly guide the dynamic thermal management schemes [5], [6]. Further more, to precisely predict the thermal impacts in VLSI physical design, an efficient and accurate thermal analysis is critical in the temperature-aware design flow.

The traditional thermal simulation algorithms are intensively studied to estimate the temperature profile for the full chip based on power density map, which consist of two categories, numerical methods and analytical methods. Numerical methods, such as finite element method (FEM) [7] and finite difference method (FDM) [8]–[10], discretize the full chip structure and solve a sparse linear system to get accurate thermal maps with fine mesh. However, millions of unknowns generated by the mesh are needed to be solved, which is very time-consuming for the large-scale chip with high integration density. Due to the rectangular shaped structure of the chip, under simplified boundary conditions, several analytical methods, such as Green's function method [11], [12] and separation of variables method [13], have been developed to compute the thermal maps accurately and efficiently because they avoid meshing the chip to solve a linear system.

However, thermal conductivity and leakage power are functions of temperature [14], [15], which causes the high nonlinearity of thermal equations and remains a challenge for traditional algorithms. Ignoring the temperature dependence can lead to significant error in thermal simulation. The thermal conductivity, leakage power and chip temperature are mutually dependent. From an analysis perspective, there is a feedback loop between them. This feedback loop repeats until thermal maps, thermal conductivity and leakage power converge. The feedback loop runs numerical thermal simulation several times, which is computationally intensive. In addition, it is difficult for analytical methods to solve the nonlinear heat conduction with temperature-dependent parameters. Therefore, fast thermal simulation considering temperature-dependent parameters is highly desired for accurate temperature-aware design.

Recently, scientific machine learning (SciML) has become a promising method to solve high dimensional and nonlinear partial differential equations (PDEs) by using deep neural networks based on the universal approximation theory [16]–[18]. There are two kinds of methods: data-driven methods and physics-informed neural networks (PINN). Data-driven methods only use the labeled data to train the machine learning (ML) model [18]. However, it is very difficult or expensive to generate the labeled data, which restricts the practical application of data-driven methods. To solve this problem, PINN was proposed to add the domain knowledge and physics law into loss function and solve nonlinear PDEs by using unsupervised learning without labeled data [16], [17]. However, the plain PINN requires a huge time to train the ML model to achieve a specific accuracy level, especially for large engineering problems.

Furthermore, boundary and initial conditions in PINN are difficult to be enforced using loss functions as they need to be satisfied completely for a valid solution. Governing equation loss and boundary/initial conditions loss are not in the same order of magnitude so that it is hard to both exactly solve the governing equations and satisfy boundary/initial conditions, which may lead to low accuracy of the solution and increase

This research was performed at UC Riverside when Dr. Liang Chen was a post-doc at UC Riverside. The work is supported in part by NSF grant under No.CCF-2007135, and in part by NSF grant under No. CCF-2113928.

the computational cost [19], [20].

In this work, we develop an efficient PINN-based unsupervised learning approach, called *ThermPINN*, to solve the differential equations for full-chip thermal analysis of VLSI chips. Our new contributions are as follows:

- First, to mitigate the long training issues of the plain PINN approach, we propose to partially enforce some physics laws based on the discrete cosine series. More importantly, the boundary conditions are implicitly enforced in this way. As a result, the number of variables in the loss function is reduced dramatically. Furthermore, due to the exponential convergence speed of the discrete cosine series, the loss function can converge rapidly by the back-propagation algorithm. Therefore, the training speed of proposed *ThermPINN* is improved significantly and can achieve 150× speedup over the plain PINN.
- Second, to consider the different design parameters and running conditions, we further parameterize ambient temperature and effective convection coefficient in *ThermPINN*, which leads to the parameterized *ThermPINN*. Then, the surrogate model obtained by the parameterized *ThermPINN* can be used for fast design space and running condition exploration and for performing uncertainty quantification (UQ).
- Third, the numerical results on alpha21264 processor show that the proposed *ThermPINN* has 2× speedup and 3× better accuracy over the state-of-the-art VarSim. The experimental results for 2-D full-chip thermal analysis of 3171 cases show that the proposed parameterized *ThermPINN* considering both training and inference time can achieve a 6× speedup over commercial COMSOL with an average Mean absolute error (AE) of 0.47 K. In terms of training time, the proposed parameterized *ThermPINN* is 11× faster than the parameterized plain PINN with similar accuracy. The UQ analysis with 5000 samples for maximum temperature propagated from ambient temperature shows that the parameterized *ThermPINN* and parameterized plain PINN are 113× and 22× faster than COMSOL, respectively.

The paper is organized as follows: Section II reviews the traditional and ML-based methods for thermal map estimation of the full chip. Section III introduces the thermal modeling for full chip and temperature dependence of thermal conductivity and leakage power. Inspired by separation of variables method, we propose a novel *ThermPINN* architecture with parameterized ambient temperature and effective convection coefficient in Section IV. Experimental results are presented in Section V. Finally, Section VI concludes this paper.

II. Relevant work

Several traditional thermal algorithms have been developed to obtain full-chip thermal maps for early design stages. Goplen *et al.* utilized the finite element method (FEM) to directly discretize temperature field for thermal-aware placements [7]. FEM method is widely used in commercial software, such as COMSOL and ANSYS. Finite difference method (FDM) was also employed to discretize the differential operator to capture temperature distribution with power density [8]. Li *et al.* presented a multigrid approach to speed up the FDM for a full-chip thermal analysis [9]. Based on FDM, Huang *et al.* proposed a compact thermal model (HotSpot) to perform thermal simulation of full chip with desired levels of abstraction [10]. Above numerical methods require discretization of the full chip, which is very time-consuming. To avoid volume meshing procedure of the chip, several analytical methods are proposed to perform fast thermal analysis for rectangle-shaped chips. Zhan *et al.* applied Green's function method to obtain thermal map efficiently by using the convolution of Green's function and power density map [11]. Then, separation of variables method was developed to represent the full chip thermal maps with truncated cosine series [13].

Several works were conducted to consider temperature-dependent thermal conductivity and leakage power. Yang *et al.* showed that temperature dependence of thermal conductivity can lead to an increase of 5 K in peak temperature [14]. Liu *et al.* employed an iterative way to calculate the temperature distribution considering the impact of temperature dependent leakage power [15]. Sultan *et al.* proposed modified Green's function to consider both temperature-dependent thermal conductivity and leakage power in full chip thermal simulation [21]. However, such an analytical method approximated the leakage power with a linear function of temperature.

Machine learning (ML)-based methods have shown great potential to solve the high dimensional and nonlinear PDEs with fast speed and high accuracy. ML-based methods applied in full chip thermal simulation are divided into two categories, data-driven methods and unsupervised learning methods. Data-driven methods require labeled data to train the neural networks. Sheriff *et al.* applied Long-Short-Term-Memory (LSTM) network to capture dynamic temperature profiles measured by infrared thermal imaging setup [22]. Jin *et al.* took the performance metrics as input to generate full-chip thermal maps by using generative adversarial networks [23]. This kind of work collects data from the real chip and is not suitable for thermal predictions in the design process. Based on the CTM, Juan *et al.* proposed a learning-based autoregressive model to estimate the thermal map of the target chip [24]. However, this model needs to be retrained when the floorplan of the target chip changes significantly. To provide a transferable ML model, Wen *et al.* divided the whole chip into several small regions (tiles) where DNN-based solvers are applied [25]. Chhabria *et al.* [26] performed thermal analysis by using convolutional encoder-decoder networks. Chen *et al.* [27] applied graph convolution networks (GCN) to represent the compact thermal model and obtained thermal maps for chips with different sizes by using the transferability of GCN. However, those data-driven methods require a database with ground truth to train the model, which restricts their applications in real problems as data generation may not be available or be very expensive to generate.

Conversely, unsupervised learning methods, especially the recently proposed *PINN* concept [16], have emerged as a solution to address the challenge of data generation. The essence of the PINN approach involves incorporating essential physics principles, including governing equations, boundary conditions, and initial conditions, into the loss functions. These augmented loss functions are then harnessed in back-propagation for training the neural network devoid of a conventional dataset. The efficacy of the PINN paradigm is well demonstrated in tackling electrostatics problems [28]. Additionally, Cai *et al.* have effectively applied the PINN methodology to a diverse array of prototype heat transfer scenarios, encompassing forced and mixed convection, as well as the two-phase Stefan problem [29]. Notably, the Central ML Team at ANSYS delves into the exploration of heat transfer

in electronic chips through PINN [30]. NVIDIA contributes to this trajectory by introducing a PINN-based PDE solver, named Modulus, that aids in the simulation and optimization of heat sink designs employing parameterized techniques [31]. Recently, Liu *et al.* proposed a DeepOHeat framework to perform a fast thermal analysis of 3D IC from the function space of several PDE configurations to the function space of the solution [32]. However, these endeavors predominantly center around resolving the heat conduction equation with constant parameters, and they encounter challenges associated with the gradual training convergence of conventional PINN, particularly when confronted with larger-scale problems.

Furthermore, boundary and initial conditions in PINN are difficult to be enforced using loss functions as the boundary (BC)/initial conditions (IC) should be satisfied absolutely for a valid solution. To address this issue, Sun *et al.* proposed to use penalty coefficients λ to differentiate the magnitudes of governing equation loss and boundary/initial conditions loss [33]. However, such penalty coefficient method still cannot ensure the complete IC/BC compliance. Then, hard constraints concept was proposed to convert multiple losses to one single loss by using the augmented Lagrangian method [19], [20]. But this method still needs to explicitly consider the boundary/initial conditions in loss functions. In this paper, we use discrete cosine neural networks (DCN) to represent the solution. The new method we proposed only considers governing equation in loss function since DCN automatically enforces the boundary conditions. At the same time, DCN has a better convergence speed than the fully-connected neural networks (FCN), which significantly reduces the training time of the PINN method.

III. Problem Formulation

A. Thermal modeling for a full chip

Functional units on the chip generate lots of heat, which leads to nonuniform temperature distributions. The steady-state temperature profile of the full chip is governed by thermal equation [11], [13]

$$\nabla \cdot (-\kappa(\mathbf{r})\nabla T(\mathbf{r})) = g(\mathbf{r}) \tag{1}$$

where $\mathbf{r}$ represents the position (x, y, z), $T(\mathbf{r})$ is the temperature, $g(\mathbf{r})$ is the power density and $\kappa(\mathbf{r})$ is the thermal conductivity. For boundary conditions, the bottom surface of the chip is set as convection boundary condition with the convection coefficient h and other surfaces are set as adiabatic boundary conditions, which can be expressed as

$$\begin{aligned} \left.\frac{\partial T}{\partial x}\right|_{x=0,a} = \left.\frac{\partial T}{\partial y}\right|_{y=0,b} = \left.\frac{\partial T}{\partial z}\right|_{z=0} = 0 \\ \kappa \left.\frac{\partial T}{\partial z}\right|_{z=c} = h(T|_{z=c} - T_0) \end{aligned} \tag{2}$$

where T_0 is the ambient temperature. The chip system consists of several layers, such as thermal interface material (TIM) layer, TSV layer, chip layer, etc. We can reduce the 3D problem to 2D problem because of the multi-layer structure. The reduced thermal equations are written as [34], [35]

$$\begin{aligned} &\frac{\partial}{\partial x}\left(\kappa(\mathbf{r})\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(\kappa(\mathbf{r})\frac{\partial T}{\partial y}\right) - m^2(T - T_0) = g(\mathbf{r}) \\ &\text{BC:} \left.\frac{\partial T}{\partial x}\right|_{x=0,a} = \left.\frac{\partial T}{\partial y}\right|_{y=0,b} = 0 \end{aligned} \tag{3}$$

where m is the effective convection coefficient, which describes the vertical heat transfer behavior.

Separation of variables method is an efficient and accurate method to solve the thermal equation (3) with constant thermal conductivity and power density [35]. Based on boundary conditions, the temperature distribution can be expressed as

$$T(x, y) = \sum_{q=0}^{Q}\sum_{p=0}^{P} C_{pq} \cos\left(\frac{p\pi}{a}x\right)\cos\left(\frac{q\pi}{b}y\right) \tag{4}$$

where P and Q represent the truncated number of series along x- and y- directions, respectively, a and b are the widths of the chip along x- and y- directions, respectively, and C_{pq} is the coefficients to be determined by governing equation of (3). If the thermal conductivity and power density are constant, the C_{pq} can be calculated analytically by using the formula in [35]. However, in the real application, thermal conductivity and power density are temperature-dependent, which is illustrated in the next subsection. The separation of variables method fails to solve this nonlinear thermal equation with temperature-dependent parameters.

B. Temperature-dependent parameters

Several parameters of the chip are impacted by temperature, such as thermal conductivity and leakage power. Thermal conductivity $\kappa(\mathbf{r})$ is position dependent in the chip. Also, thermal conductivity is a function of temperature, which is given by [14]

$$\kappa(T) = \kappa_0\left(\frac{T}{300}\right)^{-\eta} \tag{5}$$

where κ_0 is the thermal conductivity at temperature 300 K and η is a constant for the specific material.

Leakage power has a strong dependence on temperature and is approximated by a linear model [21]

$$P_{\text{leak}}(T) = P_{\text{leak,0}}(1 + \beta\Delta T) \tag{6}$$

where $P_{\text{leak,0}}$ is the leakage power at 300 K, β is the coefficient, and $\Delta T = T - T_0$. However, this linear model only works for a small range of temperature (such as 318-343 K). If the chip works beyond this range of temperature, a piece-wise linear model is employed to describe the leakage power [15].

IV. The proposed ThermPINN for full-chip thermal analysis

A. The proposed ThermPINN

Due to the temperature-dependent thermal conductivity and power density, we can not directly obtain C_{pq} analytically for the thermal equation (3). To overcome this problem, we propose a discrete cosine neural networks (DCN) to represent (4), as shown in Fig. 1(a). Based on automatic differential of DCN, we can build the governing equation of (3) and use unsupervised learning method to find C_{pq}, which is similar to recently proposed PINN [16]. The separation of spatial variables x and y forms two cosine vectors. Then $Q \times P$ matrix is obtained by the matrix product of two cosine vectors. The temperature value is calculated by the inner product of $Q \times P$ matrix and the coefficients C_{pq} matrix. Such network is called the DCN.

To consider the variations of ambient temperature T_0 and effective convection coefficient m, we parameterize these two variables, as shown in Fig. 1(a). The coefficients C_{pq} depend

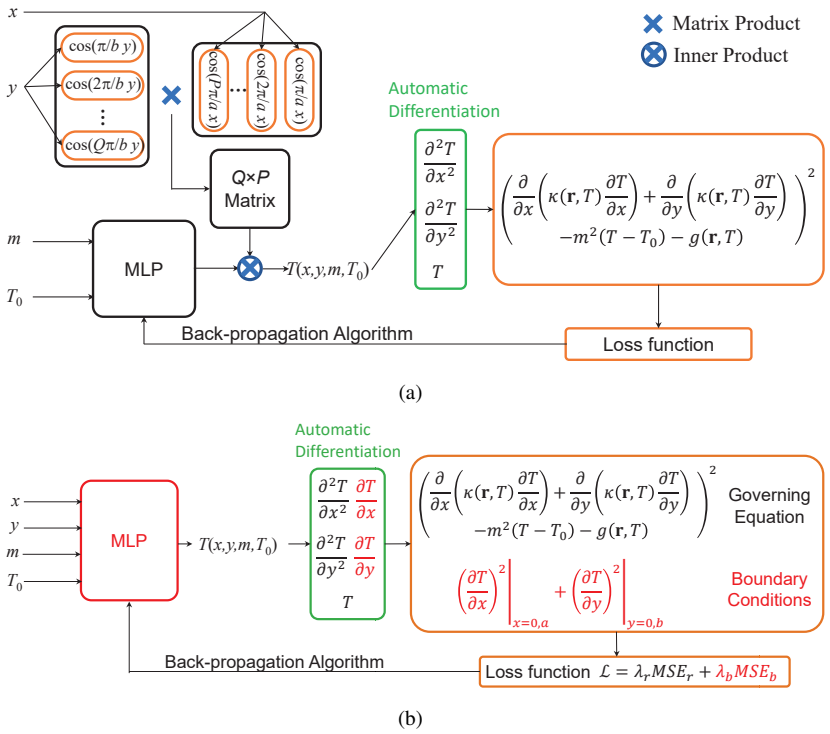

Fig. 1. Overall frameworks of (a) the proposed *ThermPINN* and (b) the plain PINN with parameterized ambient temperature T_0 and effective convection coefficient m.

on the two variables. We use multi-layer perception (MLP) to build a connection between C_{pq} and two variables. Therefore, we have

$$T(x,y,m,T_0) = \sum_{q=0}^{Q}\sum_{p=0}^{P} \text{MLP}(m,T_0)\cos\left(\frac{p\pi}{a}x\right)\cos\left(\frac{q\pi}{b}y\right) \tag{7}$$

Then, we use back-propagation algorithm to learn the parameters of MLP.

Unsupervised learning method trains the parameters of networks without using labeled data and adds the physics law into the loss function. Thermal equations (3) consist of governing equation and boundary conditions. The expression (7) automatically satisfies the boundary conditions. Therefore, we only need to build governing equation based on (7) and the loss function then can be expressed as

$$\mathcal{L}_{ThermPINN} = MSE_r = \frac{1}{N_r}\sum_{i=1}^{N_r}\left|\frac{\partial}{\partial x}\left(\kappa(T)\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(\kappa(T)\frac{\partial T}{\partial y}\right) - m^2(T-T_0) - g(T)\Big|_{(x_i,y_i,m_i,T_{0i})}\right|^2 \tag{8}$$

where MSE_r is the mean-square errors of governing equation, N_r is the number of sampling points, which are randomly selected in the combinational domains of space, ambient temperature, and effective convection coefficient.

B. The plain PINN

As shown in Fig. 1(b), the plain PINN directly uses MLP to model the temperature distribution, which is given by

$$T(x,y,m,T_0) = \text{MLP}(x,y,m,T_0) \tag{9}$$

Then, plain PINN adds all physics laws into loss function which is represented by

$$\mathcal{L}_{\text{PINN}} = \lambda_r MSE_r + \lambda_b MSE_b \tag{10}$$

where

$$MSE_b = \frac{1}{N_b}\sum_{i=1}^{N_b}\left[\left(\frac{\partial T}{\partial x}\Big|_{(x=0,a)}\right)^2 + \left(\frac{\partial T}{\partial y}\Big|_{(y=0,b)}\right)^2\right] \tag{11}$$

where MSE_b is the mean-square error of boundary conditions, λ_r and λ_b are the penalty coefficients, N_b is the numbers of sampling points, which are selected randomly in the combinational domains of space, ambient temperature and effective convection coefficient at the boundaries.

C. Comparison of proposed ThermPINN and plain PINN

As shown in Fig. 1, compared with plain PINN, the proposed *ThermPINN* only needs to consider governing equation constraints and does not require sampling points on the boundaries. The reduced input data leads to faster training speed. We only need a small number of series to obtain accurate temperature distribution. The number of learning parameters is significantly less than plain PINN. Therefore, the training speed of *ThermPINN* is much faster than that of plain PINN, which is demonstrated in Section V. In addition, the proposed *ThermPINN* can avoid the impact of penalty coefficients λ_r and λ_b on the accuracy of the physics-informed methods since the loss function of the proposed *ThermPINN* only consists of governing equations. It is very difficult for the plain PINN to find proper penalty coefficients to achieve good accuracy.

V. Experimental Results and Discussions

In this section, we demonstrate the accuracy and speed of the proposed *ThermPINN* method by using two floorplans

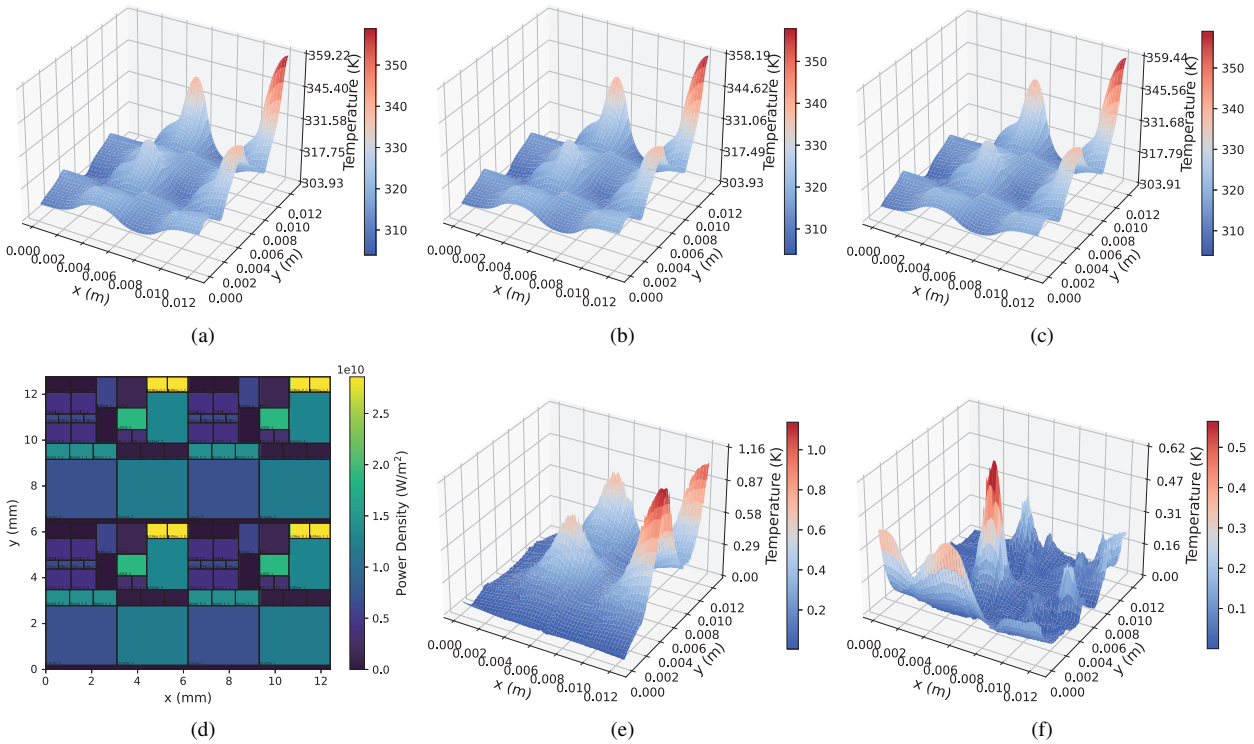

Fig. 2. Thermal maps of Alpha21264 with four cores by using (a) commercial COMSOL, (b) proposed *ThermPINN* with 60×60 eigenvalues, and (c) plain PINN. (d) Power density map of Alpha21264 with four cores. (e) The absolute error between *ThermPINN* and COMSOL. (f) The absolute error between plain PINN and COMSOL

TABLE I
ACCURACY AND SPEED COMPARISON FOR ALPHA21264 PROCESSOR

Metrics	*ThermPINN*						plain PINN [29]	VarSim [21]	COMSOL
	10×10	20×20	30×30	40×40	50×50	60×60			
Mean AE (K)	1.93	0.97	0.55	0.41	0.30	0.23	0.05	0.61	Ground truth
Max Deviation (K)	9.92	3.30	1.68	1.16	1.12	1.03	0.22	0.55	
Percent Mean AE	3.36 %	1.64%	0.93%	0.69%	0.51%	0.39%	**0.08%**	2%	
Training Speed	5.44 s	8.36 s	12.04 s	**33.78 s**	80.24 s	160.55 s	5076 s	Unkown (offline)	–
Inference Speed	0.61 ms	0.61 ms	0.64 ms	**0.59 ms**	0.65 ms	0.67 ms	1.04 ms	1.3 ms (online)	5 s

of full chip. One is the Alpha21264 processor with four CPU cores. Another more complicated floorplan is randomly generated, which consists of many function units with different sizes and power densities. To show the advantage of the proposed *ThermPINN*, we further apply the surrogate model for the uncertainty quantification analysis, which is a hyper-dimensional problem.

The proposed *ThermPINN* and plain PINN are implemented in the PyTorch platform. All programs, including training, inference and generating ground truth, are run on a Linux server with Xeon E5 2.2 GHz CPU and NVIDIA Titan RTX GPU.

A. Accuracy and speedup of ThermPINN without parameterization

For ease of analysis, we first implement the individual *ThermPINN* without parameterization to obtain temperature profile of Alpha21264 processor, as shown in Fig. 2(d). The training data has 10000 sampling points in the interior domain of the chip. For plain PINN, there are extra 500 sampling points for boundary conditions. Both thermal conductivity and leakage power are temperature-dependent, which is a nonlinear problem and can not be solved by the separation of variables method. Therefore, we propose the *ThermPINN* method to calculate the cosine coefficients C_{pq} of equation (4). The starting learning rate of the Adam optimizer for *ThermPINN* is 0.1.

Based on separation of variables methods, the number of truncated cosine series has a great impact on the accuracy and speed of the cosine series solution. Then, we explore the number $P \times Q$ of cosine series to trade off the accuracy and speed of the proposed *ThermPINN*. As illustrated in Table I, the Mean absolute error (AE) is reduced from 1.93 K to 0.23 K and the training time increases from 5.44 s to 160.55 s when the number of cosine series increases from 10×10 to 60×60. "Mean AE" represents the average absolute error $\sum_{i=1}^{10000} |T_{\text{predict,i}} - T_{\text{truth,i}}|/10000$ for one thermal map. As we can see, 40×40 is a proper number of cosine series, which has relatively less training time of 33.78 s with acceptable accuracy of Max Deviation 1.16 K. "Max Deviation" is used to represent the difference of maximum temperature between two results. As the number of cosine series continues to increase,

the accuracy is improved slightly.

We also implement plain PINN to solve the same nonlinear thermal equations. The MLP used in the plain PINN has 6 layers and 128 neurons for each layer. The inputs of MLP are position x and y. Its output is the temperature T. The starting learning rate of the Adam optimizer is 10^{-3}. Commercial COMSOL tool is employed to generate ground truth, as shown in Fig. 2(a). "Percent Mean AE" represents the ratio of Mean AE to maximum temperature rise (59 K). Based on ground truth, The mean AE and Max Deviation of plain PINN is 0.05 K and 0.22 K, respectively, which are better than that of proposed *ThermPINN* with 60×60 eigenvalues. Fig. 2(b) and Fig. 2(c) are the temperature distributions for *ThermPINN* and plain PINN, respectively. The corresponding absolute errors for *ThermPINN* and plain PINN are shown in Fig. 2(e) and Fig. 2(f), respectively. As we can see, they are in the same accuracy level, although the plain PINN has better accuracy than the *ThermPINN*. With enough accuracy, training and inference speed of *ThermPINN* (40×40) are 150× and 1.76× faster than that of plain PINN, respectively, as shown in Table I. The results show that the proposed *ThermPINN* reduces training time significantly and can be of more practical use in EDA fields by comparison to plain PINN.

Compared with the plain PINN, the *ThermPINN* has fewer parameters to be learned, which reduces both training and inference time. In addition, the cosine series used in *ThermPINN* automatically satisfy the boundary conditions. Therefore, *ThermPINN* does not require sampling points on the boundaries and just needs to learn the parameter to satisfy the constraints of governing equation. *ThermPINN* does not require the penalty coefficients to balance boundary loss and governing equation loss. The coefficients C_{pq} of the cosine series converge exponentially so that the main value of C_{pq} that *ThermPINN* learns can change the loss function rapidly, which leads to a fast convergence speed of *ThermPINN*'s loss function.

We also use the floorplan of alpha21264 processor to compare the proposed *ThermPINN* with the recently proposed VarSim, as shown in Table I. The 0.23 K Mean AE of the Proposed *ThermPINN* is smaller than 0.61 K of VarSim, although our example with four CPU cores is more complicated than the alpha21264 processor in VarSim. What's more, the inference speed of the proposed *ThermPINN* achieves 2× speedup over the VarSim. This is because the VarSim needs to double the size of the chip to consider the adiabatic boundary conditions for edges and corners.

We note that we did not compare *ThermPINN* with fast thermal simulator *HotSpot* [10] because *HotSpot* can not directly solve the nonlinear heat conduction equation with temperature-dependent parameters and requires many iterative calculations to find the final solution, which is slower than commercial software COMSOL. We did not compare the *ThermPINN* with the recently proposed DNN based method such as CNN-based method [26] and GAN-based method [23] as they are supervised learning methods.

B. Accuracy and speedup of ThermPINN with parameterization

To explore various running environments and design spaces, we parameterize the ambient temperature T_0 and effective convection coefficient m in the *ThermPINN*, which is notoriously difficult for traditional numerical methods because of high dimensionality. The ambient temperature is set in the range of 300∼310 K and effective convection coefficient ranges between 1500 and 1650. In the parameterized *ThermPINN*, a MLP with 5 layers and [2, 200, 400, 800, 1200, 1600] neurons is employed to build connection between parameters (T_0, m) and coefficients (C_{pq}). The number of cosine series is set as 40×40. The inputs data are 60000 sampling points, which consist of 10000 sampling points in the space domain and 6 combinations of T_0 and m, which are (T_0=300,m=1500), (T_0=330,m=1500), (T_0=300, m=1650), (T_0=330,m=1650) and others.

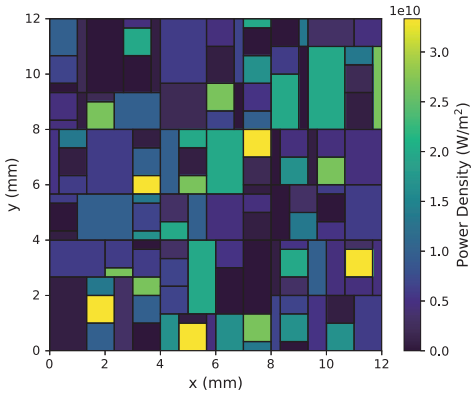

Fig. 3. A more complicated power density map.

TABLE II
ACCURACY AND SPEED COMPARISON FOR 3171 CASES

Metrics	parameterized *ThermPINN*	parameterized plain PINN	COMSOL
Max Mean AE (K)	1.21	0.39	Ground truth
Min Mean AE (K)	0.15	0.09	
Mean Mean AE (K)	0.47	0.17	
Max Deviation (K)	3.60	1.33	
Percent Mean Mean AE	0.66 %	0.24%	
Training Speed	2122 s	7.2 hours	–
Inference Speed	119 s	474 s	14008 s

To validate the accuracy and speed of the proposed parameterized *ThermPINN*, we generate a more complicated floorplan, as shown in Fig. 3. We use COMSOL to generate ground truth by sweeping parameters T_0 and m. Their intervals are 0.5 K and 1, respectively. The total number of cases are 3171 (21x151). For each case, we can obtain one thermal map of the chip by using parameterized *ThermPINN*. "Mean AE" represents mean absolute error for one thermal map. "Max Mean AE", "Min Mean AE", and "Mean Mean AE" are the maximum, minimum, and mean Mean AE for 3171 cases, respectively, as shown in Table. II. Percent Mean Mean AE represents the ratio of Mean Mean AE to maximum temperature rise (71 K). As we can see, parameterized *ThermPINN* yields Mean AE from 0.15 to 1.21 K and achieves an average Mean AE of 0.47 K. Fig. 4(b)and 4(e) show the thermal maps of both the cases with Min and Max Mean AE predicted by proposed parameterized *ThermPINN*. Ground truth also is generated by COMSOL, as shown in Fig. 4(a) and 4(d). The absolute errors between parameterized *ThermPINN* and COMSOL are shown in Fig. 4(c) and 4(f). We note that solution time including training and inference time of parameterized *ThermPINN* is 6× smaller than that

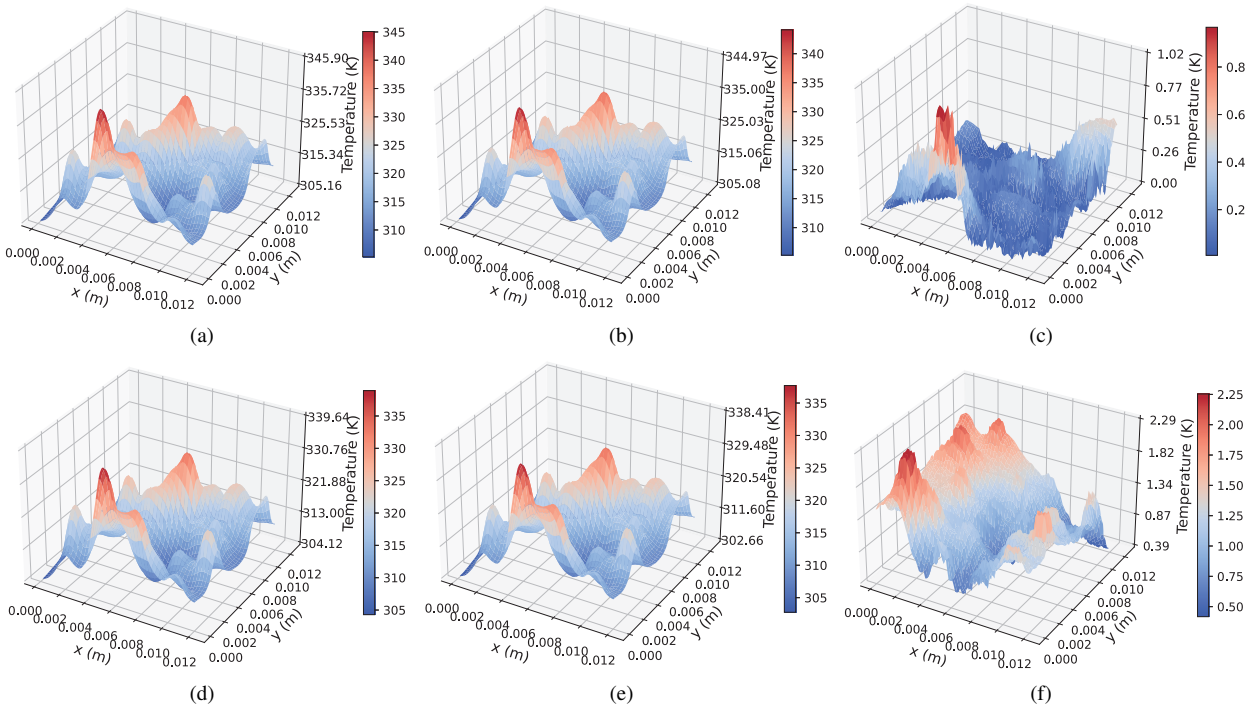

Fig. 4. Thermal maps of full chip by using (a)(d) commercial COMSOL, and (b)(e) proposed parameterized *ThermPINN* with 40×40 eigenvalues. (c)(f) The absolute error between parameterized *ThermPINN* and COMSOL. (a)-(c) The best case with a Min Mean AE of 0.15 K, m=1557 and T_0=330 K. (d)-(f) The worst case with a Max Mean AE of 1.21 K, m=1649 and T_0=330 K.

of COMSOL for this high dimensional problem. The results show the great advantage of the parameterized *ThermPINN* for various running environments and design space exploration over the traditional COMSOL.

To compare the parameterized *ThermPINN* with parameterized plain PINN, we also apply the parameterization technique for plain PINN. The MLP has 6 layers and 256 neurons for each layer. The inputs of MLP are T_0, m, x and y and the output is T. The parameterized plain PINN achieves the Mean AE from 0.09 to 0.39 K, respectively, which are better than the parameterized *ThermPINN*. But parameterized *ThermPINN* achieves 11× speedup over the parameterized plain PINN in terms of training time.

C. Uncertainty quantification based on surrogate model

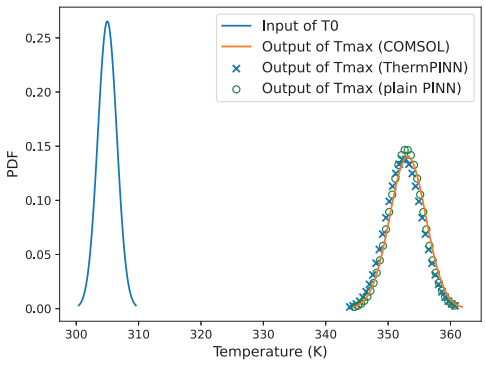

Fig. 5. Probability density of maximum temperature $T_{\max}$ propagated from a truncated Gaussian distributed ambient temperature T_0 (m=1587).

Based on the parameterized surrogate model, we can easily predict the thermal maps with different T_0 and m. Then, the surrogate model is applied for uncertainty propagation from T_0 to maximum temperature $T_{\max}$ of the full chip based on Monte Carlo (MC) simulation. We use python normal distribution function to generate 5000 MC samples for ambient temperature T_0 with a mean value of 305 and a standard deviation value of 1.5, as shown in Fig. 5. With m=1587, we obtain the maximum temperature of 5000 thermal maps as ground truth based on COMSOL, which costs 6.3 hours. We plot the distribution of maximum temperature with 5000 cases in Fig. 5. The maximum temperature distribution is also normal. Its mean value is 353.2 and the standard deviation value is 2.4. If the thermal conductivity and the leakage power are constant, the standard deviation of the maximum temperature distribution should be equal to that of the ambient temperature distribution. For our cases, the dependence of thermal conductivity and leakage power on temperature increases the standard deviation of maximum temperature distribution.

TABLE III
ACCURACY AND SPEED COMPARISON FOR 5000 SAMPLES

Metrics	parameterized *ThermPINN*	parameterized plain PINN	COMSOL
Mean AE of $T_{\max}$ (K)	0.59	0.25	Ground truth
Percent Mean AE of $T_{\max}$	0.93 %	0.39%	
Inference Speed	199 s (113×)	996 s (22×)	22568 s

To validate the accuracy and speed of the surrogate model obtained by parameterized *ThermPINN* and parameterized plain PINN, we predict the maximum temperature for 5000 cases and their distributions are shown in Fig. 5. Compared with ground truth, the Mean AEs of $T_{\max}$ for parameterized *ThermPINN* and parameterized plain PINN are 0.59 and 0.25, respectively, as shown in Table III. Percent Mean AE of $T_{\max}$ represents the ratio of Mean AE of $T_{\max}$ to maximum

temperature rise (63 K). The parameterized plain PINN has better accuracy than parameterized *ThermPINN*. Both of them can match the results of ground truth, as shown in Fig. 5. We note that parameterized *ThermPINN* can achieve 113× speedup over the COMSOL, which shows a great acceleration in UQ analysis. The parameterized plain PINN is 22× faster than COMSOL and slower than parameterized *ThermPINN*.

VI. Conclusion

In this paper, we have proposed a novel full-chip thermal numerical analysis approach based on the cosine series and DNN, called *ThermPINN*, which is faster than plain PINN for training. To consider design space exploration and various running environments using *ThermPINN*, we parameterized the ambient temperature and effective convection coefficient. Then, the obtained surrogate model is used to explore design space and perform UQ analysis. The numerical results on alpha21264 processor show that the proposed *ThermPINN* has 2× speedup and 3× better accuracy over the state-of-the-art VarSim. The experimental results for 2-D full-chip thermal analysis of 3171 cases show that the proposed parameterized *ThermPINN* considering both training and inference time can achieve a 6× speedup over commercial COMSOL with an average Mean absolute error (AE) of 0.47 K. In terms of training time, the proposed parameterized *ThermPINN* is 11× faster than the parameterized plain PINN with similar accuracy. The UQ analysis with 5000 samples for maximum temperature propagated from ambient temperature shows that the parameterized *ThermPINN* and parameterized plain PINN are 113× and 22× faster than COMSOL, respectively.

References

[1] M. Taylor, "A landscape of the new dark silicon design regime," *IEEE/ACM International Symposium on Microarchitecture*, vol. 33, no. 5, pp. 8–19, October 2013.

[2] V. Hanumaiah and S. Vrudhula, "Energy-efficient operation of multicore processors by DVFS, task migration, and active cooling," *IEEE Trans. on Computers*, vol. 63, no. 2, pp. 349–360, February 2014.

[3] Z. Liu, S. X.-D. Tan, X. Huang, and H. Wang, "Task migrations for distributed thermal management considering transient effects," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 2, pp. 397–401, 2015.

[4] H. Wang, J. Ma, S. X.-D. Tan, C. Zhang, H. Tang, K. Huang, and Z. Zhang, "Hierarchical dynamic thermal management method for high-performance many-core microprocessors," *ACM Trans. on Design Automation of Electronics Systems*, vol. 22, no. 1, pp. 1:1–1:21, July 2016.

[5] K.Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in *International Symposium on Computer Architecture*, 2003, pp. 2–13.

[6] J. Kong, S. W. Chung, and K. Skadron, "Recent thermal management techniques for microprocessors," *ACM Comput. Surv.*, vol. 44, no. 3, pp. 13:1–13:42, jun 2012. [Online]. Available: http://doi.acm.org/10.1145/2187671.2187675

[7] B. Goplen and S. Sapatnekar, "Efficient thermal placement of standard cells in 3d ics using a force directed approach," in *Proc. Int. Conf. on Computer Aided Design (ICCAD)*, 2003, pp. 86–89.

[8] C. Tsai and S. Kang, "Cell-level placement for improving substrate thermal distribution," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 2, pp. 253–266, Feb. 2000.

[9] P. Li, L. Pileggi, M. Asheghi, and R. Chandra, "Efficient full-chip thermal modeling and analysis," in *Proc. Int. Conf. on Computer Aided Design (ICCAD)*, 2004, pp. 319–326.

[10] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "Hotspot: a compact thermal modeling methodology for early-stage vlsi design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 5, pp. 501–513, May 2006.

[11] Y. Zhan and S. S. Sapatnekar, "High-efficiency green function-based thermal simulation algorithms," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 9, pp. 1661–1675, 2007.

[12] B. Wang and P. Mazumder, "Accelerated chip-level thermal analysis using multilayer green's function," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 325–344, 2007.

[13] P.-Y. Huang and Y.-M. Lee, "Full-chip thermal analysis for the early design stage via generalized integral transforms," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 5, pp. 613–626, 2009.

[14] Y. Yang, Z. P. Gu, C. Zhu, R. P. Dick, and L. Shang, "ISAC: Integrated space and time adaptive chip-package thermal analysis," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, no. 1, pp. 86–99, 2007.

[15] Y. Liu, R. P. Dick, L. Shang, and H. Yang, "Accurate temperature-dependent integrated circuit leakage power estimation is easy," in *Proc. Design, Automation and Test In Europe. (DATE)*, 2007, pp. 1–6.

[16] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[17] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nat. Rev. Phys.*, vol. 3, no. 6, pp. 422–440, 2021.

[18] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. M. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," *CoRR*, vol. abs/2010.08895, 2020.

[19] L. Lu, R. Pestourie, W. Yao, Z. Wang, F. Verdugo, and S. G. Johnson, "Physics-informed neural networks with hard constraints for inverse design," *SIAM Journal on Scientific Computing*, vol. 43, no. 6, pp. B1105–B1132, 2021.

[20] E. Schiassi, R. Furfaro, C. Leake, M. De Florio, H. Johnston, and D. Mortari, "Extreme theory of functional connections: A fast physics-informed neural network method for solving ordinary and partial differential equations," *Neurocomputing*, vol. 457, pp. 334–356, 2021.

[21] H. Sultan and S. R. Sarangi, "Varsim: A fast and accurate variability and leakage aware thermal simulator," in *Proc. Design Automation Conf. (DAC)*, 2020, pp. 1–6.

[22] S. Sadiqbatcha, Y. Zhao, J. Zhang, H. Amrouch, J. Henkel, and S. X. D. Tan, "Machine learning based online full-chip heatmap estimation," in *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2020, pp. 229–234.

[23] W. Jin, S. Sadiqbatcha, J. Zhang, and S. X.-D. Tan, "Full-chip thermal map estimation for commercial multi-core cpus with generative adversarial learning," in *Proc. Int. Conf. on Computer Aided Design (ICCAD)*. New York, NY, USA: ACM, Nov. 2020, pp. 1–9.

[24] D.-C. Juan, H. Zhou, D. Marculescu, and X. Li, "A learning-based autoregressive model for fast transient thermal analysis of chip-multiprocessors," in *Proc. Asia South Pacific Design Automation Conf. (ASPDAC)*, 2012, pp. 597–602.

[25] J. Wen, S. Pan, N. Chang, W.-T. Chuang, W. Xia, D. Zhu, A. Kumar, E.-C. Yang, K. Srinivasan, and Y.-S. Li, "Dnn-based fast static on-chip thermal solver," in *Proc. Semiconductor Thermal Meas., Modeling Manage. Symp. (SEMI-THERM)*, 2020, pp. 65–75.

[26] V. A. Chhabria, V. Ahuja, A. Prabhu, N. Patil, P. Jain, and S. S. Sapatnekar, "Thermal and ir drop analysis using convolutional encoder-decoder networks," in *Proc. Asia South Pacific Design Automation Conf. (ASPDAC)*, 2021, pp. 690–696.

[27] L. Chen, W. Jin, and S. X.-D. Tan, "Fast thermal analysis for chiplet design based on graph convolution networks," in *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2022, pp. 485–492.

[28] W. Jin, S. Peng, and S. X.-D. Tan, "Data-driven electrostatics analysis based on physics-constrained deep learning," in *Proc. Design, Automation and Test In Europe Conf. (DATE)*, Feb. 2021, pp. 1–6.

[29] S. Cai, Z. Wang, S. Wang, P. Perdikaris, and G. E. Karniadakis, "Physics-Informed Neural Networks for Heat Transfer Problems," *Journal of Heat Transfer*, vol. 143, no. 6, Apr. 2021.

[30] H. He and J. Pathak, "An unsupervised learning approach to solving heat equations on chip based on auto encoder and image gradient," 2020.

[31] O. Hennigh, S. Narasimhan, M. A. Nabian, A. Subramaniam, K. Tangsali, Z. Fang, M. Rietmann, W. Byeon, and S. Choudhry, "NVIDIA SimNet™: An ai-accelerated multi-physics simulation framework," in *Computational Science – ICCS 2021*, M. Paszynski, D. Kranzlmüller, V. V. Krzhizhanovskaya, J. J. Dongarra, and P. M. Sloot, Eds. Cham: Springer International Publishing, 2021, pp. 447–461.

[32] Z. Liu, Y. Li, J. Hu, X. Yu, S. Shiau, X. Ai, Z. Zeng, and Z. Zhang, "Deepoheat: Operator learning-based ultra-fast thermal simulation in 3d-ic design," in *Proc. Design Automation Conf. (DAC)*, 2023, pp. 1–6.

[33] L. Sun, H. Gao, S. Pan, and J.-X. Wang, "Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data," *Computer Methods in Applied Mechanics and Engineering*, vol. 361, p. 112732, 2020.

[34] C. Xu, S. K. Kolluri, K. Endo, and K. Banerjee, "Analytical thermal model for self-heating in advanced finfet devices with implications for design and reliability," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 7, pp. 1045–1058, 2013.

[35] D. Sarkar, A. Haji-Sheikh, and A. Jain, "Temperature distribution in multi-layer skin tissue in presence of a tumor," *International Journal of Heat and Mass Transfer*, vol. 91, pp. 602–610, 2015.