POLITECNICO DI TORINO

Collegio di Ingeneria Informatica, del Cinema e Meccatronica

Corso di Laurea Magistrale in

Mechatronic Engineering (Ingegneria Meccatronica)

Control Technologies for Industry 4.0

Tesi di Laurea Magistrale

A Novel Virtual Sensor Approach to Estimate Junction Temperature of Power Electronics Components for Electrical/Hybrid Vehicles

Relatore:

PROF. MASSIMO VIOLANTE

DOTT. JACOPO SINI

Candidato

Al Hassan Choucair

S309397

Novembre 2024

Table of Contents

Content of figures:

Content of tables:

IN HONOR OF THOSE WHO COULDN'T BE WITH US YET STILL INSPIRE…

27/03/2024

Abstract

As modern technology continues to evolve, the demand for more complex yet compact electronic components has steadily increased. This trend has led to significant challenges, particularly in managing heat generation within integrated circuits (ICs). Effective thermal management has become a critical concern, as excess heat can not only impair the performance of components like transistors but also shorten their operational lifespan by accelerating thermal wear. In high-stakes applications such as electric vehicle (EV) inverters, precise temperature control is vital for maintaining both the efficiency and reliability of the system. Given these challenges, this thesis focuses on developing a mathematical thermal model to estimate the junction temperature of the MOSFET transistor in EV inverters, employing both simplified and complex modeling methodologies applied on a case study presented by I&M Torino.

This thesis is divided into two primary units, each focusing on distinct modeling approaches and methodologies. The first unit explores a simplified lumped parameter model. This approach was chosen due to its computational efficiency and ability to approximate the thermal behavior of physical systems by reducing the system into a series of discrete elements. This method is particularly well-suited for real-time applications, such as those required for motor control units in EV systems. The lumped parameter model used in this thesis is represented in a Linear Time-Invariant (LTI) system framework. LTI systems are known for their analytical tractability, which enables the efficient simulation of thermal behavior in both the continuous and discrete-time domains. The construction of the LTI model involves carefully selecting input signals, such as temperature readings from an onboard NTC (negative temperature coefficient) thermistor, which feeds into the feedback loop for recalibration, improving the accuracy of the temperature predictions.

To validate this model, MATLAB and Simulink simulations are employed. These tools enable the translation of the thermal system between the continuous and discrete-time domains, a process critical for ensuring the model's adaptability to various operating environments and control systems. The model is first tested using reference data provided by Ideas & Motion Torino, the company suggesting the case study and providing supporting data. It is important to note that the reference data given by the company will not correspond to the assumed PCB structure in this study but is rather used as realistic reference behavior. Therefore, the given reference data and built model (based on assumed PCB structure) will present mismatching results. This mismatch will be mitigated in the study using curve matching techniques to rather focus on the predictive power of the model. This underscores not only the predictive power of the simplified model but also its flexibility and capacity for individual design calibration, essential in practical industrial applications.

In the second unit, the study transitions to a more complex assumed model. The goal of this approach is to extend the predictive capabilities of the initial simplified model by incorporating additional dynamic components. The complex model is introduced using a bond graph representation, which allows for the systematic and visual depiction of the energy flows within the system across multiple physical domains (electrical, thermal, mechanical). Bond graphs are particularly useful in this context because they facilitate the integration of multi-domain dynamics in a manner that is both intuitive and mathematically rigorous. The model captures the interaction between various thermal and electrical elements of the inverter system, providing a deeper understanding of the thermal behavior under more complex operating conditions.

The process of refining the complex model involves matrix building, where the system of equations is derived to represent the interactions between different components within the inverter.

Preliminary component value calculations, drawn from the case study data, are used to populate the matrices, while MATLAB simulations are again leveraged to analyze the system's behavior. The results of these preliminary simulations provide valuable insights, leading to several rounds of tweaking and modifications to the assumed model structure. This iterative process enhances the model's precision, enabling it to more accurately capture the thermal dynamics of the system under varying loads and operational scenarios. Ultimately, the refined model demonstrates significant improvements in both predictive accuracy and flexibility when tested against the case study’s thermal data.

The findings from both units underscore the importance of grey-box modeling techniques in thermal system analysis. The grey-box approach, which blends theoretical understanding with empirical data, provides a balanced framework for estimating the thermal behavior of complex systems. The lumped parameter model proves particularly effective in reducing system complexity, while the LTI-based simulations offer a robust platform for analyzing the system's real-time thermal response. Additionally, the bond graph method facilitates a deeper, multi-domain analysis, extending the applicability of the model beyond simple PCB systems to more intricate inverter designs.

This research highlights the potential of the developed thermal model in a wide range of industries. In particular, the model's ability to predict junction temperatures in MOSFET transistors within EV inverters makes it a valuable tool for optimizing thermal stability and component longevity. Such models could be applied to various sectors, including automotive, aeronautics, and other technology-driven industries where thermal management is critical. By accurately predicting thermal behavior, this modeling technique offers the potential to extend the lifecycle of critical components, thereby reducing maintenance costs and improving overall system efficiency.

The conclusions drawn from this work offers a comparison between the performance of the results of the 2 discussed units and suggest several avenues for future research. One key area involves enhancing the granularity of the model to account for more intricate internal dynamics. Expanding the model's application to other semiconductor devices, such as IGBTs and SiC-based transistors, is another promising direction. Moreover, the importance of high-quality reference data cannot be understated. The study highlights the need for precise and reliable component data, as errors in this area can significantly affect the accuracy of grey-box models. Continued refinement and validation of this thermal modeling approach could lead to more robust, real-time control strategies for thermal management in next-generation electronic systems.

Introduction

As technology continues to evolve, there is an ever-increasing demand for compactness in modern electronic systems, driven by both rising complexity and the need to accommodate various ergonomic and spatial constraints. The trend toward miniaturization is evident across a wide array of industries, and one of the most striking examples of this is the ongoing advancement of integrated circuits. Every year, chips grow more complex while shrinking in size, a phenomenon closely aligned with Moore's Law (Moore, 1965). Moore's Law, an empirical observation based on historical trends, posits that the number of transistors on an integrated circuit doubles approximately every two years, a prediction that has held remarkably true since the 1970s. This steady increase in transistor density not only allows for more powerful and efficient chips but also introduces significant challenges, one of the most pressing being the generation of heat. A visual representation Moore's law can be visualized in **Figure 1**.

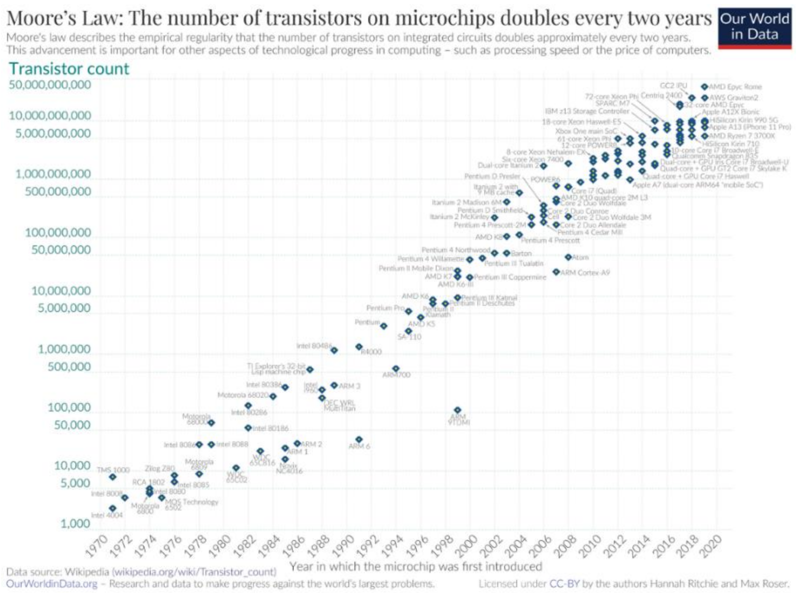


Figure 1 - Moore's Law in effect throughout production history of transistors in chips.

The continuous rise in the number of transistors and other components within ever-smaller integrated circuits intensifies the issue of heat dissipation. Excessive heat within these circuits can prove catastrophic, not only directly—by damaging or degrading components—but also indirectly, by accelerating wear and shortening the lifecycle of devices. Moreover, elevated temperatures can alter the electrical behavior of sensitive components such as transistors and fast-switching devices, impacting the overall performance of the circuit. As chips shrink to the nanoscale, managing thermal effects becomes increasingly difficult, further complicating efforts to maintain operational efficiency and longevity.

Given these challenges, thermal modeling has become an essential tool in modern electronics design. Accurate thermal models allow engineers to predict the heat distribution within a circuit, assess its impact on performance, and take necessary actions to mitigate overheating. This is particularly critical as integrated circuits become more compact and heat generation becomes a more prevalent issue. The ability to model and predict thermal behavior is key not only for the functional integrity of devices but also for optimizing the overall design and extending the lifespan of critical components.

This paper addresses the pressing need for efficient thermal modeling by presenting a case study provided by Ideas & Motion Torino. The objective is to develop a lightweight thermal model that can run in real-time on the motor control unit microcontroller featured in the study. Specifically, the model focuses on estimating the junction temperature of a MOSFET transistor within the PCBs used in electric vehicle (EV) inverters. These components play a crucial role in managing the high-power demands of EVs, making it essential to ensure their thermal stability under varying operational conditions. The importance of accurate thermal modeling in this context cannot be overstated, as temperature fluctuations can significantly impact the reliability and efficiency of these critical power electronics.

Using data provided by the company of Ideas & Motion (located in Torino), this study employs a grey-box modeling approach. This method blends empirical data with theoretical insights to form a simplified structure that captures the key thermal dynamics of the real system. The simplified structure of the chip, including the MOSFET transistor, is visualized in **Figure 2**. The grey-box model is particularly advantageous in this case, as it strikes a balance between complexity and computational efficiency, allowing it to provide accurate predictions without overwhelming the limited processing power available on the microcontroller.

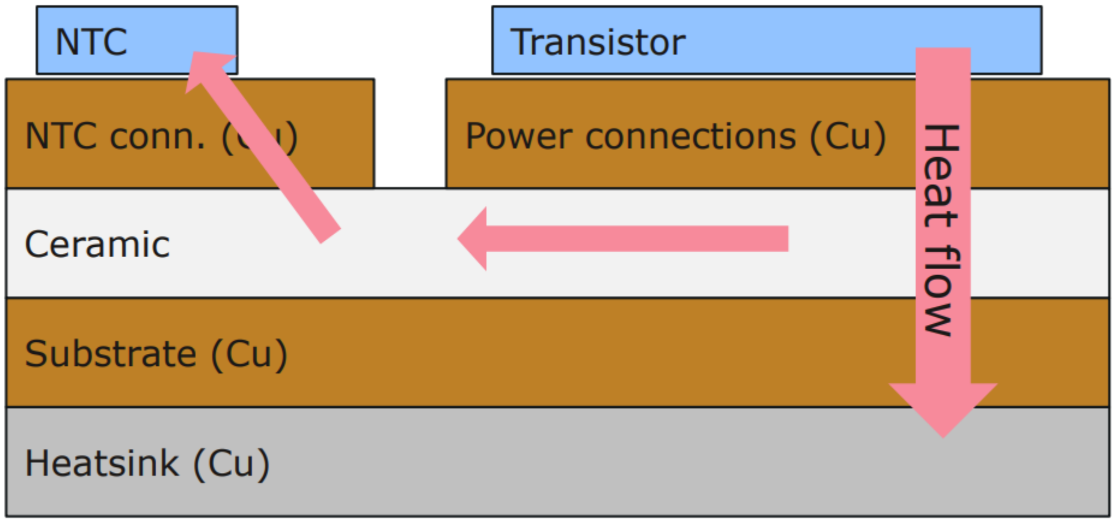


Figure 2 – Simplified structure of the studied chip. (from company slides)

The lumped parameter model, discussed in the 1st section of Unit 1, is particularly advantageous in this context, allowing the complex thermal system to be reduced into discrete, manageable elements such as thermal resistances and capacitances. This simplified structure is represented in both Foster and Cauer thermal networks, each offering unique insights into the thermal behavior of the junction. Then as outlined in the 2nd section, constructing an accurate thermal model involves creating a Linear Time-Invariant (LTI) system, which can be simulated in continuous and discrete domains using MATLAB and Simulink tools. These simulations allow for an analysis of the model's behavior over time, with initial results discussed in third section, where continuous and discrete-time domain simulations are performed and validated.

Throughout the course of this study, several assumptions and simplifications have been made to manage the inherent complexity of the system while maintaining the model’s lightweight and real-time functionality. Despite these approximations, the general structure of the model remains highly adaptable. It can be tailored to different PCB configurations, provided the core principles of simplification are applicable. This flexibility enhances the model's utility across a range of applications, making it a valuable tool for engineers working on diverse PCB designs.

On the other hand, in unit 2, the thesis introduces a more complex assumed model, exploring the limits of the simplified approach by applying advanced modeling techniques such as bond graph representations. This model involves constructing more detailed system matrices and tweaking preliminary component assumptions to improve accuracy. The application of a hybrid Foster-Cauer structure in this case allows for a more detailed representation of lateral and vertical heat flow within the MOSFET junction, ultimately providing a higher degree of predictive power.

Ultimately, this paper aims to contribute to the growing body of knowledge on thermal modeling by providing a practical, scalable solution that addresses one of the key challenges in modern electronics design—managing heat in increasingly compact and complex systems. The techniques and methodologies discussed here can be applied to a wide range of technologies, from consumer electronics to automotive and aerospace systems, offering a way to improve thermal performance and enhance the overall reliability of advanced electronic devices. The study concludes with reflections on how future research could build on this work by refining the model further, improving computational efficiency, and applying these insights to emerging technologies, as highlighted in the final chapter of the thesis. Throughout the study, a lot of different symbols/letters are used in equations and explanations. The explanations of these symbols can be found in **Table 1**.

Heat Transfer Equations	
$\dot{Q}$	*Rate of heat transfer*
$T_{surf}, T_{envr}, T_{surr}$	*Temperature of surface, environment, or surrounding*
k	*Thermal conductivity of the material*
L	*Thickness of the material*
h_{conv}, h_{τ}	*Heat transfer coefficient through convection, conduction*
A_{surf}	*Surface area through which convection occurs*
ϵ	*Emissivity of the surface*
σ	*Stefan-Boltzmann constant*
Electric Domain (Circuit theory)	
V	*Voltage*
I	*Current*
R	*Resistor (Resistance)*
C	*Capacitor (Capacitance)*
τ	*Time Constant*
t	*time*
Linear Time Invariant Systems (LTI)	
$x(t)$	*state*
$u(t)$	*input*
LQ Control Theory	
$t \in [0, t_f]$	*Time horizon over which the optimization is performed*
$J(u)$	*This is the cost function that we want to minimize. It quantifies the performance of the control strategy, typically involving terms that penalize deviations from desired behavior and the control effort.*
S	*Symmetric positive semi-definite matrix that weighs the final state in the cost function. It penalizes the state at the final time* t_f
Q	*Symmetric positive semi-definite matrix that weights the state in the integral cost function. It penalizes the state throughout the time horizon.*
R	*Symmetric positive definite matrix that weighs the control input* $u(t)$ *in the cost function. It penalizes excessive control effort.*
τ	*A dummy variable used for integration over time in the cost function.*
Bond Graph Representation	
e	*Effort*
f	*Flow*
R	*Resistive component*
C	*Capacitive component*
S_f, S_e	*Source of flow, Source of effort*
Some Subscripts used (X)	
X_{NTC}	*Referring to NTC (Negative Temperature Coefficient Resistor)*
X_J, X_{junc}	*Referring to Junction*
X_{cer}	*Referring to Ceramic*
X_{Real}, X_{EST}	*Real/Recorded Value, Estimated Value*
Other Annotations Used	
$\frac{d(\ )}{dt}, \dot{\ }$	*Time derivative or Rate*

Table 1 - Table of Symbols.

State of the Art

The development of accurate and efficient thermal models is critical for modern electronic systems, particularly as devices become more compact and power-dense. Traditional thermal modeling approaches have evolved from early methods like finite element analysis (FEA), which is widely used for detailed thermal simulations, to more computationally efficient techniques tailored for real-time applications. FEA, while precise, is computationally intensive and may not be ideal for real-time applications due to the high level of granularity required in modeling the geometry and material properties of systems (Zienkiewicz, Taylor, & Zhu, 2013). In response, alternative methods, such as lumped parameter models, have emerged for applications where computational efficiency is key.

Among these models, lumped parameter models, particularly Foster and Cauer networks, have gained prominence due to their ability to simplify complex thermal systems into manageable components while retaining predictive accuracy. Foster networks model thermal behavior by representing heat flow through thermal resistances and capacitances lumped into a simplified network structure, a practical method for simulating transient thermal behavior in systems with a simple thermal gradient (Kraus, Aziz, & Welty, 2011). Cauer networks, on the other hand, provide a more detailed representation of the thermal system by modeling both vertical and lateral heat flow through a multi-layer structure (Piumatti, Quitadamo, Reorda, & Fiori, 2020). Cauer networks, though more complex than Foster networks, are essential in systems where the spatial distribution of heat is critical, such as multi-layered PCBs or chips with significant vertical heat conduction pathways.In recent years, the need for real-time thermal monitoring in high-power applications, such as electric vehicle (EV) inverters, has driven the development of lightweight models capable of running on microcontrollers. These models are essential for predicting critical junction temperatures, such as those in MOSFET transistors, which are especially susceptible to thermal degradation. Junction temperatures have become a focal point of thermal management, as overheating can severely impact the performance, lifespan, and safety of these components. To address these challenges, grey-box modeling approaches have emerged as powerful tools, blending theoretical knowledge with empirical data. This hybrid method balances accuracy with computational efficiency, making it suitable for real-time applications where processing power is limited.

In parallel, control systems such as Kalman filters and Linear Quadratic (LQ) control have been integrated into thermal models to enhance predictive accuracy. These methods adjust model parameters dynamically, allowing real-time calibration based on sensor inputs like Negative Temperature Coefficient (NTC) thermistors. This real-time adaptability is crucial for maintaining the reliability of thermal predictions under fluctuating operational conditions. The combination of LTI system theory, predictive control algorithms, and lumped parameter models forms the backbone of modern thermal management strategies in complex systems such as those used in electric vehicles and other high-power electronics.

This thesis builds upon these state-of-the-art methodologies by constructing an LTI-based thermal model using MATLAB, designed specifically to estimate the junction temperature of MOSFET transistors in EV inverter PCBs. It employs two approaches: a simplified Cauer network and a hybrid Foster-Cauer structure (Complex Model), each tailored to specific structural and computational constraints. The Cauer network is particularly suited for capturing vertical heat flow in the PCB, while the hybrid structure incorporates aspects of both Foster and Cauer models to balance simplicity and accuracy. The data used in this study are drawn from thermal simulations provided by Ideas & Motion Torino, the industrial partner in this research. It is important to note that the given data does not correspond to the assumed/simplified structures of the PCB but will serve as a reference for realistic PCB behavior. So naturally, there will be a mismatch between the given data and the outputs presented from the built model. These errors do not compromise the validity of the model or its predictive power, and so accordingly the mismatch during simulations will be adjusted and mitigated through rigorous curve matching techniques.

The study ultimately demonstrates the robustness of the developed model, showing that it is a reliable and computationally efficient tool for real-time thermal estimation on microcontrollers, with potential applications in electric vehicles and other high-power electronic systems.

UNIT 1: Simplified Model Methodology

1. Lumped Parameter/Element Model

The lumped parameter model (also called lumped-parameter model or lumped-component model) is a widely-used modeling technique, especially in electrical and thermal systems, to simplify the representation of complex physical systems. In the electrical domain, this technique represents different components of a system, such as resistances, capacitances, and inductors, as a series of lumped elements. These lumped elements approximate the system's behavior by concentrating components into discrete points and describing them with idealized mathematical models. As highlighted by (Infineon Technologies AG, 2020), this approach contrasts with the distributed parameter model, where system behavior is distributed spatially, considering variations across the entire component structure.

This lumped modeling technique is especially useful when systems need to be simplified to facilitate real-time simulation, design optimization, or control system integration. By reducing the complexity of the system, engineers can approximate the dynamics of various components, whether electrical, mechanical, or thermal, while maintaining sufficient accuracy for practical applications (Ogata, 2010).

When applied in the thermal domain, the lumped parameter method reduces a thermal system into discrete lumps, each representing a region with negligible internal temperature differences. These lumps effectively emulate the system's thermal capacitance. Meanwhile, components that generate temperature differences due to heat transfer via conduction or convection are modeled as thermal resistors. This approach allows engineers to simulate the dynamic heat transfer processes within a system without requiring the computationally intensive solution of partial differential equations (PDEs), which are typically used in distributed models.

Through this methodology, the thermal behavior of systems can be represented using two primary network topologies: the Cauer thermal network and the Foster thermal network, both of which can be visualized in **Figure 3**. The Cauer network is derived from the physical structure of the system, with each resistor-capacitor pair corresponding to a specific physical layer or component in the system. This topology is particularly beneficial when a detailed understanding of the system's internal structure is available (Piumatti, Quitadamo, Reorda, & Fiori, 2020). On the other hand, the Foster network models the system using empirical data from thermal measurements. It represents thermal behavior as a succession of RC cells arranged in parallel, each representing observed trends in different parts of the system, rather than specific physical structures.

The simplification introduced by the lumped parameter technique reduces the state-space dimension of the system, transforming the PDEs of continuous time and space models into ordinary differential equations (ODEs). These ODEs provide a more tractable way to model the system's behavior with a finite set of parameters, making them suitable for real-time applications, especially in embedded systems where computational resources are limited (Kraus, Aziz, & Welty, 2011).

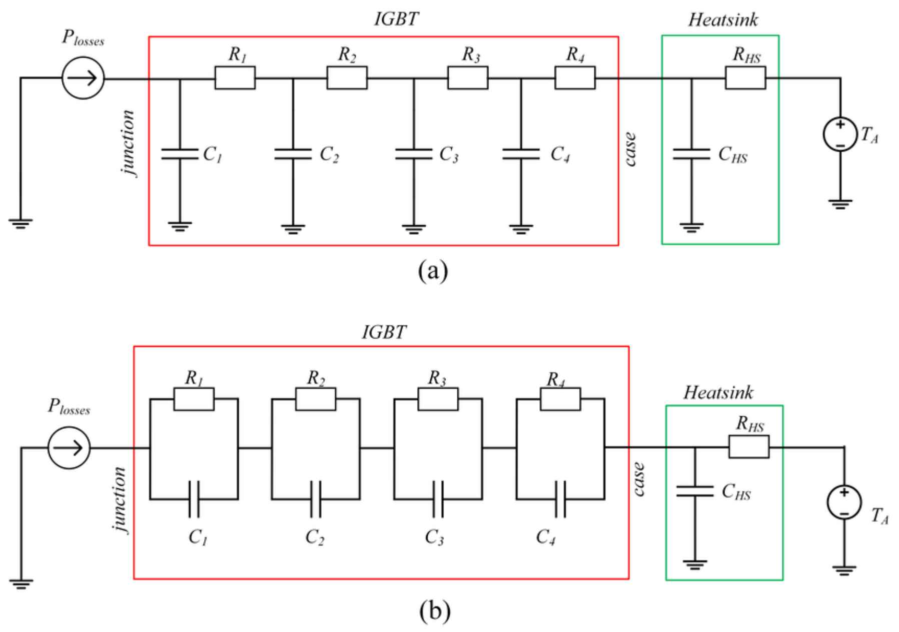
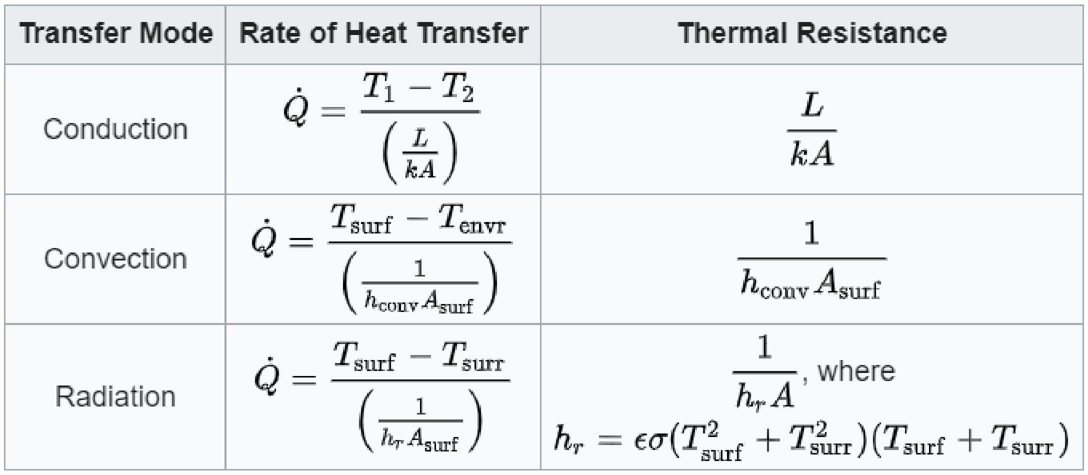


Figure 3 - (a) Cauer Thermal Network and (b) Foster Thermal Network. (Giannakis & Peftitsis, 2019)



Transfer Mode	Rate of Heat Transfer	Thermal Resistance
Conduction	$\dot{Q} = \frac{T_1 - T_2}{\left(\frac{L}{kA}\right)}$	$\frac{L}{kA}$
Convection	$\dot{Q} = \frac{T_{\text{surf}} - T_{\text{envr}}}{\left(\frac{1}{h_{\text{conv}} A_{\text{surf}}}\right)}$	$\frac{1}{h_{\text{conv}} A_{\text{surf}}}$
Radiation	$\dot{Q} = \frac{T_{\text{surf}} - T_{\text{surr}}}{\left(\frac{1}{h_r A_{\text{surf}}}\right)}$	$\frac{1}{h_r A}$, where $h_r = \epsilon\sigma(T_{\text{surf}}^2 + T_{\text{surr}}^2)(T_{\text{surf}} + T_{\text{surr}})$

Figure 4 - Equations for different heat transfer modes and their thermal resistances.

Source: Wikipedia (https://en.wikipedia.org/w/index.php?title=Lumped-element_model&oldid=1226659382)

In the Cauer approach, which is particularly useful for this study, the thermal model is derived from a detailed analysis of the system's physical structure. Each resistor-capacitor (RC) couple directly maps to a physical feature within the system, such as a specific layer in a PCB or a heat sink. This method allows for greater precision in modeling how heat flows through individual components. In retrospect, the Foster network approach is more abstract, focusing on the overall thermal response rather than the detailed physical arrangement, making it better suited for empirical or black-box modeling scenarios. Given the grey-box nature of this study, where some internal component details are known while others are estimated or inferred, the Cauer approach is deemed the most appropriate.

According to the data provided by Ideas & Motion Torino, the chips being modeled in this study can be simplified and represented using the Cauer approach, as seen in **Figure 6**.

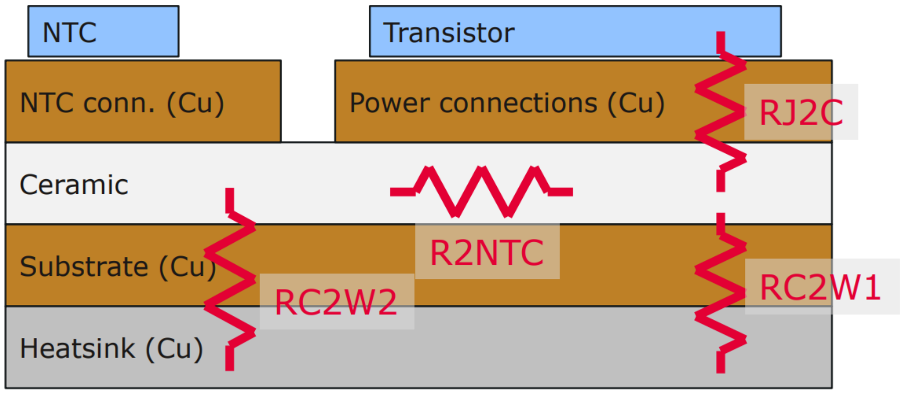


Figure 5 - Chip structure with thermal equivalent components. (from company slides)

In this case, the thermal system can be simplified into three key temperatures along the heat flow path from the transistor to the NTC resistor and the heatsink:

- T_J: Junction temperature
- T_{cer}: Ceramic layer temperature
- T_{NTC}: temperature of the NTC resistor.

These temperatures are essential to understanding how heat dissipates from the MOSFET junction to the PCB and how the NTC helps regulate the thermal control system. Accordingly, an equivalent thermal circuit assuming that the transistor behaves as the source of the heat is modeled:

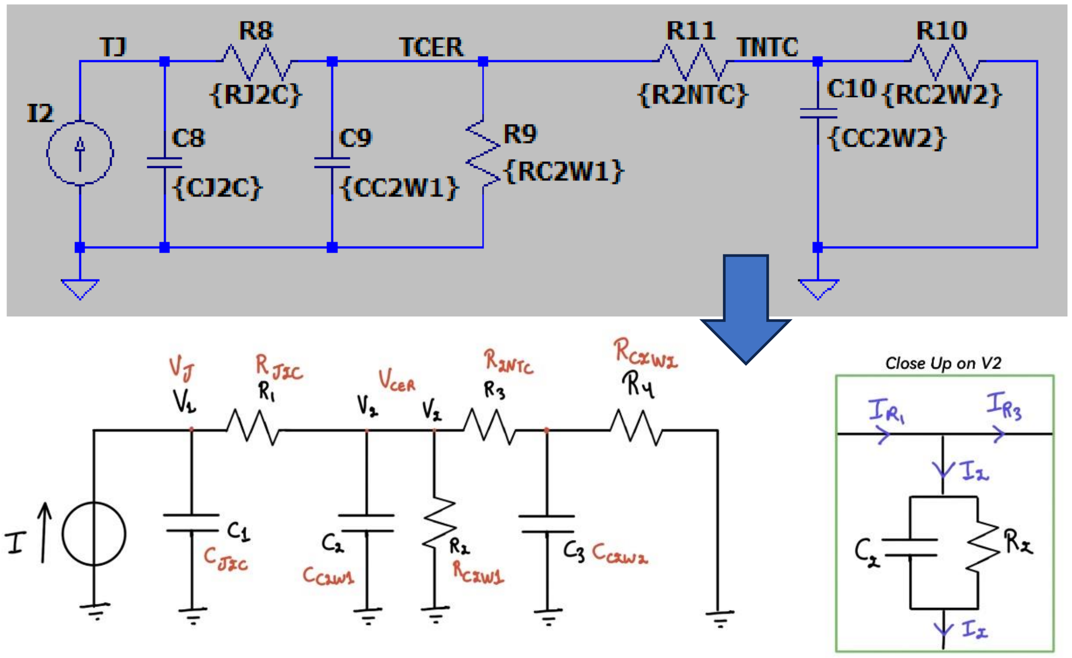


Figure 6 - (Top) LTspice equivalent circuit and (Bottom) Simplified parameter model to be used for calculations.
Note: Parameters in black are simplified naming of the red names of parameters which are provided in company slides.

The equivalent thermal circuit shown in **Figure 6** models the system using lumped parameter techniques. This equivalent circuit is constructed within LTspice, a widely-used tool for simulating electronic circuits, to study the system's dynamics. It is important to note that the circuit used in this study is a generalized assumption of the real structure of the PCB. Due to limitations in the available data, the real structure of the PCB is not fully represented. However, the simplified model adequately captures the key thermal dynamics needed for this case study. The obtained circuit which can be simulated in the electric domain can be instantaneously converted to the thermal domain through the following trend:

Current (I) → Power loss (P).
Voltage (V) → Temperature (T).
Resistor (R) → Thermal Resistance (Rth).
Capacitor (C) → Thermal Capacitance (Cth).

Once this equivalence between the electric and thermal domains is established, the next step is to derive the mathematical equations governing the dynamics of the system. These equations describe the rate of change of thermal parameters, such as temperature and heat flow. One effective approach is the construction of a Linear Time-Invariant (LTI) system, which provides a powerful framework for analyzing and simulating thermal dynamics. LTI systems are particularly advantageous because they are mathematically well-defined and can be simulated efficiently using tools like MATLAB and Simulink, allowing for detailed analysis of system behavior under different operating conditions (Ogata, 2010).

By leveraging the lumped parameter model, this study provides a robust framework for thermal analysis that can be applied to other PCB designs, provided the necessary simplifications are appropriate. The adaptability of this modeling approach ensures its utility across a wide range of applications in electronics design, especially as the need for compact and efficient systems continues to grow.

2. LTI system Construction

A system can be classified into two broad categories: continuous time systems and discrete time systems. Continuous time systems deal with signals that vary continuously over time, meaning the system's states can change values at every instant of time. These systems are described using differential equations. On the other hand, discrete time systems handle signals that change at specific, separate time intervals. In these systems, states update at distinct moments, usually represented as integer multiples of a time step, and are typically modeled using difference equations.

Both continuous and discrete time systems exhibit fundamental properties that guide their behavior and how they are analyzed. Two critical properties that play an essential role in system analysis are linearity and time invariance. Linearity refers to a system's ability to respond to a weighted combination of inputs in a manner proportional to their respective weights. Time invariance, on the other hand, means that the system's behavior and characteristics do not change over time—if the same input is applied at different times, the output remains the same except for a shift corresponding to the input shift.

When a system possesses both linearity and time invariance, it is called a linear time-invariant (LTI) system. LTI systems are crucial in both signal processing and control theory because they exhibit predictable and well-understood behavior, allowing for straightforward mathematical analysis and solution methods (Oppenheim & Willsky, 1997).

In a more detailed definition, an LTI system is a system whose output for any linear combination of inputs is the same as the linear combination of the system's individual responses to those inputs. This definition applies to both continuous and discrete domains. For example, if a system receives two inputs, the output will be the sum of the individual outputs for each input. This property of superposition is central to LTI systems, making them easier to model and predict.

As outlined by (Chumbley, Areias, Williams, & Khim, s.d.), time invariance means that the output for a given input does not depend on when the input is applied. In other words, if the input signal is delayed by a certain amount of time, the output will experience the same delay without altering its form. This property ensures that the system's behavior remains consistent over time.

In the continuous time domain, an LTI system can be represented in several ways. One common representation is the transfer function, which is the ratio of the system's output to its input, typically expressed in the Laplace domain. This method is particularly useful for analyzing how different frequency components of an input signal are modified by the system. Another important representation is the state-space form, which is expressed as a set of first-order differential equations:

$\frac{dx(t)}{dt} = A\,x(t) + B\,u(t), y(t) = C\,x(t) + D\,u(t)$, such that:

$x(t)$→ time dependent states.

$y(t)$→ time dependent output.

$u(t)$→ time dependent input.

A, B, C, D→ Matrices containing the constant coefficients of the variables in LTI.

The equations above, also called as state space representation of LTI, find the change of the state over time (derivative $\dot{x}$) and the output $y(t)$ as a function of the states $x(t)$ and inputs $u(t)$. They can also be later converted into the discrete domain when describing the system behavior in presence of real electronic components. This state-space representation is powerful because it allows the analysis of multi-input, multi-output systems and can be used for both continuous and discrete systems.

In the discrete time domain, the equivalent equations for an LTI system involve difference equations instead of differential equations. The system can also be described by its z-transform, which is the discrete-time counterpart of the Laplace transform. This allows similar analysis of how the system responds to various input frequencies.

LTI systems are particularly important because of their mathematical simplicity and the rich set of tools available for their analysis. Methods such as convolution, transfer functions, and state-space representations allow for precise predictions of system behavior, making LTI systems a cornerstone of modern signal processing, control engineering, and communications.

Moving on, the next step is to now derive the equations from **Figure 6**, and arrange them in a form identical to the one previously defined:

The state vector is set to $x = [V1, V2, V3] = \left[V_{junction}, V_{ceramic}, V_{NTC}\right]$, input vector $u = [I]$.

Therefore,

$\rightarrow \frac{dx}{dt} = \left[\frac{dx1}{dt}, \frac{dx2}{dt}, \frac{dx3}{dt}\right] = \left[\frac{dV1}{dt}, \frac{dV2}{dt}, \frac{dV3}{dt}\right]$=$[\dot{x_1}, \dot{x_2}, \dot{x_3}]$

**Applying Kirchoff's current law describing the node including current through C1:*

$Ic1 = I - \frac{V1-V2}{R1}$, *knowing that* $Ic1 = \frac{dVc1}{dt} = \frac{dV1}{dt}$

So:

$$\rightarrow \frac{dV1}{dt} = \left(-\frac{1}{R1 * C1}\right) V1 + \left(\frac{1}{R1 * C1}\right) V2 + \left(\frac{1}{C1}\right) I = \dot{x}1$$

**Applying Kirchoff's current law describing the node including current through C2:*

$I2 = \frac{V1-V2}{R1} - \frac{V2-V3}{R2}$, *but* $I2 = Ic2 + Ir2\ (parallel\ junction)$

$Ic2 + Ir2 = \frac{V1-V2}{R1} - \frac{V2-V3}{R2} \rightarrow C2 * \frac{dV2}{dt} + \frac{V2}{R2} = \frac{V1-V2}{R1} - \frac{V2-V3}{R2},$

Rearranging the terms into proper form:

$$\rightarrow \frac{dV2}{dt} = \left(\frac{1}{R1 * C1}\right) V1 + \left(-\frac{1}{R1 * C1} - \frac{1}{R2 * C2} - \frac{1}{R3 * C2}\right) V2 + \left(\frac{1}{R1 * C1}\right) V3 = \dot{x}2$$

**Applying Kirchoff's current law describing the node including current through C2:*

$Ic3 = \frac{V2-V3}{R3} - \frac{V3}{R4}$, *knowing that* $Ic3 = \frac{dVc3}{dt} = \frac{dV3}{dt}$

So:

$$\rightarrow \frac{dV3}{dt} = \left(\frac{1}{R3 * C3}\right) V2 + \left(-\frac{1}{R3 * C3} - \frac{1}{R4 * C3}\right) V3 = \dot{x}3$$

After acquiring the derivative state vector $\dot{x} = [\dot{x}1, \dot{x}2, \dot{x}3]$, the following step is building the coefficient matrices A, B, C, D:

$$A = \begin{bmatrix} -\frac{1}{R1 * C1} & \frac{1}{R1 * C1} & 0 \\ 0 & \frac{1}{R3 * C3} & -\frac{1}{R3 * C3} - \frac{1}{R4 * C3} \\ \frac{1}{R1 * C2} & -\frac{1}{R1 * C2} - \frac{1}{R2 * C2} - \frac{1}{R3 * C2} & \frac{1}{R3 * C2} \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{1}{C1} \\ 0 \\ 0 \end{bmatrix}, C = [1 \quad 0 \quad 0], D = 0.$$

The LTI system is now built successfully with all its elements defined properly. The next would be simulating it through MATLAB/Simulink software given different inputs while comparing it to given references/data.

3. MATLAB & Simulink Simulations

A priori-information analysis:

For accurately testing the dynamics of the system in ideal conditions, computer software is used. With the help of MATLAB for initialization and parameter loading, and Simulink for building the thermal model, preliminary results can be studied.

As a 1st step, the veracity of the LTI model is double checked within the original time domain. So, with the use of the company provided simulations (check **Figure 6**) an LTspice simulation is run identical to the equivalent lumped model of the chip. The LTspice provides the electric circuit equivalent of the simplified thermal model of the chip structure to be studied.

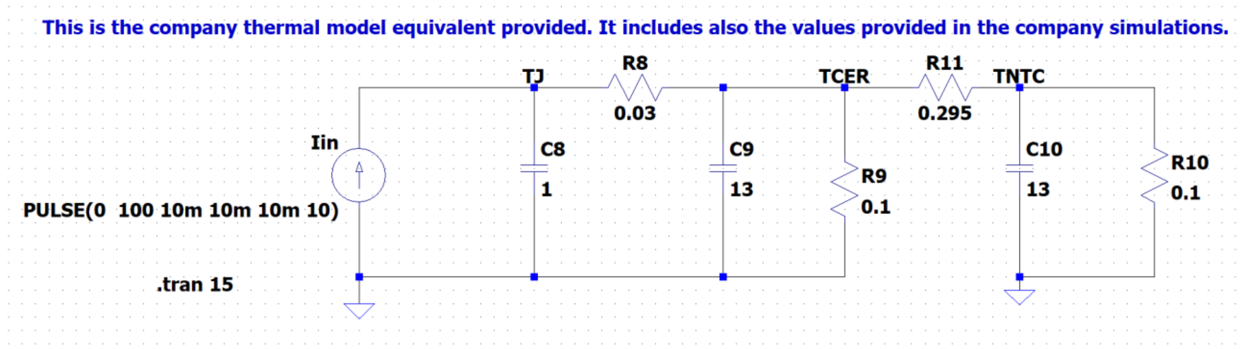

This is the company thermal model equivalent provided. It includes also the values provided in the company simulations.
TJ
R8
0.03
TCER
R11
0.295
TNTC
Iin
C8
1
C9
13
R9
0.1
C10
13
R10
0.1
PULSE(0 100 10m 10m 10m 10)
.tran 15

Figure 7 - Recreation of company LTspice simulation.

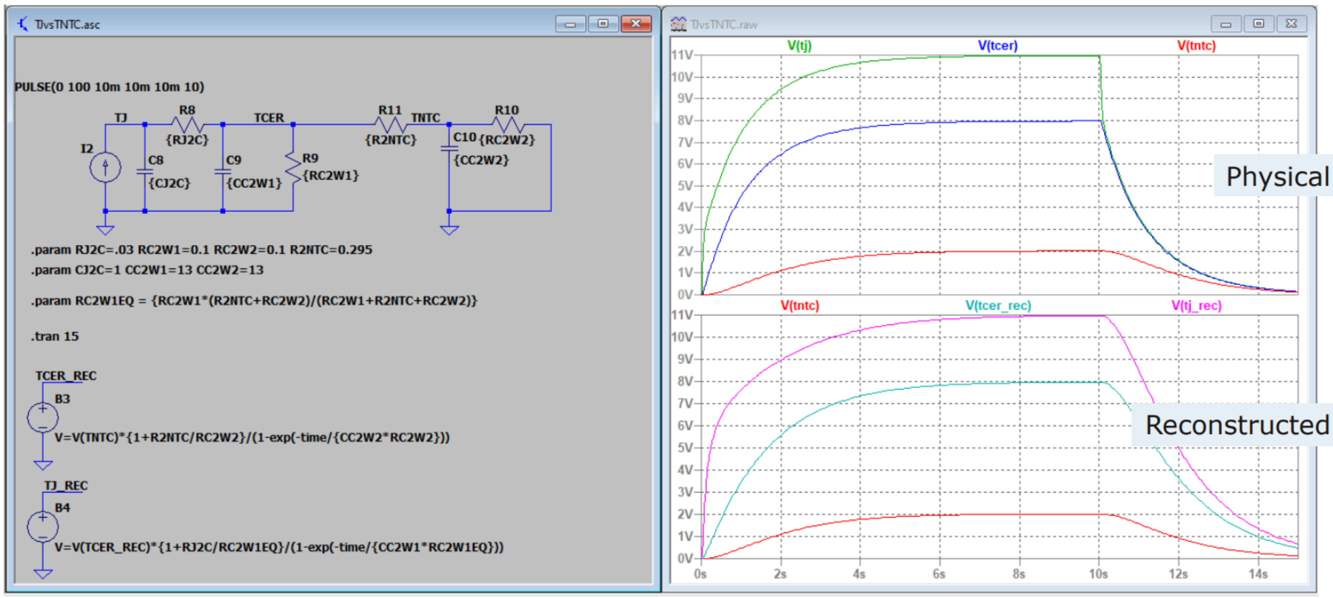
The resulting circuit, which can be observed in **Figure 7** of the LTspice simulations, show exponential increase and decrease of the respective temperatures of the junction, ceramic layer, and NTC during the transient phase of the simulation (plots can be visualized in **Figure 8**). This sharp behavior is due to the step (pulse) behavior provided by the input current. The behavior provides problems if recreated through the simplified acquired equations provided by the company.

The equations seen below (equations (1) and (2) retrieved from the company's slides and own study),

$$T_{CER} = T_{NTC,ss} = \frac{T_{NTC}(t)}{1-\exp\left(-\frac{t}{\tau_{2NTC}}\right)} \cdot G_{VD2} \qquad (1)$$

$$T_J = T_{CER,ss} = \frac{T_{CER}(t)}{1-\exp\left(-\frac{t}{\tau_{J2C}}\right)} \cdot G_{VD1} \qquad (2)$$

which can be programmed into LTspice for testing provides as seen in **Figure 8** provides a slow estimated signal that fails during the transient phase. This provides a problem already without taking into consideration any real-life unideal conditions such as: sensor noise, sensor frequency delay of NTC, inaccuracies in the NTC value, ambient temperature effect. Moreover, this estimation neglects the provided input power which is a known and reliable quantity. In fact, the supplied input power is a more reliable parameter in comparison to the NTC resistor temperature value.



```
PULSE(0 100 10m 10m 10m 10)

.param RJ2C=.03 RC2W1=0.1 RC2W2=0.1 R2NTC=0.295
.param CJ2C=1 CC2W1=13 CC2W2=13

.param RC2W1EQ = {RC2W1*(R2NTC+RC2W2)/(RC2W1+R2NTC+RC2W2)}

.tran 15

V=V(TNTC)*{1+R2NTC/RC2W2}/(1-exp(-time/{CC2W2*RC2W2}))

V=V(TCER_REC)*{1+RJ2C/RC2W1EQ}/(1-exp(-time/{CC2W1*RC2W1EQ}))
```

Figure 8 - Results of the LTspice simulation. (slide 113 of company's given slides)

A more accurate estimation would be one that could at least take the input power of the chip into consideration, which is then corrected by some factor by the value of the NTC temperature. To progressively reach this goal, the simulation of the previously calculated LTI is needed which uses solely the input power/current to estimate all the needed temperatures or respective voltages in the electric domain.

First LTI simulation within t-Domain (Continuous):

Using Simulink, the initial model is built (**Figure 9**) which takes into consideration only the input current. The LTI block, with its parameters preloaded in MATLAB, uses the received input to calculate the states x_1, x_2, and x_3. So, since the grey box model doesn't even have proper data for the equivalent thermal resistances and capacitances of the simplified circuit, generic values are placed inspired from another simple PCB structure found in previous data. As the study progresses, the components' values are replaced to match reference output curves for more accurate and reliable numbers.

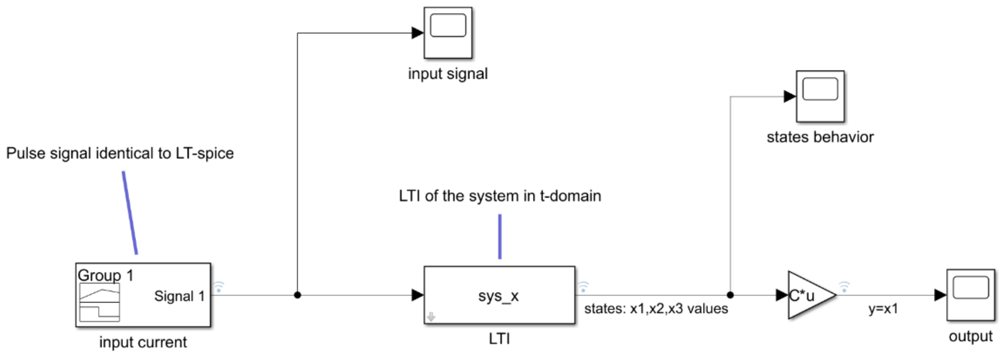


Figure 9 - Simulink simulation of preliminary LTI model.

The calculated matrices along with the values of the given parameters (in MATLAB) are seen in **Table 2**.

```
%GIVEN PARAMETERS (ASSUMED)
R_J2C=0.03;
R_C2W1=0.1;
R_C2W2=0.1;
R_2NTC=0.295;
C_J2C=1;
C_C2W1=13;
C_C2W2=13;
```

```
%LTI parameters build
A=[-1/(R_J2C*C_J2C) 1/(R_J2C*C_J2C) 0;...
1/(C_C2W1*R_J2C) -1/(C_C2W1*R_J2C)-
1/(R_C2W1*C_C2W1)-1/(R_2NTC*C_C2W1)
1/(R_2NTC*C_C2W1);...
0 1/(C_C2W2*R_2NTC) -1/(R_2NTC*C_C2W2)-
1/(R_C2W2*C_C2W2)];
B=[1/C_J2C;0;0];
C=[1 0 0];
D=0;
x0=[0;0;0]; % assumed Initial Conditions
sys=ss(A,B,C,D);

%USED in SIMULINK LTI_BLOCK
sys_x=ss(A,B,eye(3),D);
```

Table 2 - MATLAB code for LTI building and preliminary simulations.

Results: $A = \begin{bmatrix} -33.3333 & 33.3333 & 0 \\ 2.5641 & -3.5941 & 0.2608 \\ 0 & 0.2608 & -1.0300 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, C = [1 \quad 0 \quad 0], D = 0;$

The obtained state along with the provided input curves can be observed in **Figure 10**. The obtained behavior is identical to the physical LTspice results of **Figure 8**. This confirms the veracity of the LTI matrix structure and values. However, if a real-life controller is to be used, then there is a requirement for the simulations to be run in the discrete domain. And by that, the next step has been defined: Simulations in Discrete Domain. However, before the dive into discrete domain commences, a deeper analysis of the system dynamics has to be done within the continuous domain: Continuous-Peak-Continuous Simulation.

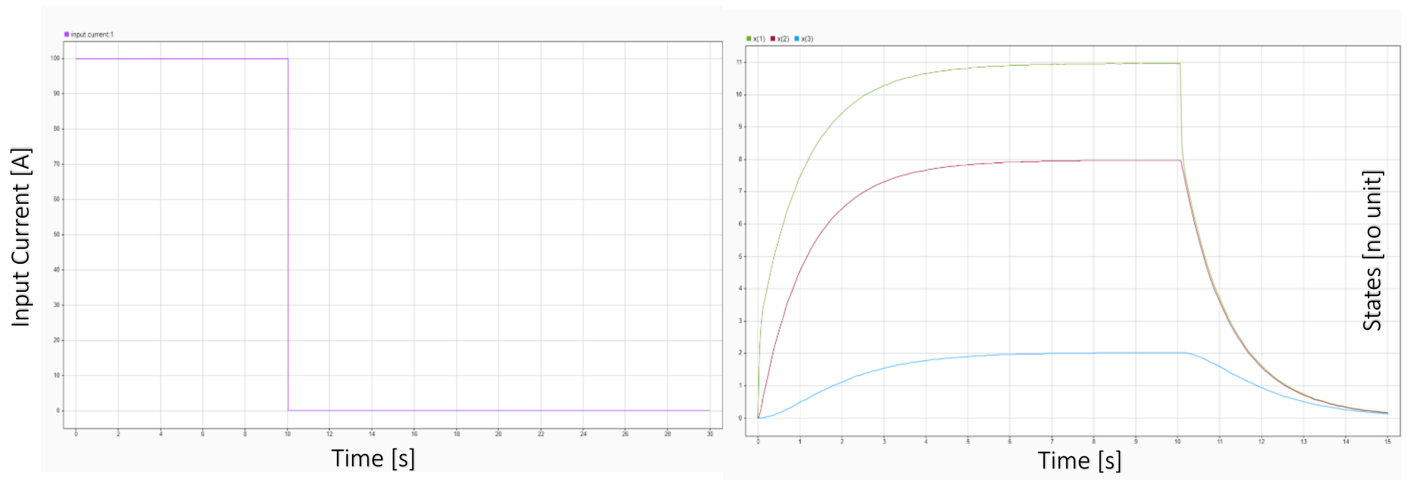


Figure 10 - (Left) Input current, (Right) States in t-domain LTI.

Continuous-Peak-Continuous Simulation:

As the name suggests, in this step the goal is to visualize the behavior of the LTI in the presence of continuous-peak-continuous input behavior. This emulates real life behavior of chips and helps study the temperature wear to analyze the life cycle of electronics.

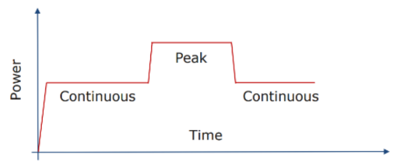


Figure 11 - Continuous-Peak-Continuous power configuration.

So, using the same configuration of **Figure 9**, the signal builder input current curve block is modified to match the shape of **Figure 11**. The output can be visualized in **Figure 12** which shows a plateau at the peak of each of the 3 phases. The sharp step increase and decrease of input is translated into exponential behavior which especially extreme during the 1st warm-up period (from off state into 1st continuous phase). This sharp increase/decrease can present a challenge when discretized in electronic components. Due to this, the next step of the study is to build the discretized model under different sampling times and fine tune it to fit the requirements accordingly.

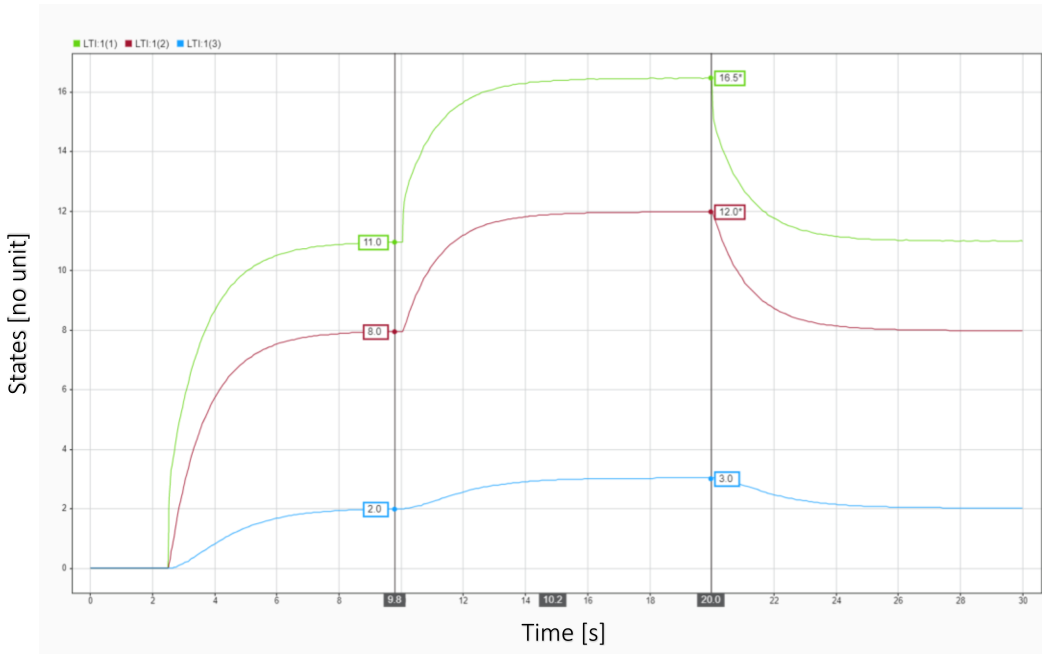


Figure 12 - Output of states (green x1, red x2, blue x3) from Continuous-Peak-Continuous input configuration.

Translation into the Discrete-time Domain (Zero-Order-Hold):

The first step into getting more realistic data is to move into the discrete domain. There are many ways to do that with the used software (MATLAB/Simulink) including but not limited to the Zero-Order-Hold block. The block is inserted into the simulation space and the previously constructed system is rebuilt within the respective discrete domain with built-in MATLAB function (dt=c2d(LTI_system,SamplingTime,'zoh'). For the function mentioned, a sampling time needs to be defined. The change of behavior with respect to different sampling times is studied along with its effect on the accuracy of the results. The following are the MATLAB initialization commands used:

```
%Now we move to discretise the model
Ts=1/100; %will be modified
dt=c2d(sys,Ts,'zoh');
Ad=dt.A;
Bd=dt.B;
Cd=dt.C;
Dd=dt.D;
sys_dt=ss(Ad,Bd,eye(3),Dd); %discretized
```

Results: **Rounded results due to precision errors and format*

$$Ad = \begin{bmatrix} 0.7199 & 0.2786 & 0.0004 \\ 0.0214 & 0.9684 & 0.0026 \\ 0.0000 & 0.0026 & 0.9898 \end{bmatrix}$$

$$Bd = \begin{bmatrix} 0.0085 \\ 0.0001 \\ 0.0000 \end{bmatrix}, Cd = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

Now, regarding the Simulink model 2 choices are available, either the built-in LTI function is used with the discrete option selected or the LTI is manually through a closed feedback loop in order to have control over each component (matrix) independently for later modification according to (Rowell). Due to the requirement of fine tuning and modification later, the latter option is selected which gives rise to **Figure 13**.

The LTI system in discrete domain looks as follows:

$$x(k+1) = Ad * x(k) + Bd * u(k), \qquad y = Cd(k) * x(k) + Dd * u(k)$$

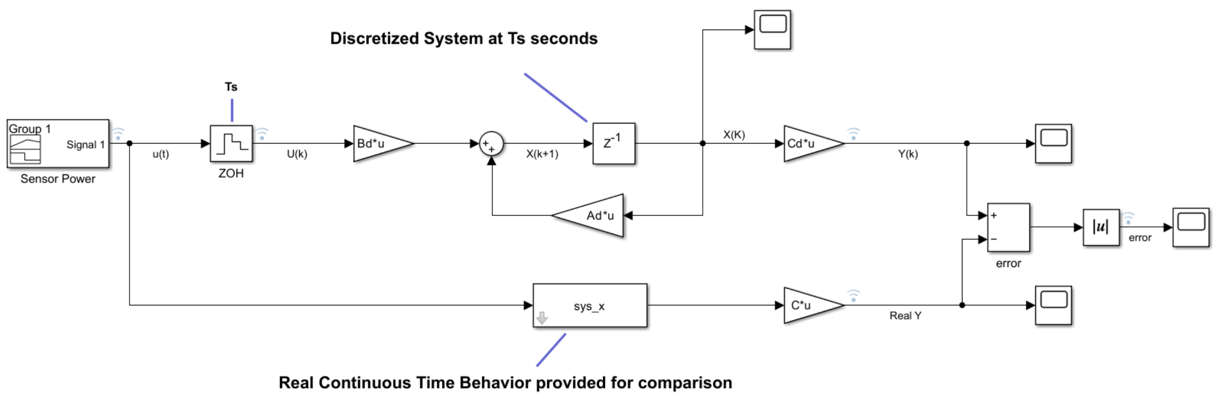


Figure 13 - Simulink Model of discretized LTI (up) being compared to continuous time domain LTI (down).

Note: (Rowell) was used as a reference to build the LTI.

The continuous LTI in t-domain that was found in the previous section is added to have a reference for comparison. Then a sum block is inserted to calculate the difference. Afterwards, the system is simulated under different sampling times to observe the effect of the controller refreshing frequency on the accuracy of the results.

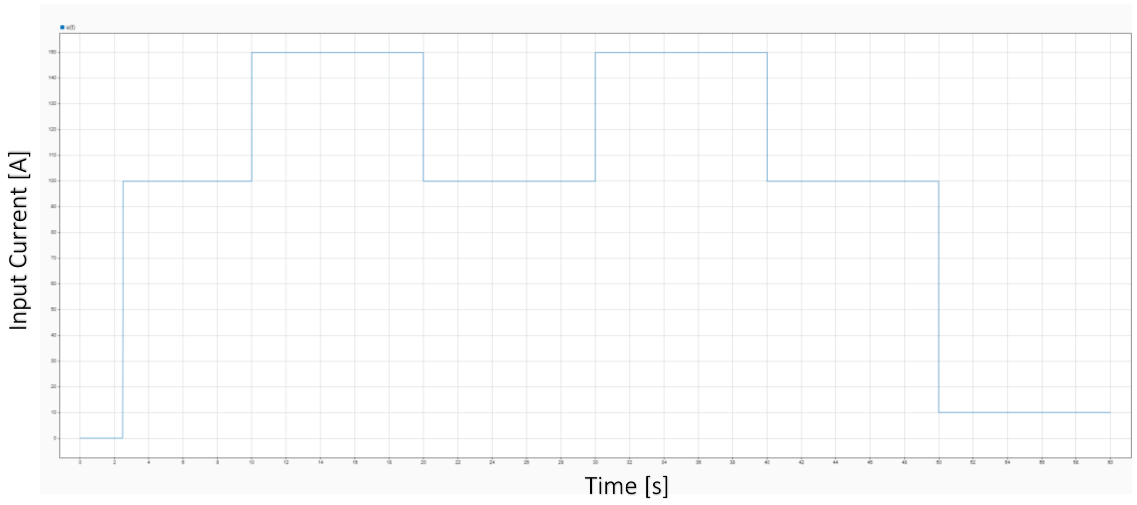


Figure 14 - Input signal generated for the study of the discretized LTI.

Using the signal generator block in Simulink, a signal that varies between 150 peak and 100 steady state units is generated. That signal will be discretized over Ts = 1; 1/10; 1/100 in separate simulations and inputted into a discrete time LTI (closed state-feedback system) outputting the Tj ($x1$) state. Results of the continuous and discrete LTI's simulated in tandem over the 3 different sampling times are reported in **Figure 15**. From the latter's simulation results it can be concluded that any sampling time faster than 0.1 seconds is fast enough to avoid major errors (errors within 10th place decimal delay). The assumed sampling domain for the grey box model will be 1/100 from now on for the controller. It must be kept in mind however that the sensor rate of data retrieval will be much slower, which will cause some problems later during the design. After discretization, it is time to begin segregating the Simulink model to clearly define a controller model.

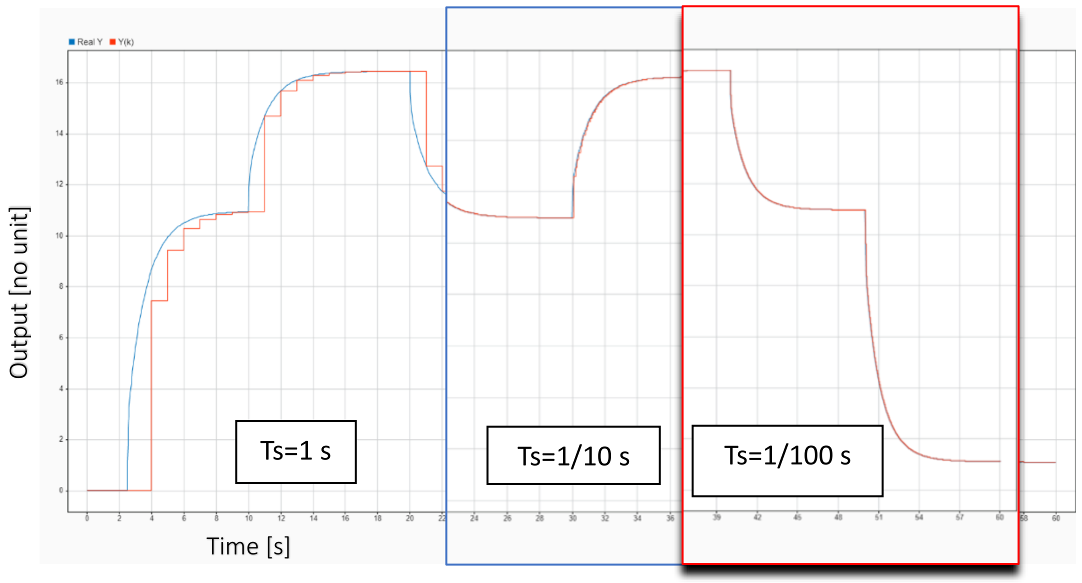


Figure 15 - Tj (x3) values for continuous domain (blue) and discrete domain (red) over Ts= 1, 0.1, 0.01 seconds respectively.

Model encapsulation:

This step focuses on distinguishing a clear controller model from the simulation space in Simulink that will be tuned and edited for later code generation. As the name implies, the model will be encapsulated from now on into a simulation space which acts as the outer environment which the controller acts in and receives its inputs. And a clear controller model which will take the environment inputs to generate the required outputs (states $x1, x2, x3$) back out to the environment (simulation space). The idea comes in compliance with the company model seen in **Figure 16**.

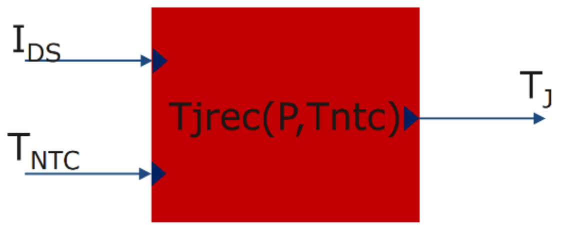


Figure 16 - 2 input, 1 output thermal model/function. (company slide 111)

For the sake of simplicity, the preliminary built model will focus only on the 1 input approach. The input power is directly inputted from the environment into the controller which will discretize it over a sampling time $Ts = \frac{1}{100} seconds$. Then the discretized input is translated into the states via the discretized LTI. The states are then outputted into the environment and their respective results compared with the reference LTI (continuous time domain). **Figure 17** and **Figure 18** show the design and results respectively.

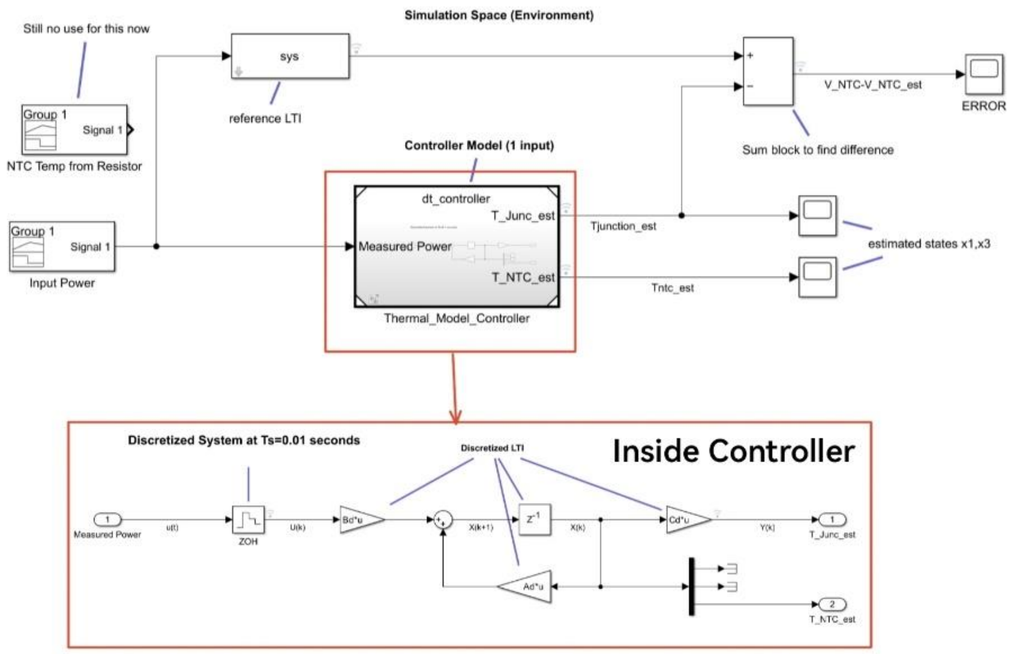


Figure 17 - Simulink model of the simulation space (up) and the controller (down).

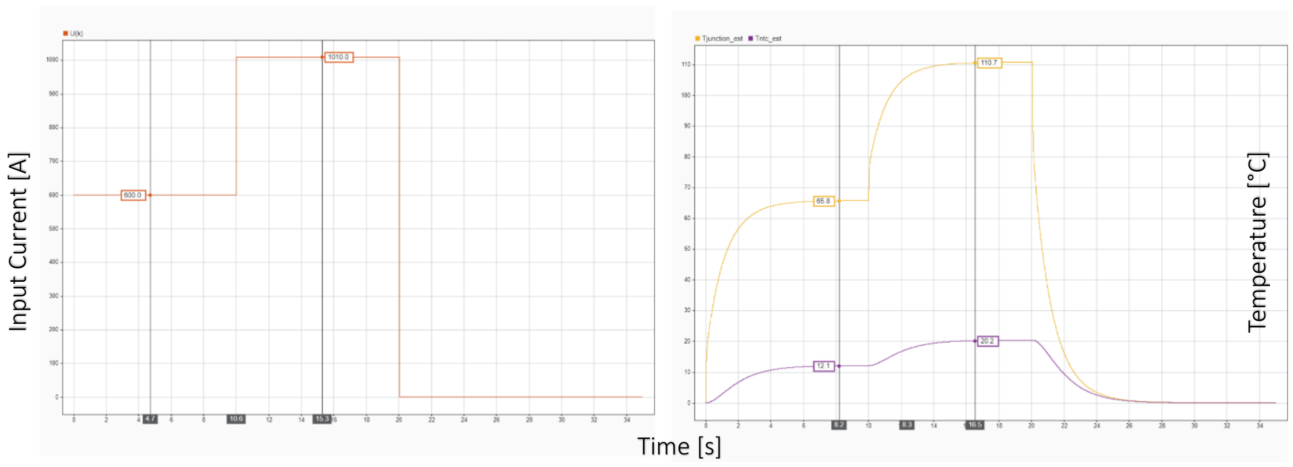


Figure 18 - (Left) Input Signal. (Right) Estimated Tjunction (yellow) and Estimated Tntc (purple).

Due to the relatively fast sampling time, the simulation produces very small error for a negligible time duration as seen in the error plot (**Figure 19**). The error plot produces trackable errors only during the 1st phase of the plot from the off state into the on state (warming up phase).

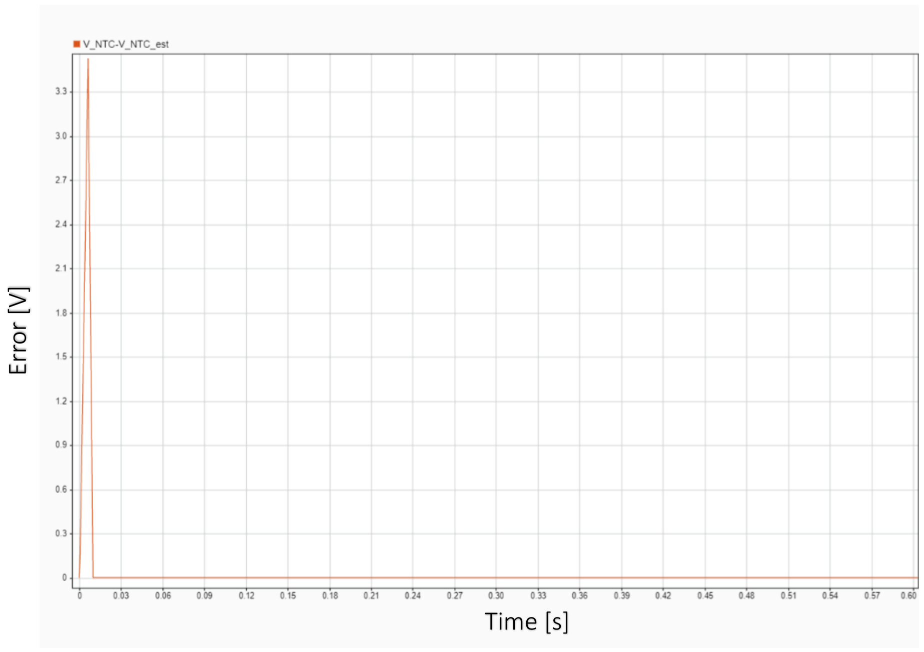


Figure 19 - Error plot representing difference between estimated NTC (discrete) and Ideal NTC values (continuous).

It can be concluded that the obtained thermal model behaves as desired within the chosen sampling time and presents very little to no noticeable error even after segregating the model. So ideally the behavior should be consistent for any simulation environment that the controller is inserted in. But this is obviously not enough due to the real-life environment complexity. The controller still lacks factors that anticipate real life factors from ambient temperature to noise…. The solution to this problem without over complicating the model lies within **Figure 16** itself– Adding a 2nd input (T_{NTC}) . The 2nd input should act like a calibrating term that helps the LTI act within the real-life ranges of the NTC actual logged temperature. So, in the next step there will be focus on inserting a 2nd input to the established controller model.

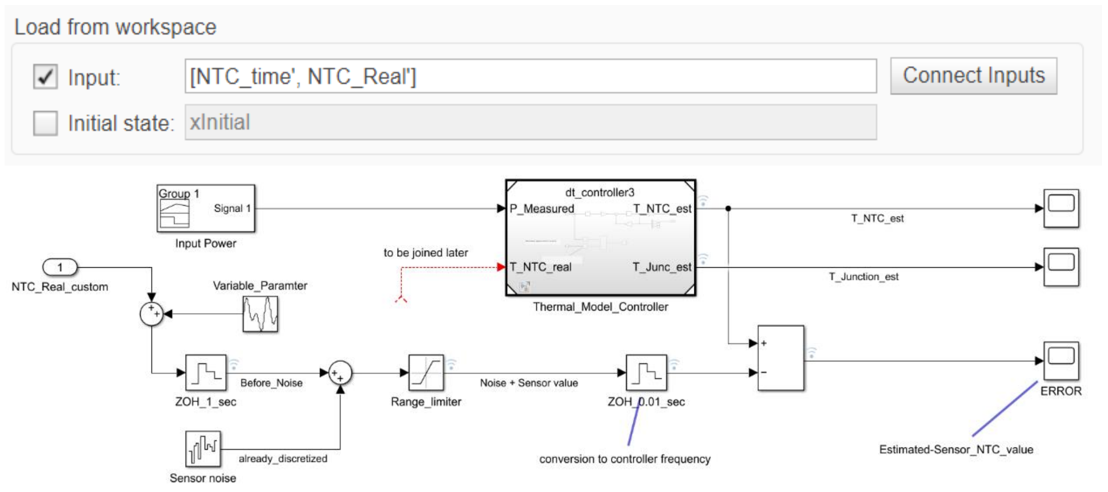
Second input (T_{NTC}) Signal incorporation:

The next step is to start working on incorporating the 2nd input signal into the model. However before incorporating it into the model, it is important to understand the purpose of this incorporation. The 2nd signal comes directly from the NTC sensor on the PCB. So, it acts like a dynamic calibrator of the model. Based on its value, the model should be able to update itself into a value closer to the real one dynamically. Ideally, this is not too difficult with most of the constraints discussed so far. However, the issue arises from a technicality – Sampling time difference. Step by step progressive work is done on creating, tuning, and inserting this signal and sampling it properly into the controller.

a. *T_{NTC} signal creation:*

For the sake of simulation, the model will incorporate a discretized slow signal simulating the input directly received from the NTC sensor to study the tracking dynamics. So, using LTspice the ideal behavior of $x3$=T_{NTC} is directly imported from a previous simulation* to use as a reference into the simulation space. This imported signal is followed by a sum block that adds a variable sign signal along with a white noise block to intentionally create unideal values that can be visualized in **Figure 21** represented as the red stepped signal. This "worsening" of the ideal NTC signal is due to the limitation in resolution and frequency of the ADC (analog to digital convertor) used on board. This setup is visualized in **Figure 20**.

Note: The aforementioned previous simulation is one of the data provided by the company which does not correspond directly to the assumed PCB structure. As mentioned in the state of the art, while keeping note of this fact, the study will carry on as planned while combatting the mismatch between the reference data and assumed structure using curve fitting techniques.*


Load from workspace

✓ Input: [NTC_time', NTC_Real']

Initial state: xInitial

Figure 20 - The Simulink configuration of the simulation space with an imported and modified NTC sensor value from MATLAB.

The NTC estimated signal output along with the created NTC sensor signal value plot due to the same current signal from **Figure 18 (left side)** are plotted in **Figure 21.** Following this new red signal observed, the LTI must update its $x3$ state value and recalculate all the states accordingly within a neighborhood closer to the sensor inputted value. So, the next step is to work with this new signal to update the LTI states.

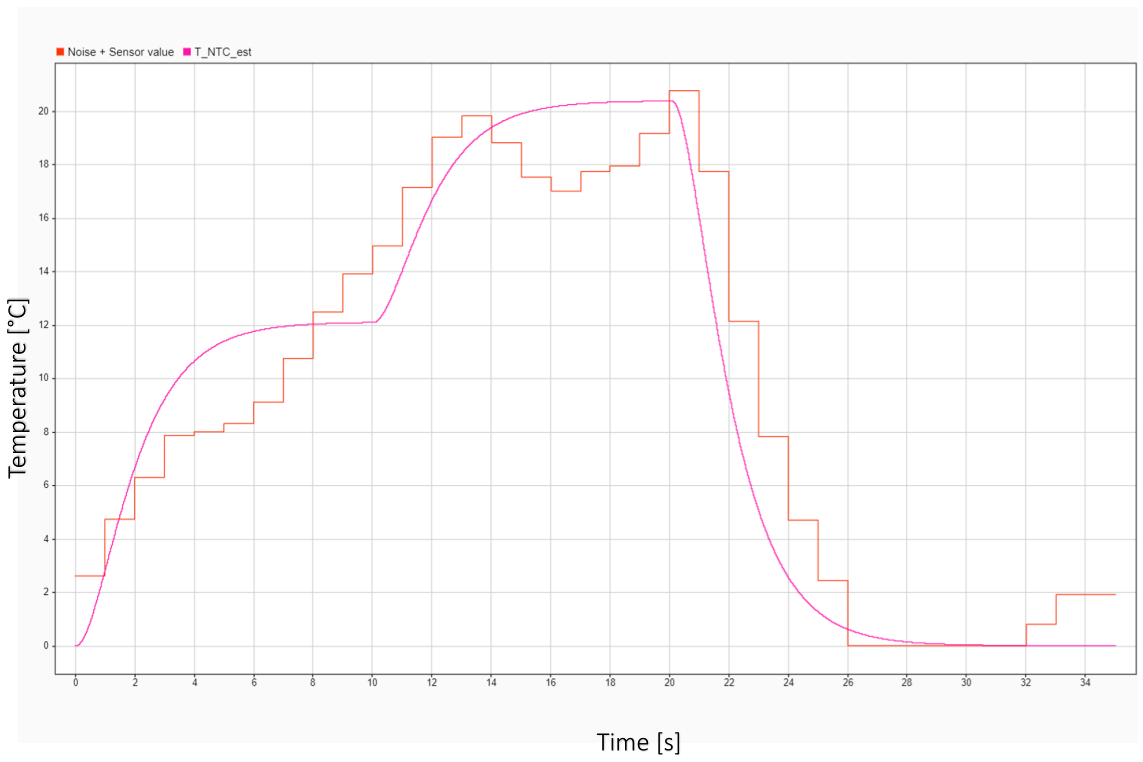


Figure 21 - Plot of created NTC sensor value for simulation and Tntc estimated with respect to time.

b. *Controller Design:*

To insert this newly created signal into the controller, the problems along with the expected goals from inserting the new input model must be identified properly. For better organization, **Table 3** is constructed. And the proposed design of the model/system is presented in **Figure 22**.

Problems	Goals
- The power input and sensor value input operate within different sampling times ($\frac{1}{100}$ *and* 1 seconds respectively) - The difference in sampling times will also propagate into affecting the desired correction factor behavior. - Creation of discontinuity around sharp changes of values due to slow frequency of NTC sensor inputs. - Increase of model complexity.	- Creation of a correction factor that somehow modifies and updates the LTI states based on real environmental data. - Insertion of initialization data directly from environment such as initial temperature of NTC

Table 3 - Table highlighting the problems and goals of incorporating a 2nd input signal into the model.

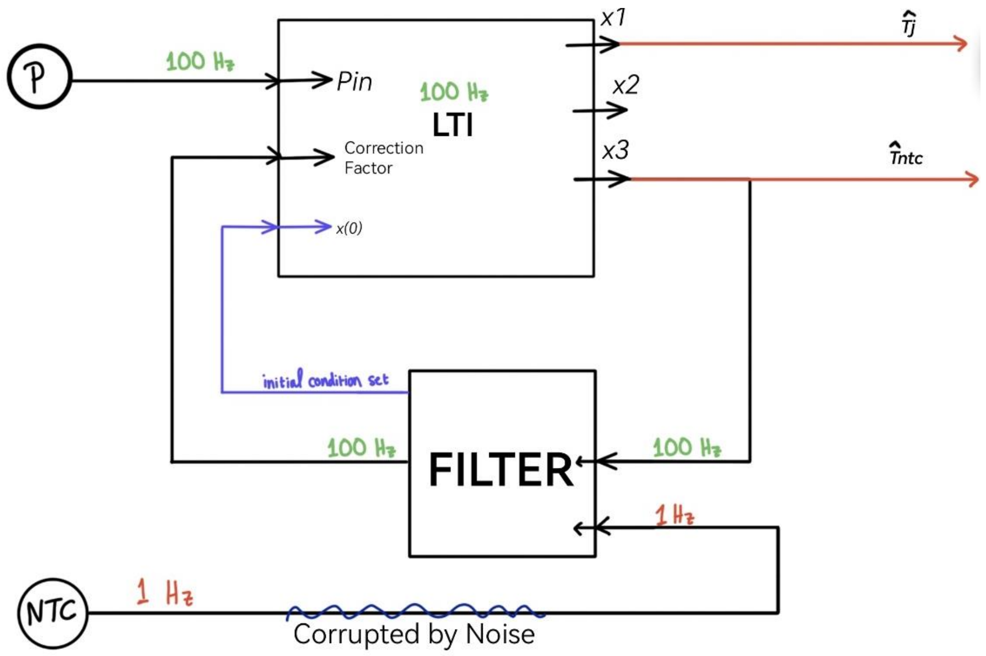


Figure 22 - Preliminary design of LTI + Filter.

The Filter receives the $\hat{T}_{\mathrm{NTC}}$ (also referred to as $T_{NTC}\ est$) from the LTI output along with the real value directly from the NTC sensor. The difference between both the values is calculated after they're both discretized over the sample sampling time $Ts = \frac{1}{100}$ seconds. The difference is used with specific gain to be defined in order to create a correction factor. This factor is then reinserted into the LTI along with the 1st value the sensor of NTC provides as an initial condition $x0 = [x1(0), x2(0), x3(0)]$. The design now has to be implemented into Simulink step by step. So, the next step will involve the design of a compensator system that forms the filter.

c. *Compensator Design*

To begin the design of a compensator system, an approach must be defined. An approach that could be used is an intuitive approach.

i. Intuitive approach:

This approach uses the direct difference between the output of the LTI $y = x3 = T_{NTC}est$ and the input imported from the sensor. It is important to note that the previously modeled LTI has the C matrix being: $C = [1\ 0\ 0]$. This made the output be $y = x1 = Tj_{est}$. So, from now on the output should be $y = x3 = T_{NTC}est$ by simply changing C to $C = [0\ 0\ 1]$ in the MATLAB code. This ensures that the difference when inserted into the $x(k+1)$ portion of the LTI will compensate all the states with every step.

In summary, LTI runs one step finding the difference between output and real value $e = T_{NTC}Real - T_{NTC}est$, then in the calculation of the next step, the difference e is summed to the rate of change compensating the previous step's one. In application, the inside of the controller block is seen in **Figure 23**.

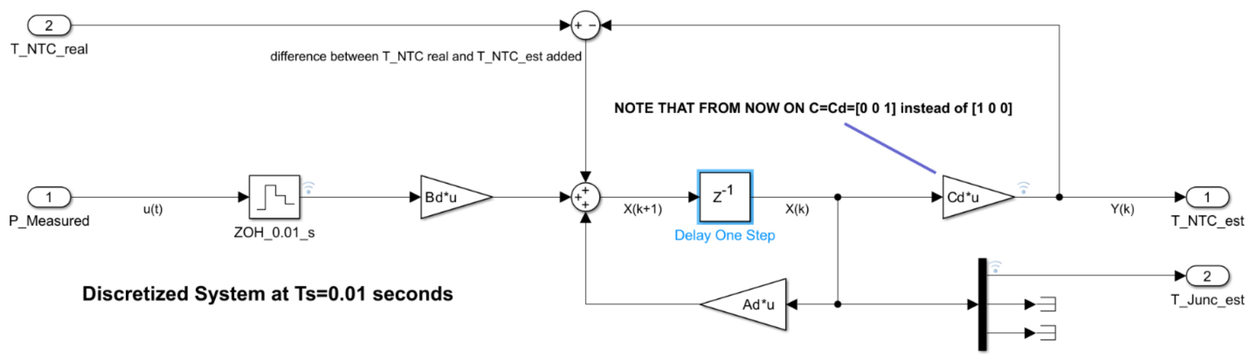

2
T_NTC_real
difference between T_NTC real and T_NTC_est added
NOTE THAT FROM NOW ON C=Cd=[0 0 1] instead of [1 0 0]
1
P_Measured
u(t)
ZOH_0.01_s
Bd*u
X(k+1)
z-1
Delay One Step
X(k)
Cd*u
Y(k)
1
T_NTC_est
2
T_Junc_est
Ad*u
Discretized System at Ts=0.01 seconds

Figure 23 - Intuitive Approach Compensator Configuration.

A problem arises however when the new $T_{NTC}est$ is plotted along with sensor value one as observed in **Figure 24**. As can be observed, the values superimpose gaining a step-like shape basically neglecting the LTI equations. This is caused by the absence of a gain after e is calculated. Thus, the full difference is summed every step making the output always identical to the value of the sensor value a step before (0.01 seconds before). The fix for this is very simple and is by adding a small gain which in this case through trial and error is chosen to be $Ts = \ 0.01\ s$.

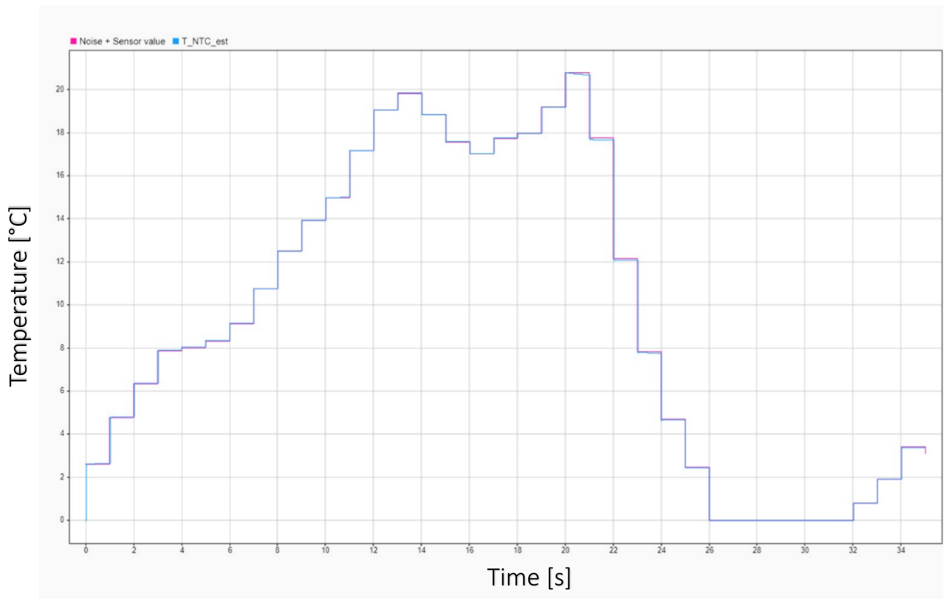


Figure 24 – Plot of given NTC values corrupted by sensor noise (added manually) and the estimated NTC output from the constructed LTI.

After the intuitive compensator is fixed the Simulink model is constructed as seen in the **Figure 25**. To better validate the fix, the simulation space is modified to compare the NTC value with also the ideal value so that the estimated temperature along with the ideal and sensor ones are observable. The updated simulation space can be visualized in **Figure 26**.

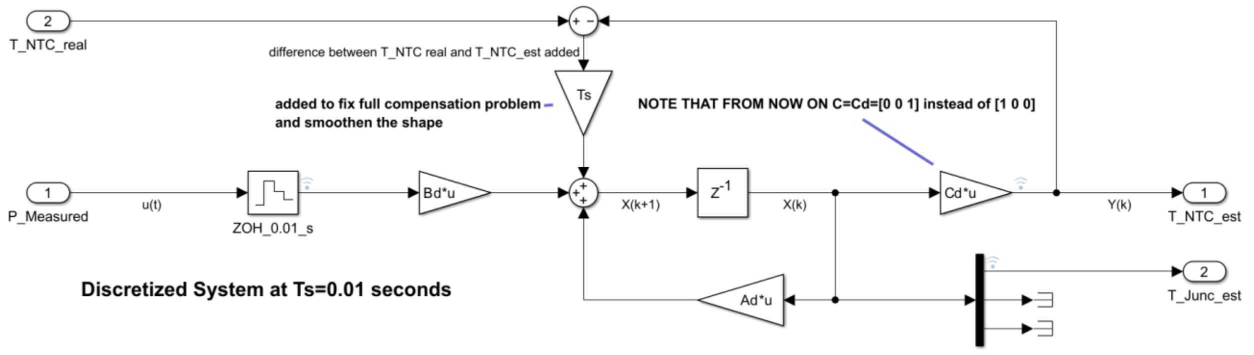

2
T_NTC_real
difference between T_NTC real and T_NTC_est added
Ts
added to fix full compensation problem and smoothen the shape
NOTE THAT FROM NOW ON C=Cd=[0 0 1] instead of [1 0 0]
1
P_Measured
u(t)
ZOH_0.01_s
Bd*u
X(k+1)
z-1
X(k)
Cd*u
Y(k)
1
T_NTC_est
Discretized System at Ts=0.01 seconds
Ad*u
2
T_Junc_est

Figure 25 - Intuitive Approach Compensator Configuration with added gain.

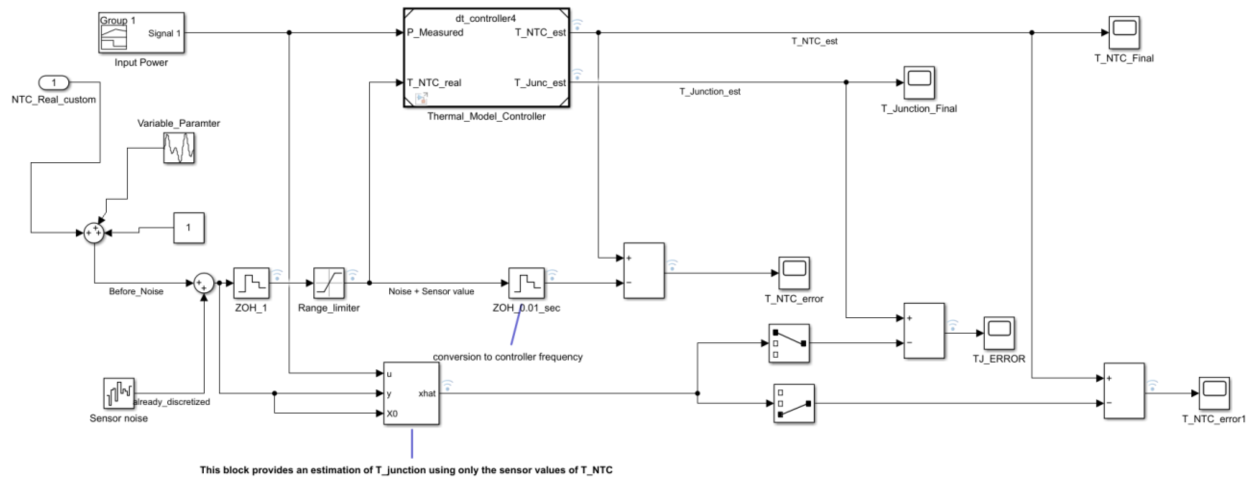


Figure 26 - New simulation space which highlights: Ideal (lowest), Sensor (middle), and estimated (upmost) temperature values.

With the updated simulation space the output plots can be visualized in **Figure 27**. The results of this plot deem this approach to be viable with great tracking ability represented in the T_NTC_EST plot sandwiched in the middle between the reference NTC curve (Noise + Sensor value) and the Kalman filter curve. The only issue arises from the lack of initial values taken into consideration which could with higher values pose a tracking problem. This could be fixed by increasing the complexity of the controller design to implement initial values as will be seen in the next section of this study where different state observer approaches are explored.

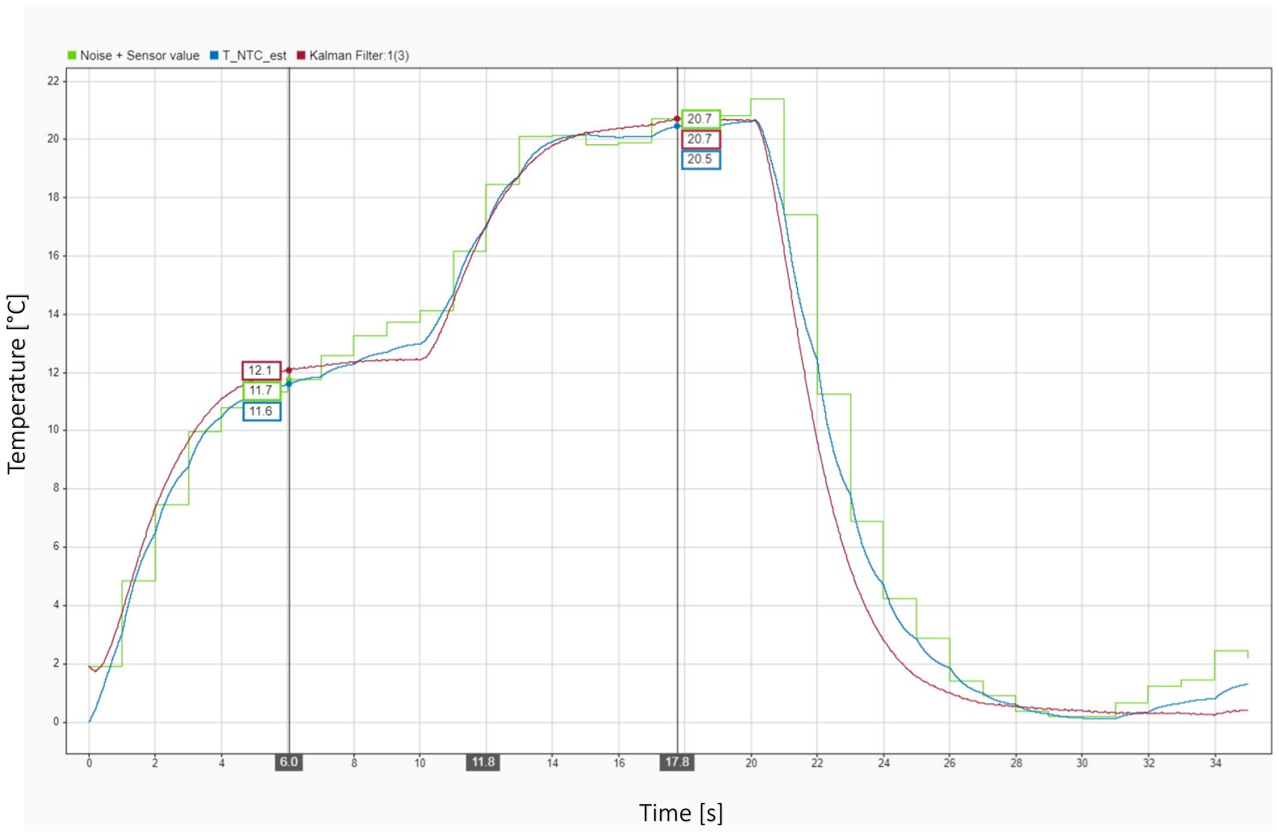


Figure 27 - NTC temperature values: (Green) Sensor, (Red) ideal, (Blue) Estimated.

ii. State Observer approach:

1. Kalman filter:

This approach is an improved version of the previous approach. To understand how it works, the Kalman filter must be defined first. A Kalman filter is a filter that estimates the states of an LTI by taking the output, initial states, and input as inputs.

Knowing that, the modeling process can be done using the built in Kalman filter block in Simulink to create a structure similar to that of **Figure 28**. The detailed simulation space design along with the Kalman prediction Block settings are presented in **Figure 29**.

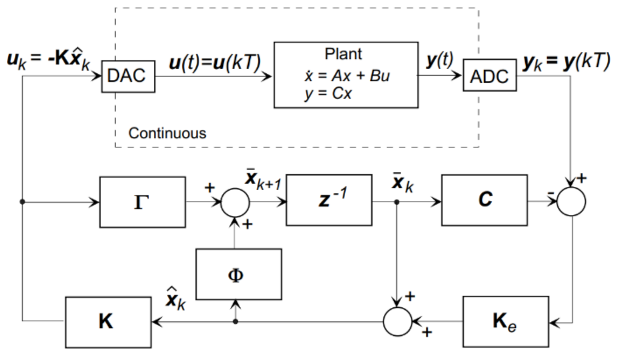
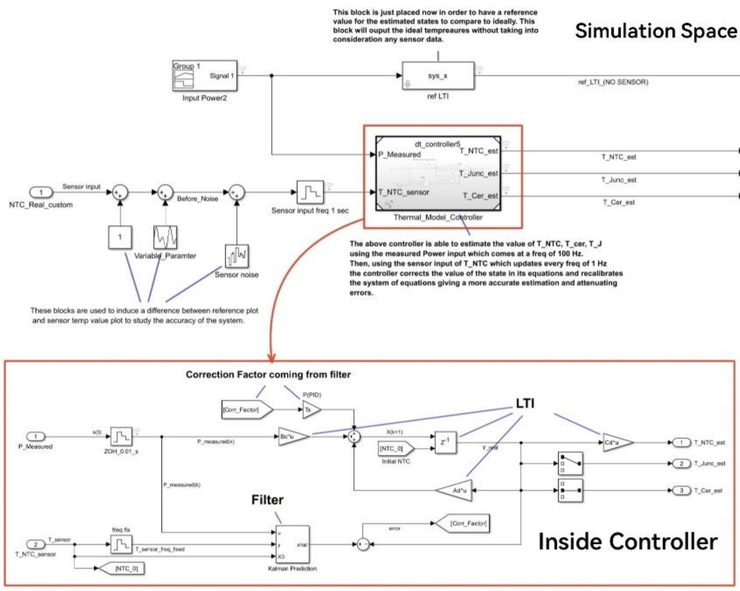


Figure 28 - Discrete-time state feedback control of a continuous plant using a current discrete observer. Source: (Rowell)

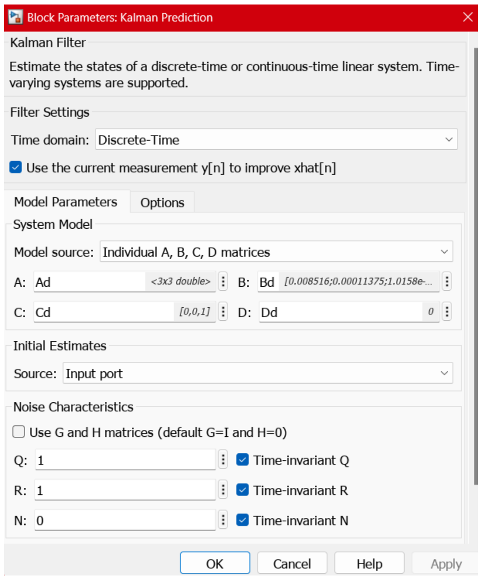
`



This block is just placed now in order to have a reference value for the estimated states to compare to ideally. This block will ouput the ideal tempreaures without taking into consideration any sensor data.

The above controller is able to estimate the value of T_NTC, T_cer, T_J using the measured Power input which comes at a freq of 100 Hz. Then, using the sensor input of T_NTC which updates every freq of 1 Hz the controller corrects the value of the state in its equations and recalibrates the system of equations giving a more accurate estimation and attenuating errors.

These blocks are used to induce a difference between reference plot and sensor temp value plot to study the accuracy of the system.


Block Parameters: Kalman Prediction

Kalman Filter

Estimate the states of a discrete-time or continuous-time linear system. Time-varying systems are supported.

Filter Settings

Time domain: Discrete-Time

[✓] Use the current measurement y[n] to improve xhat[n]

Model Parameters | Options

System Model

Model source: Individual A, B, C, D matrices

A: Ad <3x3 double> B: Bd [0.008516;0.00011375;1.0158e-...

C: Cd [0,0,1] D: Dd 0

Initial Estimates

Source: Input port

Noise Characteristics

[ ] Use G and H matrices (default G=I and H=0)

Q: 1 [✓] Time-invariant Q

R: 1 [✓] Time-invariant R

N: 0 [✓] Time-invariant N

Figure 29 - (Up) Simulation Space (Down) Controller with Kalman Filter compensator approach (Right) Kalman Filter Block settings.

This new improved structure takes into consideration the initial values $x0$ from the NTC sensor. And so, the MATLAB code is modified to also have the initial NTC values to be set around 25 °C. Then the output results of the NTC values are plotted in **Figure 30**.

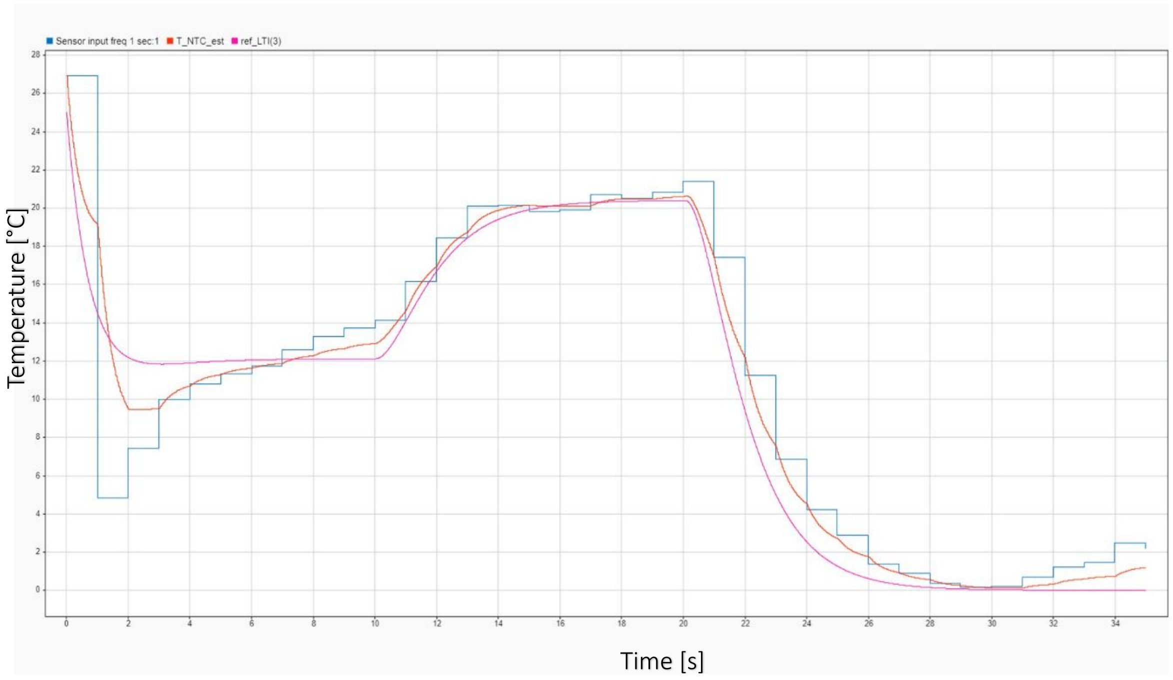


Figure 30 - NTC temperature values: (Blue) Sensor, (Pink) ideal, (Red) Estimated.

The plot is very promising, the estimated value running always sandwiched between the ideal and sensor values (when the error gain is set equal to 0.01). Furthermore, the estimated value starts at the initial values defined by the NTC sensor (25 °C). In addition to all that, the plot runs in a smooth circular fashion avoiding sharp step-like behavior and avoiding discontinuity. To study the $T_{junc}est$ properly, a similar plot is drawn for the x_1 values as observed in **Figure 31**. The junction temperature plot behaves properly as expected with an avoidance of sharp behavior even when the sensor values vary rapidly. As a direct example, the sensor value curve (blue) shows a sharp saw behavior around 1 second, but the estimated curve (red) remains smooth and between the other 2 curves.

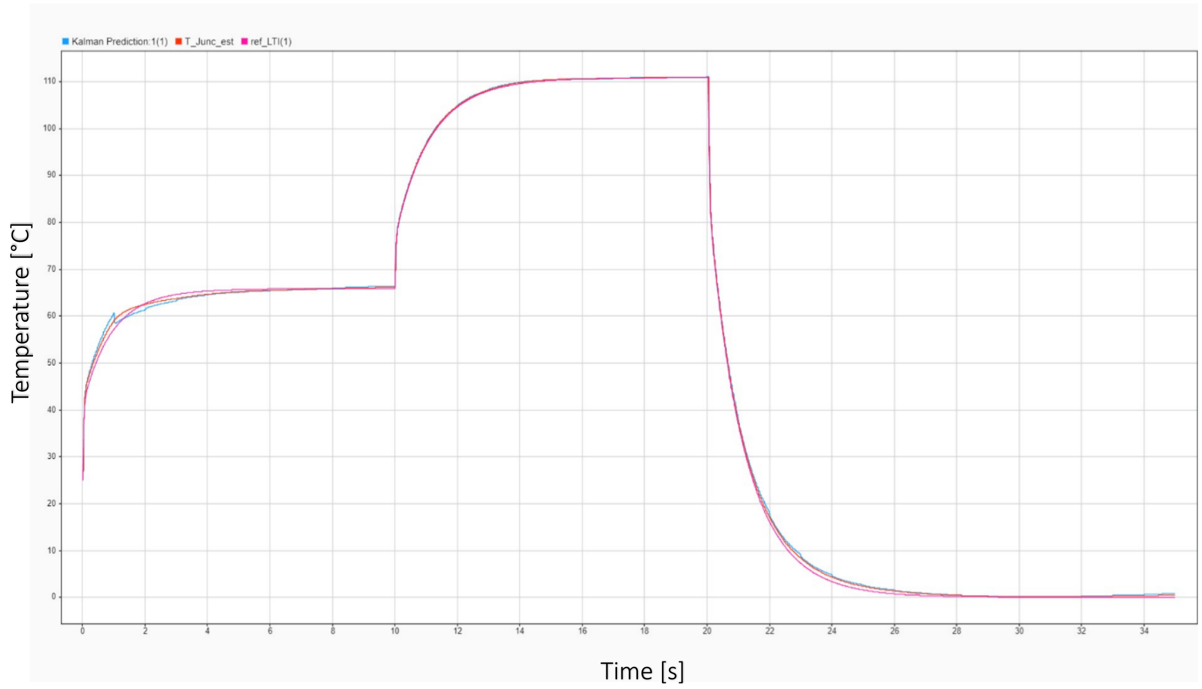


Figure 31 - Junction temperature values: (Blue) Sensor, (Pink) ideal, (Red) Estimated.

2. LQR State Observer:

This approach is remarkably similar to the Kalman filter approach in goal: Both are filters used to estimate the states based on reference output data given. This approach, even though requires more steps to build and finalize, is more accurate and is not a built-in function in MATLAB. With reference to (Anderson & Moore, 2007), the new approach is explained.

The observer to be built is a Linear Quadratic Regulator State Feedback. The latter system follows the LQ control law $u(t) = -Kx(t)$ (continuous domain). As can be observed, the optimal control problem becomes finding an appropriate input $u(t)$ defined over the time range $[0, t_f]$ as a solution to the optimization problem:

$$\min_{u(t), t \in [0,t_f]} J(u) = \min_{u(t), t \in [0,t_f]} x^T(t_f)Sx(t_f) + \int_{\tau=0}^{t_f} \left(x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau)\right)d\tau$$

$$Q, S \in \mathbb{R}^{n,n}: Q = Q^T \geq 0, S = S^T \geq 0, R \in \mathbb{R}^{p,p}: R = R^T > 0$$

This approach is called finite horizon Linear Quadratic optimal control problem and as the name suggests is finite ($[0, t_f]$). The problem can be modified to become infinite horizon by $t_f \to \infty$.

The infinite horizon LQ optimal control problem can be written as:

$$\min_{u(t), t \in [0,\infty]} J(u) = \min_{u(t), t \in [0,\infty]} \int_{\tau=0}^{\infty} \left(x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau)\right)d\tau$$

$$s.t.\ \dot{x} = Ax(t) + Bu(t)$$

For the LQ control law to apply, the provided LTI must be reachable and to also do some error tracking (integral action), the system must also be observable. If the system is reachable and observable, then states could be approximated with state feedback structure (discretized) seen in **Figure 32.**

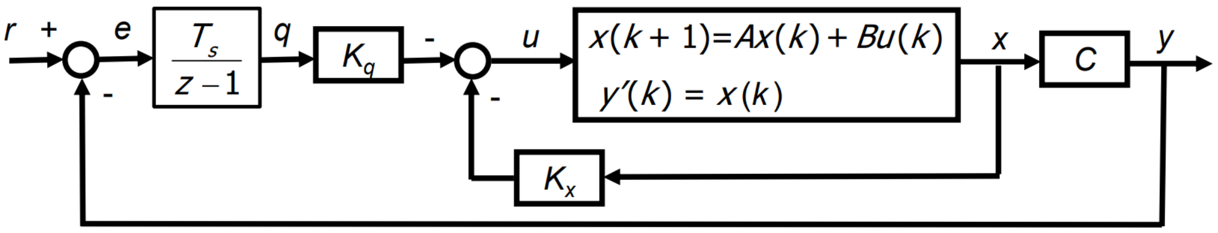


Figure 32 - LQR state feedback Design with integral action in discrete domain.

Using the last figure as a reference, the observer subsystem is built. And then following the same logic used in the Kalman filter approach, the difference between the output of the observer and LTI is found (error) and is then fed into the LTI as feedback. A structure like that of **Figure 29** is built which can be visualized in **Figure 33**.

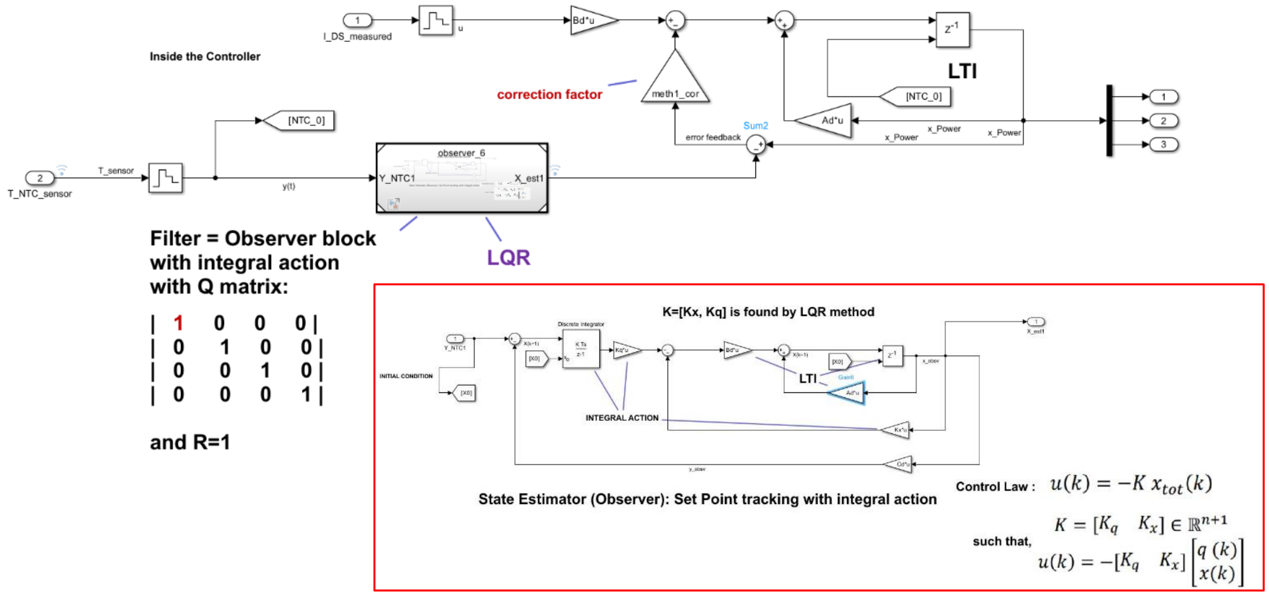


Figure 33 - Controller with LQ Regulator approach.

As the approach before, this approach requires manually setting a correction factor to regulate the error insertion into the LTI in a favorable manner. Both approaches (Kalman and LQR) produce identical results when the correction factors (error feedback terms) are set adequately. So, in the next steps of the design, both methods will be compared side by side to then select of them as the optimal approach.

Testing the Design with Provided Reference Data:

Reference data is provided to test the LTI equations and parameter values. The reference data provided for the 1st test can be summarized with the plot seen in **Figure 34**.

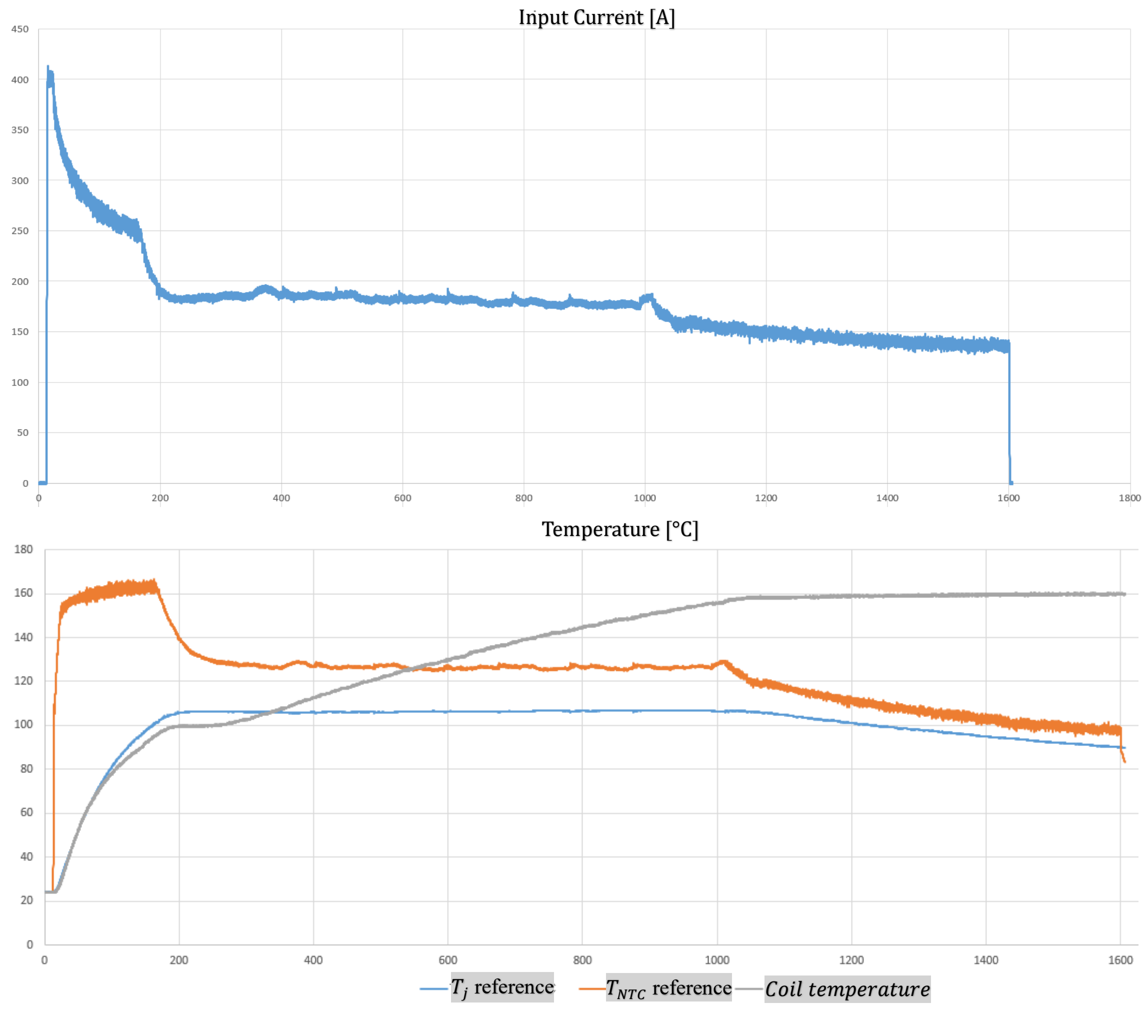
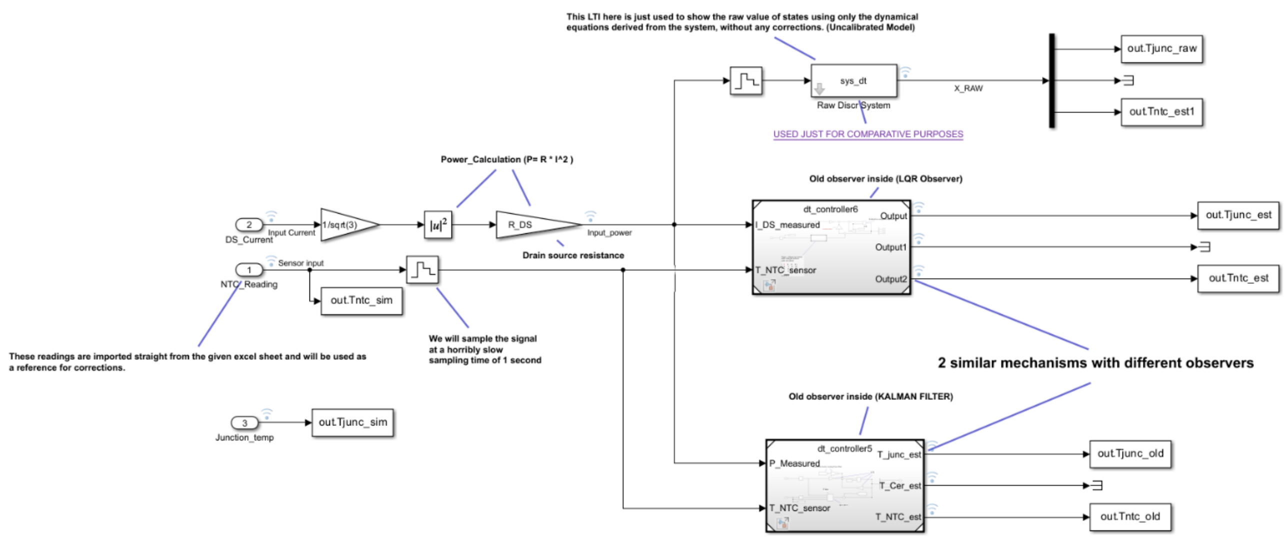


Figure 34 - Reference plots acquired from thermal simulation software (FEM):
(Up) Input Current.
(DOWN) Junction Temperature in orange and NTC Temperature in blue.

The reference data is then inserted into the Simulink simulation space created. The simulation space with 2 controllers and the reference LTI is shown in **Figure 35**. The process of testing the created models will be split into subsections: LTI parameters check, Error Gain Calibration.



This LTI here is just used to show the raw value of states using only the dynamical equations derived from the system, without any corrections. (Uncalibrated Model)
sys_dt
Raw Discr System
X_RAW
out.Tjunc_raw
out.Tntc_est1
USED JUST FOR COMPARATIVE PURPOSES
Power_Calculation (P= R * I^2)
Old observer inside (LQR Observer)
dt_controller6
I_DS_measured
T_NTC_sensor
Output
Output1
Output2
out.Tjunc_est
out.Tntc_est
DS_Current
Input Current
1/sqrt(3)
|u|²
R_DS
Input_power
Drain source resistance
NTC_Reading
Sensor input
out.Tntc_sim
We will sample the signal at a horribly slow sampling time of 1 second
These readings are imported straight from the given excel sheet and will be used as a reference for corrections.
2 similar mechanisms with different observers
Old observer inside (KALMAN FILTER)
Junction_temp
out.Tjunc_sim
dt_controller5
P_Measured
T_NTC_sensor
T_junc_est
T_Cer_est
T_NTC_est
out.Tjunc_old
out.Tntc_old

Figure 35 - Simulation space with 3 main components: LTI only, Controller with Kalman Filter, and Controller with LQ regulator.

a. LTI Parameters Check:

The LTI parameters were initially set according to company set values which are prone to error and are preliminary. So, the focus in this step is to get the values as close as possible to the real values through manual testing iteration. The simplest and most intuitive way to do this is by plotting the junction temperature output of the LTI (input power is the only input) along with the given reference values on the same grid. And then to tune the parameters progressively until the general shape and scale of the curves match. The focus here is mainly the general shape and scale because the designed controllers (yet to be tested) are used later for fine tuning.

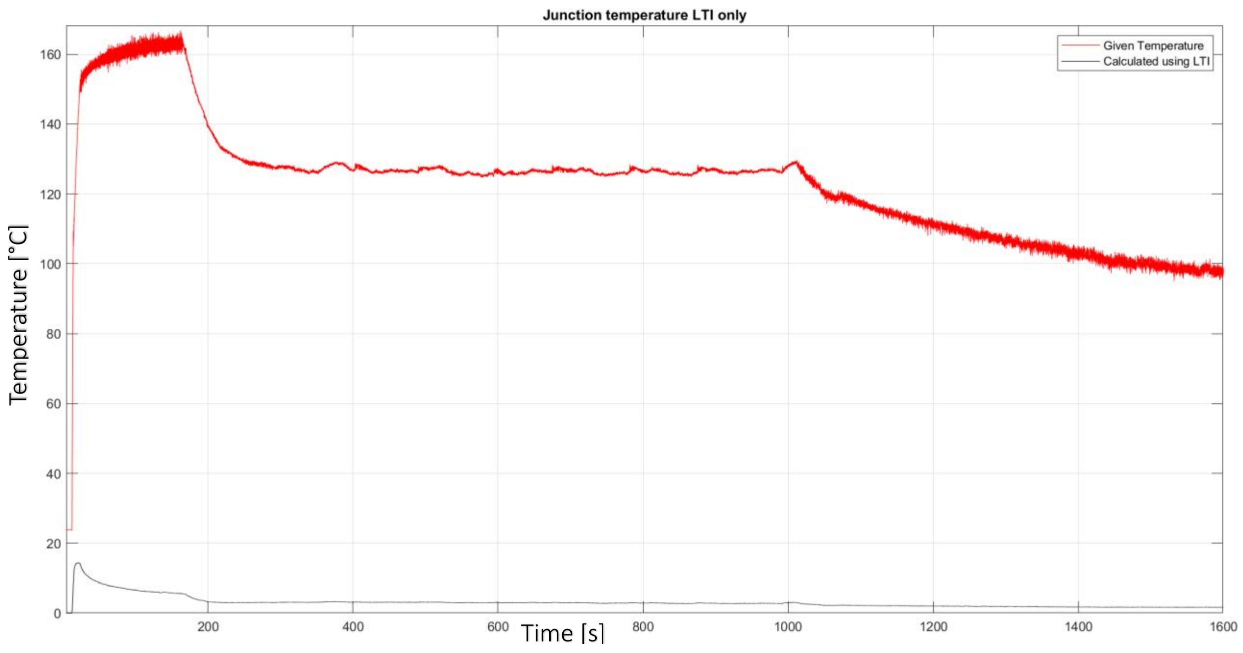


Figure 36 - Plot showing the variation of given reference junction temperature (red) and the calculated one from LTI equations (black) as function of time.

As seen in **Figure 36**, the scale of the calculated LTI is off. So, manually tuning the parameters until a more consistent behavior is achieved is needed. It should be noted that the general behavior of both plots seems to be matching.

To preserve behavior and to lower the degrees of freedom, the time constants are set to be similar to the original values (time constant= R x C). Then, the time constants and resistances are slowly tuned until more desirable values as seen in **Table 4** are reached.

Original VALUES	**MODIFIED VALUES**

%DEFINING PARAMETERS: Selected from slides
R_J2C=0.03;
R_C2W1=0.1;
R_C2W2=0.1;
R_2NTC=0.295;
C_J2C=1;
C_C2W1=13;
C_C2W2=13;
%ASSUMED drain source resistance
R_DS= 0.00245 ;%page 96 %probably truest
	% BETTER VALUES chosen manually of parameters
	tau_J2C=0.05; %0.03 %0.5 %0.2 %peak manipulation and rise time
	tau_C2W=17.5; %1.3 %21 %18 %17.5 %less peak manipulation and rise time
	R_J2C=0.5; %0.03 %0.5 %plateu sharpness
	R_C2W1=12;%2 %7 plateu shift up/down
	R_C2W2=R_C2W1;
	R_2NTC=0.1; %0.295 %2 %0.1 %fine tuning more on left side
	C_J2C=tau_J2C/R_J2C;
	C_C2W1=tau_C2W/R_C2W1;
	C_C2W2=C_C2W1;
	%ASSUMED drain source resistance
	R_DS= 0.00245 ;%page 96 %probably truest

Table 4 - Applied component modifications.

After the modifications, a new plot with observably better behavior can be observed in

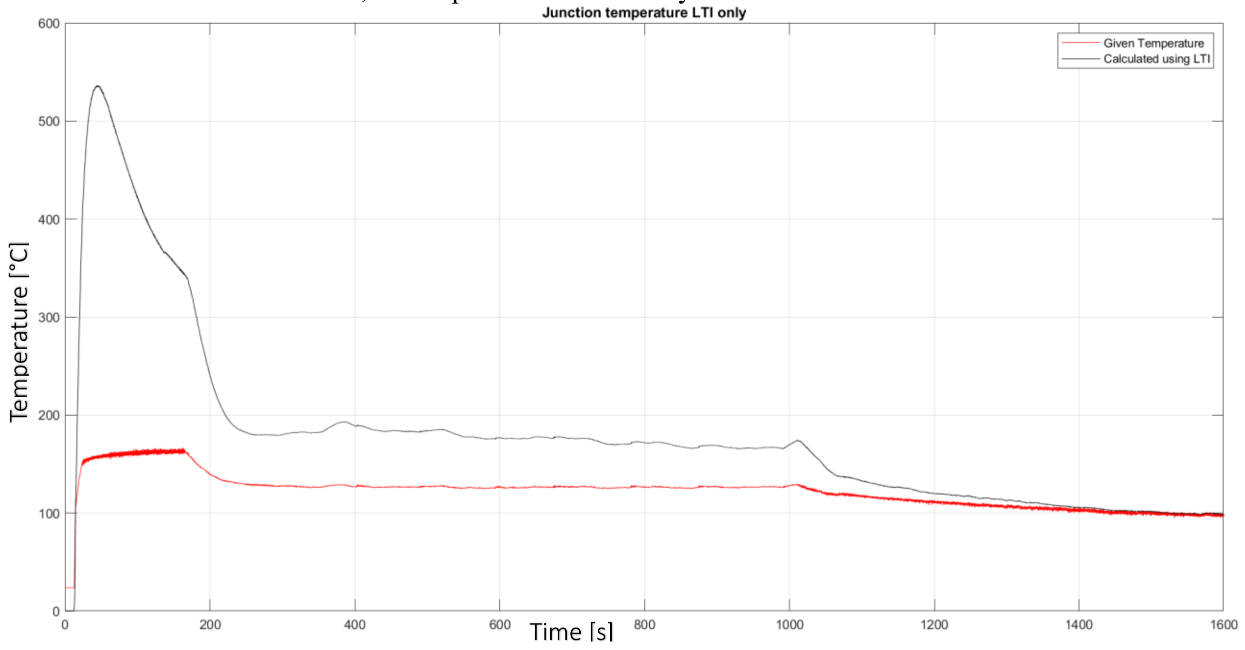
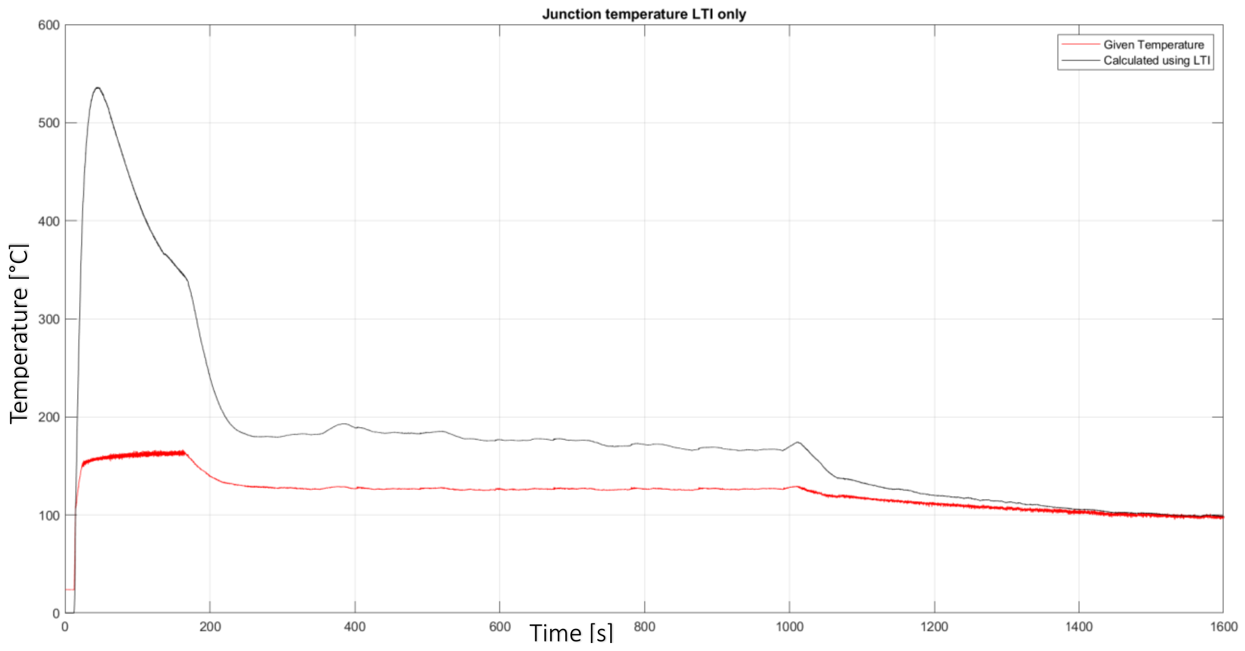


Figure 37



. The curve shows similar dynamics to the given but with obviously higher scaling of values and differences. It must be noted that however much manual change is induced to the parameters, the peak seen between 0-200 seconds seems to always overshoot due to the input current behavior introduced which the given company data fail to capture properly. Therefore, using the closed feedback loop in the controllers, a change in the error gain is introduced until the models match and the peak temperature can be estimated to a minor error. This needed change is introduced in the next step of: Error Gain Calibration.

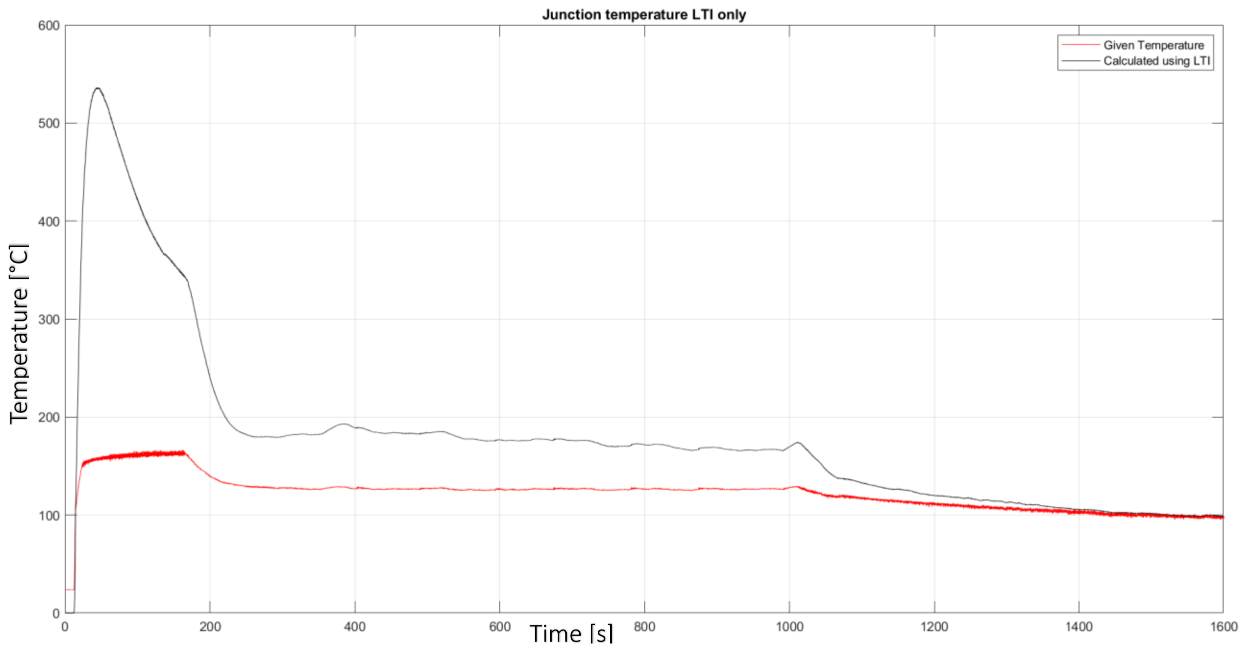
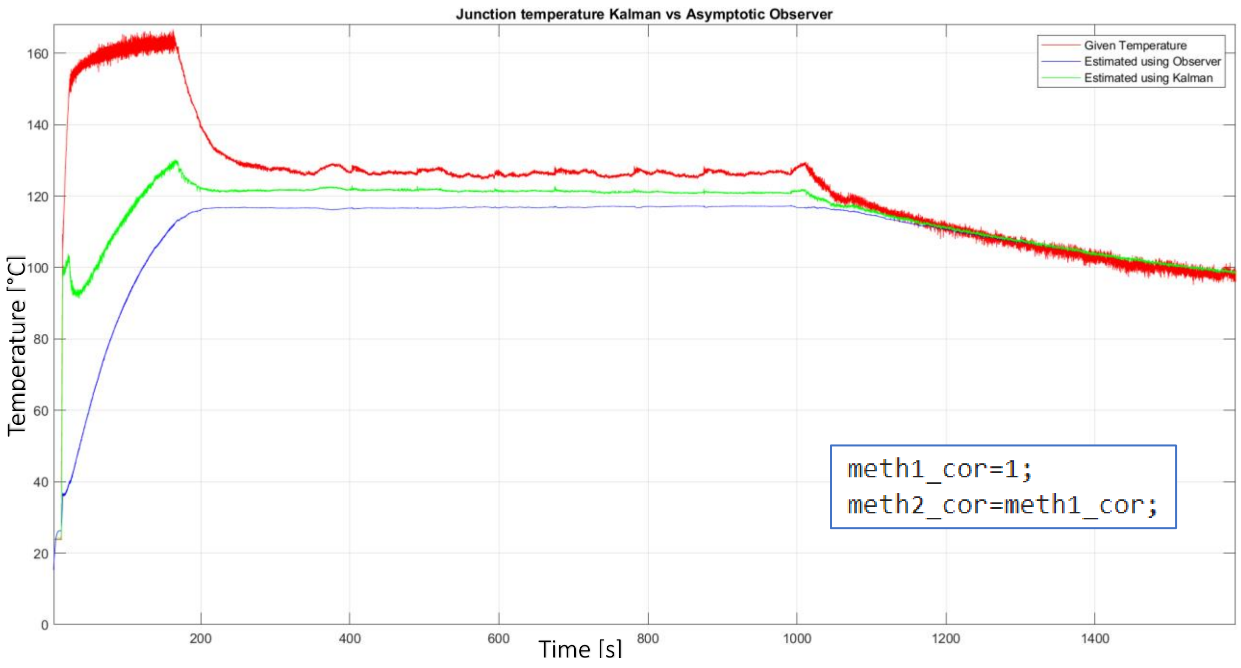


Figure 37 - Plot showing the variation of given reference junction temperature (red) and the calculated one from LTI equations (black) as function of time after modification of the parameters.

b. Error Gain Calibration:

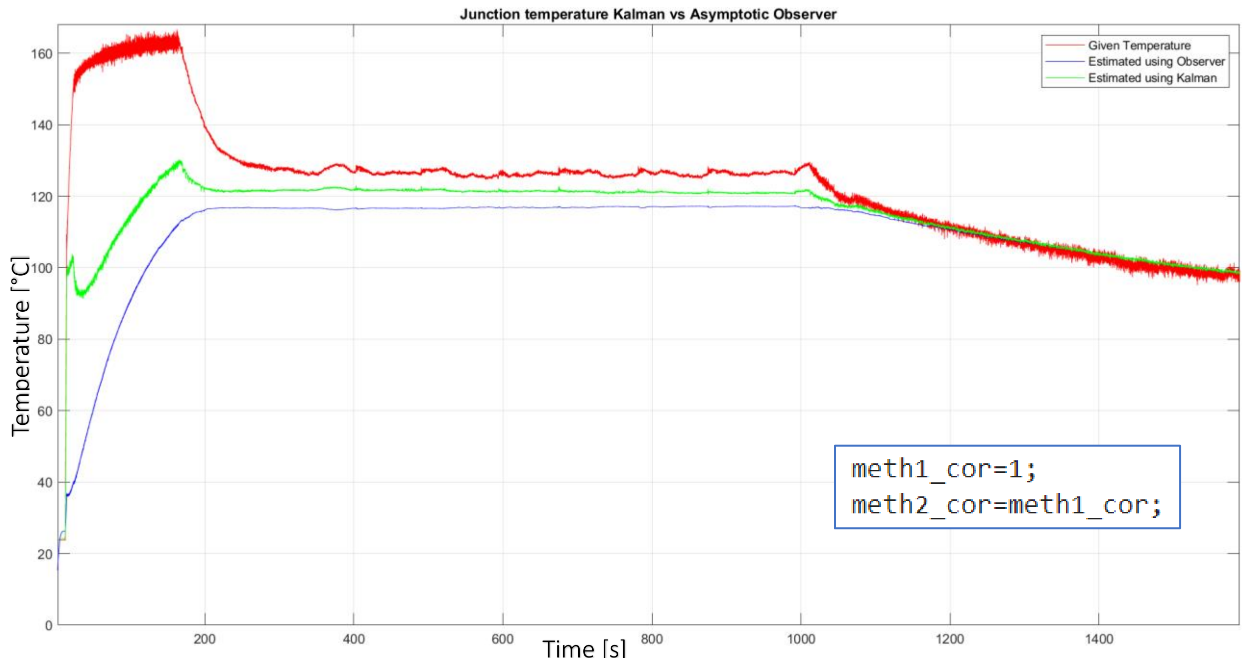
Next step is to observe the output of the controllers with the error gain still to be calibrated. The error gains of each controller (Kalman and LQR one) are: denoted by $meth1_{cor}$ and $meth2_{cor}$ respectively (check **Figure 29** and **Figure 33**). For a preliminary control test, $meth1_{cor} = meth2_{cor} = 1$.



```
meth1_cor=1;
meth2_cor=meth1_cor;
```

Figure 38 - Plot of Junction temperature outputs of Kalman and LQR controllers with the error gains set to 1.

To tackle the goal more efficiently at hand, the focus is shifted to one of the controllers: The controller with LQR with asymptotic observer. As seen in



```
meth1_cor=1;
meth2_cor=meth1_cor;
```

Figure 38, many iterations with different values of the error gain were simulated. The results seem promising and with the best result showed in cyan where error gain was of value $4.9 * 10^{-3}$. It is important to note that the general shape chosen to achieve an estimation in this case was that able to **predict the peak during the peak amperage**. Another acceptable result with a different shape would be that with error gain value of $3 * 10^{-3}$.

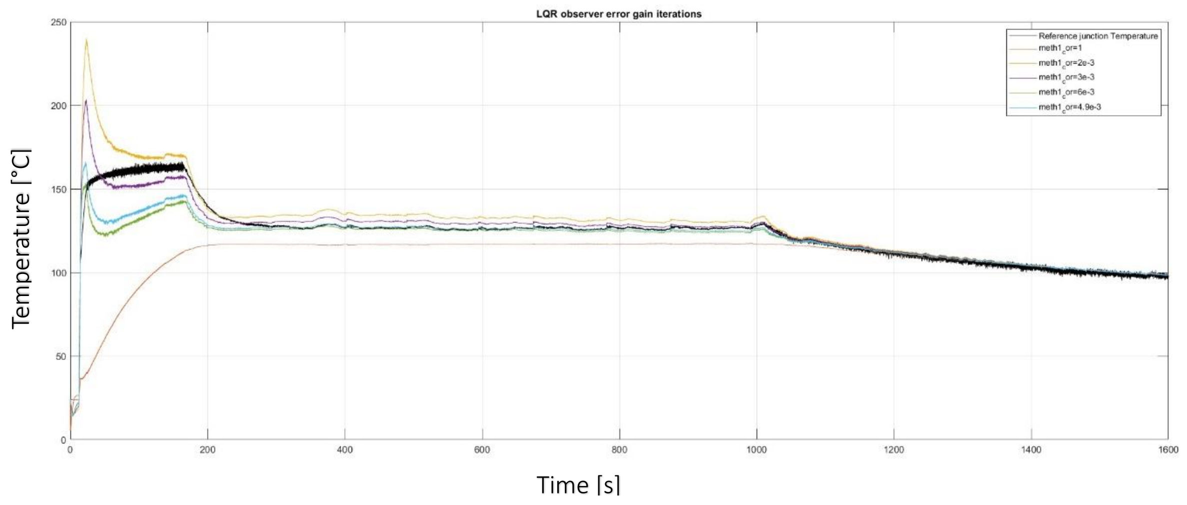


Figure 39 - Output of LQR controller with many possible choices of error gain visualized.

After fixing the value of $meth1_{cor}$ to $4.9 * 10^{-3}$. The same iteration process is applied to obtain a similar value for the Kalman controller method. The optimal value with almost the same values is obtained when $meth2_{cor} = meth1_{cor} + 0.85 * 10^{-3}$.

Analyzing the curve, it can be observed that at peak behavior **the estimators are able to momentarily predict the peak temperature well but not with the exact same dynamics**. Even though that behavior is technically erroneous, the value of the peak is acceptable. **So, within the framework of this study of estimating the peaks**, this estimation is deemed acceptable especially since the rise time in the curve and steady state values seem to match almost perfectly.

The next step is to test the same choice of error calibrator gain for another set of reference data provided also by Ideas & Motion Torino. So, using a new set of reference data, the simulation is rerun, and the corresponding plots are studied to fine tune the error calibrator gain. Both used reference input data can be visualized in **Figure 54**.

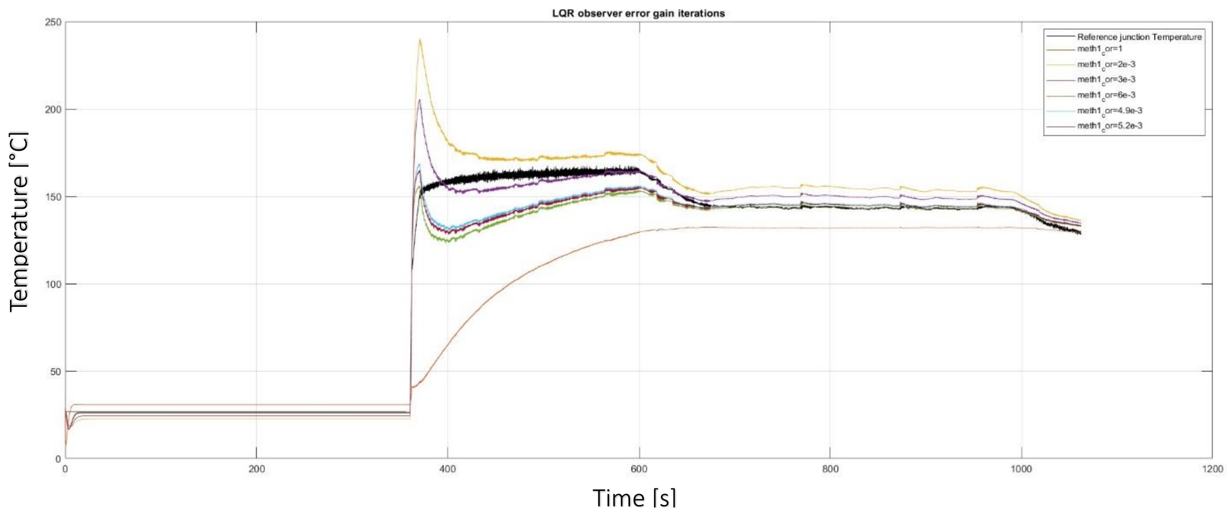


Figure 40 - Junction temperature plot for new reference data with different values of meth1_cor error gain.

As seen in **Figure 40**, the newly provided reference data shows no activity for the first 360 seconds, and then a rise to a peak. After the peak there is a slight decrease before reaching the plateau as in the simulation before. Different error gain values ($meth1_{cor}$) are used and marked with different colors. It could be observed that the best behavior and estimation of the peak is with $6 * 10^{-3}$(0.006) which behaves slightly better than the previously picked $4.9 * 10^{-3}$ value. For the next phase, a more complex case is studied. And for the sake of diversifying the study, the study will shift into try track the temperature rather than predict it more like that seen in **Figure 40** with $meth1_{cor} = 3 * 10^{-3}$.

UNIT 2: Complex Assumed Model Case

1. New Assumed Model Introduction

With reference to the steps taken before, a new more complicated model is introduced. The provided assumed model is similar to that of unit 1, but with the addition of extra levels. The model will present 3 main components being: The Junction, NTC, and Transitional component. This model will be used to test also the limits of the methodology employed in this study for estimating the junction temperature of a PCB with minimal information provided.

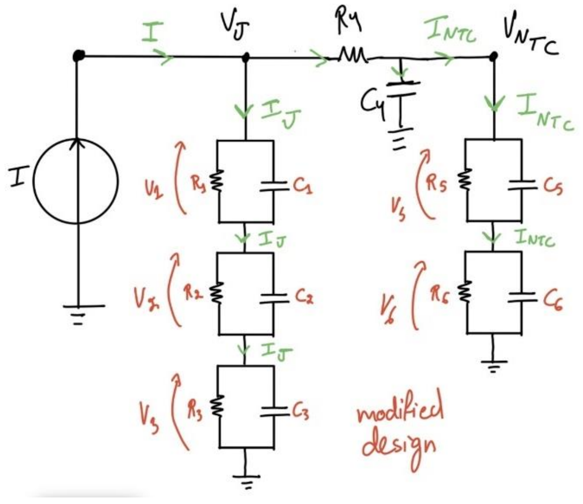


Figure 41 - The topology of the new assumed model with Foster Lumped Model equivalent structure.

As can be seen in **Figure 41**, the transitional component part (R_4, C_4) acts as a heat sink of the sort in the middle between the junction and the NTC. This shows that with this model, the behavior of the components in between the junction and the NTC are simplified meanwhile the modeling of the Junction and NTC is further complicated. Before diving deeper into the analysis, the overcomplication of NTC and Junction models observed already poses a problem regarding the weight of the program to be employed. The next step now is to distinguish the equations which govern the dynamics of this system in order to be able to construct the appropriate LTI.

2. LTI Construction

One way to find the equations that govern this circuit is to use circuit theory and treat the model as an electric circuit. However, another way to dissect and explore the circuit would be the use of Bond Graph representation which can pave the way to directly construct an LTI after isolating the system's states.

Bond Graph Representation and Modeling:

A bond graph is a powerful visual tool used to represent physical dynamic systems, providing a comprehensive framework for converting these systems into state-space representations. With the help of (Borutzky, 2011) and (Karnopp, Margolis, & Rosenberg, 2000), a brief yet important explanation of the key concepts will be presented. While bond graphs share similarities with other modeling techniques like block diagrams and signal-flow graphs, their distinctiveness lies in the representation of the two-way exchange of physical energy. This differentiates them from block diagrams and signal-flow graphs, where connections typically depict the one-way flow of information. Bond graphs are inherently versatile, capable of representing energy flow across different domains such as mechanical, electrical, hydraulic, and thermal, making them a domain-neutral modeling tool. This characteristic of seamlessly integrating multiple domains is particularly useful in studies involving complex, multi-domain systems.

The structure of bond graphs revolves around "bonds," which connect "single-port," "double-port," and "multi-port" elements. Each bond signifies the instantaneous energy flow, or power, described mathematically as the time derivative of energy $\left(\frac{dE}{dt}\right)$. This flow is captured by pairs of power variables—analogous to conjugate variables—whose product gives the instantaneous power transmitted through the bond. These power variables are categorized into two components: flow and effort. For example, in an electrical system, the flow variable corresponds to the current, while the effort variable represents voltage. The product of these two quantities gives instantaneous power flowing through the bond. This universal structure allows bond graphs to model various energy domains consistently, preserving the physical integrity of the system regardless of its complexity.

Two additional, essential features of bond graphs are the "half-arrow" sign convention and "causality." The half-arrow convention, akin to conventions used in electrical circuit diagrams and free-body diagrams, indicates the assumed direction of positive energy flow within the system. Though the direction is arbitrary, maintaining consistency throughout the analysis is crucial to avoid errors in interpretation. Causality, indicated by a vertical bar at one end of the bond (|→ or ←|), defines the cause-and-effect relationship between effort and flow. It establishes which variable (flow or effort) is dependent and which is independent, thus determining the system's dynamic behavior. Importantly, causality assignment follows specific rules, and precedence is carefully considered when applying it to the various ports within a bond graph.

When dealing with physical systems that exhibit dynamics over varying time scales, bond graphs prove particularly useful. For instance, systems with fast continuous-time behaviors can be represented as instantaneous phenomena using hybrid bond graphs, allowing for accurate modeling of complex dynamics. This makes bond graphs invaluable in analyzing multi-scale systems, where some processes are nearly instantaneous, and others evolve over extended periods.

Bond graphs were introduced by Henry Paynter and have since become a foundational tool in the modeling of dynamic systems. They are particularly useful in systems involving interactions across multiple physical domains. In these graphs, "1" and "0" junctions (represented in **Figure 42**) serve as fundamental building blocks. A "1" junction represents components that share a common flow—such as series connections in electrical systems—and can only have one effort arrow pointing outward. Conversely, "0" junctions represent components sharing a common effort, such as parallel connections in electrical systems, and have only one effort arrow pointing inward. These junctions allow for the intuitive modeling of energy distribution across components.

In light of this, the model structure assumed in the study is effectively translated into its corresponding bond graph representation as seen in **Figure 43**. For example, in the case of an electrical system, a current source I is represented as a source of flow S_f, corresponding to a current generator. The bond graph thus provides an efficient and coherent representation of the system's dynamic behavior, allowing for precise state-space modeling and analysis across various energy domains.

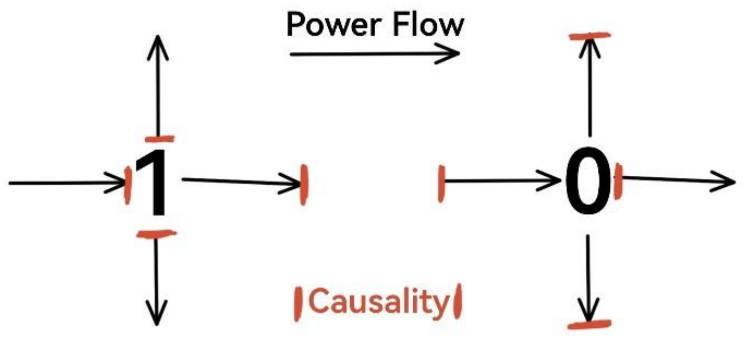


Figure 42 - (Left) Common Flow Junction, (Right) Common Effort Junction .

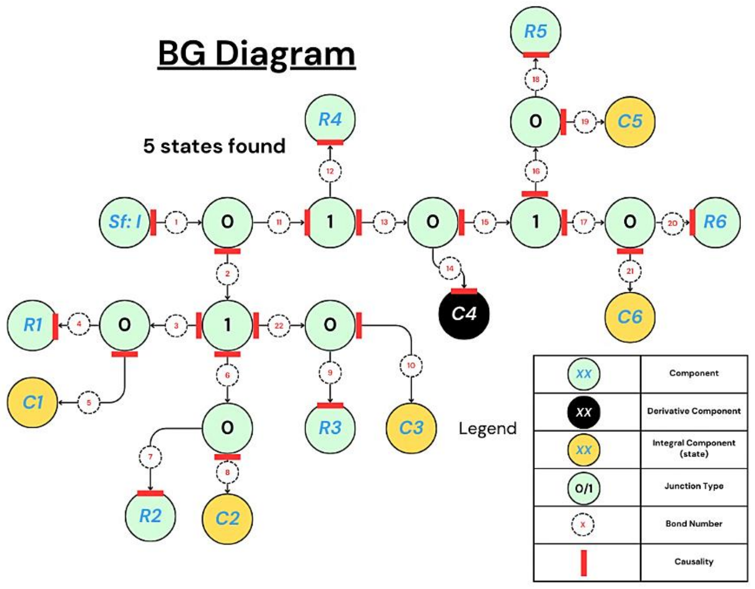


Figure 43 - Bond graph representation of the new assumed model structure with causality analysis.

With the help of the labeling seen in green, and the use of common flow and common effort laws, the equations that govern the system can be distinguished. The states that describe the dynamics are 5 and as follows: $\dot{Q}1, \dot{Q}2, \dot{Q}3, \dot{Q}5, \dot{Q}6$ which describe the charge variation of the capacitors respectively $C1, C2, C3, C5, and\ C6$. These 5 capacitors' respective efforts (Voltages/temperatures) are the goal of the analysis.

Common flow law: Flows of components around the junction are equal, and the sum of their efforts are zero. Example: f3=f2=f22=f6, e2-e3-e6-e22=0.

Common effort law: Efforts of components around the junction are equal, and the sum of their efforts are zero. Example: e3=e4=e5, f3-f4-f5=0.

- $\dot{Q}1 = f5 = f3 - f4 = f2 - \frac{e4}{R1} = I - \frac{e4}{R1} - f11 = I - \frac{e4}{R1} - \frac{e12}{R4} = I - \frac{V1}{R1} - \frac{V4}{R4}$

But $C1 * \dot{V1} = Q1$, therefore $\boldsymbol{\dot{V1} = \frac{I}{C1} - \frac{V1}{R1*C1} - \frac{V1+V2+V3-V5-V6}{R4*C1}}$ **state 1**

- $\dot{Q2} = f8 = f6 - f7 = f2 - \frac{e7}{R2} = I - \frac{e12}{R4} - \frac{e7}{R2}$

But $C2 * \dot{V2} = Q3$, therefore $\boldsymbol{\dot{V2} = \frac{I}{C2} - \frac{V2}{R2*C2} - \frac{V1+V2+V3-V5-V6}{R4*C2}}$ **state 2**

- $\dot{Q3} = f10 = f22 - f9 = f2 - \frac{e9}{R3} = I - \frac{e12}{R4} - \frac{e9}{R3}$

But $C2 * \dot{V2} = Q3$, therefore $\boldsymbol{\dot{V3} = \frac{I}{C3} - \frac{V3}{R3*C3} - \frac{V1+V2+V3-V5-V6}{R4*C3}}$ **state 3**

Relation between $\dot{V5}$ and $\dot{V6}$:

I_{NTC} is common along $V5$ and $V6$ (series connection), so $I_{R5} + I_{C5} = I_{R6} + I_{C6}$

=> $\frac{V5}{R5} + \dot{V5} * C5 = \frac{V6}{R6} + \dot{V6} * C6$, therefore $\boldsymbol{\dot{V6} = \frac{V5}{R5*C6} + \frac{C5}{C6} * \dot{V5} - \frac{V6}{R6*C6}}$ **relation found**

Relation between V_{C4} and $V5$:

$V_{C4} = V5 + V6$ (Capacitor C4 is parallel to V5 and V6) but $e14 = V_{C4}$
So, $\dot{e14} = \dot{V5} + \dot{V6}$ but, in the step before the relation between $\dot{V6}$ and $\dot{V5}$ is found.

Previous equation can be rewritten as: $\boldsymbol{\dot{e14} = \dot{V5} + \frac{C5}{C6} * \dot{V5} + \frac{V5}{R5*C6} - \frac{V6}{R6*C6}}$ **relation found**

Now, after finding the important relations needed, it is possible to calculate the remaining state equations for states $\dot{Q5}$ and $\dot{Q6}$.

- $\dot{Q5} = f19 = f16 - f18 = f15 - \frac{V5}{R5} = f13 - f14 - \frac{V5}{R5} = f11 - \frac{V5}{R5} - \dot{e14} * C4$

$$= I - f2 - \frac{V5}{R5} - C4 * \left(\dot{V5} * \left(1 + \frac{C5}{C6}\right) + \frac{V5}{R5 * C6} - \frac{V6}{R6 * C6}\right)$$

But $\dot{Q5} = \dot{V5} * C5$ therefore, the expression can be rewritten as:

$\boldsymbol{\dot{V5} = \frac{C6}{C5*C6+C4*C6+C4*C5}\left(I - \frac{V1+V2+V3-V5-V6}{R4} - \frac{V1}{R1} - \frac{V5}{R5} - \frac{C4}{R5*C6} * V5 + \frac{C4}{R6*C6} * V6\right)}$ **state 4**

But now using the previous relation found between $\dot{V6}$ and $\dot{V5}$: $\left(\dot{V6} = \frac{V5}{R5*C6} + \frac{C5}{C6} * \dot{V5} - \frac{V6}{R6*C6}\right)$

$\dot{V5} = \dot{V6} * \frac{C6}{C5} + \frac{V6}{R6*C5} - \frac{V5}{R5*C5}$ => so $\dot{e14} = \dot{V5} + \dot{V6} = \dot{V6} * \frac{C6}{C5} + \frac{V6}{R6*C5} - \frac{V5}{R5*C5} + \dot{V6}$ (useful relation later)

- $\boldsymbol{\dot{Q6} = \dot{V6} * C6} = f21 = f17 - f20 = f15 - \frac{V6}{R6} = f13 - f14 - \frac{V6}{R6}$

$$= I - \frac{V1 + V2 + V3 - V5 - V6}{R4} - \frac{V1}{R1} - \frac{V6}{R6} - \dot{e14} * C4$$

$$= I - \frac{V1+V2+V3-V5-V6}{R4} - \frac{V1}{R1} - \frac{V6}{R6} - C4 * \left(\dot{V6} * \frac{C6}{C5} + \frac{V6}{R6*C5} - \frac{V5}{R5*C5} + \dot{V6}\right)$$

So, the expression can be rewritten as:

$\boldsymbol{\dot{V6} = \frac{C5}{C6*C5+C4*C5+C4*C6} * \left(I - \frac{V1+V2+V3-V5-V6}{R4} - \frac{V1}{R1} - \frac{V6}{R6} - \frac{C4}{R6*C5} * V6 + \frac{C4}{R5*C5} * V5\right)}$ **state 5**

Matrix Building:

After isolating the 5 states equations, the next step as seen in unit 1 would be to construct the LTI based on the form: $\dot{\boldsymbol{x}} = \boldsymbol{A} * \boldsymbol{x}(\boldsymbol{t}) + \boldsymbol{B} * \boldsymbol{u}(\boldsymbol{t})$ in the continuous domain. (revisit unit 1 for more information)

The state vector in this case would be $\boldsymbol{x} = [\boldsymbol{V1}, \boldsymbol{V2}, \boldsymbol{V3}, \boldsymbol{V5}, \boldsymbol{V6}]^{\boldsymbol{T}}$

Meanwhile, the input vector $\boldsymbol{u} = [\boldsymbol{I}]$

After acquiring the derivative state vector $\dot{\boldsymbol{x}} = [\dot{\boldsymbol{x}}\boldsymbol{1}, \dot{\boldsymbol{x}}\boldsymbol{2}, \dot{\boldsymbol{x}}\boldsymbol{3}, \dot{\boldsymbol{x}}\boldsymbol{4}, \dot{\boldsymbol{x}}\boldsymbol{5}] = [\dot{\boldsymbol{V}}\boldsymbol{1}, \dot{\boldsymbol{V}}\boldsymbol{2}, \dot{\boldsymbol{V}}\boldsymbol{3}, \dot{\boldsymbol{V}}\boldsymbol{5}, \dot{\boldsymbol{V}}\boldsymbol{6}]$, the following step is to build the coefficient matrices A, B:

$$\mathbf{A} =$$

$$\begin{bmatrix}
-\frac{1}{R1*C1}-\frac{1}{R4*C1} & -\frac{1}{R4*C1} & -\frac{1}{R4*C1} & \frac{1}{R4*C1} & \frac{1}{R4*C1} \\
-\frac{1}{R4*C2} & -\frac{1}{R4*C2}-\frac{1}{R4*C2} & -\frac{1}{R4*C2} & \frac{1}{R4*C2} & \frac{1}{R4*C2} \\
-\frac{1}{R4*C3} & -\frac{1}{R4*C3} & -\frac{1}{R3*C3}-\frac{1}{R4*C3} & \frac{1}{R4*C3} & \frac{1}{R4*C3} \\
\left(\frac{C6}{C5*C6+C4*C6+C4*C5}\right)\left(-\frac{1}{R4}-\frac{1}{R1}\right) & \left(\frac{C6}{C5*C6+C4*C6+C4*C5}\right)\left(-\frac{1}{R4}\right) & \left(\frac{C6}{C5*C6+C4*C6+C4*C5}\right)\left(-\frac{1}{R4}\right) & \left(\frac{C6}{C5*C6+C4*C6+C4*C5}\right)\left(\frac{1}{R4}-\frac{1}{R5}-\frac{C4}{R5*C6}\right) & \left(\frac{C6}{C5*C6+C4*C6+C4*C5}\right)\left(\frac{1}{R4}-\frac{C4}{R6*C6}\right) \\
\left(\frac{C5}{C5*C6+C4*C6+C4*C5}\right)\left(-\frac{1}{R4}-\frac{1}{R1}\right) & \left(\frac{C5}{C5*C6+C4*C6+C4*C5}\right)\left(-\frac{1}{R4}\right) & \left(\frac{C5}{C5*C6+C4*C6+C4*C5}\right)\left(-\frac{1}{R4}\right) & \left(\frac{C5}{C5*C6+C4*C6+C4*C5}\right)\left(\frac{1}{R4}-\frac{C4}{R5*C5}\right) & \left(\frac{C5}{C5*C6+C4*C6+C4*C5}\right)\left(\frac{1}{R4}-\frac{1}{R6}-\frac{C4}{R6*C5}\right)
\end{bmatrix}$$

$$B = \begin{bmatrix} \frac{1}{C1} \\ \frac{1}{C2} \\ \frac{1}{C3} \\ \left(\frac{C6}{C5*C6+C4*C6+C4*C5}\right) \\ \left(\frac{C5}{C5*C6+C4*C6+C4*C5}\right) \end{bmatrix}$$

Now to determine the output of the LTI, 2 choices are present for the C coefficient vector based on the desired output:

$y(t) = C * x(t) = [1\ 1\ 1\ 0\ 0] * x(t)$ for $y = T_{junction} = x1 + x2 + x3$

$y(t) = C * x(t) = [0\ 0\ 0\ 1\ 1] * x(t)$ for $y = T_{NTC} = x4 + x5$

Note that the selection of C depends on the output needed from the feedback loop. So, for example, during the estimation of junction temperature, T_{NTC} is collected from the NTC and is reinserted through either Kalman or LQR filters for continuously recalibrating the LTI (as seen in unit 1). In that case the filter uses the LTI where the coefficient matrix $C = [0\ 0\ 0\ 1\ 1]$. Meanwhile for the recorded final estimated value of junction temperature calculation, $C = [1\ 1\ 1\ 0\ 0]$.

Preliminary Component Value Calculations and Assumptions:

The next step is to review the given data regarding the components of the circuit and the calculations and curve fits done by the company for this new assumed structure. So here are the assumed preliminary values:

$T_{Junction}$			
Power, W	3.85		
Stage	1 (R1,C1)	2(R2,C2)	3(R3,C3)
Tinf, o C	5.804	4.432	7.906
tau, s	1.24	0.12	33.82

Rth, K/W	1.508	1.151	2.054
Cth, J/K	0.822536	0.104242	16.46939

T_{NTC}			
Power, W	3.85		
Stage	1(R5,C5)	2(R6,C6)	3 (EMPTY)
Tinf, o C	223953	5.397	0
tau, s	1.80E+06	3.268	1

Rth, K/W	58169.6	1.402	0.000
Cth, J/K	30.94399	2.331258	#DIV/0!

Table 5 - Table displaying the thermal stages' component values for the assumed structure.

The data seen in **Table 5** is found by the company through curve fitting the curve that is plotted of the assumed PCB structure simulated in Finite Element Analysis software (FEM). The curves can be visualized in **Figure 45**. It is important to note that the given data is then used to calculate the missing component from the table which the transitional stage (bridge) also known as $R4$ *and* $C4$ from the assumed circuit structure of **Figure 41**. Since the provided data from the company is missing details regarding the calculations and how the exact structure used in the Finite Element Analysis, the best way to move on is to calculate $R4$ using circuit theory and then find $C4$ with trial-and-error curve fitting.

Estimating best value for R4:

To estimate the value of R4 (at least have an initial estimate), the circuit is assumed to be at steady state where the capacitors start acting like connecting wires. Thus, the circuit structure is further simplified as seen in **Figure 44**.

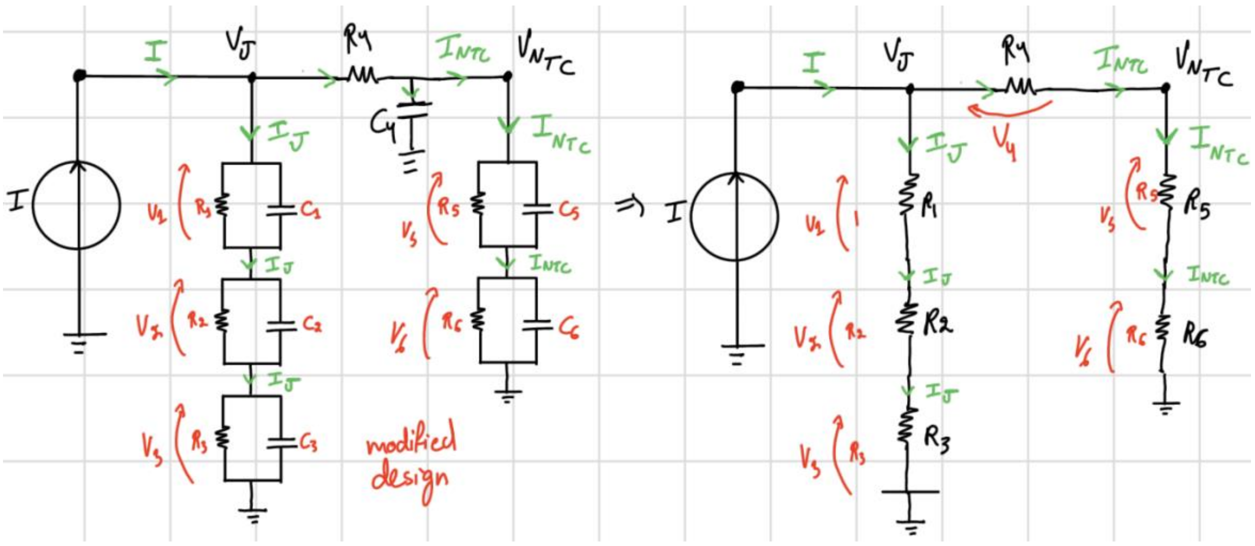


Figure 44 - Simplification of the circuit at steady state (High frequency).

Applying KVL: $V_J - V4 - V_{NTC} = 0 \Rightarrow V4 = V_J - V_{NTC}$

But according to Ohm's Law and Current splitting rule: $V4 = R4 * I4 = R4 * I * \left(\frac{R1+R2+R3}{R1+R2+R3+R4+R5+R6}\right)$

So, the expression can be rewritten as:

$$\boldsymbol{R4 = V4 * \left(\frac{R1 + R2 + R3 + R5 + R6}{I * (R1 + R2 + R3) - V4}\right)}$$

Following the given values of the **Table 5** and the corresponding curves (**Figure 45**) at T= 10 seconds, the values are assigned as: $V_J = 12.62\ V, V_{NTC} = 6.38\ V,\ I = 3.85\ A$.

Then $V4 = V_J - V_{NTC} = 12.62\ V - 6.38\ V = 6.24\ V$

Therefore, $\boldsymbol{R4} = \mathbf{27825.5245\ \Omega}$**.**

<u>Analyzing Company Given Data:</u>

From analyzing the circuit structure, it is clear that the calculation of components $R4\ and\ C4$ affect the voltage drop left at the NTC branch. So, that indicates that variations of $R4, C4$ values and assumed structure (parallel or series) greatly affects the voltage drop values. This will be apparent later when trying to curve fit the data. The company has provided, through graphing the Finite Element Analysis done through its software (FEM), 2 plots are produced visualized in **Figure 45**.

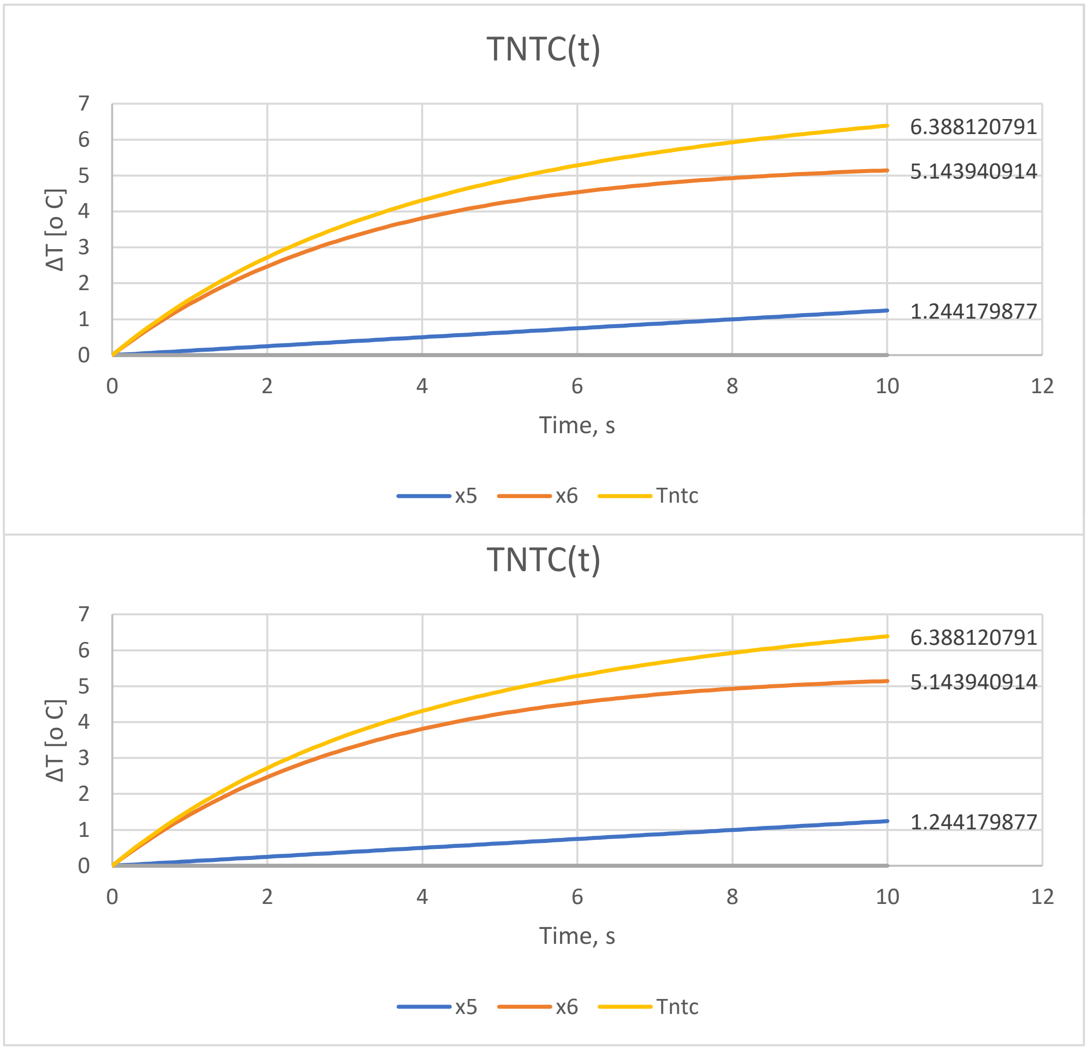


Figure 45 – Curves representing the data collected from FEM analysis

After collecting and graphing the company calculations and data, the next step would be running the same data given within the previously built model of unit 1 with the adoption of the newly structured LTI.

MATLAB Data Insertion and Preliminary Observations:

Just as done before in unit 1, the data now will be inserted into MATLAB with the appropriate new LTI structure and then discretized to be compared to the given curves of **Figure 45**. So, the MATLAB code for the LTI structure is firstly as follows in **Table 6**.

MATLAB CODE for LTI Section

```
clc;
clear all;
Ts=1/100; %controller sampling time
Ts2=1; %sensor sampling time
%parameters
R1=1.508;
tau1=1.24;
R2=1.151;
tau2=0.12;
R3=2.054;
tau3=33.82;
R4=27825.52447;
R5=58169.6;
tau5=1.8e6;
R6=1.402;
tau6=3.268;
C1=0.822; %0.822536182
C1=tau1/R1; %shifts everything except left transient peak exp2:*1.6 exp10:*10
C2=0.104; %0.104241877
C2=tau2/R2;%modifies the transient on both sides EXP:*0.1
C3=16.469; %16.46939034
C3=tau3/R3;
C4=1e-9; %ASSUMED
C5=30.944;
C5=tau5/R5;
C6=2.33;
C6=tau6/R6;
R_DS=  0.00245 ;%page 96 %probably truest: Drain Source Resistance (same as unit 1)
%matrices
A=[-1/(R1*C1)-1/(R4*C1),-1/(R4*C1),-1/(R4*C1),1/(R4*C1),1/(R4*C1);
   -1/(R4*C2),-1/(R2*C2)-1/(R4*C2),-1/(R4*C2),1/(R4*C2),1/(R4*C2);
   -1/(R4*C3),-1/(R4*C3),-1/(R3*C3)-1/(R4*C3),1/(R4*C3),1/(R4*C3);
   (C6/(C5*C6+C4*C6+C4*C5))*(-1/R4-1/R1),(C6/(C5*C6+C4*C6+C4*C5))*(-
1/R4),(C6/(C5*C6+C4*C6+C4*C5))*(-1/R4),(C6/(C5*C6+C4*C6+C4*C5))*(1/R4-1/R5-
C4/(R5*C6)),(C6/(C5*C6+C4*C6+C4*C5))*(1/R4-C4/(R6*C6));
   (C5/(C5*C6+C4*C6+C4*C5))*(-1/R4-1/R1),(C5/(C5*C6+C4*C6+C4*C5))*(-
1/R4),(C5/(C5*C6+C4*C6+C4*C5))*(-1/R4),(C5/(C5*C6+C4*C6+C4*C5))*(1/R4-
C4/(R5*C5)),(C5/(C5*C6+C4*C6+C4*C5))*(1/R4-1/R6-C4/(R6*C5))];
B=[1/C1;1/C2;1/C3;(C6/(C5*C6+C4*C6+C4*C5));(C5/(C5*C6+C4*C6+C4*C5))];
C=[0 0 0 1 1]; %makes NTC output
% C= [1 1 1 0 0]; %makes Junction output
D=0;
sys_x=ss(A,B,eye(5),0);
sys=ss(A,B,C,0);
%discretization of LTI
dt=c2d(sys,Ts,'zoh');
```

```
Ad=dt.A;Bd=dt.B;Cd=dt.C;
sys_dt=ss(Ad,Bd,eye(5),0,Ts);
out=sim("data_check.slx"); %loading the data into the simulation space
```
Table 6 - MATLAB code for LTI section.

The output result as seen in **Table 7:**

A					B
-8.06E-01	-4.37E-05	-4.37E-05	4.37E-05	4.37E-05	1.22E+00
-3.45E-04	-8.33E+00	-3.45E-04	3.45E-04	3.45E-04	9.59E+00
-2.18E-06	-2.18E-06	-2.96E-02	2.18E-06	2.18E-06	6.07E-02
-2.14E-02	-1.16E-06	-1.16E-06	6.06E-07	1.16E-06	3.23E-02
-2.85E-01	-1.54E-05	-1.54E-05	1.54E-05	-3.06E-01	4.29E-01
C	0	0	0	1	1

Table 7 - Values of Matrices Ad, Bd, and Cd.

Values after discretization as seen in **Table 8**:

A_d					B_d
9.92E-01	-4.18E-07	-4.35E-07	4.35E-07	4.35E-07	1.21E-02
-3.30E-06	9.20E-01	-3.31E-06	3.31E-06	3.30E-06	9.20E-02
-2.18E-08	-2.09E-08	1.00E+00	2.18E-08	2.18E-08	6.07E-04
-2.13E-04	-1.11E-08	-1.16E-08	1.00E+00	1.15E-08	3.22E-04
-2.83E-03	-1.47E-07	-1.53E-07	1.53E-07	9.97E-01	4.27E-03
C_d	0	0	0	1	1

Table 8 - Discretized values of Matrices Ad, Bd, and Cd.

Then, with this data, the LTI block is run in Simulink and the states are separated appropriately to create similar curves to **Figure 45**. The Simulink simulation space and resulting curves can be visualized in **Figure 46**.

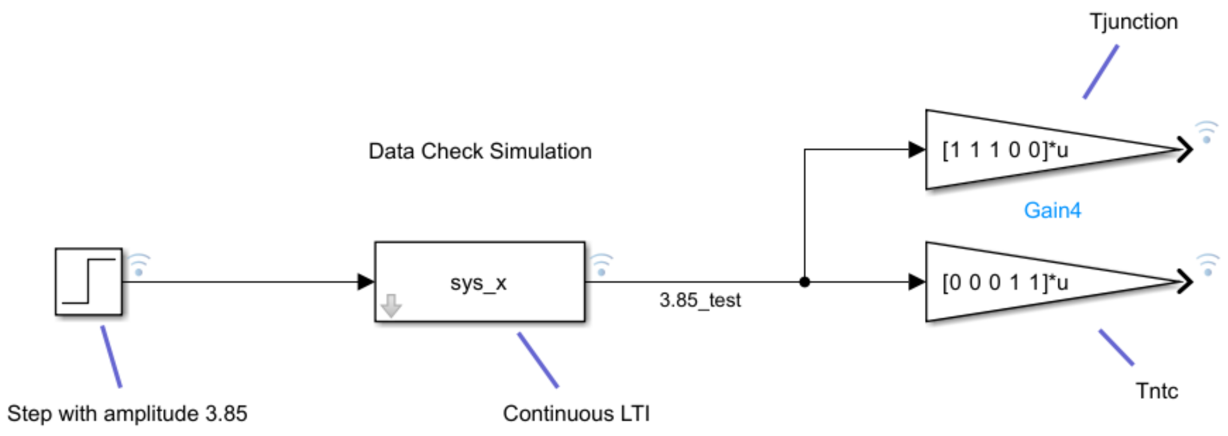


Figure 46 - Simulink model of the simulation space used to test the new LTI model.

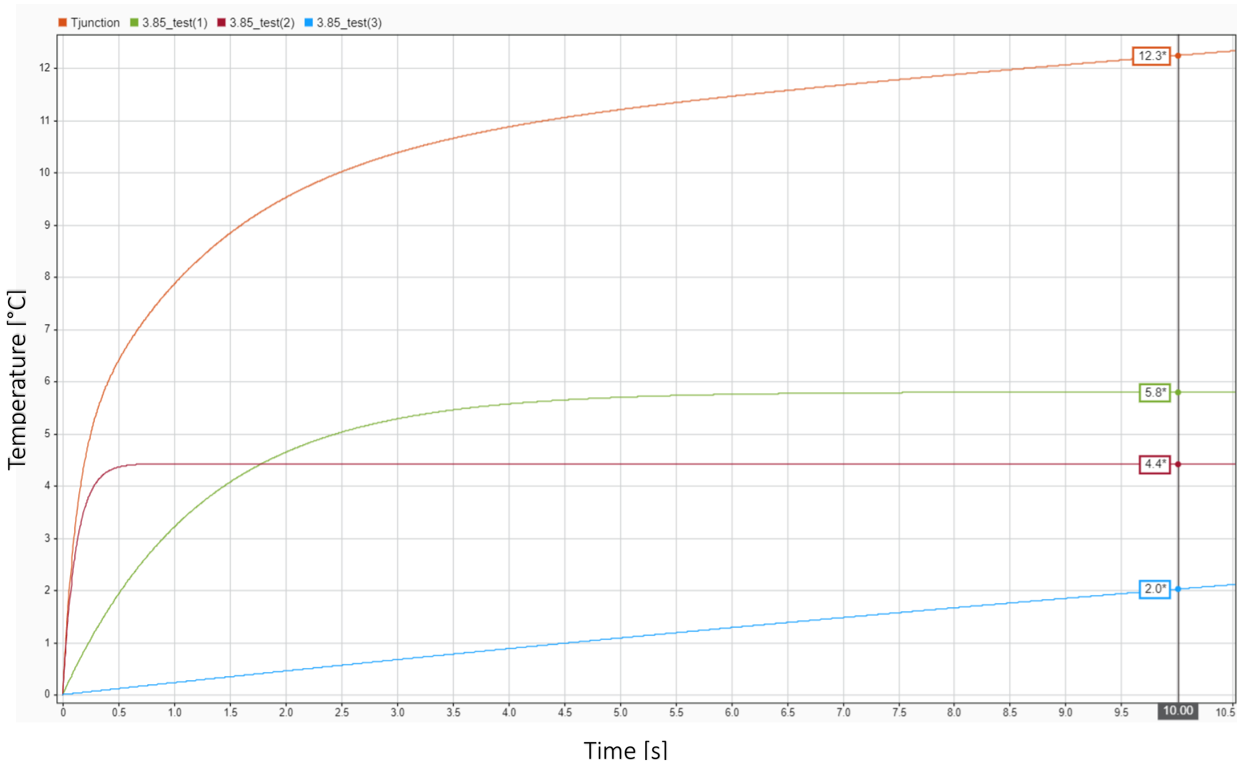


Figure 47 – Simulink plot of $T_{junction}$ along with all of its states (x_1, x_2, x_3)

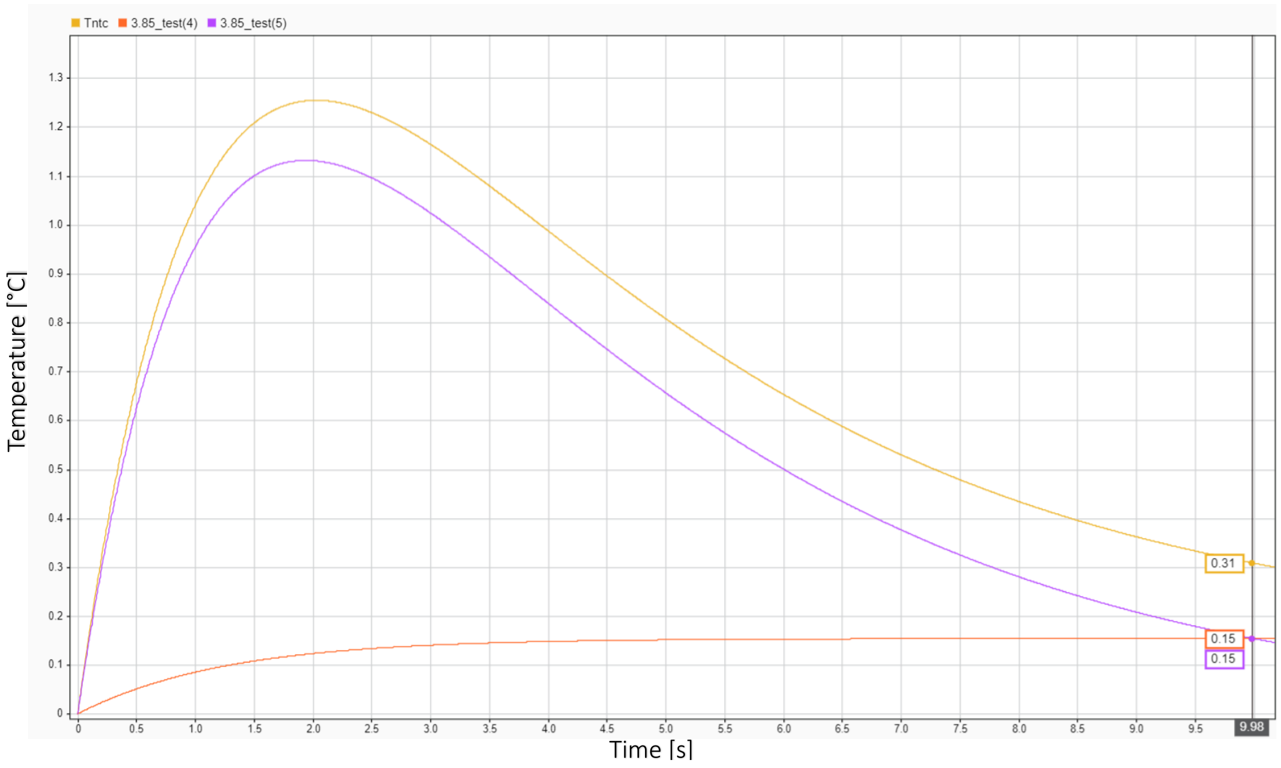


Figure 48 - Simulink plot of T_{NTC} along with all its states (x_4, x_5)

The plots of **Figure 47** and **Figure 48** shown previously provide 2 main conclusions. $T_{junction}$ data is consistent with the given assumed data but T_{NTC} on the other hand shows a mismatch. The most probable reason for the mismatch can be traced to the inaccurate modeling done by the company (misinterpretation of model structure) on the transitional section present in between the NTC and the junction. The voltage drop dynamics (Temperature drop) at that midsection directly influence the final value obtained at the NTC. This will possibly corrupt the calibration mechanism employed in the Simulink feedback mechanism during the insertion of NTC values into the system. However, the study will be resumed in order to study the effect of this mismatch in detail and see if there is a way to manually compensate for that error and possibly still save the model.

So, the next step would be to load the augmented matrices into MATLAB and construct the compensator mechanism along with the appropriate feedback matrices.

MATLAB CODE for compensator

```
Aaug=[1 -Ts*Cd;zeros(length(A),1) Ad];
Baug=[0;Bd];
Caug=[0 Cd];
Q=diag([10 10 1 1 1 1]); %started at Q= diag([10 10 1 1 1 1]) which failed to find a
%solution and then found the result Q=diag([10 10 1 1 1 1])
R=1;
Cq=chol(Q);
Mo=rank(obsv(Aaug,Cq));% if Mo = dim(Aaug)=6 , then the system is observable
K=dlqr(Aaug,Baug,Q,R);%discrete LQR
Kq=K(1);
Kx=K(2:end);
%% compensator gain
meth1_cor=0.025e-2; %0.075 %0.0547 exp1:0.0585 exp2:*1.5 0.0995e-3
meth2_cor=meth1_cor;
%since the error reinserted into LTI from Kalman is tiny, meth2_cor
%is useless in this case
out=sim("new_sim_space1.slx"); %simulating the given data from unit 1
%% plotting section: Just for having a good visual representation of the data
figure(1),subplot(1,2,1);
plot(out.Tjunc_est.time(),out.Tjunc_est.data(),'r');hold on;
plot(out.Tjunc_sim.time(),out.Tjunc_sim.data(),'b');
grid on;legend("Estimated Junction temp (LQR)","Given real junction temp");
title 'Estimated Result VS Given Result Tjunc';
subplot(1,2,2);
error=abs(out.Tjunc_est.data()-out.Tjunc_sim.data());
plot(out.Tjunc_est.time(),error,'k');hold on;
grid on;legend("Error plot)");yline(5,'--r');
title 'Error plot';
```

Table 9 - MATLAB code for compensator.

This code from **Table 9** is run and compiled into the simulation space which is built from unit 1 (same model built) which can be seen in the **Figure 49**. The results of the code can be seen as graphed in **Figure 50** and are quite clearly off. The behavior suggests that the given variables need tweaking along with the need also for manually changing the compensator gain manually until reaching a more satisfactory curve result. Moreover, in this run due to the NTC mismatch that has been mentioned before, there will be a focus only on the LQR compensator structure rather than the Kalman filter.

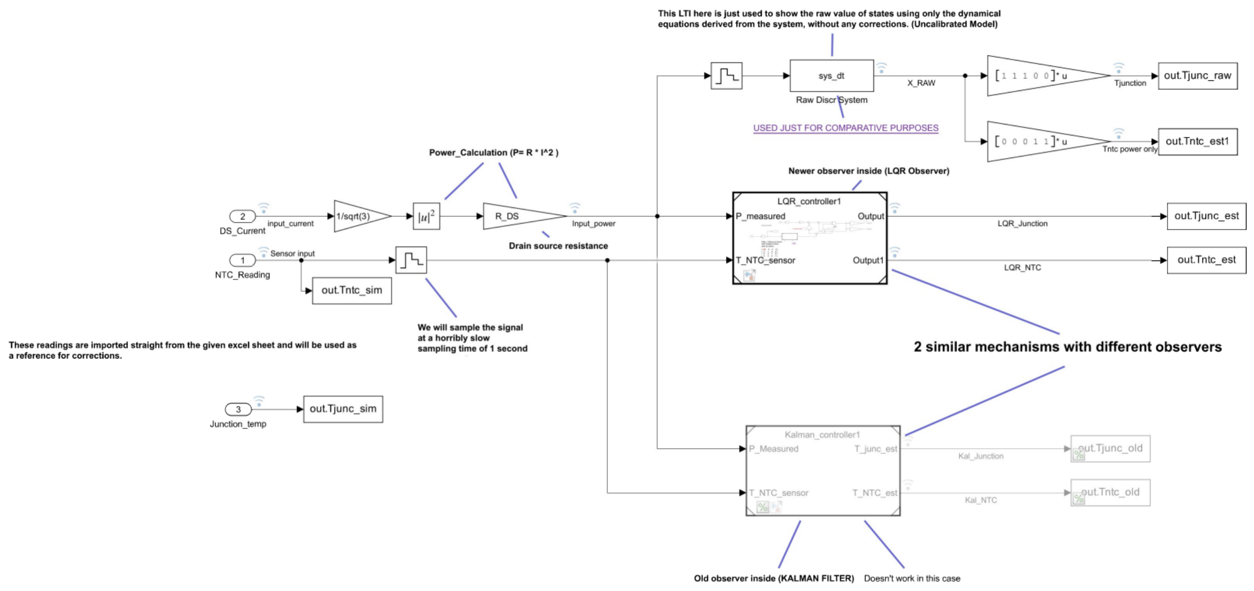

This LTI here is just used to show the raw value of states using only the dynamical equations derived from the system, without any corrections. (Uncalibrated Model)
sys_dt
Raw Discr System
X_RAW
[1 1 1 0 0]* u
Tjunction
out.Tjunc_raw
[0 0 0 1 1]* u
Tntc power only
out.Tntc_est1
USED JUST FOR COMPARATIVE PURPOSES
Power_Calculation (P= R * I^2)
Newer observer inside (LQR Observer)
LQR_controller1
P_measured
Output
T_NTC_sensor
Output1
2
DS_Current
input_current
1/sqrt(3)
|u|²
R_DS
Input_power
Drain source resistance
1
NTC_Reading
Sensor input
out.Tntc_sim
LQR_Junction
out.Tjunc_est
LQR_NTC
out.Tntc_est
We will sample the signal at a horribly slow sampling time of 1 second
These readings are imported straight from the given excel sheet and will be used as a reference for corrections.
2 similar mechanisms with different observers
3
Junction_temp
out.Tjunc_sim
Kalman_controller1
P_Measured
T_junc_est
T_NTC_sensor
T_NTC_est
Kal_Junction
out.Tjunc_old
Kal_NTC
out.Tntc_old
Old observer inside (KALMAN FILTER)
Doesn't work in this case

Figure 49 - Simulation space for the new assumed structure.

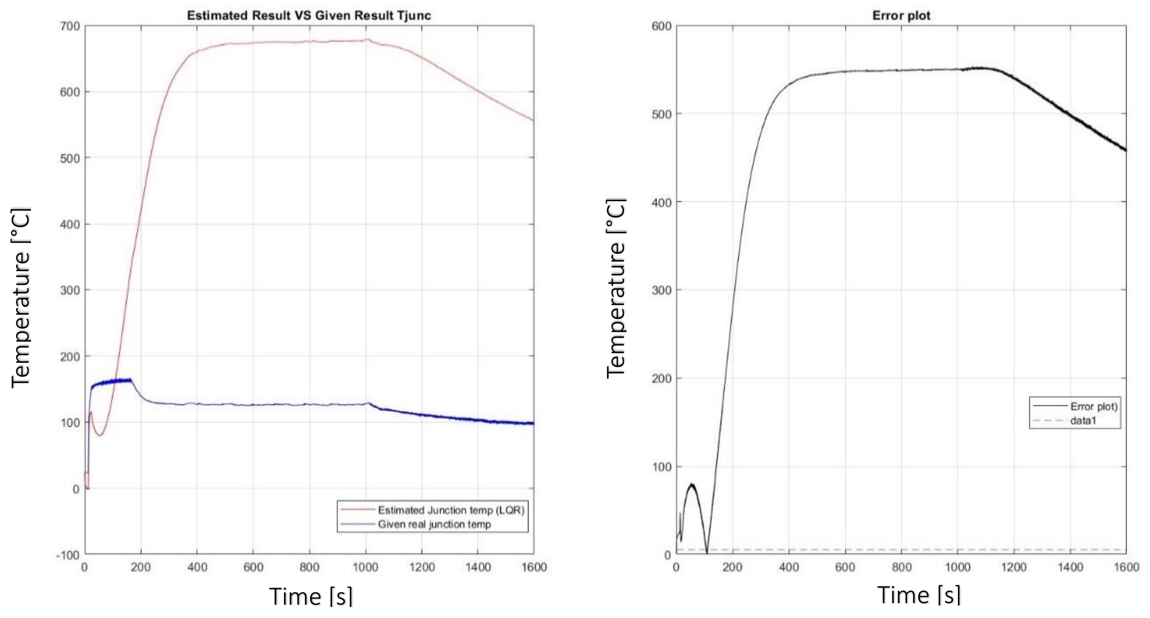


Figure 50 - Output plots of the first simulation with new circuit structure using the model of unit 1.

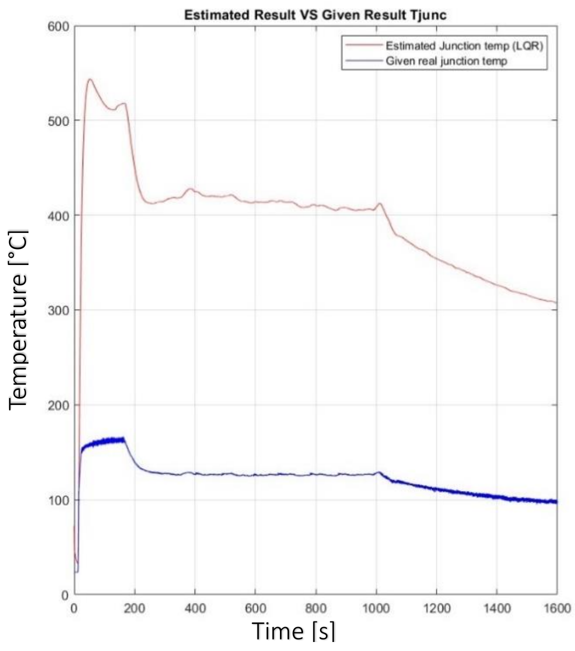
3. Tweaking and Modifications

As the title suggests, this section covers the tweaks and changes that are introduced to match the outputted $T_{junction}$ curve to the given reference one from FEM software. So, through rigorous trial and error the following conclusions can be drawn from some of the component values and their effect on the final curve visualized in **Table 10**:

Component	Effect of change	Initial Value	Final Value
$R1$	shifts steady state strongly and others weakly	1.508	1.508
$R2$	affects peak strongly but shifts steady state too	1.151	0.1151
$R3$	affects transients strongly	2.054	8.2160
$R4$	--	27825.52447	27825.52447
$R5$	shifts all but peak	58169.6	58169.6
$R6$	modifies left side minimum	1.402	1.402
$C1$	shifts everything except left transient peak	0.822	11.1008
$C2$	modifies the transient on both sides	0.104	1.0426
$C3$	modifying this modifies strictly the left side of curve	16.469	4.1164
$C4$	ASSUMED	1e-9	1e-9
$C5$	--	30.944	10.1589
$C6$	tilts curve counter-clockwise slightly	2.33	2.331
Q	Affects the shape of LQR feedback gain Kq and Kx which can affect accuracy of data (minor changes)	**diag([10 10 1 1 1 1])**	**diag([1000 100 1 1 1 1])**
$meth1_{cor}$	LQR method calibrator gain	**0.025e-2 (assumed)**	**2.1e-4**

Table 10 - Table showcasing the effect of changing some component values on the output plots.

The observed results after the changes are as follows:



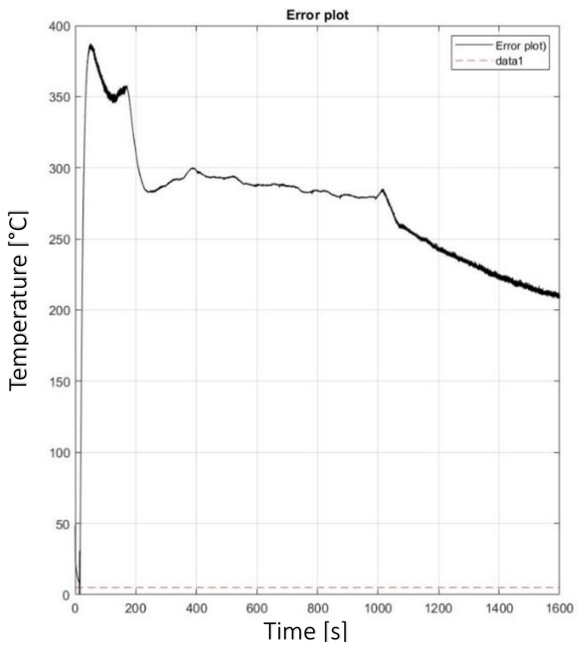


Figure 51 - Simulation result of new assumed circuit structure after component tweaking.

The results are already much better than the previous ones from the dynamics side. It is very clear that the observed error with respect to the reference is proportional and can be mitigated by simply adding a gain to the $T_{junction}$ output that is manually selected through trial and error method again. And so a gain is injected called $comp = 0.315$ within the controller with LQR inside as seen in **Figure 52**.

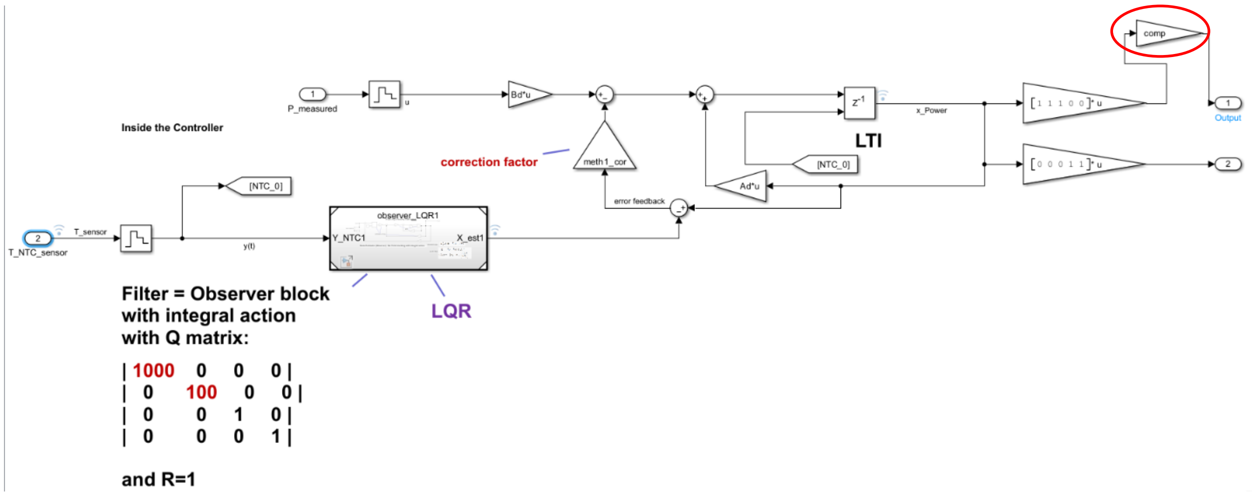


Figure 52 - Simulink structure of controller with the newly added "comp" gain.

After the final addition, the new results are plotted in **Figure 53**. These results seem optimal, and as seen on the right side of that figure, in the error plot, the curve lies mainly below the set threshold of 5 degrees. This threshold is exceeded understandably in the first phase of the simulation where a sudden increase of the current to peak which causes sharp oscillations and behavior is observed. The shape observed is hence realistic and can be accepted. Moreover, if the correct data was provided from the start regarding the components of the thermal circuit, this simulation would have provided more accurate results with minimal change and intervention.

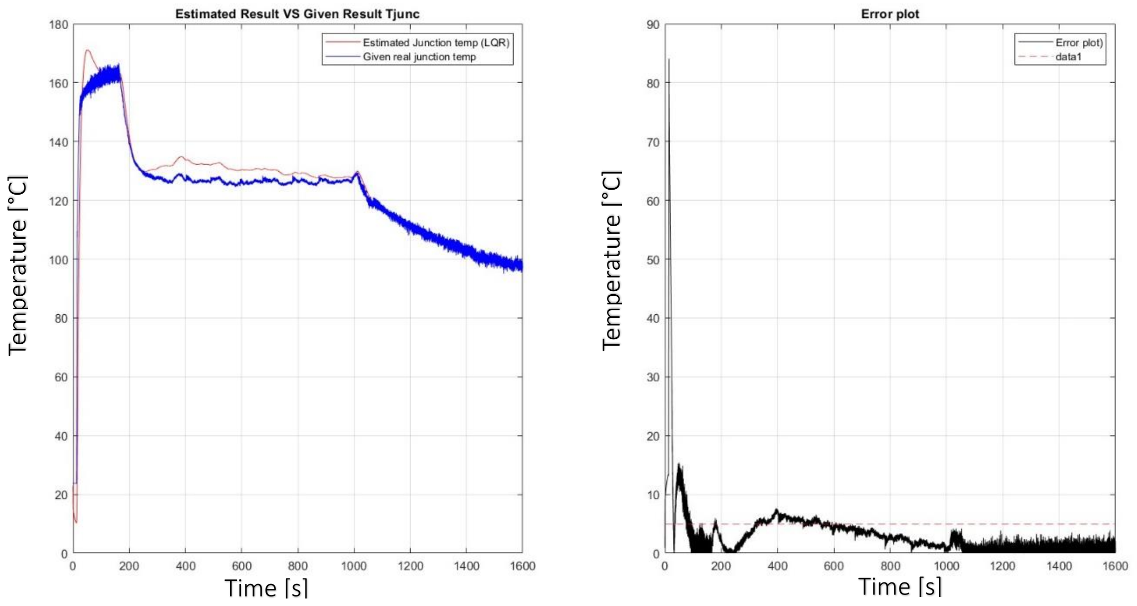


Figure 53 - Results of Simulink model after addition of the "comp" gain.

Moreover, as seen in unit 1, the company provided a second simulation with a different input power pattern which shall be called: Simulation B. Applying the same given model with the new assumed model provides the results seen in **Figure 54** and **Figure 55**.

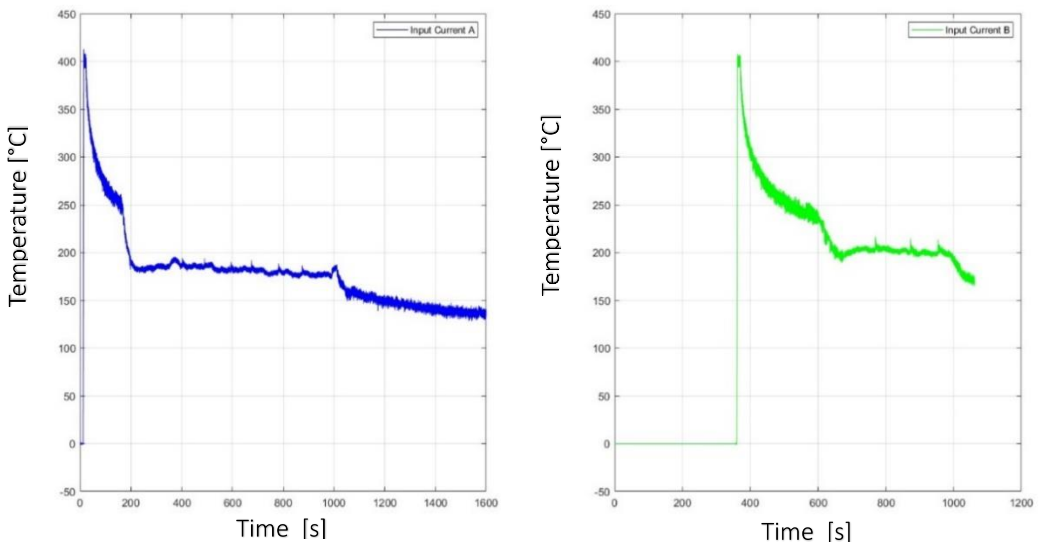


Figure 54 - The 2 different input current (drain-source) case studies.

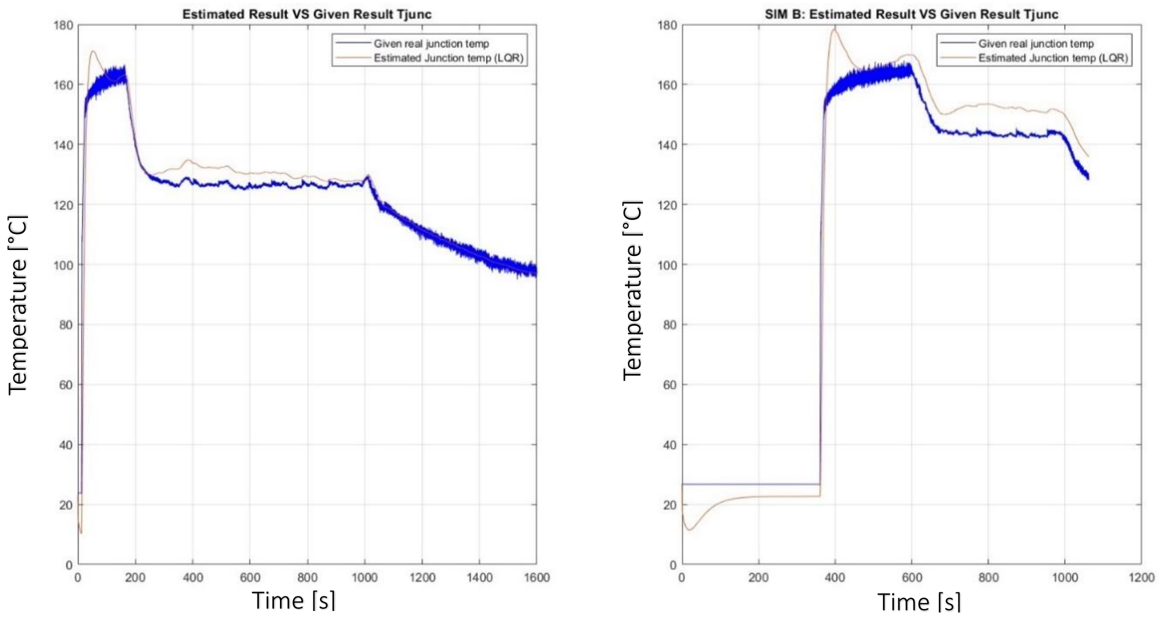


Figure 55 - Simulations A & B side by side after components modifications.

These results clearly show that with the modification of the input power pattern, the output estimation algorithm is off by a degree. This degree of error can be traced due to the inaccuracies within the given component data, which either way does not undermine the accuracy in mimicking the behavior of the curve due to the thermal dynamics involved. The error degree can be seen in the plots of **Figure 56**.

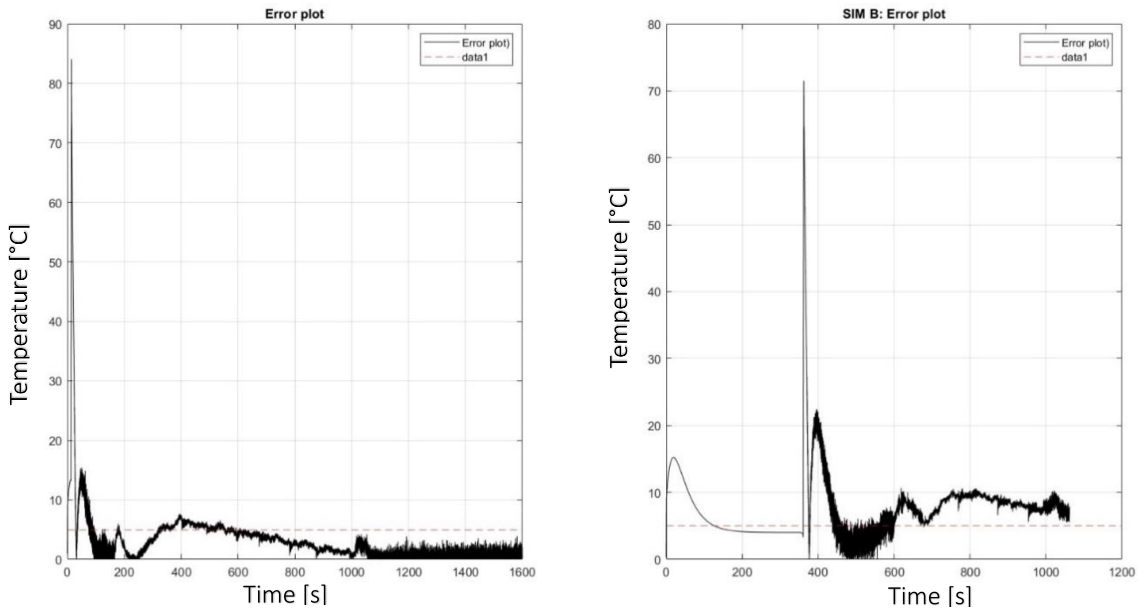


Figure 56 - Error plot of simulations A and B after modification.

The error plot on the right side (B) is crossing the set threshold of 5 degrees of error. So, the mathematical model can be reworked and enhanced to suit both simulations a bit better given the faulty data used from the start. So, in the final section of this study, an error mitigation step is added in order to optimize the faulty model as much as possible to satisfy better the requirements from both simulations.

Final Optimization:

As mentioned before; to better satisfy the error threshold of simulation B, another data modification step is introduced to mitigate the errors observed. So, through curve fitting and trial and error, the following rigorous data modification is obtained:

Component	Effect of change	Secondary Value	Final Value
$R1$	shifts steady state strongly and others weakly	1.508	1.508
$R2$	affects peak strongly but shifts steady state too	0.1151	2.3020
$R3$	affects transients strongly	8.2160	8.2160
$R4$	--	27825.52447	27825.52447
$R5$	shifts all but peak	58169.6	78528.96
$R6$	modifies left side minimum	1.402	9.8140
$C1$	shifts everything except left transient peak	11.1008	11.1008
$C2$	modifies the transient on both sides	1.0426	0.0521
$C3$	modifying this modifies strictly the left side of curve	4.1164	4.1164
$C4$	ASSUMED	1e-9	1e-9
$C5$	--	10.1589	7.4128
$C6$	tilts curve counter-clockwise slightly	2.331	0.3330
Q	Affects the shape of LQR feedback gain Kq and Kx which can affect accuracy of data (minor changes)	**diag([1000 100 1 1 1 1])**	Rigorously copied Kx and Kq from MATLAB of past simulation due to high complexity of LQR
$meth1_{cor}$	LQR method calibrator gain	**2.1e-4**	**2.1e-4**

The results of such simulations were not satisfactory enough but were noted nevertheless to understand the effects of component modifications on the output of the model. These results are plotted in **Figure 57** and **Figure 58**.

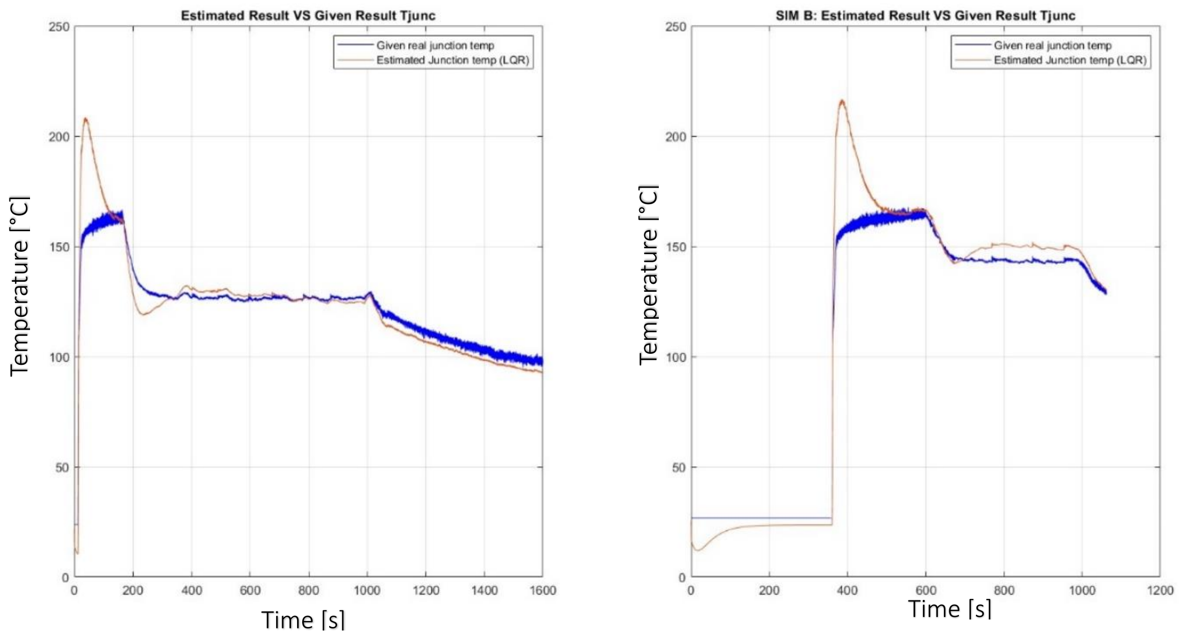


Figure 57 - Output plots of simulations A and B after final modifications.

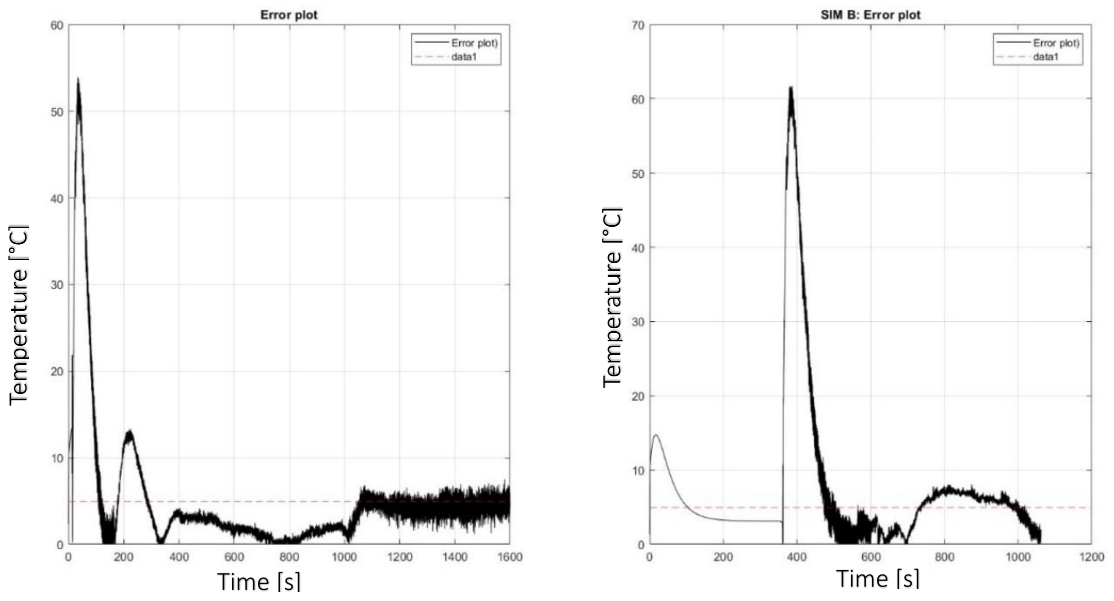


Figure 58 – Error plot of simulations A and B after final modifications.

Conclusion of Thesis and Research

In this work, a simplified mathematical thermal model for power electronics was developed which is demonstrated through a detailed case study focused on estimating the junction temperature of the onboard transistor. The model leverages an onboard NTC (negative temperature coefficient thermistor), which recalibrates the LTI feedback control system, allowing for more accurate and dynamic temperature predictions. Despite facing challenges with corrupted and unsalvageable reference data, the model proved its flexibility by achieving a close match to this corrupted data through rigorous curve fitting. This adaptability showcases the robustness of the model, not only in its predictive accuracy but also in its ability to maintain proper thermal dynamics, even in less-than-ideal data conditions.

To compare the results of both units, another plot is visualized in **Figure 59**. As mentioned at the end of Unit 1, the general shape desired is varied between both cases to diversify the study. In that of Unit 1, the aim was to obtain a peak estimation before the actual peak is attained (check the green curve). On the other hand, with the curve of Unit 2, the aim was to obtain a more "live" tracking of the temperature. Thus, the curve attempts to estimate the temperature even at the peak of the provided amperage. Both results are relatively good considering the limited data and time the thesis had at hand. The error plots of these curves can also be visualized in **Figure 60**. Even though both models exhibit similar behavior, that of Unit 1 is still considered the more favorable model due to the simplicity of the model in comparison to that of Unit 2 without sacrificing flexibility and accuracy.

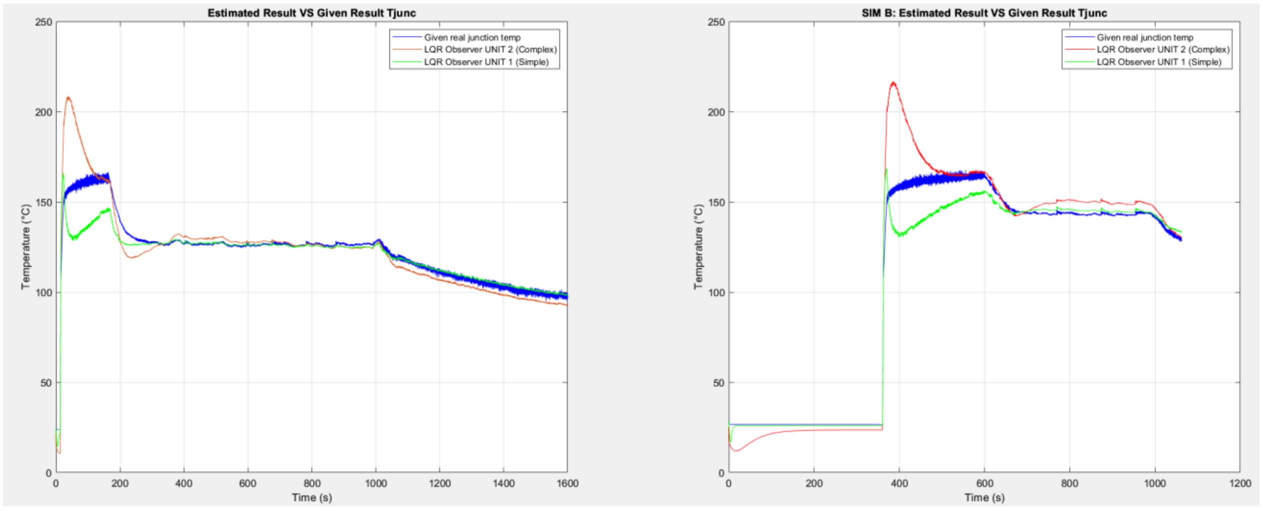


Figure 59 - Comparison between the model of unit 1 (Simple Model) and that of Unit 2 (Complex Model) with 2 different simulation data (Sim A on left and Sim B on right).

The results of this study highlight the reliability and versatility of the thermal model in estimating real-time thermal behavior and adapting to varying conditions through recalibration. This makes it a valuable tool for ensuring thermal stability in PCB designs, particularly in applications requiring precise temperature control, such as sensitive electronics and high-performance computing systems. The flexibility demonstrated by the model suggests its applicability in a wide range of scenarios where component behavior must be monitored and managed in real time, ensuring both operational efficiency and the longevity of components.

Moreover, this modeling technique has potential applications across various technological and mobility sectors, including automotive, aerospace, and electronics manufacturing. By providing a framework for studying the lifecycle of critical components, such as transistor junctions, this model can contribute to the design of more thermally efficient systems. Its predictive capabilities could help mitigate thermal wear, reducing the frequency of component replacements and improving the overall durability and efficiency of complex systems. Such improvements could have far-reaching implications, contributing to more sustainable and cost-effective production processes across industries by extending the lifespan of key components.

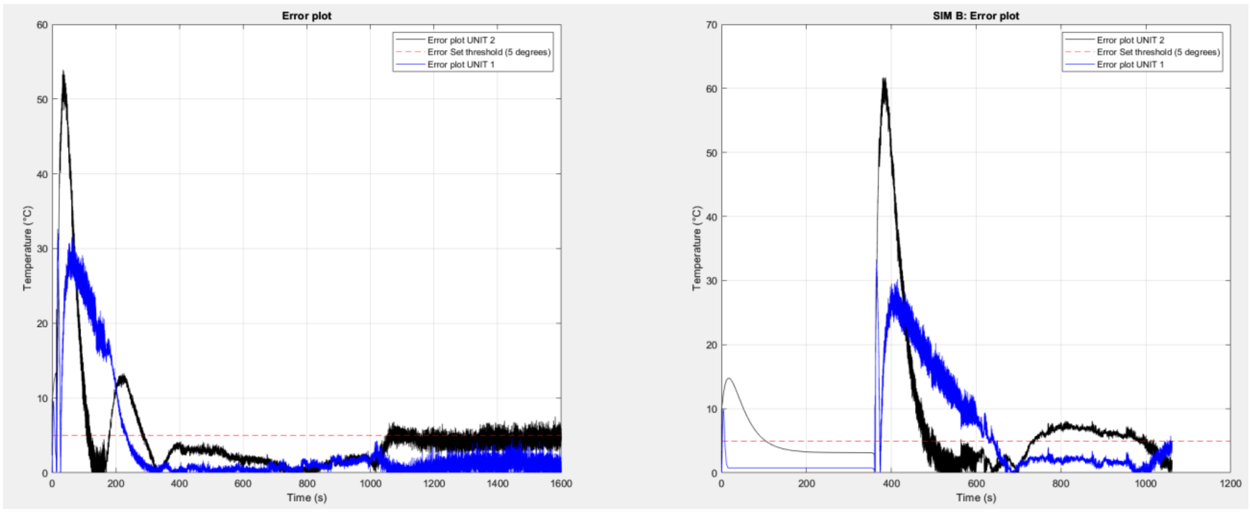


Figure 60 - Error plots of the compared models from Unit 1 and Unit 2 during simulations Sim A(Left) and Sim B(Right).

For future work, this study voices the critical importance of ensuring the accuracy and integrity of component data. Compromised or inaccurate data can significantly affect the performance of grey-box models like the one presented here, potentially leading to erroneous predictions. Addressing these challenges in future research will be essential for further enhancing the model's accuracy and expanding its range of applications. In this study, it is shown that through reference data, it is possible to curve fit components to mitigate errors without compromising the predictive capabilities of the model. Additionally, exploring more complex system configurations and dynamic environments would allow for the continued refinement of this approach, pushing the boundaries of what can be achieved with thermal modeling in diverse technological settings. Moreover, a future study could possibly attempt to generate the code on PCBs to study the "heaviness" of the generated code on the chip and attempt to optimize it.

Bibliography

Anderson, B. D., & Moore, J. B. (2007). *Optimal Control: Linear Quadratic Methods.* Courier Corporation.

Borutzky, W. (2011, August 5). *Bond Graph Modelling of Engineering Systems.* New York: Springer New York. Retrieved from Wikipedia: https://en.wikipedia.org/w/index.php?title=Bond_graph&oldid=1238720973

Chumbley, A., Areias, J., Williams, C., & Khim, J. (n.d.). *Brilliant.org*. Retrieved from Linear Time Invariant Systems | Brilliant Math & Science Wiki: https://brilliant.org/wiki/linear-time-invariant-systems/

Giannakis, A., & Peftitsis, D. (2019). *Electro-thermal Design of a Solid-State MVDC.* Norwegian University of Science and Technology.

Infineon Technologies AG. (2020, April 14). *Transient thermal measurements and thermal.* Munich: Infineon Technologies AG. Retrieved from https://www.infineon.com/

Karnopp, D. C., Margolis, D. L., & Rosenberg, R. C. (2000). *System Dynamics : Modeling and Simulation of Mechatronic Systems.* New York: Wiley-Interscience.

Kraus, A. D., Aziz, A., & Welty, J. (2011). *Extended Surface Heat Transfer.* Hoboken, NJ: Wiley-Interscience.

Moore, G. E. (1965). *Cramming more components onto integrated circuits.*

Ogata, K. (2010). *Modern Control Engineering.* Upper Saddle River, NJ: Prentice Hall.

Oppenheim, A. V., & Willsky, A. S. (1997). *Signals and Systems.* Upper Saddle River: NJ: Prentice Hall.

Piumatti, D., Quitadamo, M. V., Reorda, M. S., & Fiori, F. (2020). *Testing Heatsink Faults in Power Transistors by means of Thermal Model.* Torino.

Rowell, D. (n.d.). *Advanced System Dynamics and Control Discrete Time Observers and LQG Control.* MASSACHUSETTS INSTITUTE OF TECHNOLOGY.

Zienkiewicz, O., Taylor, R., & Zhu, J. (2013). *The Finite Element Method: Its Basis and Fundamentals.* The Boulevard, Langford Lane, Kidlington, Oxford, OX5 1GB, UK: Elsevier.