Instituto Tecnológico
y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Matemáticas y Física

MAESTRÍA EN CIENCIA DE DATOS

Solución de predicción de temperaturas usando datos de un simulador térmico

TESIS que para obtener el **GRADO** de
MAESTRO EN CIENCIA DE DATOS

Presenta: **YARED ISMAEL FLORES JIMÉNEZ**

Director **DR. FERNANDO IGNACIO BECERRA LÓPEZ**

Tlaquepaque, Jalisco. Noviembre de 2023.

ACKNOWLEDGMENTS

It is not a simple task to recall all of those that have supported you in every new venture because it happens more often than not, they are so many.

In the first place, I would like to express my gratitude to Edna for her understanding throughout all these two years. I deeply appreciate her patience and support every time I asked for help, that meant a lot to me.

Also, to the academic body of the Data Science master program. The endeavor to start my practice in the field of Data Science would have never been possible without their kindness at moments when I was not precisely the smartest person in the classroom.

To my family and friends that always have believed in me. I would not have been able to accomplish anything in my life without their support.

AGRADECIMIENTOS

No es una tarea sencilla recordar a todos aquellos que te han apoyado en cada nuevo emprendimiento porque sucede la mayoría de las veces, son muchísimos.

En primer lugar, quisiera expresar mi agradecimiento a Edna por su comprensión a lo largo de estos dos años. Agradezco profundamente su paciencia y apoyo cada vez que pedí ayuda, eso significó mucho para mí.

Asimismo, al cuerpo académico de la maestría en Ciencia de Datos. El empeño de iniciar mi práctica en el campo de la Ciencia de Datos nunca hubiera sido posible sin su amabilidad en momentos en los que no era precisamente la persona más inteligente del aula.

A mi familia y amigos que siempre han creído en mí. No habría podido lograr nada en mi vida sin su apoyo.

DEDICATIONS

To my family, Edna, my professors and the privilege to be part of this time now …

DEDICATORIA

A mi familia, Edna, mis profesores y la oportunidad de ser parte de este tiempo ahora …

ABSTRACT

The industry of integrated circuits is experiencing a moment of fierce change. As is, the methods used in all stages implied in its design process. The present work presents a method to predict temperatures for System on Chip (SoC) *chiplet* part with quite simple power map and a single thermal interface material using Machine Learning (ML) and its offspring Deep Learning (DL).

The SoC part is represented as a response surface of a 2D model geometry surface used for a set of experiments to determine the relevant factors for the temperature prediction. In addition to the experiment design, a deployment strategy to implement a continuous integration and deployment process to be used for the target organization is also proposed.

The idea is to achieve the principle of productive ML that states that models should be constantly learning by automating new data ingestion into the training process to enhance model performance in each of the cycle updates.

The project proposes a method to strengthen the established thermal processes of the target organization by using ML tools and provide an alternative to speed up thermal model analysis using new available techniques derived from ML and Deep Learning.

RESUMEN

La industria de los circuitos integrados vive un momento de fuerte cambio, de igual manera, los métodos utilizados en todas las etapas implicadas en su proceso de diseño. El presente trabajo tiene la intención de presentar un método para lograr la predicción de temperatura de un chiplet parte de un *System-on-Chip* (SoC) con un mapa de potencia bastante simple y un material de interfaz térmica haciendo uso de *Machine Learning* (ML) y su descendencia *Deep Learning*.

La parte chiplet del SoC se representa como geometrías de un modelo 2D y es la superficie de respuesta utilizada en un conjunto de experimentos que permiten identificar los factores relevantes en la predicción de su temperatura. Además del diseño de experimentos, también se propone una estrategia de implementación del proceso de Integración Continua de ML por usarse en la organización destino (también conocido como MLOps).

La idea es lograr que el principio de ML que establece que los modelos deben aprender constantemente al automatizar la ingesta de nuevos datos, el proceso de entrenamiento que incentiven la mejora del rendimiento del modelo en cada una de las actualizaciones del ciclo del proceso.

El proyecto es, en esencia, un esfuerzo por proponer un método para fortalecer los procesos térmicos establecidos de la organización objetivo mediante el uso de herramientas ML y proporcionar una alternativa para acelerar el análisis del modelo térmico utilizando las nuevas técnicas disponibles, como ML y Deep Learning.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACHRONYMS AND ABBREVIATIONS

IC		Integrated Circuit
ML		Machine Learning
SoC		System of Chip
NN		Neural Network
PINNs		Physics-Informed Neural Networks
FEM		Finite Element Method
FVM		Finite Volume Method
TIM		Thermal Interface Material
CFD		Computation Fluid Dynamics
CMOS		Complementary metal–oxide–semiconductor
MLOps		Machine Learning Operations

1 INTRODUCTION

With the accelerated evolution of the design of integrated circuits (IC) from Small-Scale Integration (SSI) to the point of Very Large-Scale Integration (VLSI) the addition of requirements connected to optimization science during the design stages of the product are resolutely changing the process of how the companies next products are meant to beat the same from their competitors. The prevalent technologies used in the design of integrated circuits are suffering liabilities on product performance in part due to the increment in circuits failures associated to thermal issues.

Part of the reason, relies on the fact that modern consumers and market demands faster, and more function-capable devices. This represents big challenges as the increase of transistor's density and more sophisticated IC packaging methods which have impacted power and temperature upsurge. Therefore, areas like thermal management are becoming increasingly critical. Some of the challenges for this field now and in the close future will consist mainly of reducing the cost for thermal model analysis by producing more efficient solutions faster and "make room" for new design and innovation and spend less time fixing defects and issues.

Inputs from other knowledge's fields are becoming more relevant in order to enhance thermal management processes. The integration of Machine Learning and Deep Learning can help in high regard for the purpose. These technologies could enable thermal designers to better analyze their systems and identify areas of opportunity to optimize their designs for future projects.

1.1. Justification

Thermal issues are a key factor for performance and reliability in chips [1]. Hence, accurate prediction of the maximum temperature on chips become important for the performance and consistency of chip-packaging systems [2].

Efforts to improve the simulation and temperature predictions with methods like Machine Learning (ML), Numerical, Finite Element Method (FEM) and Finite Volume Method (FVM) and Analytical are currently some of the most common choices.

However, while FEM and Analytical methods have proved their value, they still are not enough to satisfy the level demand and complexity of modern electronic thermal design. ML could be of great value to expand the existing toolkit of options to construct solutions for problems with certain kind of difficulty.

Therefore, it is the conviction of this work to be of significance to show how techniques associated to Data Science and ML can support thermal design process to fulfill its goal to produce thermal efficient devices.

1.2. Problem description

As current methods for thermal analysis of ICs are computational expensive, this represents an opportunity to innovate in the field, for instance like in the case of temperature prediction of System of Chip (SoC) parts.

Although, the company from where the project is originated, owns a good-enough software thermal simulator tool, the internal management-leadership voices pushing for accelerating an optimizing the thermal analysis of new upcoming products has made company's thermal teams to consider data science and machine learning to extend their current toolset of options.

The idea is to reinforce the thermal analysis process of the company with a machine learning solution capable to learn from data produced by the simulation tool to enhance the current thermal design-analysis process (Figure 1 Thermal Design Analysis Process)

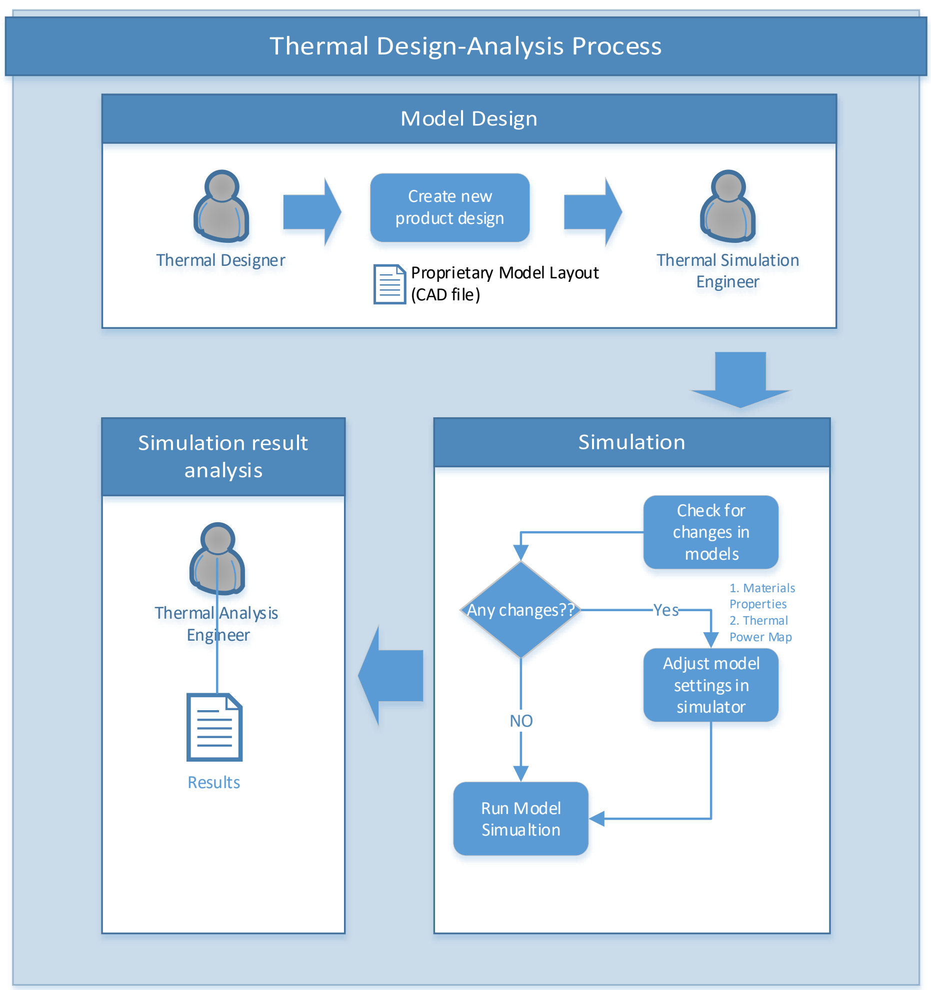

Figure 1 Thermal Design Analysis Process

For this project's scope, it was decided that in order to prove the utility of applying ML in the thermal process, the focus would be on constructing a prediction model based on a specific SoC chiplet. This chiplet referred from now on as the *experiment part or response surface* of the experiment will be the foundational stone that will help to build a prediction model using temperature grids (Figure 3 Temperature Grid extracted from SoC chiplet (simulated)) as input (Figure 2 Alternative Thermal Design-Analysis Process.)

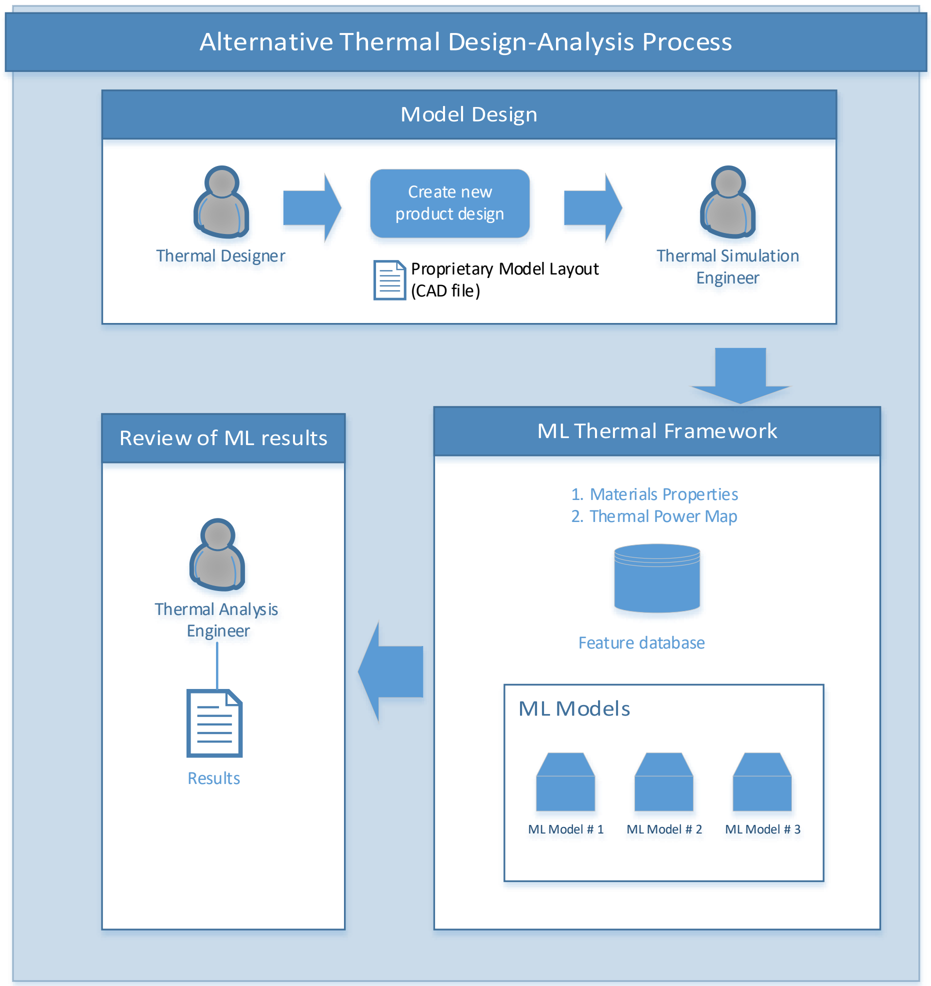

Figure 2 Alternative Thermal Design-Analysis Process

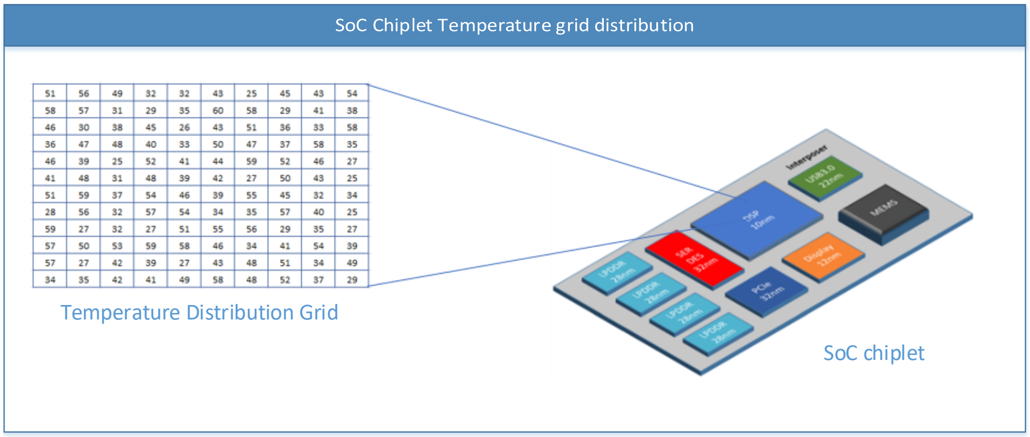

Figure 3 Temperature Grid extracted from SoC chiplet (simulated)

1.3. Hypothesis

Prove the worth of Machine Learning in the field of thermal design by predicting temperatures on SoC surface areas that can help engineers and designers to eventually become more efficient for future designs execution by learning from generated data from on-going projects execution.

1.4. Assumptions and relevant aspects of this work

This work is fundamentally an effort to demonstrate the way machine learning can contribute in the thermal design process for integrated circuits.

The idea is not to replace the simulation software used by engineers and designers in the target organization, but to service from the generated data of these tools to learn more from the temperature profiles of diverse chips combination of materials, dimensions and other associated variables.

Since the thermal simulator encapsulates the complexities of physics calculation for the temperature grids, the intention is to analyze these grids to find the right ML model that approximates it by using the most viable set of variables. Once the right model is identified, plugging it into already existing data analysis pipelines would be a straightforward task.

1.5. Objectives

1.5.1. General Objective:

- Build a solid Machine Learning (ML) prediction model capable to help thermal engineering staff to accelerate thermal analysis for upcoming company's products s

1.5.2. Specific Objectives:

- Construct a viable database for the SoC chiplet temperature prediction model
- Determine the factors (features) that are more relevant for a temperature prediction for a specific SoC chiplet
- Construct a machine learning prediction model capable to predict temperatures for the project's selected chiplet

2 STATE OF THE ART

2.1 Machine Learning on thermal prediction

As it was earlier stated, the incursion of Machine Learning in the process of Thermal Design has already made its own progress throughout the most recent years.
Well documented efforts to replace existing simulation and temperature predictions strategies using ML have been running with the idea to accelerate the existing process. This section briefly describes such efforts.

The article *A Thermal Machine Learning Solver for Chip Simulation* [2] explains very well, a new strategy to create a solver for thermal simulations by bringing a new version of an existing algorithm inspired by the *Composable Autoencoder Machine Learning Simulator* (CoAEMLSim) to make it capable to handle global system parameters:

In the case of the proposed Thermal ML Solver some of these system parameters are:

1. Heat Transfer Coefficients (HTCs)
2. Power maps distributions and
3. Die thickness

Other consider parameters kept as constants are:

1. Inter connection Layer Thickness
2. Insulation Layer Thickness
3. Si Substrate thermal conductivity
4. Interconnection layer conductivity

In the paper is proposed a way to model high dimensional power maps and HTCs. It happens that managing multiple system parameters with varying distributions for simulating physical systems under ML represented a roadblock that data and engineers have been facing to make feasible the use of ML into this domain of thermal analysis.

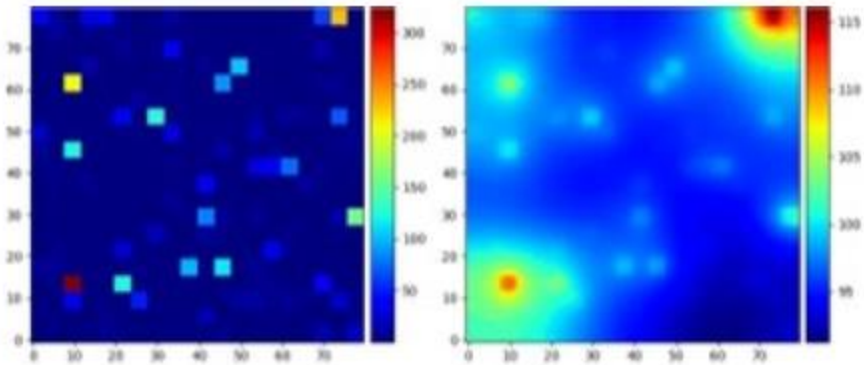

Figure 4 Example of a Power Maps Distribution taken from [2]

An important component in the solution is the **autoencoder** used in the Convolutional Neural Network (CNN) which is key element, simply because the mapping between the power map (Figure 4) with the other system parameters with the final temperature distribution is resolved by them.

In summary, the paper, provides a good indication of how CNNs are used in implementation of a thermal simulator (Figure 5 Partial view of solution).

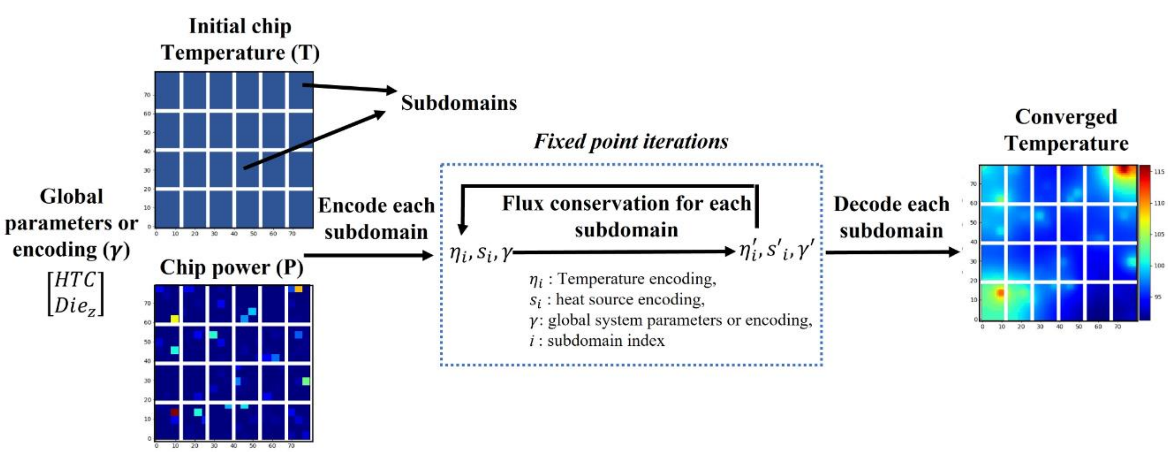

Figure 5 Partial view of solution taken from [2]

Another example is the effort documented in the article: *Approximating the Steady-State Temperature of 3D Electronic Systems with Convolutional Neural Networks* [3]. Similarly, to what [2] the purpose was to find an alternative method to run simulations for thermal design but with slightly differences. In this method, the idea is to randomly generate electronic circuits with finite element solutions, the steady state temperature is estimated a fully convolutional neural network. The underlying idea of this thermal strategy is the simulation of a thermal design by combining physics and Neural Networks.

In both articles the goal is to increase computation speed by:

1. Reducing the number of required design iterations
2. Accelerate the evaluation of individuals designs

Methods used in this solution are:

1. Fully convolutional neural networks FCNN to approximate the 3d electronic systems.
2. Large datasets required for supervised learning
3. Random system generation to create the virtual electronic circuits.
4. Generation of FEM solutions to obtain the temperature solutions
5. Voxelization. For postprocessing the systems and the FEM solutions were converted to 3d images per system as input for the NN.

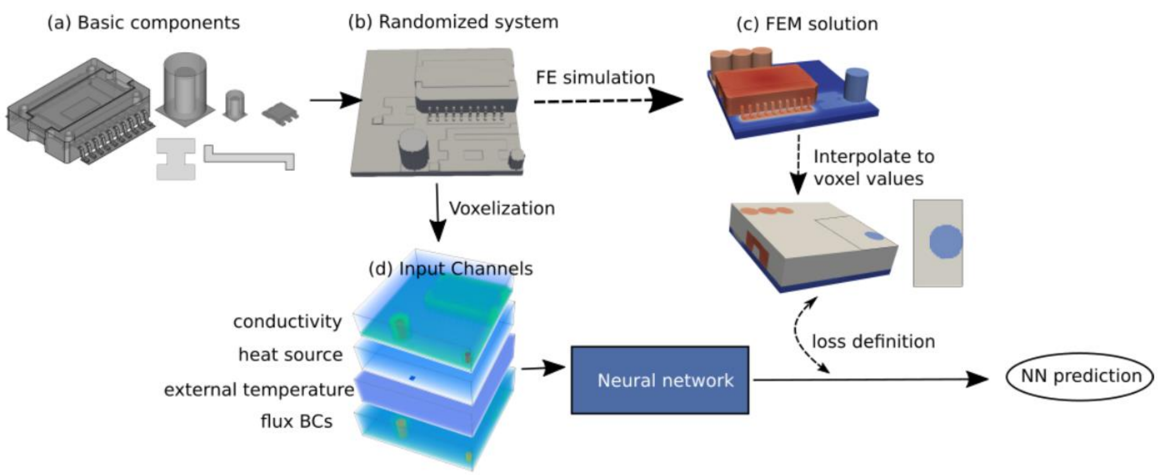

Figure 6 Flow of solution taken from [4]

2.2 Other methods

Although, the interest is to denote the influence of Machine Learning (ML) on thermal analysis and temperature estimation on integrated circuits (IC), it is pertinent to mention other tried non-ML methodologies. Before ML, solutions were envisioned by bearing in mind numerical and analytical methods tactics. For the former, Finite Element Method (FEM) and Finite Volume Method (FVM) being the most common options.

The challenge with these methods is the ability to manage various features or parameters sizes for electronic packages, turning at times into a burdensome option for parametric analysis with varying geometries as well [4].

An alternative way, that allows the use of a broader set of system parameters and power map distributions, has come in the flavor of analytical models. A worth-to-mention effort is the one by Sikka and *Muzychka*. In their work a superimposed source spreading resistance was defined to predict the junction temperature distribution in a chip package with a non-uniform power map using a Fourier series to create a coefficient method. Another effort is the one achieved by *Bagnal* based on *Muzychka's* work using a two-layered Fourier series solution for multilayer system, beating numerical methods like FEM.

After citing these previous contributions, it is imperative to understand that thermal analysis is still innovating from other disciplines to conceptualize new ways to analyze thermal conditions for integrated circuits that guarantee less resource consume and optimal performance for the end-solution.

3 LITERATURE REVIEW

3.1 Integrated Circuits (IC)

3.1.1 Integrated Circuit Design

Integrated circuit design, or IC design, is a part of a larger body of knowledge known as electronics engineering. In the discipline of electronics engineering, there is a process known as circuit design. The goal of circuit design is to assemble a collection of interconnected circuit elements that perform a specific function. The ability to add or multiply numbers is a simple example. The development of a microprocessor that executes computer instructions to perform complex tasks is another example [5] .

3.1.2 What is an Integrated Circuit (IC)

A monolithic integrated circuit (IC) is a complete circuit or group of circuits manufactured in a single piece of silicon, a typical physical size being 1.25 mm square. Such a circuit may contain fifty or more components such as transistors or resistors [6].

3.1.3 Microprocessor

Computers and microprocessors are general-purpose programmable systems which perform sequential processing operations. Classically, they are constructed using general-purpose functional units such as a central processing unit or CPU, a memory unit, and an input/output subsystem [7] .

3.1.4 Die

A die in the context of integrated circuits is a small block of semiconducting material, on which a given functional circuit is fabricated. Typically, integrated circuits are produced in large batches on a single wafer of Electronic Grade Silicon (EGS) or other semiconductor, through processes such as photolithography. The wafer is cut ("diced") into many pieces, each containing one copy of the circuit. Each of these pieces is called a die [8].

3.1.5 System on chip (SoC)

At their core, SoCs are microchips that contain all the necessary electronic circuits for a fully functional system on a single integrated circuit. In other words, the CPU, internal memory, I/O ports, analog inputs and output, as well as additional application-specific circuit blocks, are all designed to be integrated on the same chip. SoCs differentiate themselves from traditional devices and PC architectures, where a separate chip is used for the CPU, GPU, RAM, and other essential functional components [9]. Various SoCs are developed depending on their intended device. For example, SoCs on smartphones or other IoT devices may also incorporate Wi-Fi and cellular network modems. In the traditional approach, SoCs use shorter wiring between circuit blocks to reduce power expenditure and increase efficiencies

3.1.6 Chiplets

Chiplets are small, modular chips that can be combined to form a complete system-on-chip (SoC). They are designed to be used in a chiplet-based architecture, in which multiple chiplets are connected together to create a single, complex integrated circuit. [10]

3.2 Thermal Theory

3.2.1 Thermal Design

The use of appropriate heat transfer techniques, possibly along with some mechanical and electrical design modifications, to sufficiently cool an electronic device or equipment is called thermal design. [11]

However, it is important to recognize that thermal design is not a one-shot design task in which the thermal engineer proposes a thermal design once and forever. Instead, similar to other design disciplines such as mechanical, electrical, power, industrial, and so forth, thermal design is a process that goes through multiple phases and levels of details as the design of the product evolves.
Thermal engineers need to be involved during the whole product design cycle to propose thermally acceptable mechanical and electrical layouts; to check whether proposed mechanical, electrical, or other design changes can be accommodated while the product can be cooled appropriately; or to provide recommendations to achieve a thermally feasible product after implementing those proposed changes [11].

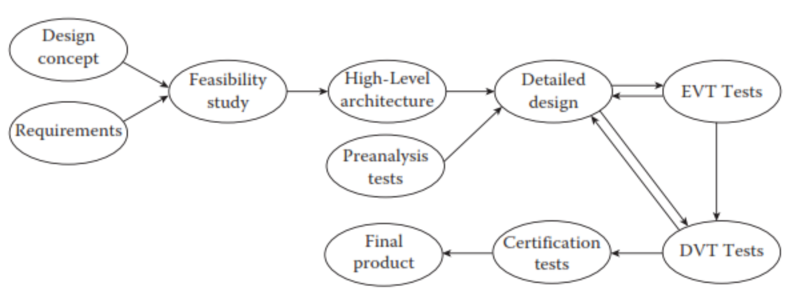

Figure 7 Thermal Process taken from [12]

3.2.2 Energy Transfer and Heat Transfer

The energy transfer that takes place between two objects or systems, because of temperature difference between them, is called energy transfer as heat or simply heat transfer and is denoted by Q. It is seen that heat transfer is nothing other than a form of energy transfer as a consequence of temperature difference between two systems [11].

Energy transfer per unit time is called power and is denoted by P. Two common units of power are watt (W) and horsepower (hp).

- One watt is equal to 1 Joule of energy transfer in one second
- and one horsepower is equal to 746 W.

Sometimes we are interested in learning how fast heat is transferred between two objects. The relationship between heat transfer rate and heat transfer is given by the following integral.

$$Q = \int_{t_1}^{t_2} Q\ x\ dt$$

In some cases, we are interested in heat transfer rate per unit area. This is called heat flux and is denoted by q.

3.2.3 Equation State

The state of a system or its condition is described by values of its properties such as mass, volume, pressure, temperature, internal energy, kinetic or potential energy, polarization, magnetization, and so forth [11].

It has been shown that not all the properties of a system are independent from each other. Consider a simple compressible substance That is a substance free from any magnetic or electric force. The state of such a substance is specified by the values of two of its independently variable properties. For example, internal energy per unit mass and pressure of such a simple compressible. Substance can be determined once its temperature and density $mass\ per\ unit\ volume;\ \rho = \frac{m}{V}$ are known [11].

$$u = u(T, \rho)$$
$$P = P(T, \rho)$$

These equations are called equations of state.

3.2.4 Thermal Analysis of Integrated Circuits

There are two different approaches for performance of thermal analysis of IC's: analytical methods and numerical methods.

The goal of analytical methods is to obtain a mathematical expression that writes the temperature inside the analyzed region as a function of all the variables that may affect it: power dissipated by the heat sources, its location, thermal properties of the materials, boundary conditions, etc.

The great advantage of this technique is that it facilitates performance of parametric analysis, that is, analysis of temperature behavior as a function of one of the variables present in the obtained expression. Its main drawback is that it can only be used when the geometry of the region under analysis can be easily described in one of the three coordinate systems (rectangular, cylindrical or spherical), and when the number of heat sources is small and the dissipated power is either a constant, a step function or a periodic function [12].

Numerical methods discretize the region under analysis into a mesh of nodes and generate a set of linear equations in which the unknown quantities are the temperatures of the different nodes. The main advantage of this method is that it limits neither the geometry description of the region under analysis, the number of heat sources nor the time description of its power dissipation. An additional advantage of these methods is that they allow the coupling of simulations from different domains: thermal, optical, mechanical, etc. The restrictions of this technique are imposed by the computational resources available to solve the linear equation system [12].

3.2.4.1 Analytical Methods

Generally, analytic solutions of the heat conduction equation are classified into three main categories [12]:

- Closed form solutions.
- Fourier series summation (separation of variables).
- Approximated solutions.

3.2.4.2 Numerical Methods

Numerical methods discretize the region under analysis into a mesh of nodes and generate a set of linear equations in which the unknown quantities are the temperatures of the different nodes.
There are three different approaches to obtain this set of linear equations: the Finite Element Method (FEM), the Finite Difference Method (FDM) and the Boundary Element Method (BEM). In this text, we will introduce the Finite Difference Method [12].

3.2.5 Heat Transfer and Its Relation to the Thermodynamics

Prior to the 19th century, heat was envisioned as a liquid that flowed from hotter to colder objects. This imagined substanceless and weightless fluid was called caloric, and no distinction was made between heat and temperature until the writings of Joseph Black (1728-1799). It was not until J. P. Joule published a definitive paper in 1847 that the idea of caloric was abandoned. Joule showed that heat is a form of energy. Moreover, after the experimental results of Rumford, Helmholtz, Joule and others, it was demonstrated that any of the various forms of energy can be transformed into another [12].

3.2.5.1 Thermodynamics

Thermodynamics is the field of science that studies the connection between heat and work and the conversion of one into the other. There are two major laws concerning thermodynamics. The First Law of Thermodynamics is the law of the conservation of energy. When heat is transformed into any other form of energy, or when other forms of energy are transformed into heat, the total amount of energy (taking into account all the forms) in the system is constant [11].

3.2.5.2 First Law of Thermodynamics

The first law of thermodynamics states that energy is not generated or destroyed; it only changes from one form to another or transfers from one system to another [11].

$$E_n - E_{out} = E_f - E_i$$

The difference between final and initial energies of a system is called change of energy of that system or energy accumulation in that system [11].

$$\Delta E_{system} = E_f - E_i$$

This is a simple yet powerful equation that is the basis of all the energy and heat transfer analysis, it is also known as energy balance equation [11].

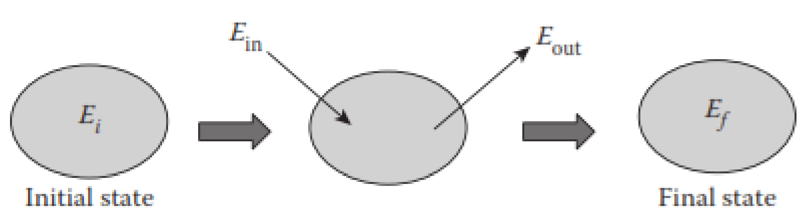

Figure 8 First Law of Thermodynamics taken from [11]

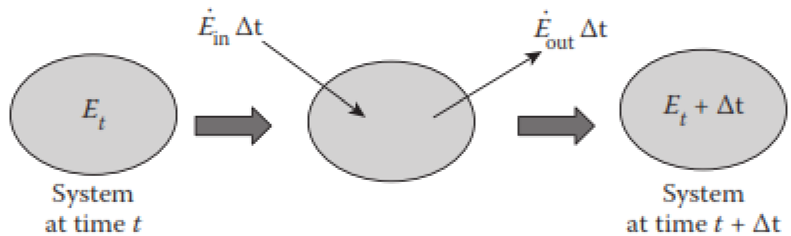

Figure 9 First Law of Thermodynamics (2) taken from [11]

If the energy of a system does not change with time, it is called a steady-state system. In this case:

$$E_{in} = E_{out}$$

Engineering systems can be classified into two groups. Some systems do not allow any mass flow in or out of the system. Such system is called closed system, fixed mass system or control mass. Note that energy may enter or exit a control mass. Many engineering systems, on the other hand, involve some form of mass flow in and out. Such a system is called control volume [11].

3.2.5.3 The Second Law of Thermodynamics

It states that some heat is lost when heat is converted into mechanical energy in a thermal converting machine it is mandatory that part of the heat energy is used just to heat (increase of temperature) the engine. The percentage of heat dedicated to work is called the thermal efficiency of the engine. It was Sadi Carnot (1796-1832) who conducted theoretical studies of the efficiency of heat engines, to model the most efficient heat engine possible [12].

3.2.5.4 Heat Coefficient

A specific heat coefficient can be defined for each material. It is the amount of heat (energy) required to raise the temperature of 1 gram of substance 1 degree Celsius (1°C), (see Table 1 for a list of specific heat coefficients for different elements). Note that this concept is parallel to the capacitance per unit of volume in an electrical conductor [12].

3.2.5.5 Mechanism of Heat Transfer

Heat transfers between bodies or regions of a body of different temperature. The heat flow always takes the direction from the body or region of higher temperature to that of lower temperature (this is another conclusion of the Second Law of Thermodynamics).
The basic mechanisms or modes that model or explain heat transference are conduction, convection, and radiation [12]

3.2.6 Heat Transfer Mechanisms

The mechanism by which heat transfer occurs depends on whether there is any material medium between the two objects and if that medium is moving or not. Three different mechanisms or modes of heat transfer are conduction, convection, and radiation [11].

3.2.7 Conduction Heat transfer

Heat transfer between two objects, or across a single object, which happens through a material medium and does not involve any fluid motion is called conduction heat transfer. Conduction heat transfer is the energy transfer from more energetic particles of a substance to the adjacent less energetic ones as a result of interaction between these particles. The physical mechanism in which this energy transfer happens is different in different materials [11].

In metals conduction heat transfer is due to the energy transfer between free electrons. Fourier's law of heat conduction

$$Q_{\mathrm{cond}} = -\mathrm{kA}\frac{dT}{dx}$$

the k in the equation is called thermal conductivity of material and its unit in SI system of units is W/m°C. As its role in this equation shows, it is a measure of how good a material conducts heat.

Thermal conductivity of a material, in general, is not a constant value. Metals are good electrical and thermal conductors while polymers are poor electrical and thermal conductors.

3.2.8 Convection Heat Transfer

Heat transfer between an object and the adjacent moving fluid (liquid or gas) is called convection heat transfer.

There are two ways that convection heat transfer is created around an object. If the fluid motion is generated by a fan or a pump or wind, it will be called forced convection.

On the other hand, if the fluid motion is generated by a density difference due to a temperature difference in it, the convection is called natural or free convection.
Natural convection is the heat transfer mechanism responsible for heating the whole air inside a room by a heater located at one corner of that room.
Convection flows may be laminar or turbulent. Laminar flows are slow, orderly, and streamlined flows while turbulent flows are faster, disordered, and fluctuating flows in which bulks of fluids move from one region to another and mix with each other in a random manner [11].

Convection heat transfer rate is proportional to the surface area of the object, which is exposed to the moving fluid, A, and the temperature difference between the object and fluid, $T_s - T_\infty$

$$Q_{conv} = \mathrm{h}A(T_s - T_\infty)$$

This equation is known as Newton's law of cooling. The h is called convection heat transfer coefficient and its unit in the SI system of units is $\mathrm{W/m°C}$. Unlike thermal conductivity, convection heat transfer coefficient is not a material property. Convection heat transfer coefficient depends on fluid properties, flow velocity, temperature difference between the surface and the fluid, acceleration of gravity, flow geometry, surface geometry, and type of the flow [11].

3.2.9 Radiation Heat Transfer

There are situations in which there is no medium between two objects that are at different temperatures and are somehow facing each other. In these situations, heat transfer happens through the exchange of electromagnetic wave of photons and is known as radiation heat transfer [11].

It must be mentioned that radiation heat transfer happens between every two objects with different temperatures, along with other modes of heat transfer, as long as the medium between the two objects is not opaque [11].

3.2.10 Thermal Interface Material (TIM)

Thermal Interface Material (TIM) refers to a thermally conductive material inserted between components to efficiently dissipate the heat generated inside electronic devices. It is generally inserted between a heat-generating element such as an integrated circuit (IC) and a heat-dissipating component such as a heat spreader or heat sink [13].

3.3 Design of Experiments

Observing a system or process while it is in operation is an important part of the learning process and is an integral part of understanding and learning about how systems and processes work [14].

In general, experiments are used to study the performance of processes and systems figuring out:

1. which variables are most influential on the response y
2. where to set the influential x's so that y is almost always near the desired nominal value
3. where to set the influential x's so that variability in y is small
4. where to set the influential x's so that the effects of the uncontrollable variables $z_1, z_2, \ldots, z_q$ are minimized.

Experiments often involve several factors. Usually, an objective of the experimenter is to decide the influence that these factors have on the output response of the system. The general approach to planning and conducting the experiment is called the strategy of experimentation. An experimenter can use several strategies [14].

3.4 Experimentation Strategies

3.4.1 Best-guess approach

It is the approach that consists of selecting an arbitrary combination of these factors, test them, and see what happens.

3.4.2 OFAT (One-factor-at-a-time)

The OFAT method consists of selecting a starting point, or baseline set of levels, for each factor, and then successively varying each factor over its range with the other factors held constant at the baseline level. After all tests are performed, a series of graphs are usually constructed showing how the response variable is affected by varying each factor with all other factors held constant [14]. The major disadvantage of the OFAT strategy is that it fails to consider any possible interaction between the factors. Interactions between factors are quite common, and if they occur, the one-factor-at-a-time strategy will usually produce poor results [4].

3.4.3 Factorial Experiment

The correct approach to dealing with several factors is to conduct a factorial experiment. This is experimental strategy in which factors are varied together, instead of one at a time.

One especially important feature of the factorial experiment is clear from this simple example; namely, factorials make the most efficient use of the experimental data. No other strategy of experimentation makes such an efficient use of the data. This is an important and useful feature of factorials [14]

3.4.4 Guidelines for Designing Experiments [14]

1. Recognition of and statement of the problem
2. Pre-experimental Planning
3. Choice of factors, levels, and ranges
4. Choice of experimental design
5. Performing the experiment
6. Statistical analysis of the data
7. Conclusions and recommendations

3.4.5 Fractional Factorial

A Fractional Factorial Experiment (FFE) uses subset of combinations from a Full Factorial experiment. They are applicable when there are too many inputs to screen practically or cost or time would be excessive.

This type of Design of Experiments (DOE) involves less time than One-Factor at a Time (OFAT) and a Full Fractional Factorial, but this choice will result in less data and some interactions will be confounded (or aliased). This means that the effect of the factor cannot be mathematically distinguished from the effect of another factor.

Most processes are driven by main effects and lower order interactions so choose the higher order interactions for confounding. Lower confounding is found with higher resolution.

If a half FFE is determined to be most practical and economical where there are two levels and five factors, then there will be a combination of 16 runs analyzed. Usually, higher order interactions are omitted to focus on the main effects.

3.4.6 Analysis of Variance (ANOVA)

ANOVA is used to decompose the variation of the response to show the effect from each factor, interactions, and experimental error (or unexplained residual).
The DOE will quantify the factor interactions and offer a prediction equation. ANOVA will help show which factors and combinations are statistically significant and which are not thus giving direction to the priority of improvements [15].

DOE Assumptions since ANOVA is used to analyze the data:

1. The residuals are independent.

2. The residuals have equal variance.
3. The residuals are normally distributed.
4. All inputs (factors) are independent of one another.

Most prediction equations will be linear and reliable when using only two levels. This saves time and money while obtaining a prediction equation. Prediction equations are useful to analyze what-if scenarios. Many times, data cannot be collected at all levels and factors so a prediction equation can be used to estimate the output [15].

3.5 The Automation of ML Lifecycle

3.5.1 What is Machine Learning Operations (MLOps)

MLOps can be thought of as the intersection between machine learning and Development Operations (DevOps) practices. Devops, refers to a set of practices that combines the work processes of software developers with those of operational teams to create a common set of practices that functions as a hybrid of the two roles [16].

Similarly, MLOps adopts DevOps principles and applies them to machine learning models in place of software, uniting the development cycles followed by data scientists and machine learning engineers with that of operational teams to help ensure continuous delivery of high-performance machine learning models [16].

This is why MLOps is so crucial. It makes it significantly easier to deploy and maintain your machine learning solutions by automating most of the hard parts for you, massively expediting the development and maintenance processes. With a fully automated setup, teams can keep up with the latest in machine learning technology and deploy new models quickly. “Services can maintain their high level of performance and perhaps even improve on this front as teams can deploy newer, more promising model architectures” [16].

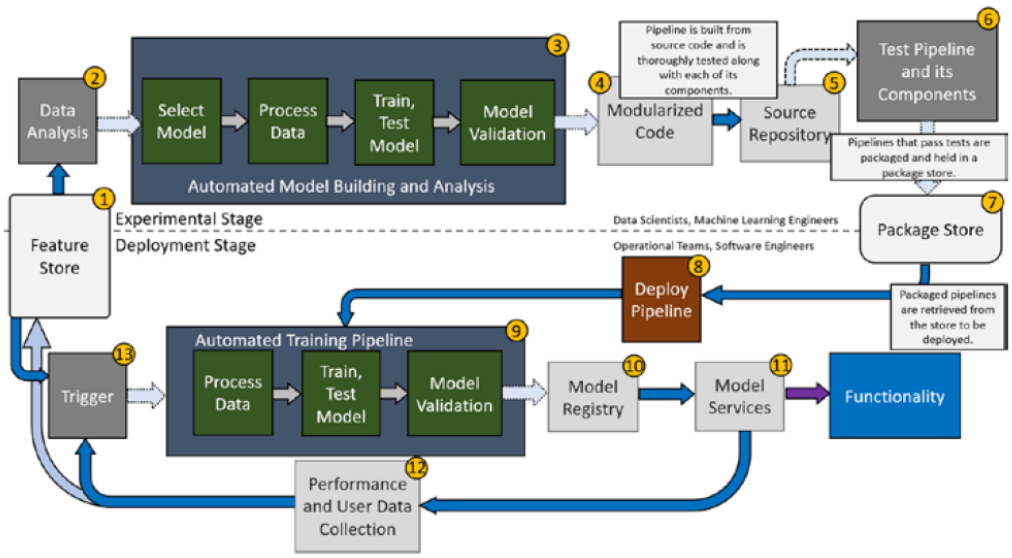

Figure 10 MLOps Process taken from [16]

3.5.2 Pipelines

One way to think about a pipeline is that it is a specific, often sequential procedure that dictates the flow of information as it passes through [16].

The concept of a software pipeline is intuitive enough. If you have a series of steps chained together in your code, so that the next step consumes or uses the output of the previous step or steps, then you have a pipeline [17].

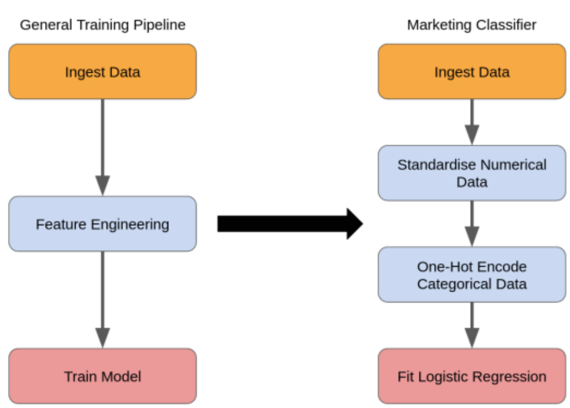

Figure 11 Example of a Training Pipeline taken from [17]

4 METHODOLOGY

This work proposes a methodology that was split into four big steps or phases as shown in Figure 12.

1. Run of Design of Experiments (DOE) that will support the decision for which factors-levels are relevant to use for a ML temperature prediction model of the selected SoC chiplet (Experiment Part).
2. Dataset Generation for analysis and ML experiments (sample dataset). Once factors and levels were defined, the same, will be used to launch the project data process generation.
3. Data analysis and ML model generation
4. Design and execution of MLOps pipelines to automate ML Process Lifecyle

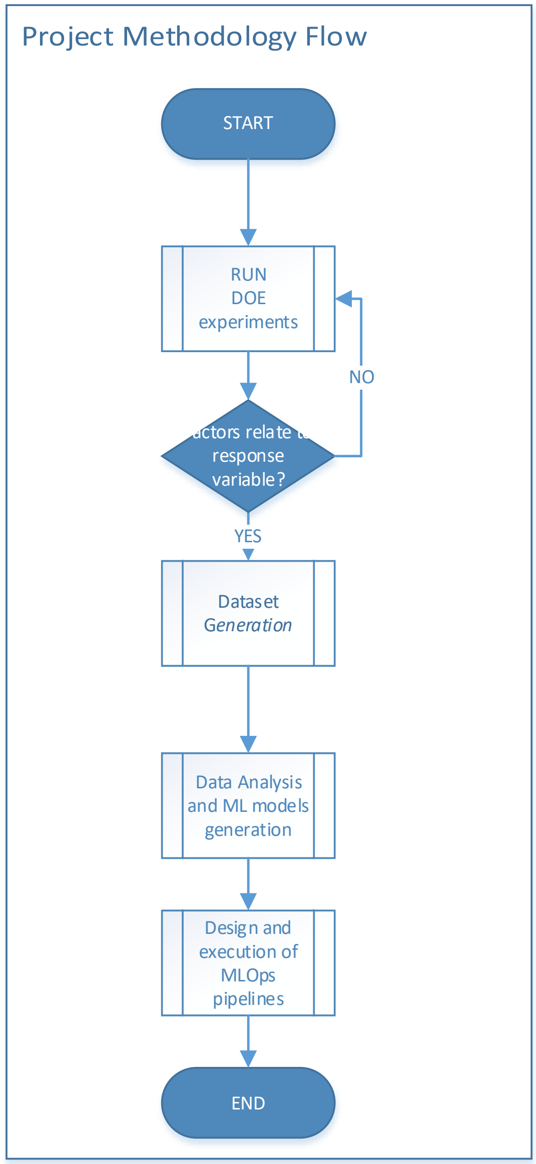

Figure 12 Methodology Flow.

4.1.1 Run of DOE.

As it was briefly explained during the section *1*, the first major challenge to deal with was the selection of the most fitting variables or predictors for the sample data generation.
Since no pre-existing data for the analysis inhabited any of the repositories of the target organization, the selection of a methodology to justify the data selection was of high importance. DOE fit perfectly for the purpose, due to data extractions execution were essentially experiments to choose those features that could best explain the temperature on the component specific surface area.
In order to run the designed set of DOE experiments, connecting with the proprietary software simulator was essential. Fortunately for the project, the software simulator came

packaged with an Application Programming Interface (API) and Command Line Interface (CLI) programs that facilitated the implementation of the automation for the DOE experiments using Python programming language (Figure 13).

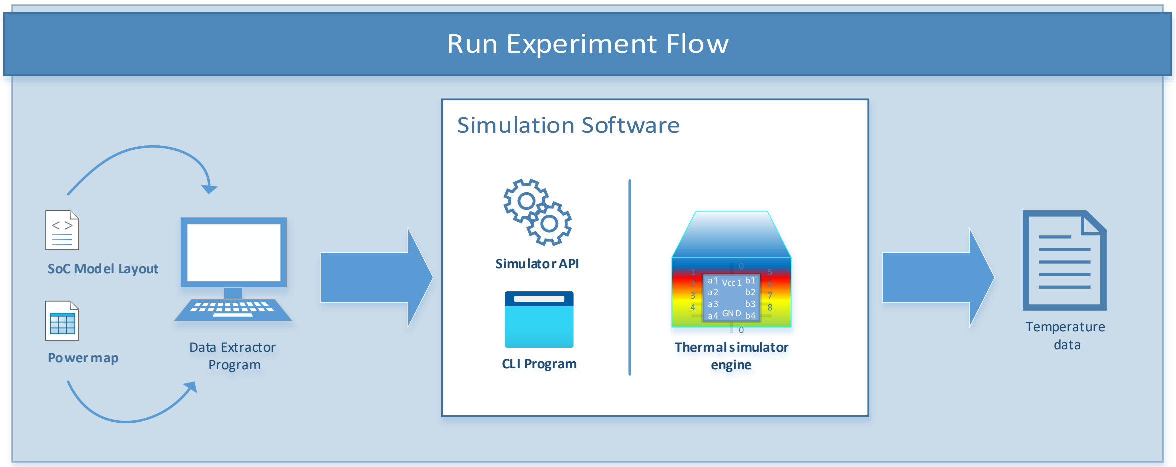

Figure 13 Run Experiment Flow

In summary, the Python Data Extractor program is responsible for:

1. Compute the combinations for the DOE Fractional Factorial (Runs)
2. Setup of the DOE experiments *Runs (Please see* Table 1 Setup of the DOE Experiments*)*
    a. Load the SoC Model Layout (proprietary CAD file of the SoC Model)
    b. Load the Power Map (Power map distribution file)
3. Connect to the Thermal Simulator Engine via the CLI and the API
4. Tweak SoC Model and Power Maps according to the factor and level of the experiment setup for the experiment run.
    a. Adjust settings of the Model and make them match to the experiment factor-levels of the experiment.
    b. Update the power map file with the power stimuli point of the *experiment part* from response surface.
5. Compile the temperature result data (a grid of temperature saved as csv file)
6. Tabulate the temperature's grid for later use in the data analysis stages.

For this to work the SoC die chiplet part (*experiment part*) was split into five different sections for power stimuli points[1] that were applied. What this means is that the Power Map distribution file will only hold one entry power for the experiment part, having the option to

[1] Power stimuli point is the point on the response surface where the power will be applied during thermal simulation

be applied to five different zones of the response surface for each of the experiment runs (Figure 14).

4.1.1.1 Setup of DOE Experiments

ID	**Run**	**Factors**								**Response**
		Levels								
		$part\_size$		ind_var_2		…		ind_var_n		
		min	max	*min*	*max*	*min*	*max*	*min*	*max*	
	Top-Point Section	3550	4200	…	…	…		…		…
1	…	…	…	…	…	…		…		…
n	…	…	…	…	…	…		…		…
	Down-Point Section	…	…	…	…	…		…		…
1	…	…	…	…	…	…		…		…
n	…	…	…	…	…	…		…		
	Left-Point Section	…	…	…	…	…		…		…
1	…	…	…	…	…	…		…		…
n	…	…	…	…	…	…		…		…
	Right-Point Section	…	…	…	…	…		…		…
1	…	…	…	…	…	…		…		…
n	…	…	…	…	…	…		…		…

Table 1 Setup of the DOE Experiments

A more detailed flow of the process is provided in the next section. Please check, Figure 15 DOE Runner Program Flow , for further details.

4.1.1.2 SoC/Die/Chiplet View (Sections)

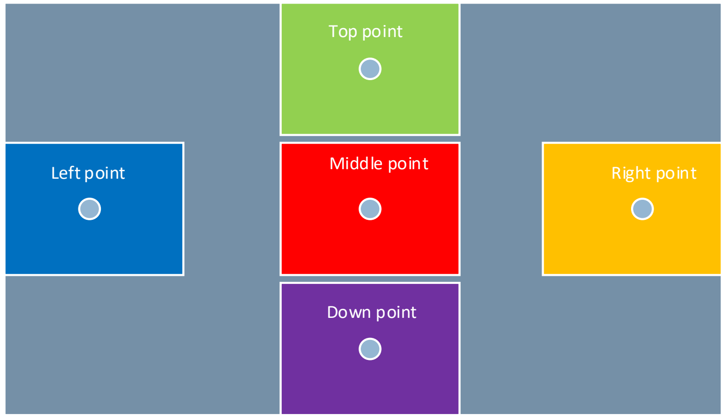

Figure 14 SoC Experiment Part/Response Surface - Defined Stimuli points

4.1.1.3 Steps of the DOE Program Flow

4.1.1.3.1 Init DOE experiment

Figure 15 show the *inputs* required for the full process are set. Inputs for this process are:

1. SoC Model Layout (cad file)
    a. The xml file that defines the outline of the SoC design
2. Factors and Levels
    a. The selected factors and levels specified for the ongoing experiment.
3. Power Map
    a. The power distribution file for the Target Path required for the simulation engine.

4.1.1.3.2 Setup DOE Factors

In this phase, the fractional factorial is computed using the picked factors.

4.1.1.3.3 Load of Model Layout

To run the simulation a cad file containing the outline geometry-based model of the SoC is taken in.

4.1.1.3.4 Setup Target Part

It is during this step, that is explicitly decided which SoC-Die-chiplet part the experiment is targeting. An SoC is comprised by several components, it is a key part of the present experiment to indicate which model's part, the power map will be centered during the simulation phase.

4.1.1.3.5 Load Factor Levels

The factor's levels of every run represent the model's settings to switch during the experiment, e.g., the next factor-level combination:

> **Factor**: part_size_x
> **level**: .3550

will modify the experiment-part size-x value to 0.3550

4.1.1.3.6 Load Power Map

The power map is the file of powers (in watts) that will be used by the simulator engine to calculate the final temperatures after the simulation is accomplished.

4.1.1.3.7 Call Simulator Engine

This is the phase in which the Simulator engine is triggered, once all the settings: experiment factor-levels, power map values and model layout are specified.

4.1.1.3.8 Write Experiments Results

The process is the step when the csv result is written. The file is a grid of temperature distribution.

4.1.1.3.9 Tabulate Results

In the Tabulate step process the grid is tabulated by embedding the factor-levels employed in the run.

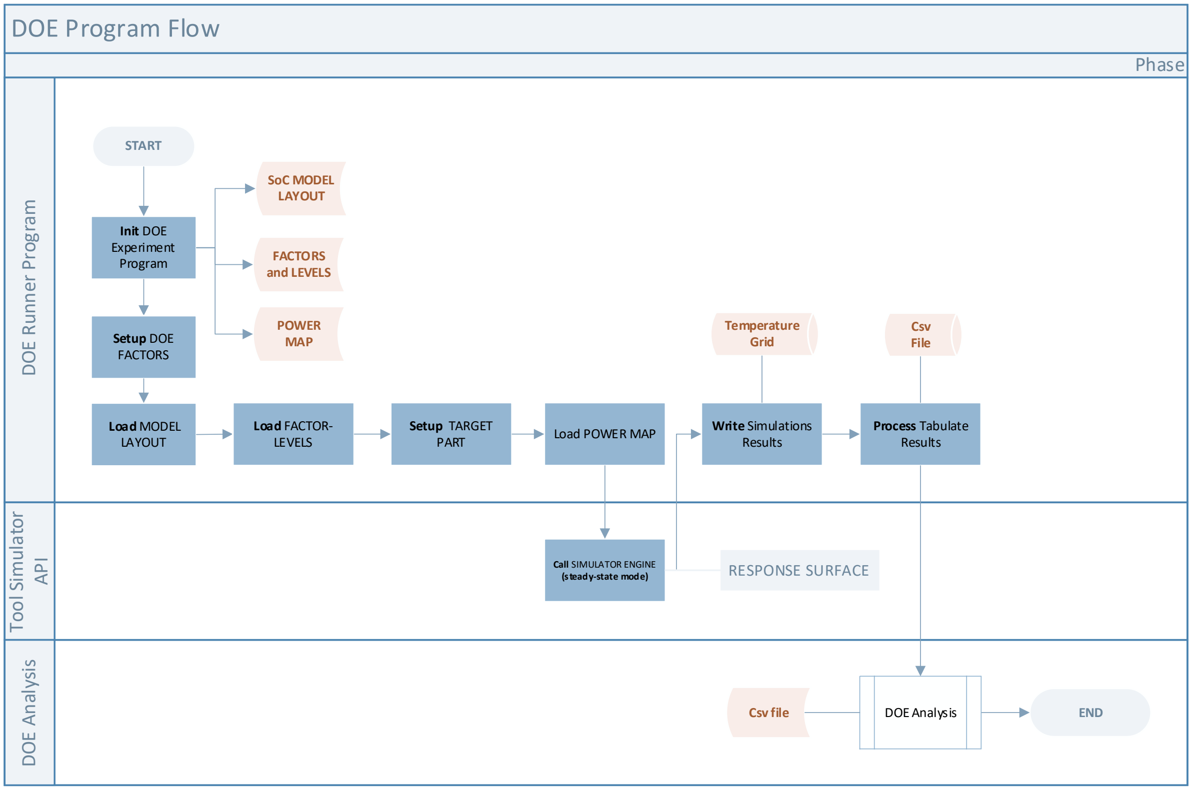

Figure 15 DOE Runner Program Flow

4.1.1.4 Factors used in the DOE set of experiments.

Factor Name	Levels	Description
part_size_x	x-axis value for the part	This represents the “width” value of the part in the given space
part_size_y	y-axis value for the part	This represents the “height” value of the part in the given space
part_size_z	z-depth value for the part	This represents the “depth” value of the part in the given space
part_material	The TIM for the target part	The thermal interface material of the Target Part
part_kx	Conductivity of part’s TIM in the x-axis	Conductivity of the TIM in the x-axis
part_ky	Conductivity of part’s TIM in the y-axis	Conductivity of the TIM in the y-axis
part_kz	Conductivity of part’s TIM in the z-axis	Conductivity of the TIM in the z-axis
part_k	Conductivity of part’s TIM	Conductivity of the part’s TIM
part_rho	Density of part’s TIM	Density of the part’s TIM
part_cp	Specific Heat of part’s TIM	Specific Heat part’s TIM

4.1.1.5 DOE Analysis (Analysis of Variance)

As far as the result for each experiment has been collected, the next step is to run the analysis of variance to verify if any of the selected factors made some sense to explain the response variable (temperature).

Dataset Generation for analysis and ML experiments

Once the DOE step was considered complete, and the factors and level that better explained the response variable are determined, the next step was the dataset generation for the data analysis and ML experiments.

4.1.2 Dataset Generation Program Flow

4.1.2.1 Init Extractor

In this phase the input of the Sample Data Generation Program is setup:

1. Sample Size
2. SoC Model Layout
3. Power Map
4. Factor/Variables range of values.

4.1.2.2 Set Sample Size

An important part of the data generation process is to specify the size of the sample dataset to use in the model construction stage.

4.1.2.3 Assign factor-variables value

This is the step when the selected variables or factors are assigned with a value according to what was obtained during the DOE phase. For instance, if it was decided that factor *part_k* with levels min: .025 and max .05 has a relevant effect over the response, then the variable *part_k* will get randomly assigned values between .025 and .05.

4.1.2.4 Set Model Layout

The Model Layout, as mentioned, represent the SoC model CAD file the Experiment chiplet part belongs to.

4.1.2.5 Setup Experiment Part

For the Data Generation Program to work well, it is important to specify the Experiment part the simulation will be centered.

4.1.2.6 Load Power Map

The Power map is a csv file containing the power distribution *zoner or points* for each of the components of the thermal model.

4.1.2.7 Call Simulation Engine

If everything goes fine after experiment tweaks and adjustments to the model the simulation will be initiated by the Simulation Engine

4.1.2.8 Write Results

For every flow execution of the sample data generation process, the step that follows is to write the temperature distribution grid.

4.1.2.9 Process Tabulate Results

Once all the executions for the data generation process flow were completed, the step that follows is the tabulation of the results for their subsequent analysis.

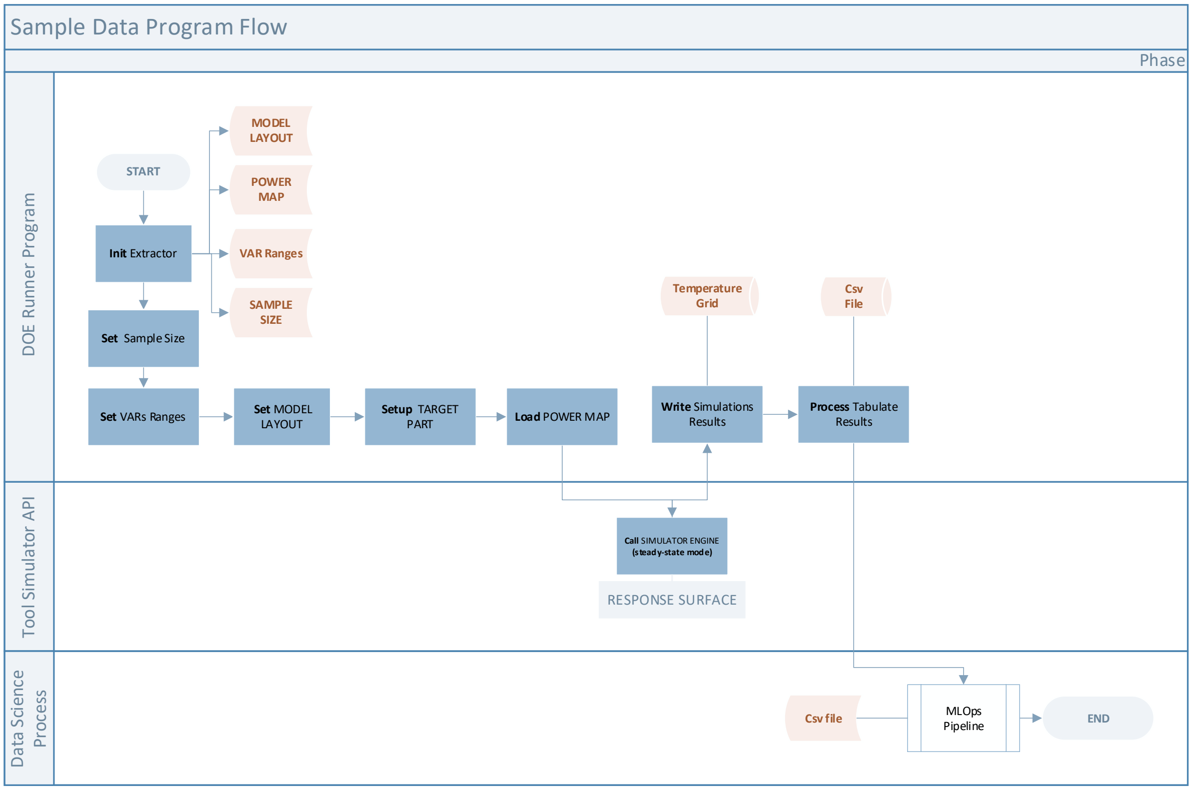

Figure 16 Sample Data Program Flow.

4.1.3 *Data analysis and ML model generation*

This is the phase where data science/machine learning stuff takes place, after the sample dataset was generated, next tasks were performed on data:

1. Statistics
   a. Descriptive Statistics
   b. Inferential Statistics
2. Processing
   a. Feature Engineering
      i. Scaling, normalization and
      ii. Data transformation
3. Training
4. Test and Validation

4.1.3.1 Design and execution of MLOps pipelines to automate ML Process Lifecyle

It was desired for the present work to include a proof of concept about the way the ML process could be automated. A basic proposal is presented following the framework recommended by [16] Please Figure 17 MLOps pipeline workflow reference from .

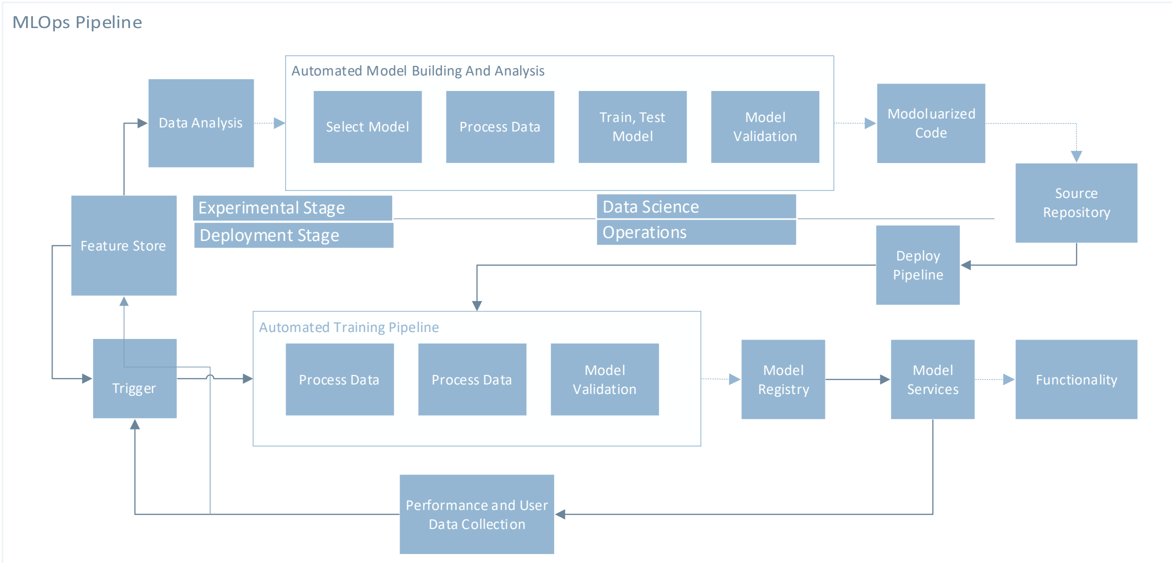

Figure 17 MLOps pipeline workflow reference from [16]

5 RESULTS

5.1 Results

5.1.1 DOE results

After DOE experiments ended, ANOVA analysis was used to determine variables that better explains the response variable. ANOVA analysis result is captured in Table 2 - ANOVA results for DOE experiments.

ANOVA Results for each of the DOE experiments (Stimuli points)

ANOVA Stimuli point at the Top Section

```
                         Df Sum Sq Mean Sq F value  Pr(>F)
part_cp                   1   77.4    77.4   11.69 0.04185 *
part_k                    1  587.0   587.0   88.73 0.00254 **
part_rho                  1    6.6     6.6    1.00 0.39101
part_material_applied_power  1  225.6   225.6   34.10 0.01001 *
Residuals                 3   19.8     6.6
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA Stimuli point at the Left Section

```
                         Df Sum Sq Mean Sq F value  Pr(>F)
part_cp                   1   95.9    95.9   13.13 0.03615 *
part_k                    1  676.3   676.3   92.62 0.00238 **
part_rho                  1    7.3     7.3    1.00 0.39101
part_material_applied_power  1  285.0   285.0   39.04 0.00827 **
Residuals                 3   21.9     7.3
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA Stimuli point at the Middle Section

```
                         Df Sum Sq Mean Sq F value  Pr(>F)
part_cp                   1  123.2   123.2   14.71 0.03123 *
part_k                    1  809.9   809.9   96.74 0.00223 **
part_rho                  1    8.4     8.4    1.00 0.39102
part_material_applied_power  1  372.7   372.7   44.52 0.00687 **
Residuals                 3   25.1     8.4
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA Stimuli point at the Right Section

```
                         Df Sum Sq Mean Sq F value  Pr(>F)
part_cp                   1   54.4    54.4   9.116 0.05680 .
part_k                    1  485.3   485.3  81.346 0.00288 **
part_rho                  1    6.0     6.0   1.000 0.39100
part_material_applied_power  1  151.4   151.4  25.385 0.01507 *
Residuals                 3   17.9     6.0
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA Stimuli point at the Down Section

```
                         Df Sum Sq Mean Sq F value  Pr(>F)
part_cp                   1   70.3    70.3   10.59 0.04731 *
part_k                    1  568.4   568.4   85.65 0.00267 **
part_rho                  1    6.6     6.6    1.00 0.39099
part_material_applied_power  1  201.4   201.4   30.36 0.01177 *
Residuals                 3   19.9     6.6
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Table 2 - ANOVA results for DOE experiments

Based on Table 2 - ANOVA results for DOE experiments, the variables that more influenced the response were:

Factors
part_cp
part_k
part_material_applied_power

This is demonstrated after performing the ANOVA Hypothesis testing, which states:

$$H_0: \mu_1 = \mu_2 = \mu_3 \ldots = \mu\_n$$
$$H_1: \text{Means are not all equal}$$

This Hypothesis testing can also be understood for our interest in the following way:

$$H_0: variances\ are\ equal$$
$$H_1: at\ least\ one\ variance\ is\ different$$

Which means at least one variable from the selected for the experiment has an inference over the response variable.

Since three of the four variables for the experiment produced a F value lower than 5% null hypothesis is rejected.

5.1.2 Data Generation results

To arrive to this part, it was important to determine the factors or predictors to use for the data generation. Since this has been solved, the generation process is the next task.

```python
from  random import uniform
import pandas as pd
import json
import os

sample_size_n=200
part_power_applied=0.439587

list_of_permutations = []

class Material:
    def __init__(self,name, cp, rho, k):
        self.name = name
        self.cp  = cp
        self.rho = rho
        self.k   = k

#Material (Silicon) default attribute values
silicon = Material(name = "Silicon", cp  = 700E0, rho = 2.33E3, k   = 120E0 )
cp_lower = silicon.cp - (silicon.cp * .60)
cp_upper = silicon.cp +  (silicon.cp * .60)

k_lower = silicon.k - (silicon.k * .60)
k_upper = silicon.k + (silicon.k *.60)

power_lower = part_power_applied - (part_power_applied * .60)
power_upper = part_power_applied + (part_power_applied * .60)

for n in range(sample_size_n):
    list_of_permutations.append({  "part_cp" : uniform(cp_lower,cp_upper),
                                    "part_k" : uniform(k_lower,k_upper),
                                    "part_material_applied_power": uniform(power_lower, power_upper)})

data = pd.DataFrame.from_dict(list_of_permutations)
print(data)
factorials_file_path = os.path.join(os.path.dirname(os.path.abspath(__file__)),"factorials.json")
with open(factorials_file_path, "+w") as f:
    str_json = json.dumps(list_of_permutations)
    f.write(str_json)
```

Figure 18 Code of Data Generation Program

1. A size of 200 runs against the simulator were run
2. During the experiment, the values of levels for the experiment were produced by applying TIM [2](in this case Silicon) properties:
   a. Conductivity (part_k) --> 120E0
   b. Specific heat (part_cp) --> 700E0
   c. Power (part_power_applied) --> 0.439587

5.1.2.1.1 Values calculation for sample generation

A calculation to get the Min, Max boundary values, which for the data generation were:

$$part\_k_{lower} = 48E0$$
$$part\_k_{upper} = 192E0$$

[2] Thermal Interface Material

$$part\_cp_{lower} \ = 280E0$$
$$part\_cp_{upper} \ = 1120E0$$

and for the power applied on part:

$$part\_power\_applied_{lower} \ = 0.1758348$$
$$part\_power\_applied_{upper} \ = 0.7033392$$

3. The factor/levels different combinations for the sample size of 200 temperature grids were calculated using the python's function random.*uniform* function using the lower(min) and upper (max) levels

```
for n in range(sample_size_n):
    list_of_permutations.append({  "part_cp" : uniform(cp_lower,cp_upper),
                                   "part_k" : uniform(k_lower,k_upper),
                                   "part_material_applied_power": uniform(power_lower, power_upper)})
```

Figure 19 Data Generation section – permutations

The output of this process, produced 200 different temperature grids like the one in Figure 20 Example of a Grid of Temperatures:

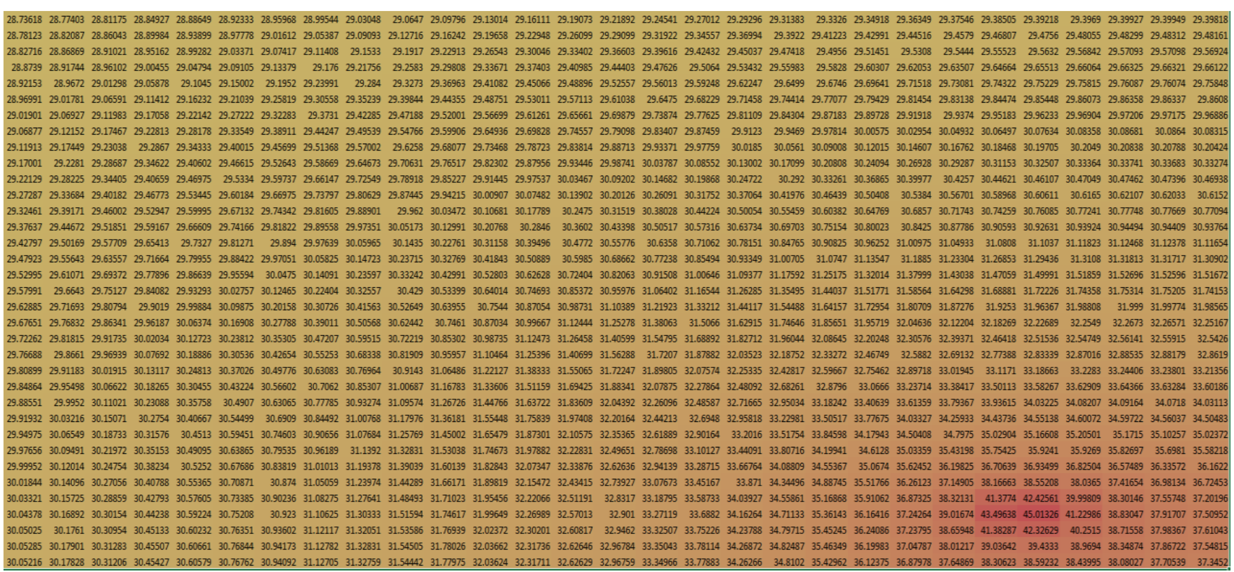

Figure 20 Example of a Grid of Temperatures

To produce the dataset for the ML experiments two decision were taken:

1. To trim each of the temperature grid to seize the portion of temperatures more meaningful for the predictive model
2. After the grids were trimmed, tabulate and merge them to obtain the final dataset

5.1.2.2 Trim each of the temperature grid to seize the portion of temperatures more meaningful for the predictive model

It happens, due the settings of the current problem, the grid of temperature produced had a large area of data points that barely varied between them. This was originated because of the type of power map applied to the simulation (with only one point of power to a single part from the model).

For this reason, the idea to check data variability using quantiles to get rid of those data points with less variability to achieve a model easier to get trained was the main focus. Please check Figure 21 Temperature quartiles and Table 3 Temperature data Standard deviation vs quartiles.

```
count    7031.000000
mean       26.665534
std         0.917899
min        25.900187
25%        26.173960
50%        26.390068
75%        26.787751
max        38.455223
Name: temperature, dtype: float64
```

Figure 21 Temperature quartiles

	Group Quartile 1	Group Quartile 2	Group Quartile 3	Group Quartile 4
Standard deviation	0.06991	0.06279	0.11281	1.27935

Table 3 Temperature data Standard deviation vs quartiles

It can be observed from Table 3 Temperature data Standard deviation vs quartiles from Group Quartile 4 is one with larger standard deviation with means more variability. This pattern was repeated for the 200 files. The final remaining task after the size reduction of the data grids was to merge all the files into one.

5.1.2.3 Data Pre-processing and Feature Engineering

5.1.2.3.1 The calculation of the thermal gradient

So far, three predictors were defined (*part_k*, *part_cp*, *power_applied_part*) and the response (temperature) to form part of the candidate dataset for the machine learning experiments. From Literature Section It was researched the concept the thermal gradient. Reminding the definition of the concept would be of significant help:

"… thermal gradient is defined as the ratio of the temperature difference and the distance between two points (equivalently, it's the change in temperature over a given length). Thermal gradients can be calculated by knowing the temperature at two points and the distance between the two points."

$$TG = (TB - TA)/DX$$

Where**:**
TG: Thermal Gradient
TB: Temperature of point B
TA: Temperature of point A
DX: Distance between the two points

Due to the grid of temperature files is in fact a collection of temperature points that maps the response surface, it made total sense to us to include the thermal gradient for each of the points of the grid with respect of the stimuli point (the point where the power was applied on the part).

5.1.2.3.2 Scaling the Features

Either *part_cp* or *part_k* variables were scaled using sklearn[3] MinMaxScaler. Decision was taken greatly in part because of the way these two predictors were generated during the Data Generation results step. The *power_part_applied* variable remained untouched.

5.1.2.3.3 Final view of the dataset for the ML experiments

Please check the final version of the dataset in: Figure 22 Dataset final version.

[3] Sklearn is a Python library used to implement machine learning model and statistical modeling

	part_cp	part_k	part_material_applied_power	thermal_gradient	temperature
5406	0.889257	0.944323	0.571621	0.209585	27.001633
3982	0.233200	0.077872	0.366923	0.347619	28.568987
4627	0.728660	0.542651	0.433338	0.603663	31.437918
6379	0.467730	0.395452	0.655082	0.678873	30.066177
3736	0.009903	0.142318	0.389456	0.577396	31.590117
...	...	...	...	...	...
4074	0.967895	0.657583	0.483506	0.513538	30.887196
3812	0.377950	0.000000	0.578890	0.576209	29.643425
6025	0.073550	0.993336	0.334578	0.471254	29.349747
6818	0.694469	0.984456	0.360673	0.574836	30.744915
5674	0.655537	0.382916	0.191516	0.855266	31.540007

Figure 22 Dataset final version

5.1.3 ML Model Training and Results

For the executions of the ML experiments, all the ML models were encapsulated as pipelines and run via docker containers. Their results stored in the MLflow platform local storage. The advantage of doing it this way was that every metric and plots produced by the ML model experiment is stored and save for future consult.

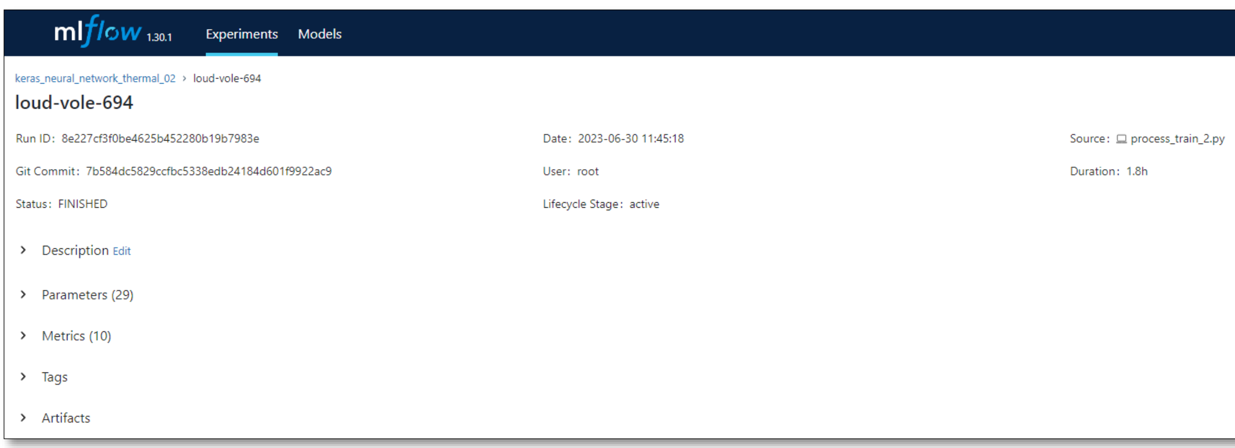

Figure 23 MLflow model experiment result

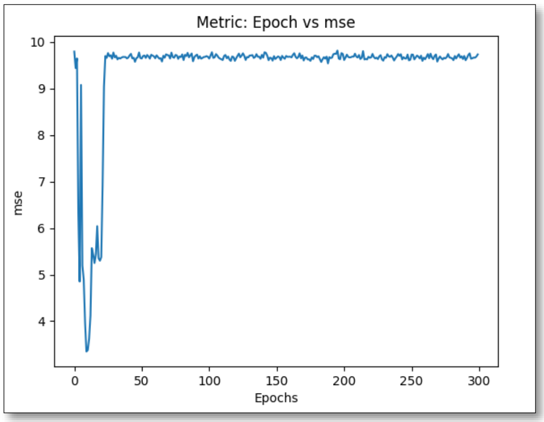

Figure 24 Plot MSE vs epochs

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 10)                50

 dense_1 (Dense)             (None, 50)                550

 dense_2 (Dense)             (None, 50)                2550

 dense_3 (Dense)             (None, 1)                 51

=================================================================
Total params: 3,201
Trainable params: 3,201
Non-trainable params: 0
_________________________________________________________________
```

Figure 25 Neural Network Architecture

5.1.4 Hyperparameters Tunning

For the Hyperparameter Tunning a new library maintained by Facebook was used: Ax. Figure 26 shows the code for the hyperparameter tunning.

```python
def process(self, experiment_name):
    from ax.service.ax_client import AxClient
    from ax.utils.notebook.plotting import render, init_notebook_plotting
    from mlops.projects.thesis.nn.pipelines.parameters import parameters
    init_notebook_plotting()
    ax_client = AxClient()
    ax_client.create_experiment(
        name="keras_experiment",
        parameters=parameters,
        objective_name='keras_cv',
    minimize=True)

    self.pipeline = ProcessorBase(self.data_path, model = None, mlflow= None)
    self.pipeline.step_compile_data_file()
    def evaluate(parameters):
        return { "keras_cv": self.pipeline.step_keras_mlp_cv_score(parameters) }


    for i in range(25):
        parameter, trial_index = ax_client.get_next_trial()
        print(parameter)
        ax_client.complete_trial(trial_index=trial_index, raw_data=evaluate(parameter))

    ax_client.get_trials_data_frame().sort_values('trial_index')
    best_parameters, values = ax_client.get_best_parameters()

    # the best set of parameters.
    for k in best_parameters.items():
        print(k)

    print()

    # the best score achieved.
    means, covariances = values
    print(means)

    # save results to json file.
    ax_client.save_to_json_file()

    # restore the client from json file. Handy if you want to do more trials or if your computer crashed in the middle of the trials.
    restored_ax_client = AxClient.load_from_json_file()
```

Figure 26 Portion of the code for hyperparameter tunning

5.1.4.1 Ax Hyperparameter Tunning Experiment results

Experiment 1 (epochs = 1)

learning rate	0.0022606556526699
dropout_rate	0.0800264815585484
num_hidden_layers	2
neurons_per_layer	64
batch_size'	7
activation	tanh
optimizer	adam

Experiment 2 (epochs = 5)

learning rate	0.002574266670049544
dropout_rate	0.0905928978022987
num_hidden_layers	1
neurons_per_layer	277
batch_size'	64
activation	tanh
optimizer	sgd

Table 4 Example of the optimized hyperparameters by Ax.

5.1.5 ML Training Experiments Results

5.1.5.1 Best Experiment

In the initial training execution, applying a similar network architecture than hyperparameter Tunning Experiment # 1 but considerably larger epochs and batch sizes as shown in Table 5 and Figure 27:

learning rate	0.09
dropout_rate	0.0800264815
num_hidden_layers	4
neurons_per_layer	[10,50,50]
batch_size'	500
activation	Tanh
'optimizer'	Adam

Table 5 Tuned Hyperparameters

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_29 (Dense)            (None, 10)                50

 dense_30 (Dense)            (None, 50)                550

 dense_31 (Dense)            (None, 50)                2550

 dense_32 (Dense)            (None, 1)                 51

=================================================================
Total params: 3,201
Trainable params: 3,201
Non-trainable params: 0
_________________________________________________________________
```

Figure 27 Neural Network Architecture

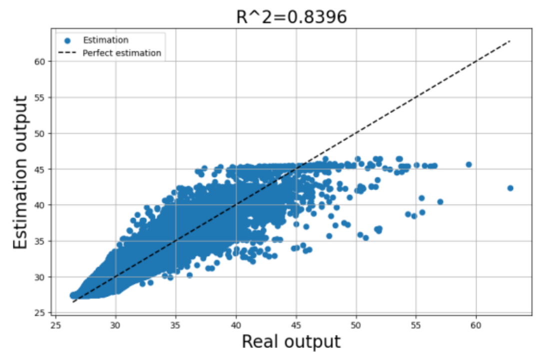

Figure 28 ML Training Experiment R^2

Currently the best score of R^2 hit for any of our trained models is shown in Figure 28 with a value of 0.8396. The model is yest being tuned up to level up the metrics.

5.1.6 MLOps implementation

The implementation of MLOps consists in setting up the MLflow [4] as the MLOps platform. There are six modes MLflow can be setup to run and store artifacts:

1. Scenario 1: MLflow on localhost
2. Scenario 2: MLflow on localhost with SQLite
3. Scenario 3: MLflow on localhost with Tracking Server
4. Scenario 4: MLflow with remote Tracking Server, backend, and artifact stores
5. Scenario 5: MLflow Tracking Server enabled with proxied artifact storage access
6. Scenario 6: MLflow Tracking Server used exclusively as proxied access host for artifact storage Access

For this project Scenario 4 was selected.
Docker containers and Docker Compose were employed to setup local next services:

1. A Tracking DB service deployed as docker container using PostgreSQL
2. An sFTP service via docker container was built for the storage artifact role of MLflow
3. The MLflow server listening to port 5000 for any requests coming from the MLflow pipeline container
4. A git repository containing the code of models' pipelines (processing, training pipelines)

Please see Figure 29 Thermal MLOps Implementation for further details

[4] MLflow is a versatile, expandable, open-source platform for managing workflows and artifacts across the machine learning lifecycle. **More information at**: https://mlflow.org/docs/latest/what-is-mlflow.html

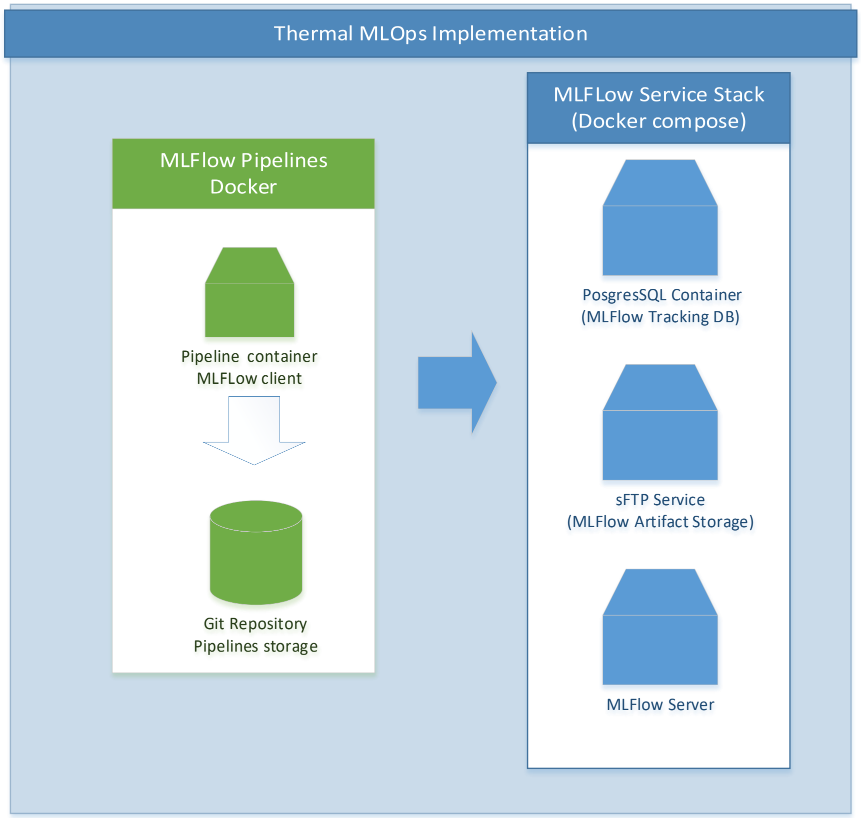

Figure 29 Thermal MLOps Implementation

6 CONCLUSIONS

6.1 Conclusions

The results presented in this work are step ahead for the thermal engineering teams in the target organization to expand their analysis tools options for their thermal models due to the following reasons:

- With the Data Generation software tool, the data extraction has been automated and is possible to generate sample data for data analysis and machine learning experiments
- The initial version of the Neural Network has shown that is possible to create deep learning models with the data extracted from the thermal simulated tool.

6.2 Future Work

Incorporating machine learning for analyzing temperature distributions and infer from the neural network model is a big step for the purpose to orient more decisions considering the data analysis world. Though, there are more things ahead to work to reach out the envisioned level of sophistication. It is considered, the next steps this effort should move on are:

1. The ML analysis must be extended to make it capable to support a more complex thermal model setting, this means more chiplets or SoC parts involved in the analysis with the idea to infer the temperature of the entire SoC
2. The inclusion of more complex power maps and surface dimensions

7 BIBLIOGRAPHY

[1] R. Chandra, "A thermal-aware IC design methodology," 13 06 2005. [Online]. Available: https://www.eetimes.com/a-thermal-aware-ic-design-methodology/.

[2] R. Ranade, H. He, J. Pathak, N. Chang, K. Akhilesh and J. Wen, "A Thermal Machine Learning Solver For Chip Simulation," *IEEE,* p. 7, 2022.

[3] M. Stipsitz and H. Sanchis-Alepuz, "Approximating the Steady-State Temperature of 3D ElectronicSystems with Convolutional Neural Networks," *Mathematical and Computational Applications,* p. 7, 2022.

[4] T. Liu, M. Iyengar, C. Malone and G. Kenneth , "Analytical Modeling for Prediction of Chip Package-level Thermal Performance," *IEEE Explore,* p. 8, 2016.

[5] Synopsys, "What is IC Design?," [Online]. Available: https://www.synopsys.com/glossary/what-is-ic-design.html.

[6] N. M. Morris, Semiconductor Devices, MacMillan, 1976.

[7] M. A. Laughton and D. Warne, Electrical Engineer's Reference Book, Newnes, 2002.

[8] J. . E. Ayers, Digital Integrated Circuits, Taylor & Francis, 2003, 2003.

[9] S. Panchapakesan, "Why System-On-Chip is the Next Generation of Computing," [Online]. Available: https://www.synopsys.com/cloud/insights/system-on-chip.html.

[10] "The Ultimate Guide to Chiplets," [Online]. Available: https://anysilicon.com/the-ultimate-guide-to-chiplets/#:~:text=Chiplets%20are%20small%2C%20modular%20chips,a%20single%2C%20complex%20integrated%20circuit..

[11] Y. Shabany, Heat Transfer, Thermal Management of Electronics, CRC Press, 2010.

[12] J. Altet and A. Rubio, Thermal Testing of Integrated Circuits, Springer, 2002.

[13] Dexerials, "Heat Dissipation Principle of TIM (Thermal Interface Material)," 10 03 2010. [Online]. Available: https://techtimes.dexerials.jp/en/electronics/heat-dissipation-principle-of-tim-thermal-interface-material/.

[14] D. C. Montgomery, Design and Analysis of Experiments, Wiley, 2017.

[15] six-sigma-material.com, "Design of Experiments (DOE)," [Online]. Available: https://www.six-sigma-material.com/Design-of-Experiments.html.

[16] S. Alla and S. Kalyan Adari, Beginning MLOps with MLFlow, Apress, 2021.

[17] A. McMahon, Machine Learning Ennginneering with Python, Birmingham, UK: Packt, 2021.

[18] M. Stipitz and H. Sanchis-Alepuz, "Approximating the Steady-State Temperature of 3D Electronic," *Mathemtical and Computational Applications,* p. 15, 2022.

[19] "Interposer," [Online]. Available: https://anysilicon.com/semipedia/interposer/.