

D4.3. Definition And Implementation of a Holistic Digital Twin of the IMD



Reinventing High-performance pOwer converters for heavy-Duty electric trAnSport

Grant Agreement Number 101056896

Deliverable name: Definition and implementation of a holistic digital twin of the IMD
Deliverable number: 4.3
Deliverable type: R — Document, report
Work Package: WP4: Software design and development of digital tools
Lead beneficiary: AU
Contact person: Arman Fathollahi; arman.f@ece.au.dk
Dissemination Level: PU
Due date for deliverable: April 30, 2025



**Funded by the
European Union**

DOCUMENT CONTROL PAGE

Author(s):	Arman Fathollahi (AU),
Contributor(s):	Corneliu Barbu (AU), Uffe Jakobsen (AU), Mathias Løgstrup (AU)
Reviewer(s):	Thibault Lemaire (Valeo), Jose Sáez (NION), Andrés Muñoz (UPC)
Version number:	v.6
Contractual delivery date:	30-04-2025
Actual delivery date:	30-04-2025
Status:	Submitted

REVISION HISTORY

Version	Date	Author/Reviewer	Notes
v.0	15 – 04 – 2025	Arman Fathollahi (AU)	Creation, First Draft
v.1	21 – 04 – 2025	Arman Fathollahi (AU)	Updates and draft review
v.2	25 – 04 – 2025	Andres Muñoz (UPC)	Review and format corrections
v.3	23 – 04 – 2025	Jose Sáez (NION)	Review and format corrections
v.4	28 – 04 – 2025	Thibault Lemaire (Valeo)	Review and format corrections
v.5	29 – 04 – 2025	Arman Fathollahi (AU) and Mathias Løgstrup (AU)	Updates, draft revisions and finalization
v.6	30 – 04 – 2025	Luis Romeral (UPC)	Final version. Submitted

ACKNOWLEDGEMENTS

The work described in this publication was subsidised by Horizon Europe (HORIZON) framework through the Grant Agreement Number 101056896.

DISCLAIMER

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or CINEA. Neither the European Union nor the granting authority can be held responsible for them.

TABLE OF CONTENTS

LIST OF FIGURES 4

LIST OF TABLES..... 6

EXECUTIVE SUMMARY 7

1 INTRODUCTION 8

 1.1 DESCRIPTION OF THE DOCUMENT AND PURSUE 8

 1.2 WPs AND TASKS RELATED WITH THE DELIVERABLE 8

2 RHODAS Digital Twin Frameworks 9

 2.1 RHODAS Inverter Digital Twin 10

 2.1.1 Sensitivity analysis of the GaN and SiC switches junction temperature
 19

 2.2 RHODAS digital twin for e-motor 31

 2.3 RHODAS digital twin for Gearbox..... 45

3 CONCLUSION 60

REFERENCES..... 61

Appendix A – SiC-CAB450M12XM3 from Wolfspeed Cree..... 62

Appendix B – GaN-GS66516T from GAN systems..... 71

LIST OF FIGURES

Figure 2.1: Digital Twin Framework for the T-type Inverter, e-Motor, and Gearbox in the RHODAS Project.....	9
Figure 2.2: Inverter Digital Twin Framework: Power Loss Calculation, Equivalent Thermal Circuit Model, and Tuning Scheme.....	10
Figure 2.3: Thermal Equivalent Circuit of Sic-CAB450M12XM3 Designed Based on Datasheet Specifications from Wolfspeed Cree.....	11
Figure 2.4: Thermal Equivalent Circuit of GaN-GS66516T Designed Based on Datasheet Specifications from GAN systems.....	12
Figure 2.5: Thermal equivalent circuit of T-type power converter with GaN and SiC.....	13
Figure 2.6: Simulation procedure verification by re-simulating a published paper [1].....	14
Figure 2.7: Simulation result of re-simulating reference [1].....	15
Figure 2.8: The simulated thermal equivalent circuit of the T-type power converter in MATLAB.....	15
Figure 2.9: MATLAB Simulink model of the GaN thermal equivalent circuits simulating by AU team...	16
Figure 2.10: The layer-by-layer Cauer model as provided in the datasheet for the GS66516T GaN transistor.....	17
Figure 2.11: MATLAB Simulink model of the SiC thermal equivalent circuits simulating by AU team ..	18
Figure 2.12: The configuration of the temperature-dependent thermal resistance.....	19
Figure 2.13: Sensitivity Analysis of the GaN Switches Junction Temperature: Variation in Switching Losses (Switching Frequency).....	20
Figure 2.14: Sensitivity Analysis of the SiC Switches Junction Temperature: Variation in Switching Losses (Switching Frequency).....	20
Figure 2.15: Sensitivity Analysis of the GaN Switches Junction Temperature: Utilizing Various Types of Thermal Interface Materials	21
Figure 2.16: Sensitivity Analysis of the SiC Switches Junction Temperature: Utilizing Various Types of Thermal Interface Materials	21
Figure 2.17: Sensitivity Analysis of the GaN Switches Junction Temperature: Variation in RthHS-A-GaN.....	22
Figure 2.18: Sensitivity Analysis of the SiC Switches Junction Temperature: Variation in RthHS-A-SiC.....	22
Figure 2.19: Sensitivity Analysis of the GaN Switches Junction Temperature: Variation in TA.....	23
Figure 2.20: Sensitivity Analysis of the SiC Switches Junction Temperature: Variation in TA	23
Figure 2.21: Comprehensive Analysis Was Conducted to Assess both The Accuracy and Computational Efficiency of Each Interpolation Method	25
Figure 2.22: The Junction Temperature Results of Both Gan and Sic Transistors	26
Figure 2.23: The Junction Temperature Results of Both Gan and Sic Transistors with Variable Input.	27
Figure 2.24: The Junction Temperature Results to Demonstrate the Individualized Thermal Parameter Handling for each Gan Transistor	28
Figure 2.25: The Power Loss Profile with CBPWM As a Function of m_i and I at $F_s=75f_s = 75f_s=75\text{ kHz}$	30
Figure 2.26: The Design of the main () Function for the High-Power C++ Code.....	30
Figure 2.27: The Design of the main () Function for the Low-Power C++ Code.....	31
Figure 2.28: Simulink Model of the RHODaS Electric Motor Representing Thermal and Physical Behavior.....	32
Figure 2.29: The Effect of Torque, Speed, and Temperatures on Stator Losses.....	32
Figure 2.30: The Effect of Torque, Speed, and Temperatures on Rotor Losses.....	33
Figure 2.31: 3D Surface Visualization of the Influence of Input Parameter Interactions on Stator Losses.....	34
Figure 2.32: 3D Surface Visualization of the Influence of Input Parameter Interactions on Rotor Losses.....	35
Figure 2.33: Thermal Equivalent Circuit Representing Heat Transfer Between Stator, Rotor, and Ambient Environment.....	35
Figure 2.34: Temperature Evolution Over Time in the Thermal Equivalent Circuit Simulation.....	36

Figure 2.35: Simulated Rotor and Stator Temperature Progression Using Runge-Kutta Solution of the Thermal Model.....	37
Figure 2.36: Initialization Section of the rk4_thermal_sim() Function Defining Thermal Parameters in C++	38
Figure 2.37: Implementation of the 4th-Order Runge-Kutta Method for Thermal Simulation in C++.....	39
Figure 2.38: Interpolated Stator and Rotor Power Losses at the Specified Operating Point.....	39
Figure 2.39: Comparison of Power Loss Values from Interpolation and Original Simulink Data.....	40
Figure 2.40: Comparison of Thermal Simulation Results from MATLAB and C++ Implementations.....	40
Figure 2.41: Nonlinear Increase of Stator and Rotor Losses with Torque.....	41
Figure 2.42: Comparison of Stator and Rotor Temperature Responses to Torque.....	42
Figure 2.43: Increase of Stator and Rotor Losses with Speed.....	42
Figure 2.44: Nonlinear Increase of Stator and Rotor Losses with Torque.....	43
Figure 2.45: Effect of Copper Temperature on Stator and Rotor Losses.....	43
Figure 2.46: Stator and Rotor Temperature Response to Initial Copper Temperature.....	44
Figure 2.47: Effect of Rotor Temperature on Rotor and Stator Losses.....	44
Figure 2.48: Nonlinear Increase of Stator and Rotor Losses with Torque.....	45
Figure 2.49: Sample View of the Valeo Dataset Used for Gearbox Temperature Modeling.....	46
Figure 2.50: Simplified Gearbox Thermal Model Based on Worst-Case Temperature Scenario.....	46
Figure 2.51: Simulated Gearbox Surface Temperature Response at 10, 125 W Power Loss.....	47
Figure 2.52: Visualization of Interpolation Parameters and Their Relationship to Total Power Loss....	48
Figure 2.53: 3D Plot of Interpolation Variables Versus Total Power Loss.....	49
Figure 2.54: Temperature Progression Based on Numerical Implementation with Power Loss Input of 10, 125 W.....	52
Figure 2.55: Power Losses and gearbox Temperature Response to Torque Variations.....	57
Figure 2.56: Power Losses and gearbox Temperature Response to internal Temperature Variations	58
Figure 2.57: Power Losses and gearbox Temperature Response to shaft speed Variations.....	58
Figure 2.58: Power Losses and gearbox Temperature Response to ambient Temperature Variations	59
Figure 2.59: Overview of Inputs and Outputs for the Inverter, E-Motor, and Gearbox Digital Twins.....	59

LIST OF TABLES

Table 2.1: Different functionalities of the RHODAS IOTP 46

Table 2.2: Parameters and Conditions for Low Power Loading First Test Case 49

Table 2.3: Parameter Values for Test Case (load_case272) in the Dataset 50

Table 2.4: Parameter Values for Test Case (load_case411) in the Dataset 50

Table 2.5: Parameter Values for Test Case (load_case411) in the Dataset 50

Table 2.6: Parameter Comparison of Adjacent Cases (load_case130 and load_case108)..... 51

Table 2.7: Parameters for High Power Loading Condition (Case 3)..... 51

Table 2.8: Parameter Values of Adjacent Cases (load_case130 and load_case108) for Case 3 Comparison 51

Table 2.9: Data Sets for Power Losses and gearbox Temperature Response to Torque Variations.... 56

Table 2.10: Data Sets for Power Losses and gearbox Temperature Response to internal Temperature Variations 56

Table 2.11: Data Sets for Power Losses and gearbox Temperature Response to shaft speed Variations 56

Table 2.12: Data Sets for Power Losses and gearbox Temperature Response to ambient Temperature Variations 56

EXECUTIVE SUMMARY

This report presents a comprehensive overview of the activities carried out in Work Package 4 (WP4), specifically Task 4.3, of the RHODAS project, which focuses on the design and development of a digital twin framework for key components of electric powertrains.

The developed framework comprises three individual digital twins: the inverter digital twin, the e-motor digital twin, and the gearbox digital twin. The inverter digital twin models the electro-thermal behavior of power electronics modules (SiC and GaN) to support performance monitoring and failure prediction. The e-motor digital twin captures the thermal behavior of the rotor and stator of the RHODAS e-motor, enabling accurate simulation under various driving conditions. The gearbox digital twin is based on a data-driven thermal equivalent circuit model that estimates the gearbox oil temperature using a range of input data.

All digital twin models are implemented in both MATLAB/Simulink and C++ to enable seamless integration into the RHODAS cloud platform, supporting both online and offline thermal monitoring and efficient computation.

The deliverable also includes extensive sensitivity analyses to investigate the behavior of the e-motor, power inverter, and gearbox under different operating modes. This outcome delivers a robust and scalable digital twin architecture that enhances monitoring, diagnostics, and predictive maintenance capabilities within the RHODAS electric vehicle platform.

1 INTRODUCTION

1.1 DESCRIPTION OF THE DOCUMENT AND PURSUE

This document provides a comprehensive overview of the digital twin developments carried out in Task 4.3 of the RHODAS project. The focus is on the creation and integration of digital twin models for three key subsystems within the RHODAS electromechanical powertrain: the T-type inverter, e-Motor, and gearbox. These components are intended to be integrated within a single system, and their thermal behavior and interactions are critical to overall system performance. Accordingly, the digital twin activities have been structured into three dedicated parts, each focusing on one of these subsystems. The first part of the document details the development of the inverter digital twin. This section describes the modeling of thermal behavior based on power loss estimations and the application of thermal equivalent circuits. It also outlines the integration of real-time data into the digital twin platform to enable accurate virtual testing and condition monitoring of the T-type power converter.

The second part presents the e-Motor digital twin. This section focuses on the numerical simulation of the motor's thermal dynamics using mathematical modeling techniques, including a simplified Thermal Equivalent Circuit Model (TECM). The e-Motor model is structured using experimental data and is designed to simulate the thermal behavior of both the stator and rotor parts, capturing the thermal interaction between the different components.

The third part of the document describes the gearbox digital twin, developed based on both simulation results and testbench data provided by the gearbox team. This section includes the methodology for dataset integration, the design of a thermal model using the TECM approach, and the application of numerical methods such as the 4th-order Runge-Kutta method and 3D interpolation. The model is implemented in C++ and supports both offline and online computation, enabling predictive maintenance and system diagnostics. Each of these digital twin models is part of an overarching architecture that captures power loss calculations and thermal behavior, supporting comprehensive system-level analysis. By developing these digital twins, the RHODAS platform aims to bridge the gap between physical hardware and virtual simulation, offering a scalable and modular solution for advanced monitoring, predictive analytics, and system optimization in power electronics and electromechanical systems.

1.2 WPs AND TASKS RELATED WITH THE DELIVERABLE

This deliverable pertains to the activities carried out under Task 4.3, titled Integration of System Models into a Complete Digital Twin, which is part of WP4 (Software Design and Development of Digital Tools) within the RHODAS project.

2 RHODAS Digital Twin Frameworks

The digital twin plays a transformative role in engineering by enabling the creation of virtual replicas of physical systems, which can be used for monitoring, analysis, and optimization. This technology allows engineers to simulate performance, predict failures and test design modifications without the risks and costs associated with physical prototypes. By integrating data from sensors and other sources, digital twins provide valuable insights into system behavior under various conditions, supporting data-driven decision-making throughout the product lifecycle. As a result, they enhance efficiency, reduce downtime and contribute to more sustainable and innovative engineering solutions.

Since one part of the RHODAS project aims to develop a T-type power converter that will be integrated with the e-Motor and a gearbox, the digital twin activities in Task 4.3 have been specifically focused on these three essential components. Recognizing the critical interaction inside these subsystems, three digital twin frameworks have been designed by AU team to accurately represent and simulate their thermal behavior. These frameworks enable detailed modeling, performance analysis and virtual testing of the T-type inverter, the e-motor and the gearbox within a digital environment. As illustrated in Figure 2.1, the digital twin architecture captures both power loss calculations and the thermal equivalent circuit model of these critical parts. In the following sections, the three distinct digital twin models are described in greater detail.

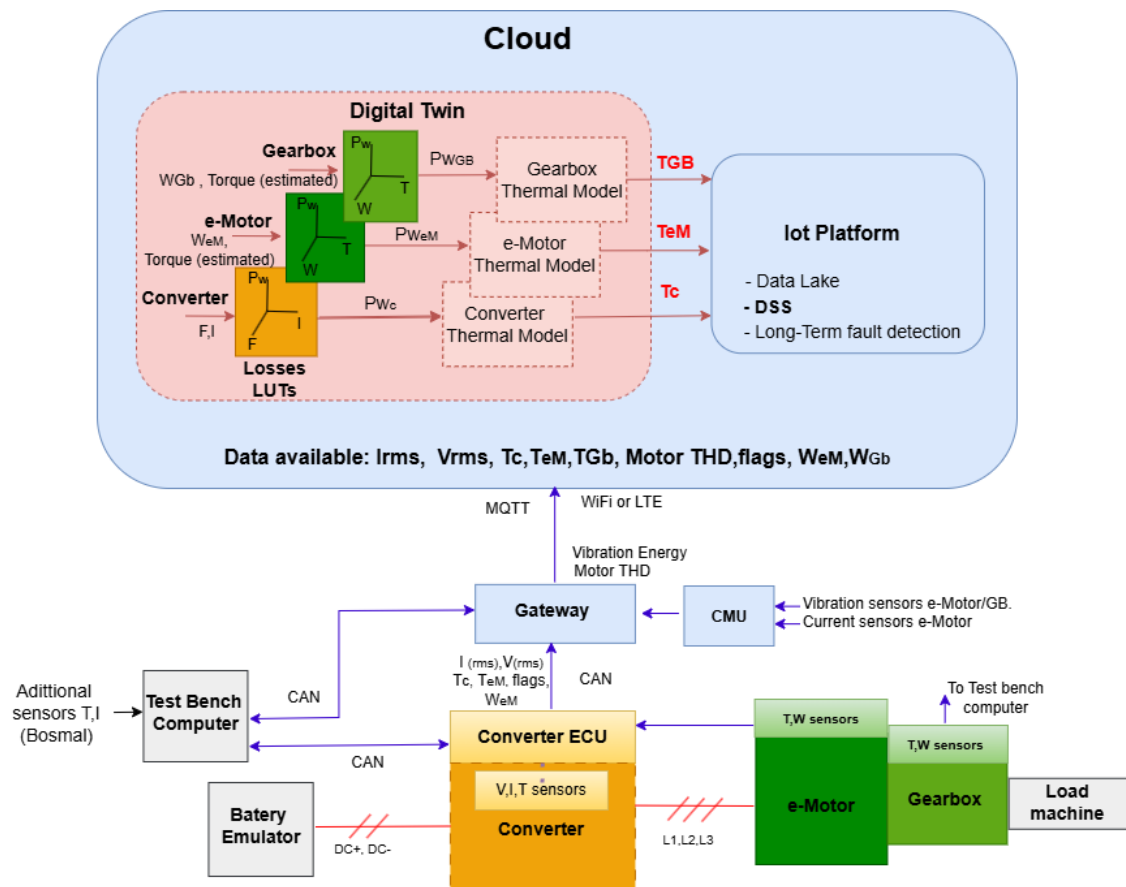


Figure 2.1: Digital Twin Framework for the T-type Inverter, e-Motor, and Gearbox in the RHODAS Project.

2.1 RHODAS INVERTER DIGITAL TWIN

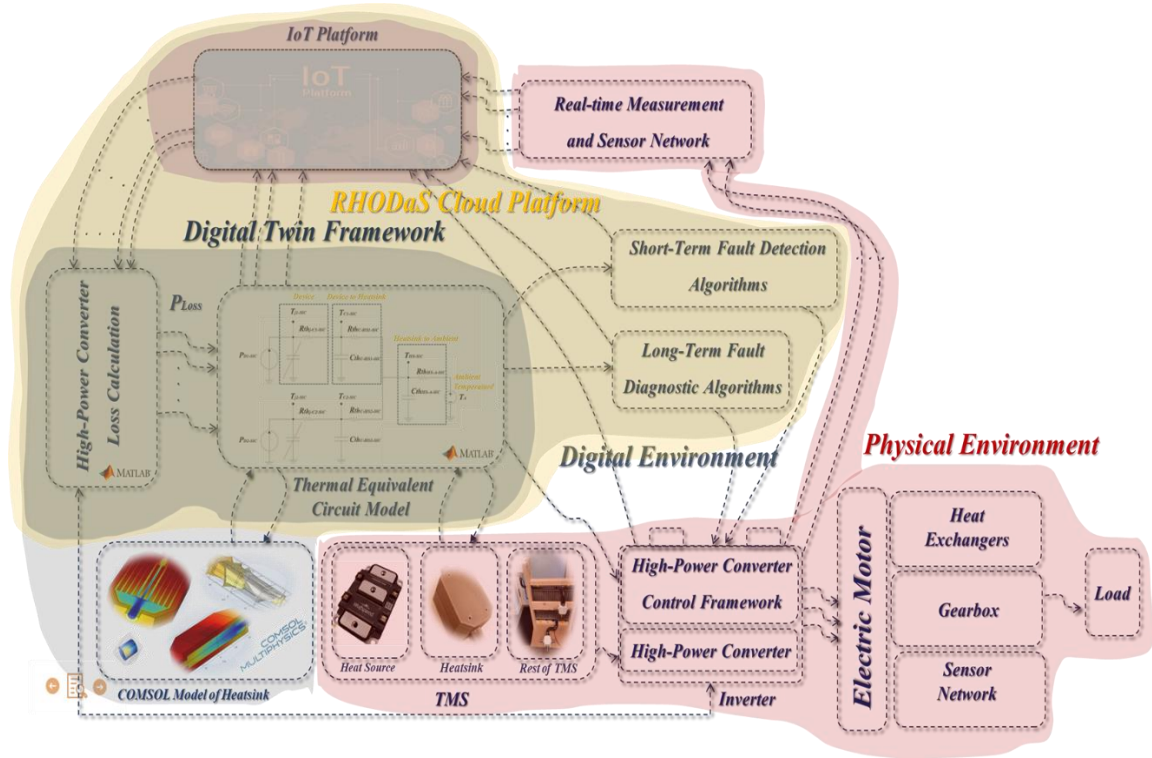


Figure 2.2: Inverter Digital Twin Framework: Power Loss Calculation, Equivalent Thermal Circuit Model, and Tuning Scheme.

In the RHODAS inverter, where wide bandgap (WBG) devices such as GaN and SiC switches are employed, monitoring and accounting for the junction temperature of these high-frequency switches is critically important. In response to this requirement, the AU team has developed a digital twin framework comprising two main components. The first component involves estimating the power losses of the GaN and SiC modules using mathematical interpolation techniques. These estimations are derived from experimental data collected from the RHODAS converter under various operating conditions. By accurately modeling the power losses, this step provides an essential foundation for thermal behavior analysis. The second component integrates the calculated power losses into equivalent circuit models of the GaN and SiC modules. These models are carefully designed based on both manufacturer datasheets and experimental measurements, ensuring a high level of accuracy. Once these two components are connected, the differential equations governing the thermal and electrical behaviour of the system are extracted and implemented into C++ code. This code is then integrated into the RHODAS cloud infrastructure as a component of the overall RHODAS IoT platform. This digital twin framework offers an effective balance between computational complexity and accuracy, making it suitable for both online and offline applications. As illustrated in Figure 2.2, the parameters of the equivalent circuit model can be tuned using high-fidelity simulation tools such as COMSOL Multiphysics, in conjunction with real-world data obtained from the final experimental phase of the RHODAS project.

Therefore, the digital twin architecture of the inverter is structured around two main components:

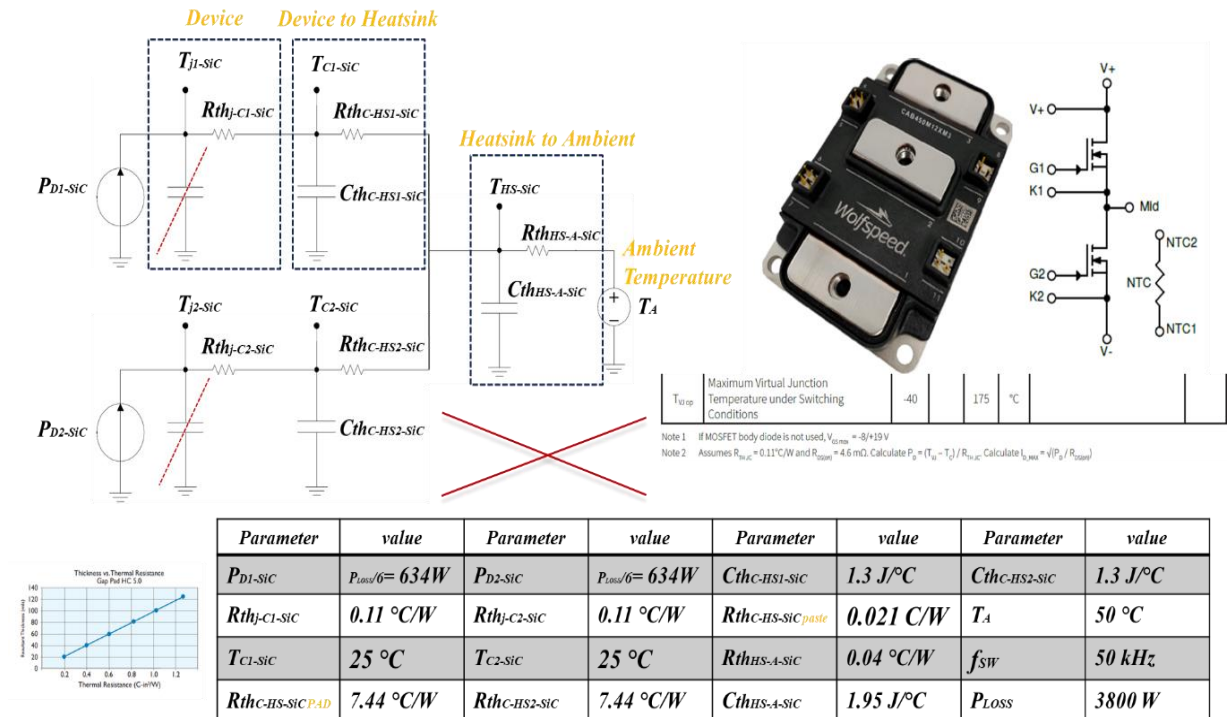
i) Power Loss Estimation via Interpolation-Based Modelling

The first stage of the framework focuses on calculating the power losses in the GaN and SiC modules. This is accomplished using mathematical interpolation techniques based on experimental data collected from the real operation of the RHODAS converter. Specifically, a 3D cubic spline interpolation algorithm has been implemented in C++ to estimate power loss as a function of three key parameters: current (I), modulation index (mi), and switching frequency (fs). This approach allows for accurate and continuous loss estimation across a wide range of operating conditions without the need to precompute extensive lookup tables. For example, for the low-power loading of converter employing Carrier-Based PWM (CBPWM), the interpolation model was validated against simulation results provided by UPC, demonstrating a high level of agreement.

ii) Thermal Modeling through Equivalent Circuit Representation

In the second stage, the estimated power losses are fed into a detailed thermal model of the inverter, implemented using lumped parameter Cauer-type equivalent circuits. These circuits were designed specifically for both GaN and SiC modules, incorporating data derived from component datasheets, COMSOL Multiphysics simulations, and practical experimental validation.

In the process of designing the equivalent circuit models for the SiC and GaN modules, as depicted in Figures 2.3 and 2.4, all components, including thermal resistances, thermal capacitances and other parameters, were calculated based on the datasheet specifications. These datasheets are the result of extensive experimental characterization and in-depth investigations conducted by the module manufacturers. As such, the parameters provided reliably reflect the real-world thermal behaviour of the modules under practical operating conditions. The datasheets for the SiC and GaN modules are included in Appendices A and B, respectively.



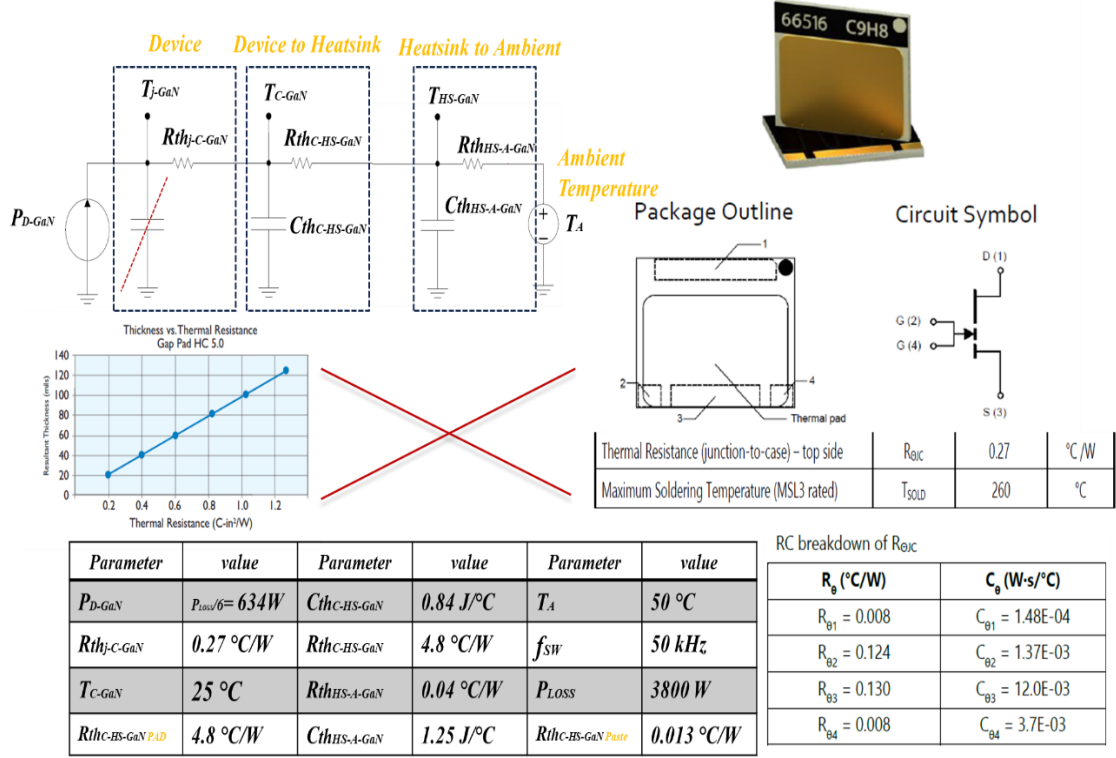


Figure 2.4: Thermal Equivalent Circuit of GaN-GS66516T Designed Based on Datasheet Specifications from GAN systems

Figure 2.5 presents the finalized thermal equivalent circuit model for the RHODaS high-power inverter. This comprehensive model was constructed by first developing individual thermal models for each system component and then integrating them based on the actual topology of the inverter to accurately reflect the overall thermal dynamics of the system.

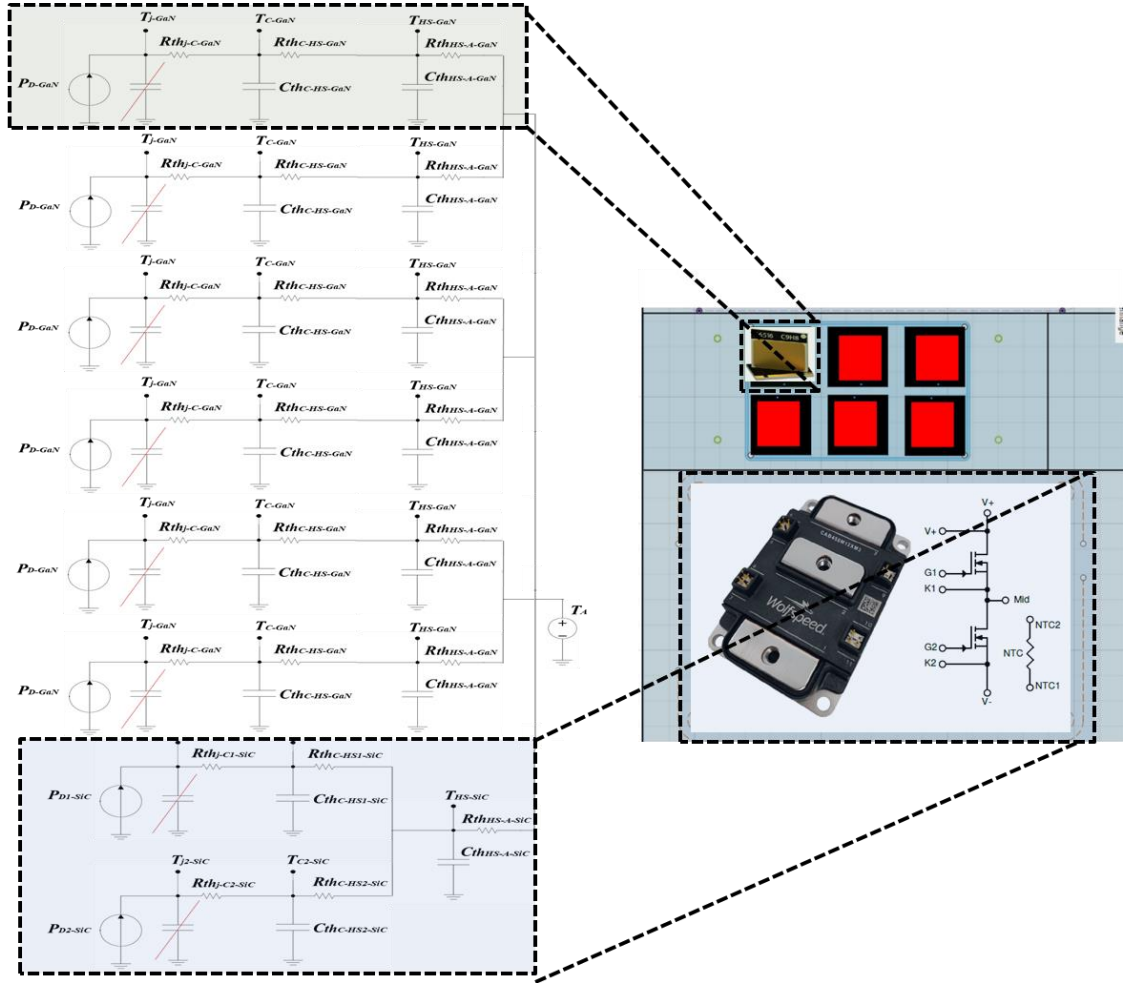


Figure 2.5: Thermal equivalent circuit of T-type power converter with GaN and SiC.

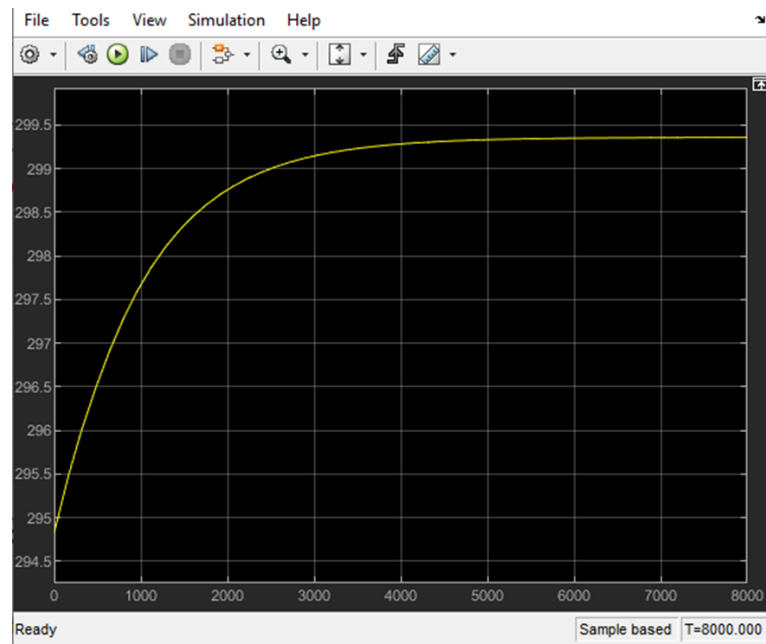
To validate the accuracy of our simulation methodology, the AU-team re-simulated a previously published study prior to developing the thermal equivalent circuit model for the supercapacitor system. The reference publication used for this verification is shown in Figure 2.6. A comparison between our simulation results and those reported in the study is illustrated in Figure 2.7, clearly demonstrating a high level of similarity and confirming the accuracy of our simulation framework.

Following this successful validation, we proceeded to simulate the thermal equivalent circuit model of the T-type power converter utilizing GaN and SiC technologies, as depicted in Figure 2.8.

^a Institut FEMTO-ST CNRS UMR 6174; Université de Franche-Comté, FCLAB bat F, 90010 Belfort, France
^b Vrije Universiteit Brussel, Pleinlaan 2, B-1050 Brussels, Belgium

Parameters	Provided by Maxwell	Measured
Operating temperature range	-40 °C, +65 °C	-
Equivalent series resistance	ESR = 0.47 (mΩ)	ESR = 0.39 (mΩ)
Thermal resistance	R _{th} = 4.5 (°C/W)	R _{th} = 4.2 (°C/W)
Thermal capacitance	-	C _{th} = 268 (J/°C)

a) Simulation result of reference [2]



b) Simulation result of AU team simulations in MATLAB (Temperature [°C] vs Time [s])

Figure 2.7: Simulation result of re-simulating reference [1]

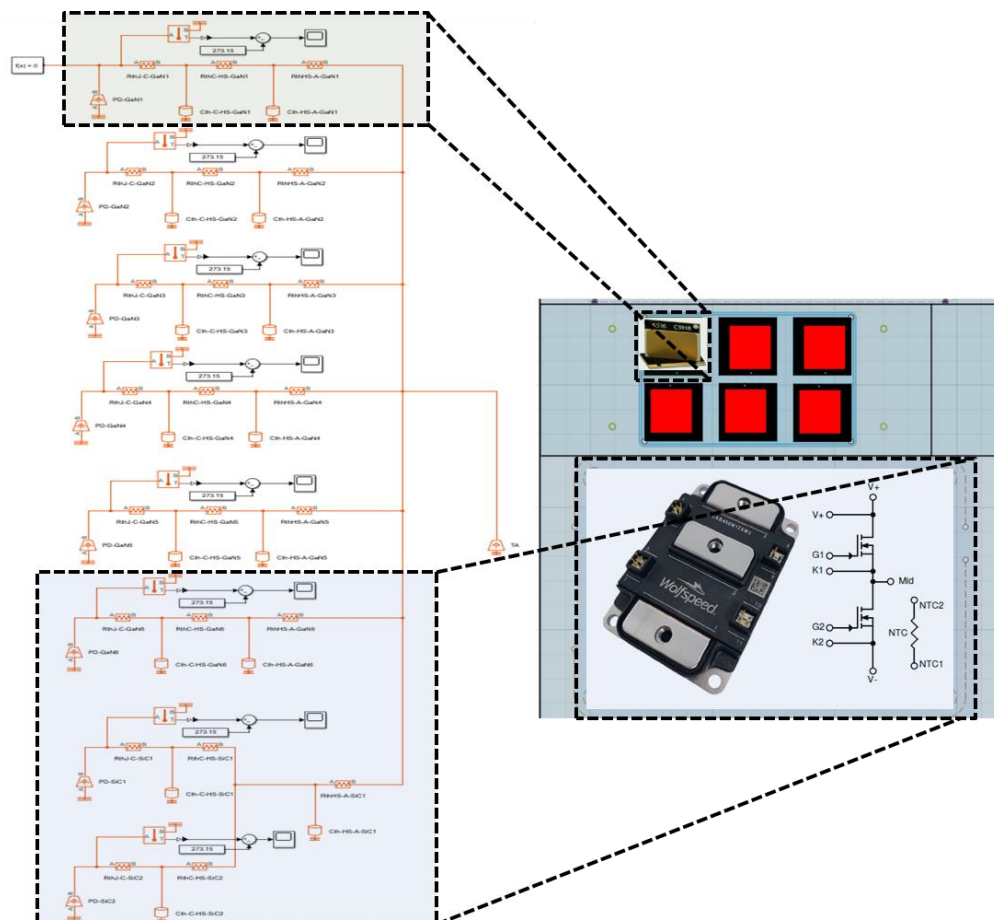


Figure 2.8: The simulated thermal equivalent circuit of the T-type power converter in MATLAB

For the thermal equivalent circuit diagram of the GaN transistor, it is assumed that thermal spreading is negligible, implying the absence of significant thermal coupling between adjacent transistors. As a result, each GaN transistor can be treated independently, allowing for the formulation of an individual equivalent thermal circuit for each device. This assumption facilitates the derivation of differential equations governing the temperatures at each layer of the structure.

It is important to note that the model under consideration is a lumped parameter Cauer model. Consequently, the differential equations do not incorporate spatial dimensions, and temperature estimations are limited to discrete thermal nodes. These nodes correspond to the interfaces between the heat sink and the thermal interface material (TIM), the TIM and the case, and, notably, the junction—regarded as the location of the heat source.

Presented in Figure 2.9, and Figure 2.10 are the MATLAB Simulink models of the GaN and SiC thermal equivalent circuits simulating by AU team:

For the GaN:

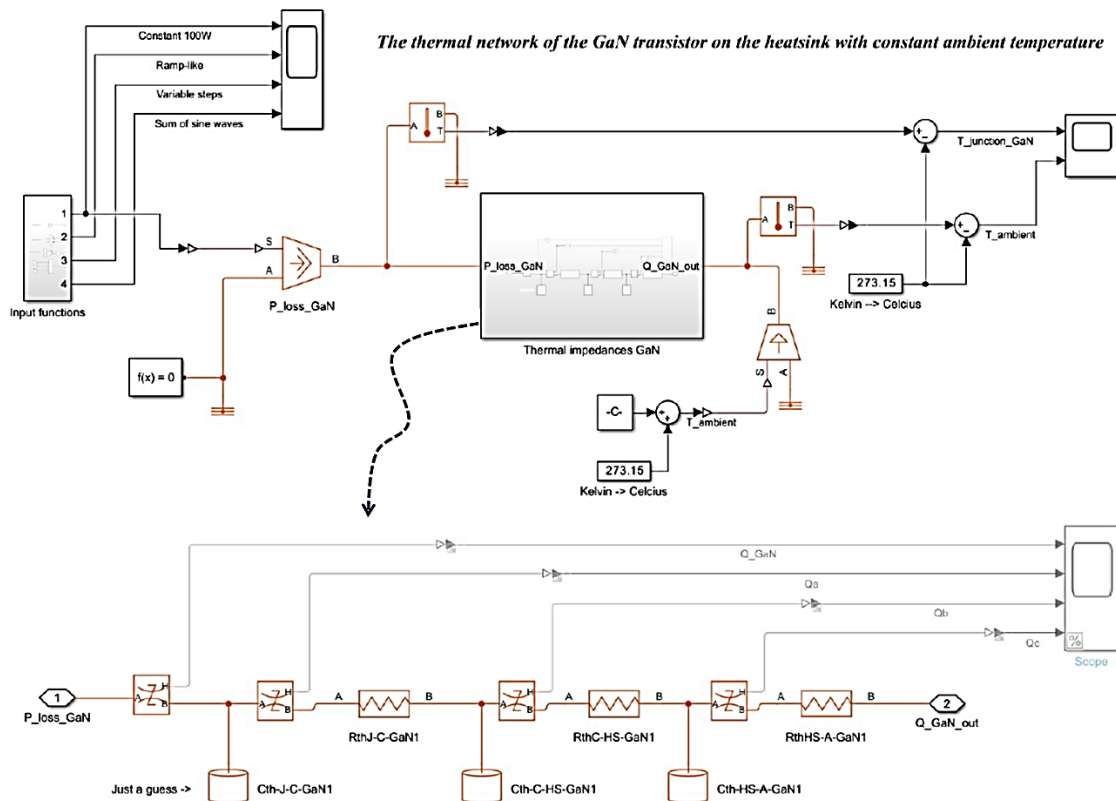


Figure 2.9: MATLAB Simulink model of the GaN thermal equivalent circuits simulating by AU team

The Simscape implementation of the thermal equivalent circuit utilizes two primary component types: thermal resistance and thermal mass. For instance, the label "Rth-J-C-GaN1" denotes the thermal resistance from junction to case for a single GaN transistor.

The employed Cauer model is relatively simple. Due to the lack of detailed information regarding the internal packaging structure of the GaN device, the model considers the overall junction-to-case thermal resistance, which is specified as 0.27 K/W. This simplification is necessary in the absence of more granular packaging data. However, reference documentation for a comparable GaN device—specifically the GS66516T from GaN Systems—has been obtained. The datasheet for this component includes a detailed Cauer model for the junction-to-case thermal path, represented by four discrete layers, each characterized by individual thermal resistance and capacitance parameters.

Below, in Figure 2.10, is an image illustrating the layer-by-layer Cauer model as provided in the datasheet for the GS66516T GaN transistor:

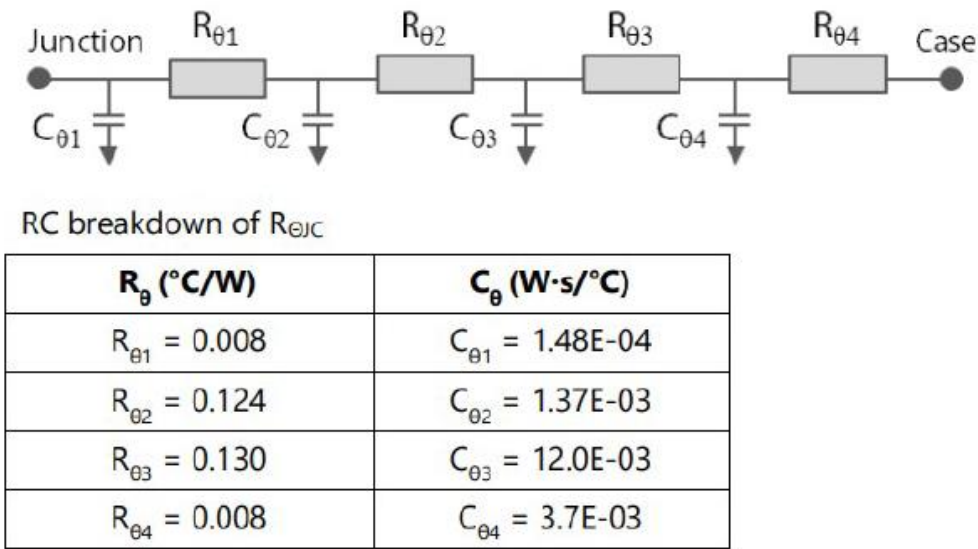


Figure 2.10: The layer-by-layer Cauer model as provided in the datasheet for the GS66516T GaN transistor

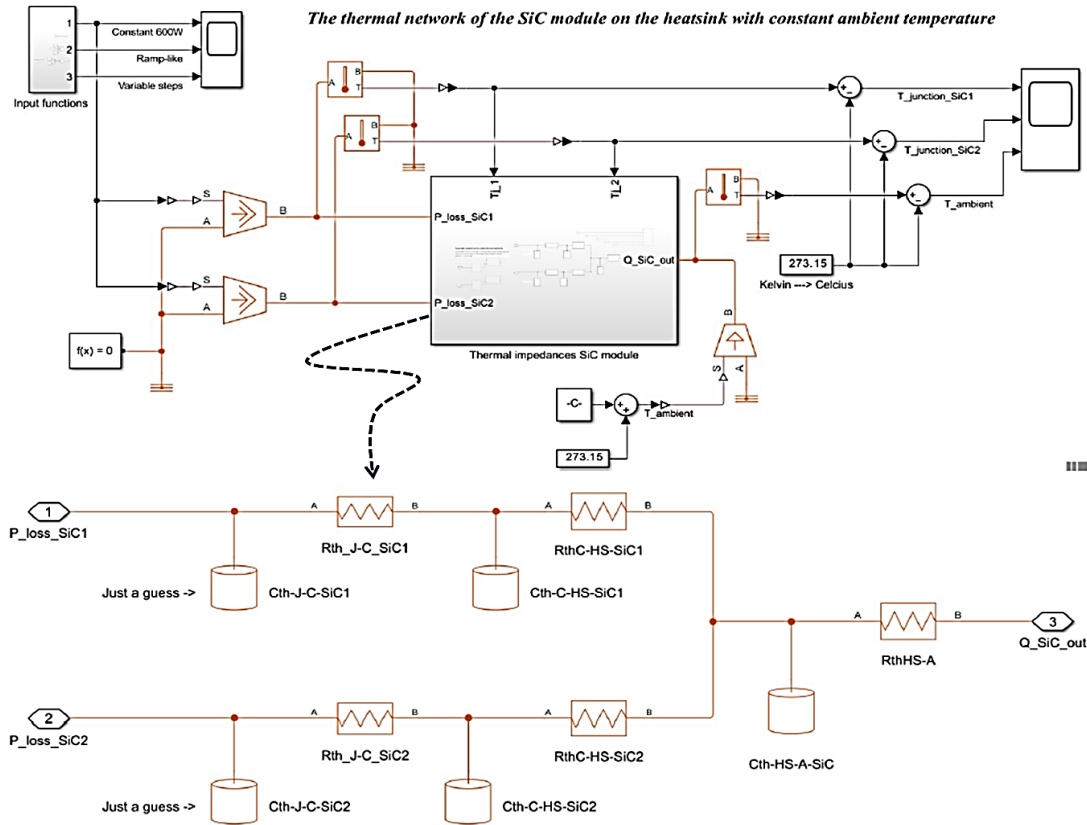


Figure 2.11: MATLAB Simulink model of the SiC thermal equivalent circuits simulating by AU team

The underlying modeling approach remains consistent with that of the GaN device. However, since the SiC power module (CAB450M12XM3 from Wolfspeed) integrates two SiC transistors, these have been configured to share a common heatsink-to-ambient thermal resistance within the thermal equivalent circuit.

In contrast to the GaN datasheet, the datasheet for the SiC module does not provide details regarding the internal layer structure of the packaging. Consequently, further investigation into the physical composition of the module has not been pursued. Nevertheless, a relevant study published by Aalborg University presents a detailed Cauer model for a conventional SiC power module, offering significant insights. Notably, the study includes the modeling of temperature-dependent thermal parameters across the multilayer structure. Inspired by this, the current thermal model incorporates a mechanism to account for temperature dependency in the thermal resistance. Note that, the label "Rth-J-C-GaN1" represents the thermal resistance from the junction to the case of a single GaN transistor. Similarly, "RthC-HS-GaN1" denotes the thermal resistance from the case to the heatsink, and "RthHS-A-GaN1" refers to the thermal resistance from the heatsink to the ambient environment. In parallel with each thermal resistance, there are corresponding thermal capacitances—"CthJ-C-GaN1", "CthC-HS-GaN1", and "CthHS-A-GaN1".

A fitted function representing the thermal conductivity of the SiC die has been implemented based on data extracted from the article in reference 2 titled "Thermal Characterization of SiC MOSFET Module Suitable for High-Temperature Computationally-Efficient Thermal-Profile Prediction". The function was derived using MATLAB's Curve Fitter Toolbox. In Figure 2.12, an image is provided illustrating the configuration of the temperature-dependent thermal resistance as implemented in the model.

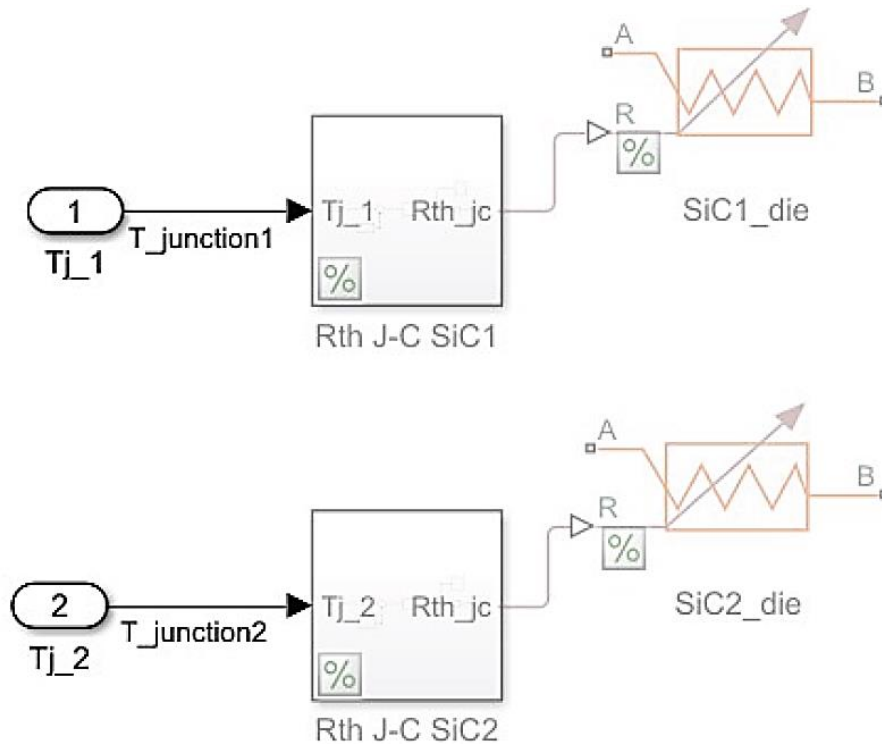


Figure 2.12: The configuration of the temperature-dependent thermal resistance

The implemented Simscape component for modeling temperature-dependent behavior is a variable thermal resistance. The corresponding Simulink block defines this resistance as a function of the junction temperature, enabling dynamic adjustment of thermal resistance in response to temperature changes.

2.1.1 SENSITIVITY ANALYSIS OF THE GAN AND SiC SWITCHES JUNCTION TEMPERATURE

As part of Task T4.3, a comprehensive sensitivity analysis was performed focusing on the junction temperature of both GaN and SiC switches. The objective of this analysis was to delineate the operational boundaries of the thermal management system and to evaluate the influence of various thermal system parameters on the junction temperature, a critical performance and reliability indicator for power semiconductor devices.

Through systematic variation of selected parameters, the sensitivity analysis provides insight into how changes in thermal resistances, heat capacities, ambient conditions, and power dissipation impact the thermal response of the system. This approach enables identification of the most influential parameters, thereby supporting design optimization and risk assessment in thermal design.

The consolidated results of the sensitivity analysis, reflecting the thermal response under different parametric variations, are presented in Figures 2.13–2.20. These figures collectively demonstrate the thermal system's robustness and highlight the key factors that govern the thermal behavior of the GaN and SiC devices under realistic operating conditions.

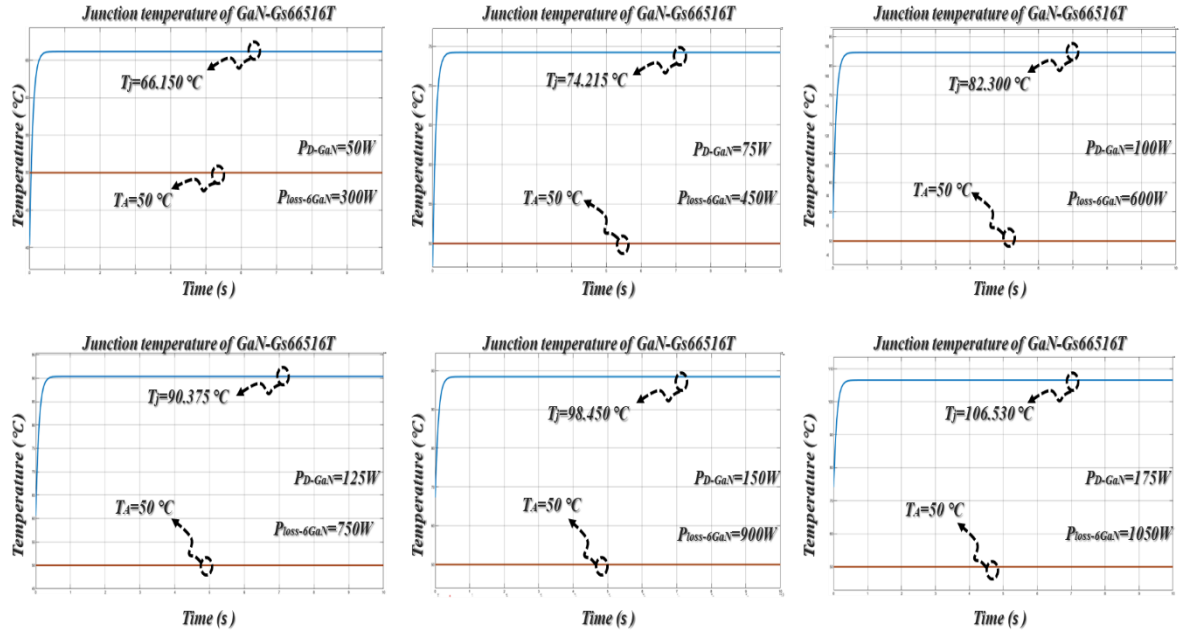


Figure 2.13: Sensitivity Analysis of the GaN Switches Junction Temperature: Variation in Switching Losses (Switching Frequency)

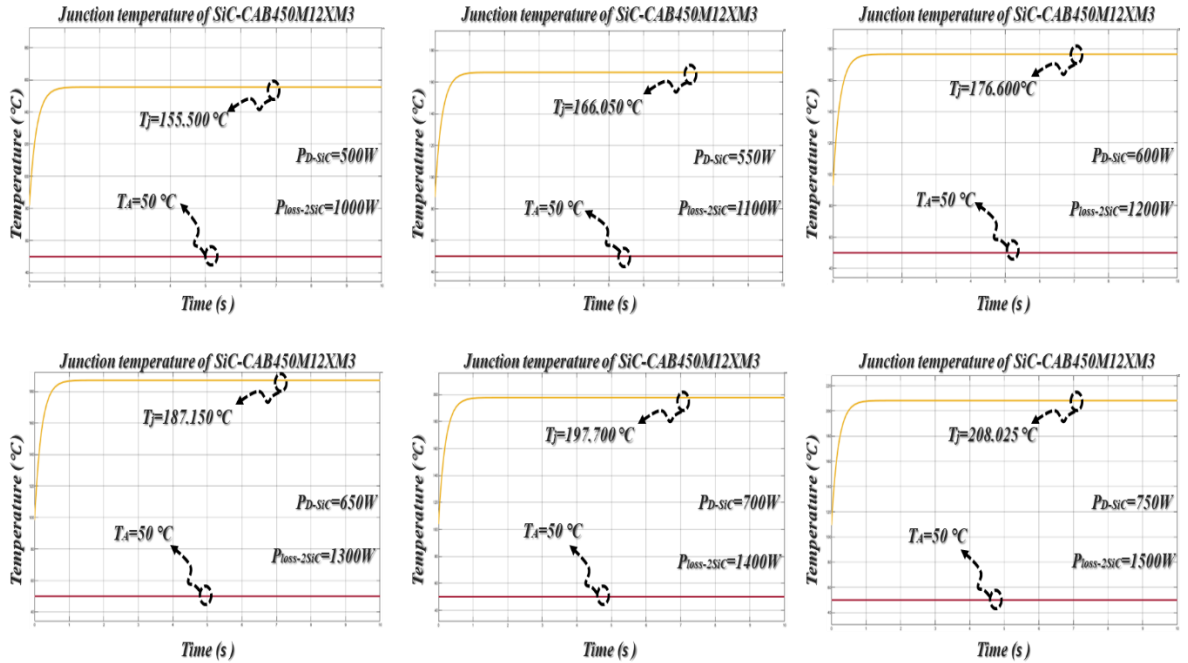


Figure 2.14: Sensitivity Analysis of the SiC Switches Junction Temperature: Variation in Switching Losses (Switching Frequency)

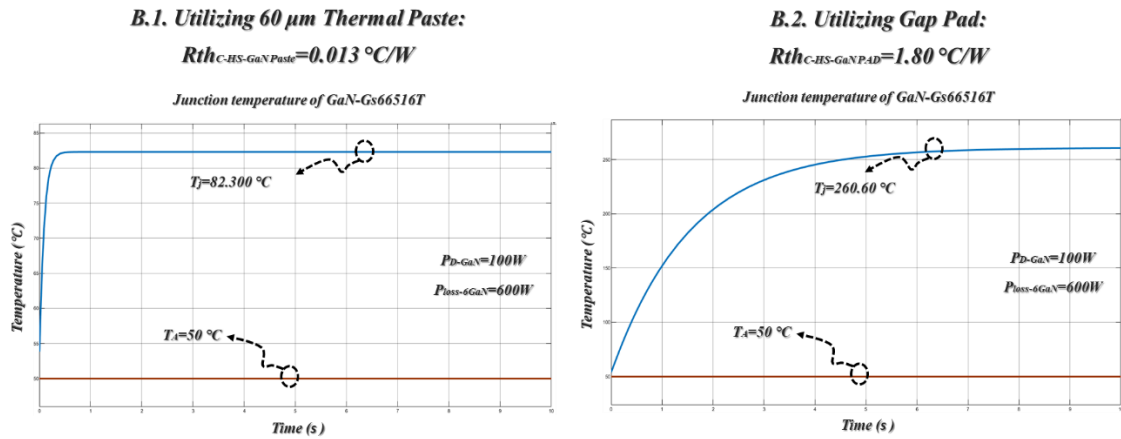


Figure 2.15: Sensitivity Analysis of the GaN Switches Junction Temperature: Utilizing Various Types of Thermal Interface Materials

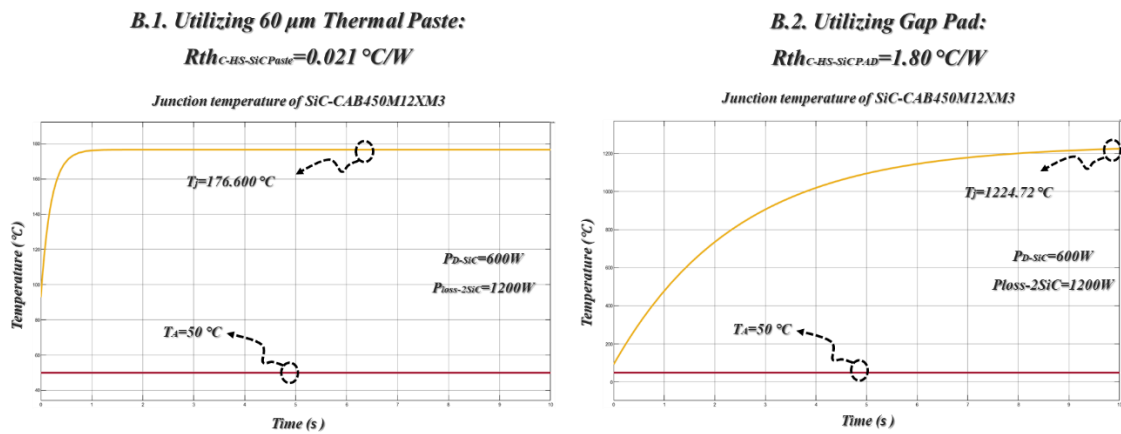


Figure 2.16: Sensitivity Analysis of the SiC Switches Junction Temperature: Utilizing Various Types of Thermal Interface Materials

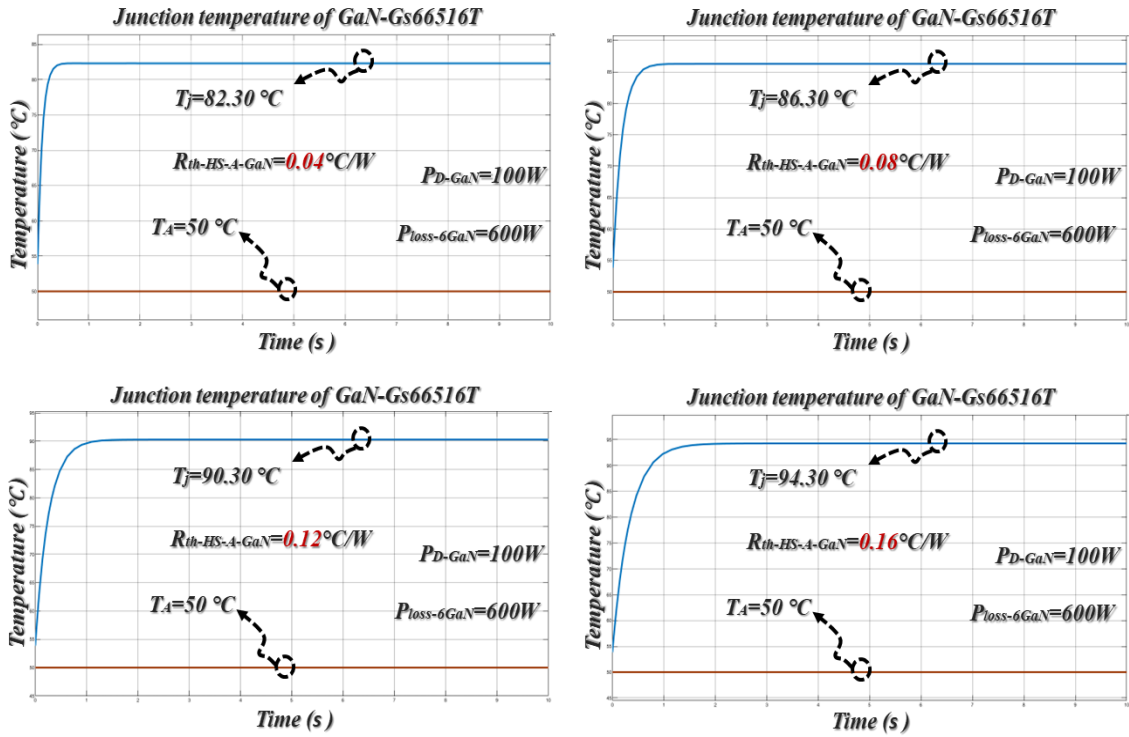


Figure 2.17: Sensitivity Analysis of the GaN Switches Junction Temperature: Variation in $R_{thHS-A-GaN}$

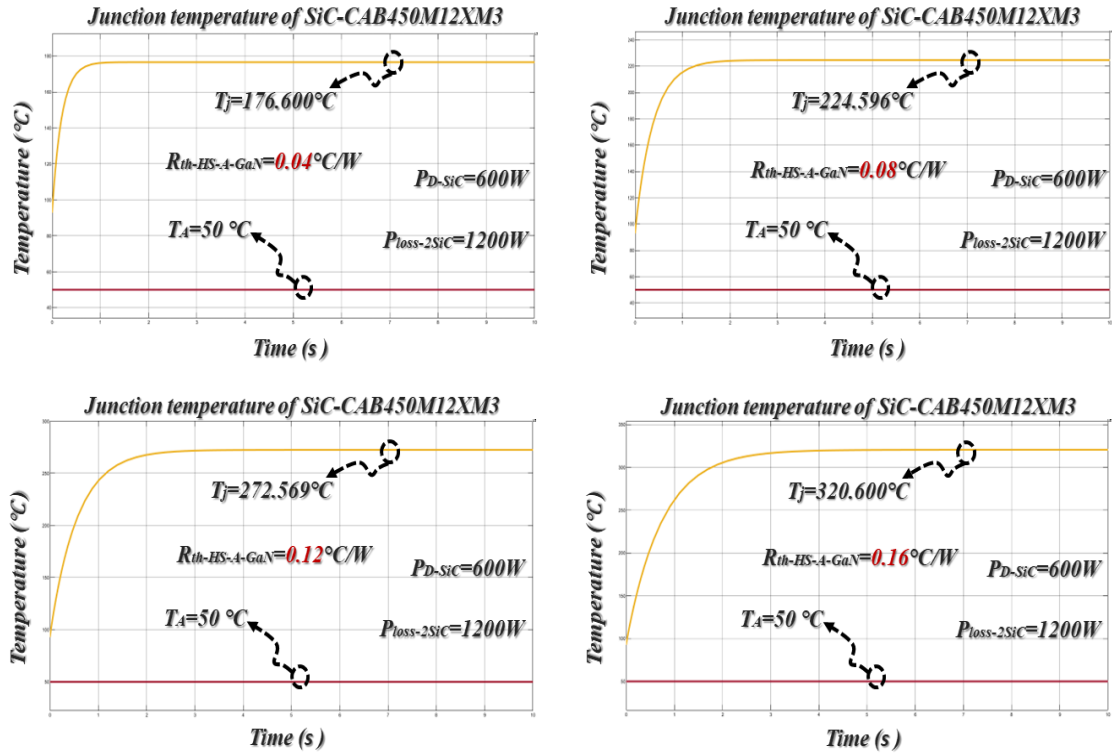


Figure 2.18: Sensitivity Analysis of the SiC Switches Junction Temperature: Variation in $R_{thHS-A-SiC}$

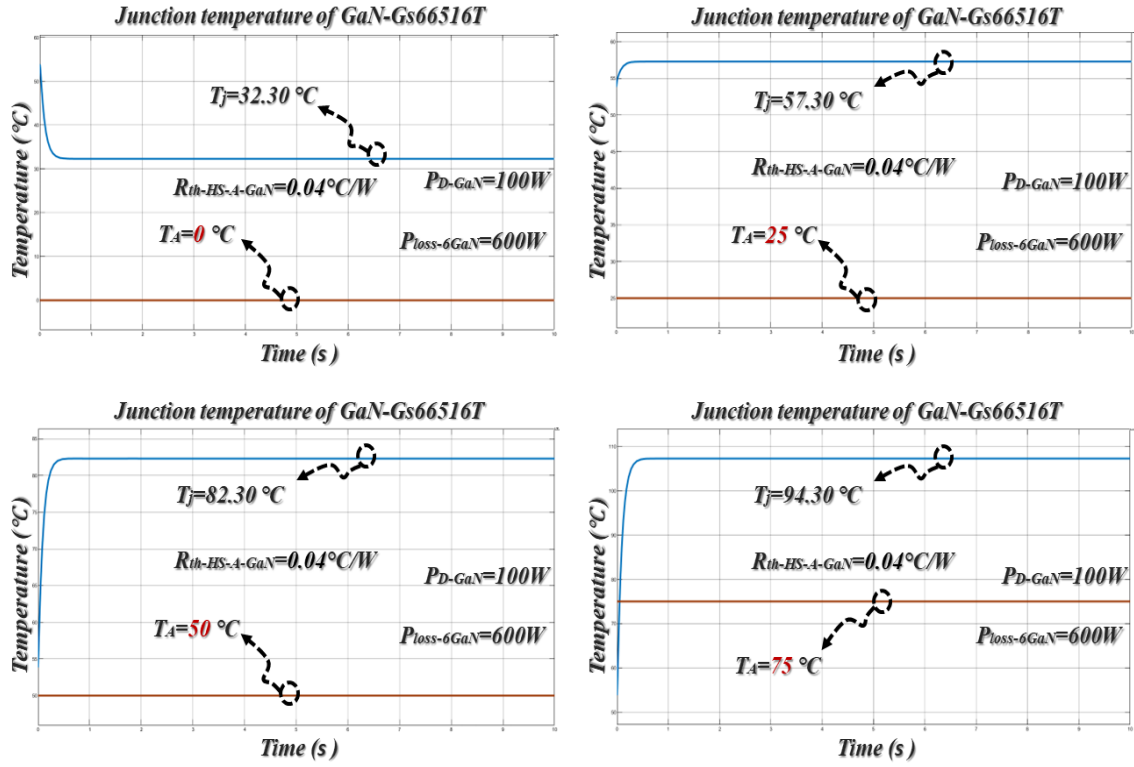


Figure 2.19: Sensitivity Analysis of the GaN Switches Junction Temperature: Variation in T_A

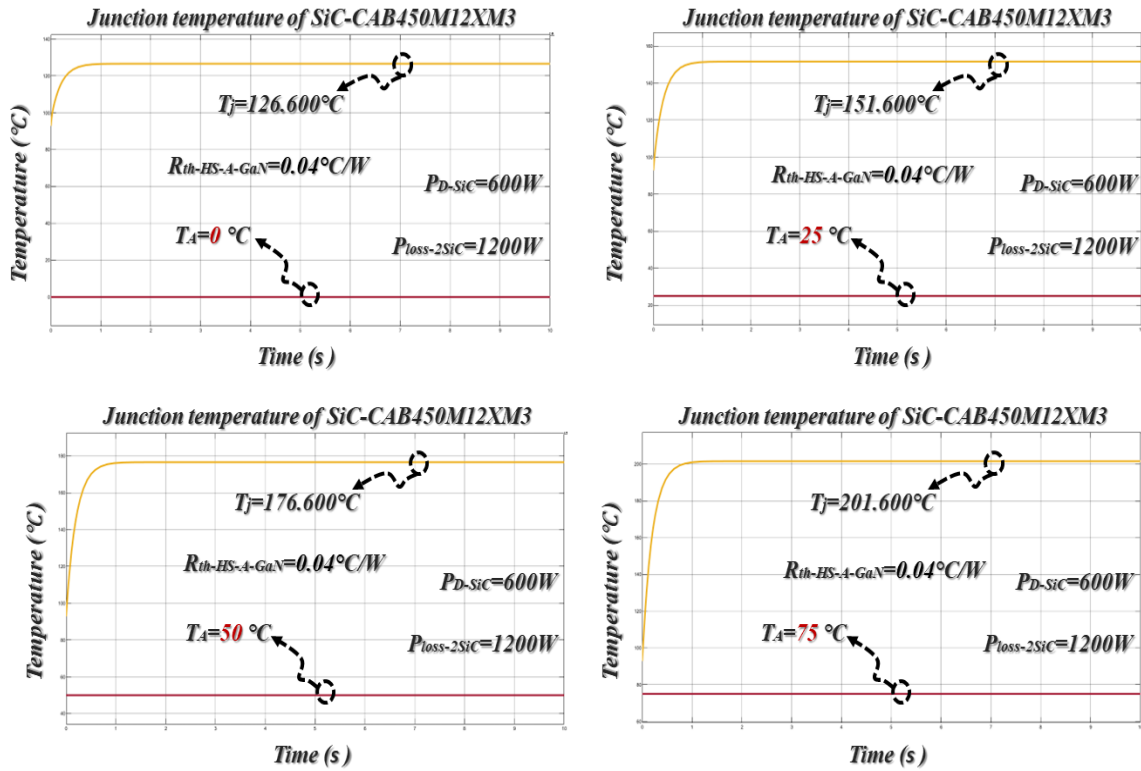


Figure 2.20: Sensitivity Analysis of the SiC Switches Junction Temperature: Variation in T_A

With thermal equivalent circuit diagrams established for both the GaN and SiC devices, the next step involves expressing these models through differential equations. This mathematical representation is essential for developing a C++ implementation capable of solving the thermal profiles numerically over time.

The derivation of these differential equations is documented in T4.3. The method is grounded in the electrical analogy between thermal and electrical domains, wherein:

- Temperature is analogous to voltage
- Heat flow corresponds to current
- Thermal resistance parallels electrical resistance
- Thermal capacitance is equivalent to electrical capacitance

This analogy allows the thermal behavior of the system to be described using equations structurally similar to those in RC electrical circuits. Each node in the network contributes a first-order differential equation, and the full system can be represented as a set of coupled ordinary differential equations (ODEs), describing the time evolution of temperature across the different layers and interfaces.

These equations serve as the basis for the numerical solver to be implemented in C++, where methods such as Euler or Runge-Kutta integration schemes may be used to simulate transient thermal behavior under various operating conditions.

As previously illustrated, the calculation of power losses in the GaN and SiC modules is based on real measurement data. To enable accurate estimation across a range of operating conditions, an interpolation technique is employed.

To identify the most suitable interpolation method, a variety of interpolation techniques were simulated and evaluated. As depicted in Figure 2.21, a comprehensive analysis was conducted to assess both the accuracy and computational efficiency of each method. This evaluation is essential to ensure that the selected technique offers a balanced trade-off between numerical precision and processing speed, particularly for real-time or iterative simulation environments. The outcome of this analysis informs the selection of the interpolation approach used in the current thermal model, contributing to more reliable and computationally efficient estimation of switching and conduction losses in the GaN and SiC modules.

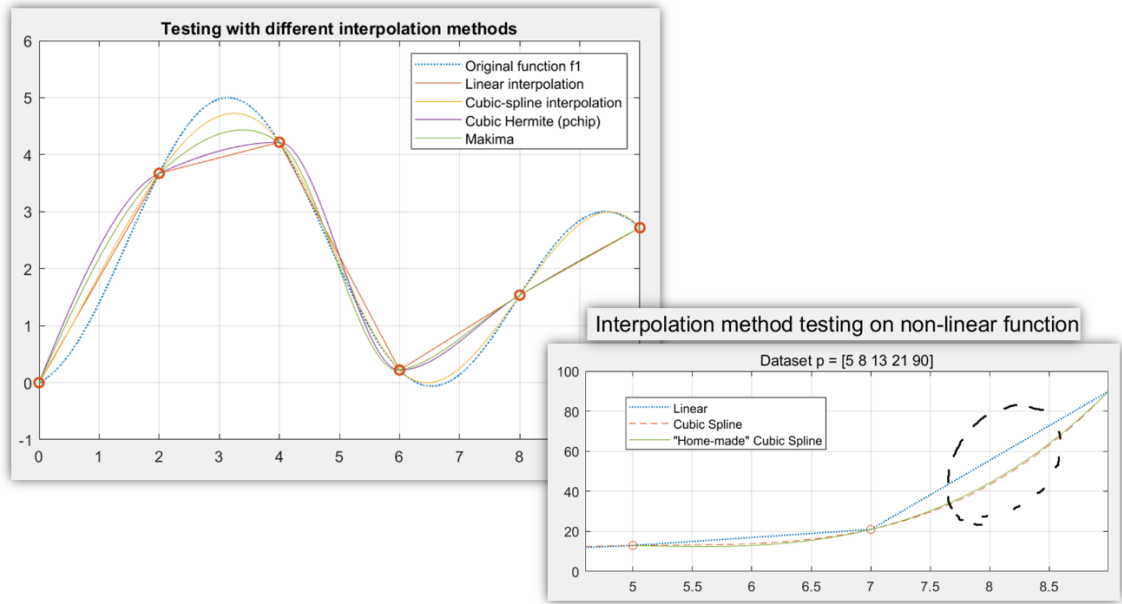


Figure 2.21: Comprehensive Analysis Was Conducted to Assess both The Accuracy and Computational Efficiency of Each Interpolation Method

By using Kirchhoff's Current Law, we obtain this system of linear differential equations for the GaN and SiC:

For the GaN part we can obtain:

$$\begin{bmatrix} \frac{dT_a}{dt} \\ \frac{dT_b}{dt} \\ \frac{dT_c}{dt} \end{bmatrix} = \begin{bmatrix} \frac{-1}{R_1 \cdot C_1} & \frac{1}{R_1 \cdot C_1} & 0 \\ \frac{1}{R_1 \cdot C_2} - \left(\frac{1}{R_1 \cdot C_2} + \frac{1}{R_2 \cdot C_2} \right) & \frac{1}{R_2 \cdot C_2} & 0 \\ 0 & \frac{1}{R_2 \cdot C_3} - \left(\frac{1}{R_2 \cdot C_3} + \frac{1}{R_3 \cdot C_3} \right) & \frac{1}{R_3 \cdot C_3} \end{bmatrix} \cdot \begin{bmatrix} T_a \\ T_b \\ T_c \end{bmatrix} + \begin{bmatrix} \frac{1}{C_1} \cdot Q_{in} \\ 0 \\ \frac{1}{R_3 \cdot C_3} \cdot T_{amb} \end{bmatrix}$$

For the SiC part we can obtain:

$$\begin{bmatrix} \frac{dT_{a1}}{dt} \\ \frac{dT_{b1}}{dt} \\ \frac{dT_{a2}}{dt} \\ \frac{dT_{b2}}{dt} \\ \frac{dT_c}{dt} \end{bmatrix} = \begin{bmatrix} \frac{1}{R_a \cdot C_a} & \frac{1}{R_a \cdot C_a} & 0 & 0 & 0 \\ \frac{1}{R_a \cdot C_b} - \left(\frac{1}{R_a \cdot C_b} + \frac{1}{R_b \cdot C_b} \right) & 0 & 0 & \frac{1}{R_b \cdot C_b} & 0 \\ 0 & 0 & \frac{1}{R_a \cdot C_a} & \frac{1}{R_a \cdot C_a} & 0 \\ 0 & 0 & \frac{1}{R_a \cdot C_b} - \left(\frac{1}{R_a \cdot C_b} + \frac{1}{R_b \cdot C_b} \right) & \frac{1}{R_b \cdot C_b} & 0 \\ 0 & \frac{1}{R_b \cdot C_c} & 0 & \frac{1}{R_b \cdot C_c} - \left(\frac{2}{R_b \cdot C_c} + \frac{1}{R_c \cdot C_c} \right) & \frac{1}{R_c \cdot C_c} \end{bmatrix} \cdot \begin{bmatrix} T_{a1} \\ T_{b1} \\ T_{a2} \\ T_{b2} \\ T_c \end{bmatrix} + \begin{bmatrix} \frac{1}{C_a} \cdot Q_{in1} \\ 0 \\ \frac{1}{C_a} \cdot Q_{in2} \\ 0 \\ \frac{1}{R_c \cdot C_c} \cdot T_{amb} \end{bmatrix}$$

The obtained differential equations are solved by employing the Runge-Kutta method of order 4. The following section of the program addresses the solution of the system of differential equations corresponding to the GaN model:

```

54 // //step size for approximation
55 // float h = 0.001;
56
57 //Approximation for T_a
58
59 float a1 = -k1*Ta_GaN + k1*Tb_GaN + k6*p_loss_transistor_GaN;
60 float b1 = -k1*(Ta_GaN + (h*0.5)*a1) + k1*(Tb_GaN + (h*0.5)*a1) + k6*p_loss_transistor_GaN;
61 float c1 = -k1*(Ta_GaN + (h*0.5)*b1) + k1*(Tb_GaN + (h*0.5)*b1) + k6*p_loss_transistor_GaN;
62 float d1 = -k1*(Ta_GaN + h*c1) + k1*(Tb_GaN + h*c1) + k6*p_loss_transistor_GaN;
63
64 Ta_GaN = Ta_GaN + (h/6.0)*(a1+2*b1+2*c1+d1);
65
66
67 //Approximation for T_b
68
69 float a2 = k2*Ta_GaN - (k2+k3)*Tb_GaN + k3*Tc_GaN;
70 float b2 = k2*(Ta_GaN + (h*0.5)*a2) - (k2+k3)*(Tb_GaN + (h*0.5)*a2) + k3*(Tc_GaN + (h*0.5)*a2);
71 float c2 = k2*(Ta_GaN + (h*0.5)*b2) - (k2+k3)*(Tb_GaN + (h*0.5)*b2) + k3*(Tc_GaN + (h*0.5)*b2);
72 float d2 = k2*(Ta_GaN + h*c2) - (k2+k3)*(Tb_GaN + h*c2) + k3*(Tc_GaN + h*c2);
73
74 Tb_GaN = Tb_GaN + (h/6.0)*(a2+2*b2+2*c2+d2);
75
76
77 //Approximation for T_c
78
79 float a3 = k4*Tb_GaN - (k4+k5)*Tc_GaN + k5*T_amb;
80 float b3 = k4*(Tb_GaN + (h*0.5)*a3) - (k4+k5)*(Tc_GaN + (h*0.5)*a3) + k5*T_amb;
81 float c3 = k4*(Tb_GaN + (h*0.5)*b3) - (k4+k5)*(Tc_GaN + (h*0.5)*b3) + k5*T_amb;
82 float d3 = k4*(Tb_GaN + h*c3) - (k4+k5)*(Tc_GaN + h*c3) + k5*T_amb;
83
84 Tc_GaN = Tc_GaN + (h/6.0)*(a3+2*b3+2*c3+d3);
85

```

Lines 59–62 of the program implement the iterative numerical solution of the first differential equation, expressed as:

$$\frac{dT_a}{dt} = \frac{-1}{R_1 \cdot C_1} \cdot T_a + \frac{1}{R_1 \cdot C_1} \cdot T_b + \frac{1}{C_1} \cdot Q_{in}$$

Lines 69–72 correspond to the numerical solution of the second differential equation, while lines 79–82 address the third. The Runge-Kutta method of order 4 is employed for solving these equations. An identical approach is used for the SiC model, which comprises five differential equations rather than three.

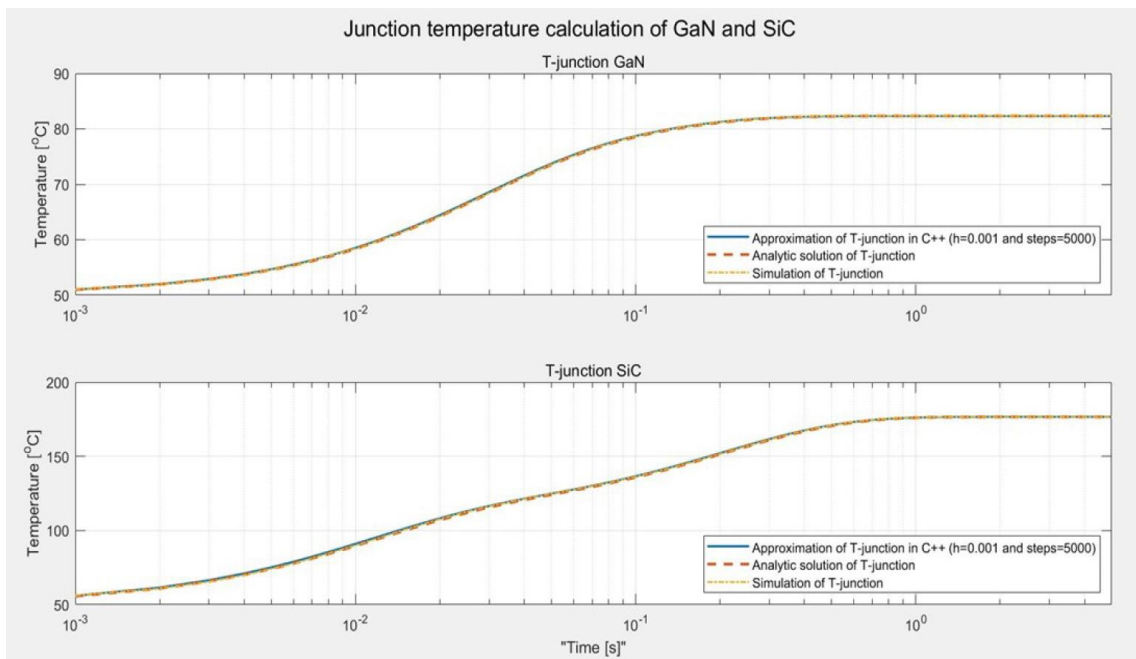


Figure 2.22: The Junction Temperature Results of Both Gan and Sic Transistors

As shown in figure 2.22 the junction temperature of both GaN and SiC transistors as a function of a 100 W power loss in the GaN and 600 W in the SiC. The resulting temperature profiles are compared to MATLAB Simulink simulations and analytical solutions. The impact of applying a step-function input is also demonstrated in figure 2.23.

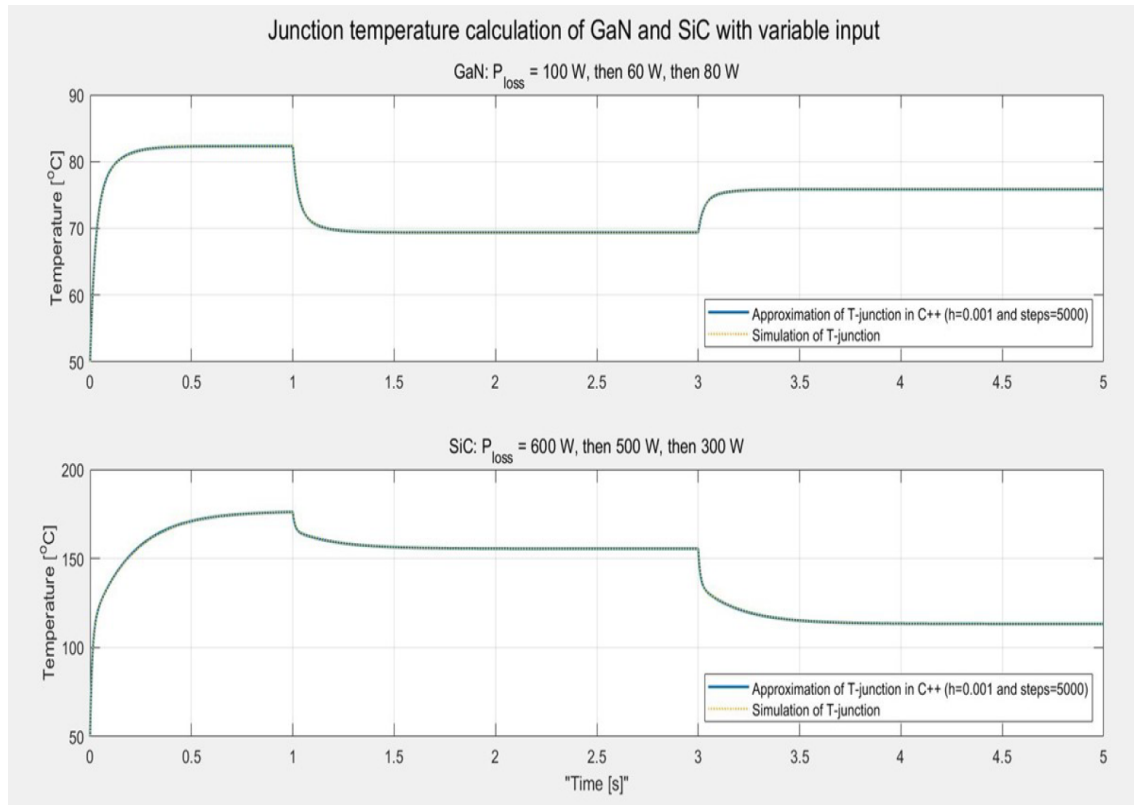


Figure 2.23: The Junction Temperature Results of Both Gan and Sic Transistors with Variable Input

It is essential to define the initial condition, namely the initial temperature. The simulations are based on the assumption that the power module is at ambient temperature prior to the start of operation. Ambient temperature is included as an input parameter to the function.

To incorporate temperature-dependent thermal impedances, the temperature calculation function is implemented to allow parameter updates based on previously computed values. Parameters such as R_{th1} , C_{th2} and others that exhibit temperature dependence are updated accordingly.

An example is provided below illustrating the update of parameters " R_{th1_1} " and " R_{th1_2} " within the SiC model, representing the thermal resistances from the junction to the case of two SiC transistors:

```
116 // Parameters for the SiC equivalent
117 float Rth1_1 = data[0]*180/(a*exp(b*Ta1_SiC) + c*exp(d*Ta1_SiC));
118 float Rth1_2 = data[0]*180/(a*exp(b*Ta2_SiC) + c*exp(d*Ta2_SiC));
119 float Rth2 = data[1];
120 float Rth3 = data[2];
121 float Cth1 = data[3];
122 float Cth2 = data[4];
123 float Cth3 = data[5];
```

The two-term exponential function used is derived via MATLAB's Curve Fitter Toolbox.

The main function (*main()*) in the file *main_LowPower.cpp* includes calls to the class member function *calc_T_junction_GaN()* from the following object instances: GaN_1 through GaN_6:

```

94 T_GaN1 = GaN_1.calc_T_junction_GaN(Ta_GaN1, Tb_GaN1, Tc_GaN1, p_loss_GaN, T_amb, GaN_eq::Data_GaN_1);
95 T_GaN2 = GaN_2.calc_T_junction_GaN(Ta_GaN2, Tb_GaN2, Tc_GaN2, p_loss_GaN, T_amb, GaN_eq::Data_GaN_2);
96 T_GaN3 = GaN_3.calc_T_junction_GaN(Ta_GaN3, Tb_GaN3, Tc_GaN3, p_loss_GaN, T_amb, GaN_eq::Data_GaN_3);
97 T_GaN4 = GaN_4.calc_T_junction_GaN(Ta_GaN4, Tb_GaN4, Tc_GaN4, p_loss_GaN, T_amb, GaN_eq::Data_GaN_4);
98 T_GaN5 = GaN_5.calc_T_junction_GaN(Ta_GaN5, Tb_GaN5, Tc_GaN5, p_loss_GaN, T_amb, GaN_eq::Data_GaN_5);
99 T_GaN6 = GaN_6.calc_T_junction_GaN(Ta_GaN6, Tb_GaN6, Tc_GaN6, p_loss_GaN, T_amb, GaN_eq::Data_GaN_6);

```

The use of C++ enables modular and reusable implementation, allowing each function to operate independently with distinct parameters. This facilitates accurate modeling in cases where power losses differ among transistors, resulting in varying local interface temperatures.

The function *calc_T_junction_GaN()* accepts input parameters specific to each GaN transistor, such as thermal impedances contained in data structures like *Data_GaN_1*. The class implementing this functionality is named *GaN_eq*.

Figure 2.24 is provided to demonstrate the individualized thermal parameter handling for each GaN transistor.

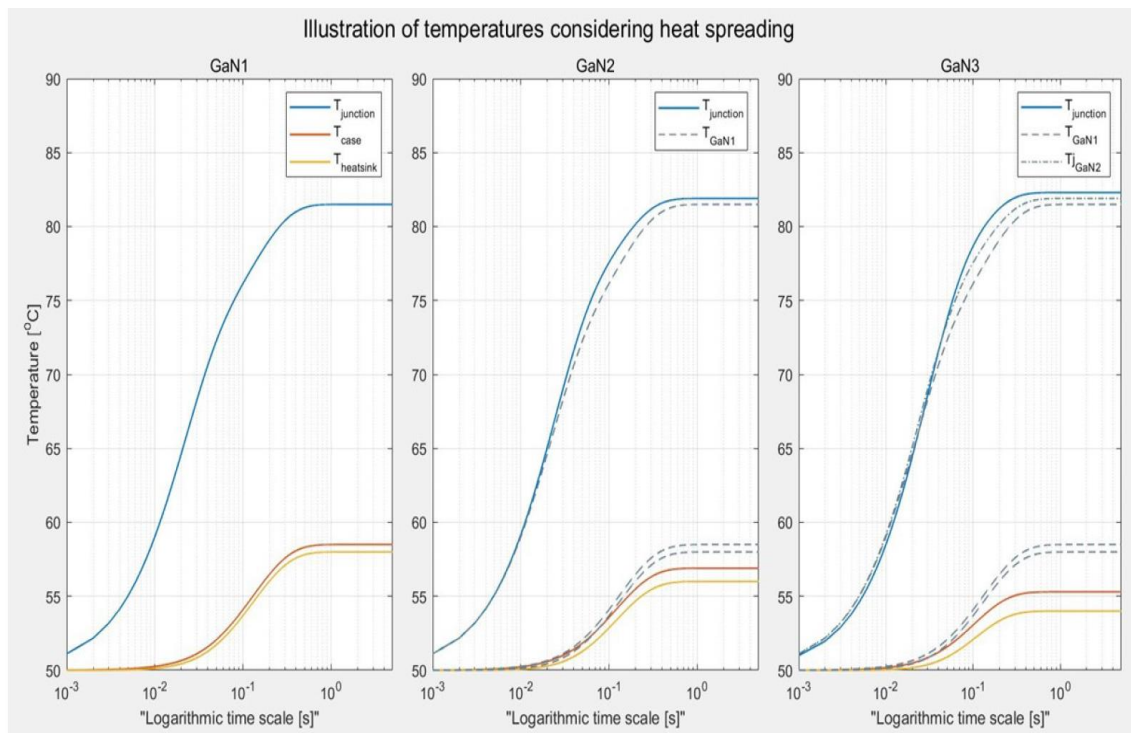


Figure 2.24: The Junction Temperature Results to Demonstrate the Individualized Thermal Parameter Handling for each GaN Transistor

A corresponding header file defines the classes associated with temperature calculation for the GaN model, as follows:


```

17
18 class Gall_eq { //Junction temperature calc. for GaN
19
20 public:
21
22 // struct Data {
23 //     string ThermalParam[6];
24 //     static float Data_Gall_1[6];
25 //     static float Data_Gall_2[6];
26 //     static float Data_Gall_3[6];
27 //     static float Data_Gall_4[6];
28 //     static float Data_Gall_5[6];
29 //     static float Data_Gall_6[6];
30 // };
31
32 float calc_T_junction_GaN(float& Ta_GaN, float& Tb_GaN, float& Tc_GaN, float p_loss_transistor_GaN, float T_amb, float Data[6]);
33
34 };

```

The same structural and functional approach is applied in the function *calc_T_junction_SiC()* for SiC modeling.

Given the requirements of WP4 and Task 4.3 to model both high-power and low-power loading conditions of the converter, separate C++ programs have been developed. The file naming convention distinguishes the two models, using suffixes such as *_HighPower.cpp* and *_LowPower.cpp*.

With both temperature modeling frameworks in place, the next step involves the inclusion of power loss calculations. Simulation data provided by UPC includes power losses generated by two modulation techniques: Space-Vector PWM (SVPWM) and Carrier-Based PWM (CBPWM). The data suggests that SVPWM is used for the high-power converter, while CBPWM is applied to the low-power operation mode. GaN transistors are not included in the high-power operation mode, as SVPWM is not applied to them.

The power loss is modeled as a function of three variables: current (I), modulation index (mi), and frequency (fs , assumed to be sample frequency rather than switching frequency). A dedicated function using cubic spline interpolation has been implemented in C++ to compute power losses for arbitrary input values. The decision to perform interpolation programmatically, rather than generating a lookup table via MATLAB, ensures dynamic adaptability to input parameters.

Although the interpolation spans three dimensions, the process follows a structured approach: interpolation is first performed along the x-axis (mi) for each combination of y and z (I and fs), followed by interpolation along the y-axis for each fs , and finally along the z-axis. This results in an interpolated value for the power loss at the desired coordinate ($mi0, I0, fs0$), (mi_0, I_0, fs_0), and ($mi0, I0, fs0$).

A 3D plot is provided in figure 2.25 to visualize the power loss profile with CBPWM as a function of mi and I at $fs=75fs = 75fs=75$ kHz, alongside a comparison with results generated in MATLAB by UPC.

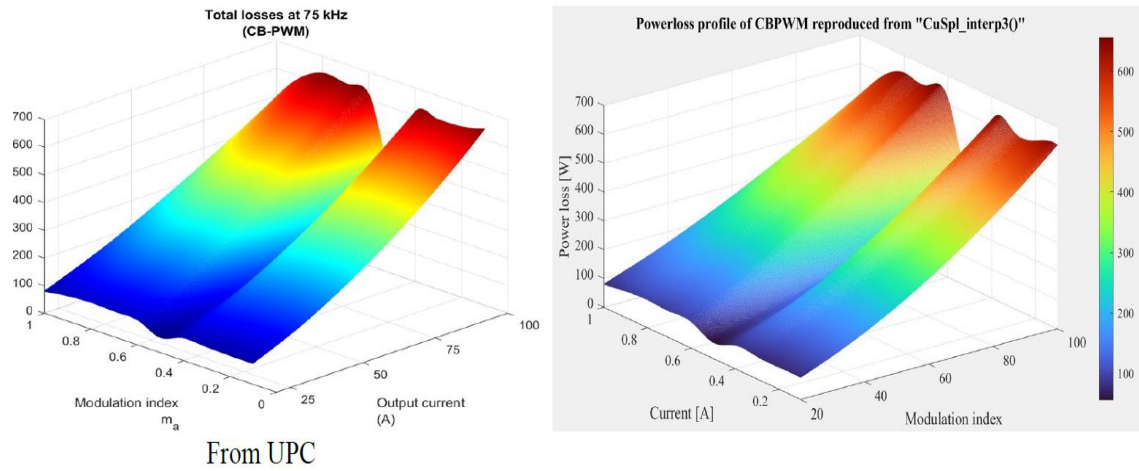


Figure 2.25: The Power Loss Profile with CBPWM As a Function of m_i and I at $F_s=75f_s = 75f_s=75 \text{ kHz}$

Finally, two diagrams illustrate the structure of the *main()* function in the respective C++ programs for the high-power and low-power converters. These diagrams have been created using *Draw.io* for clarity.

A folder titled “RHODaS WP4” has been compiled, containing supporting materials including relevant articles, MATLAB files and simulation models, documentation of the development process for the digital twin and results demonstrating the current status of the C++ implementation.

For the high-power loading conditions the design of functions are shown in Fig. 2.26.

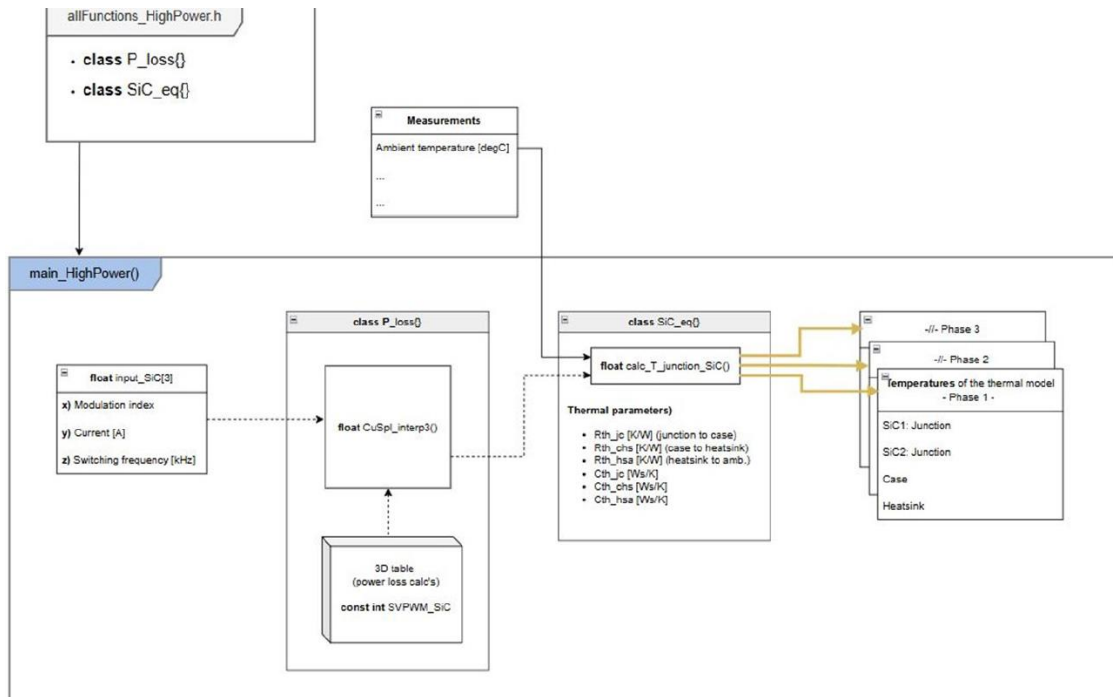


Figure 2.26: The Design of the *main ()* Function for the High-Power C++ Code

For the low-power loading conditions the design of functions are shown in Fig. 2.27.

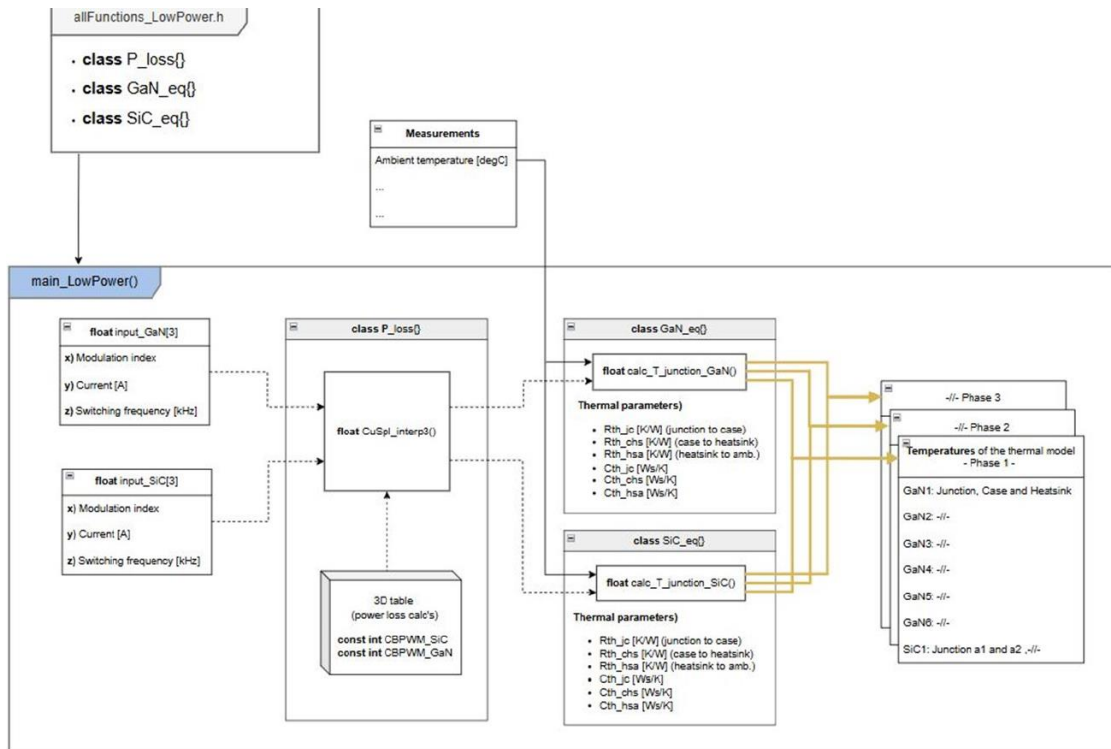


Figure 2.27: The Design of the main () Function for the Low-Power C++ Code

2.2 RHODAS DIGITAL TWIN FOR E-MOTOR

This section presents the development process and implementation details of a digital twin for the RHODaS electric motor, designed as part of the RHODaS project. The purpose of the digital twin is to simulate the thermal behavior of the motor under various operating conditions. The system is implemented in C++, employing both four-dimensional linear interpolation and the 4th-order Runge-Kutta method to accurately simulate temperature dynamics over time.

The foundation for the digital twin is a Simulink model provided by Valeo, which represents the physical and thermal behavior of the electric motor. This model includes six inputs and two outputs:

The four variable inputs:

- PWM_Torque_ref (Torque)
- PWM_Speed (Speed)
- PWM_Trotor (Rotor temperature)
- PWM_Tcopper (Stator temperature)

Two fixed inputs, which are held constant throughout simulations.

- Vdc
- Tamb_air_C

The model outputs:

- P1_stator: power losses in the stator.
- P2_rotor: power losses in the rotor.

The model can be seen on Figure 2.28 below:

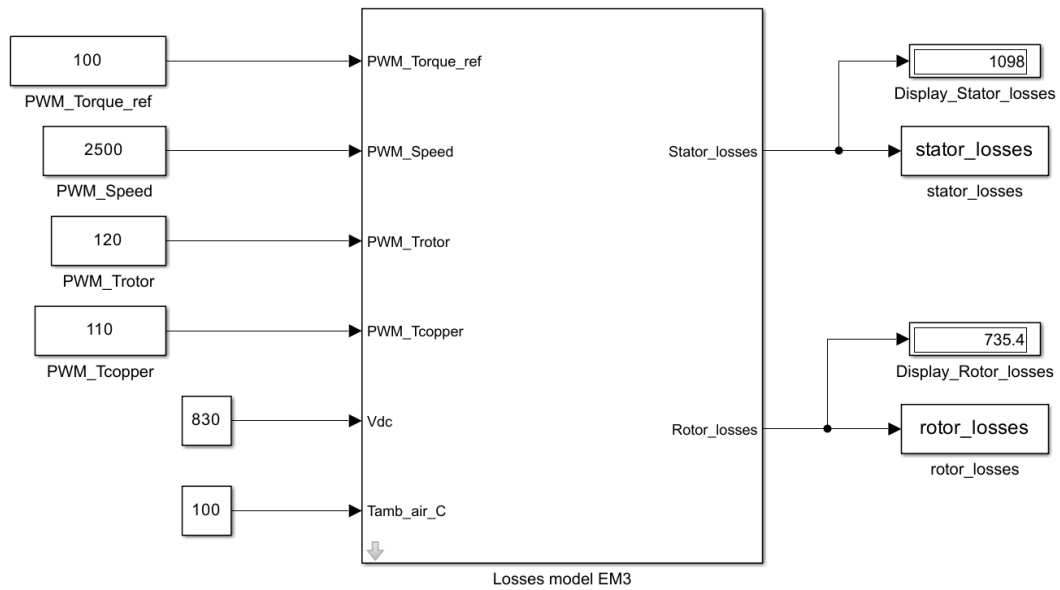


Figure 2.28: Simulink Model of the RHODaS Electric Motor Representing Thermal and Physical Behavior

In order to build a reliable interpolation model, a large dataset was generated using a custom script. The script systematically varies one input parameter at a time while evaluating all possible combinations, based on the following ranges:

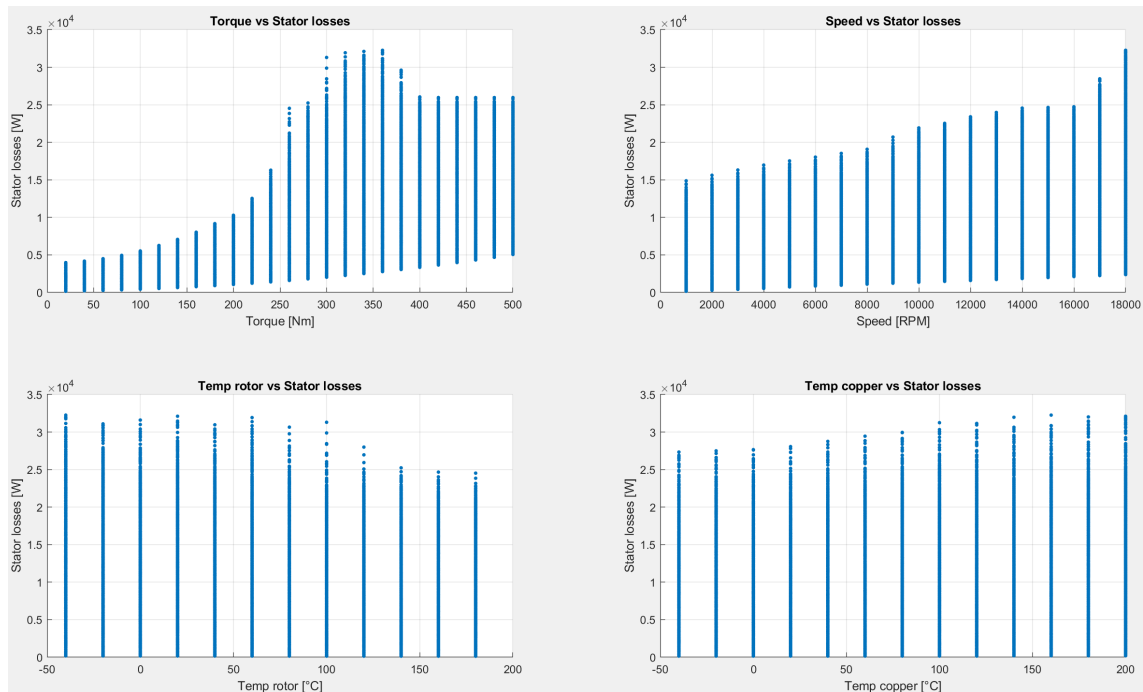


Figure 2.29: The Effect of Torque, Speed, and Temperatures on Stator Losses

- Torque range: 20 to 500 (steps 20)
- Speed range: 1000 to 18000 (steps 1000)
- Temperature rotor: -40 to 180 (steps 20)

- Temperature stator: -40 to 200 (steps 20)

This results in a total of 70,200 unique data points.

Figure 2.29 and Figure 2.30 show a series of 2D plots that illustrate how each of the four input parameters, torque, speed, rotor temperature, and stator (copper) temperature, affects the stator losses and rotor losses, respectively.

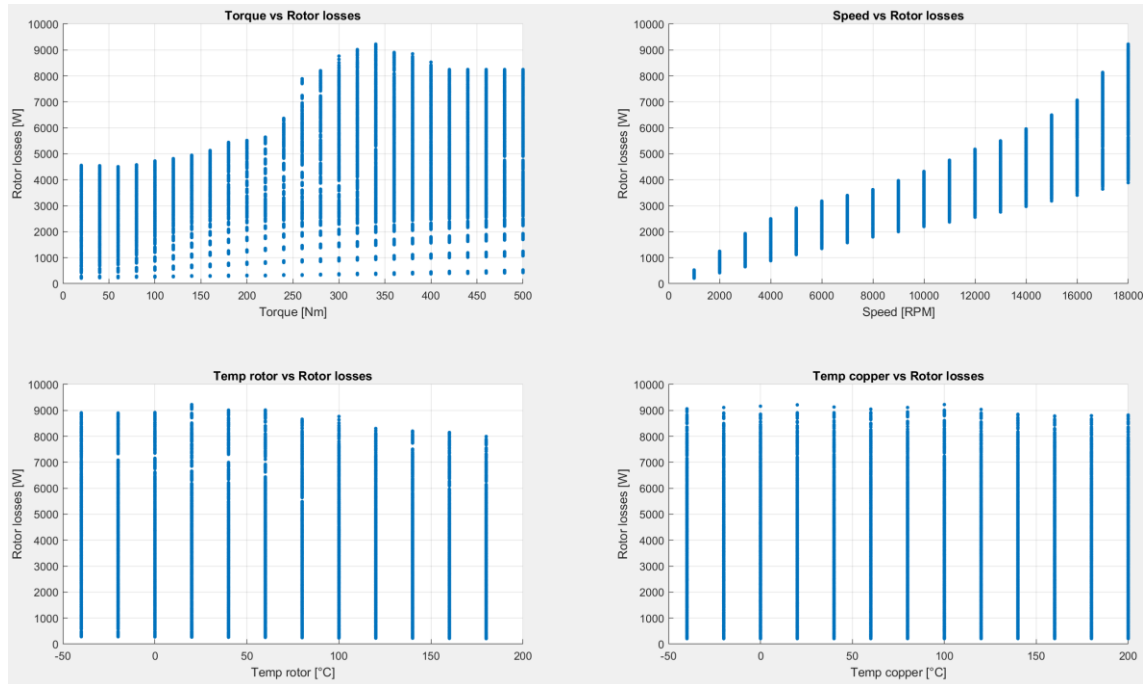


Figure 2.30: The Effect of Torque, Speed, and Temperatures on Rotor Losses

Each subplot isolates one input parameter while implicitly showing the distribution of output values across its range. The plots reveal clear nonlinear trends, such as the increase in losses with torque and speed, as well as the temperature dependency of both stator and rotor losses.

Figure 2.31 and Figure 2.32 shows this analysis by presenting 3D surface plots of the same relationships. These figures capture the interactions between pairs of input parameters and how they jointly influence the losses in the stator and rotor.

The visualizations demonstrate the complex dependency of power losses on operating conditions, thereby justifying the use of a multi-dimensional interpolation approach in the digital twin model.

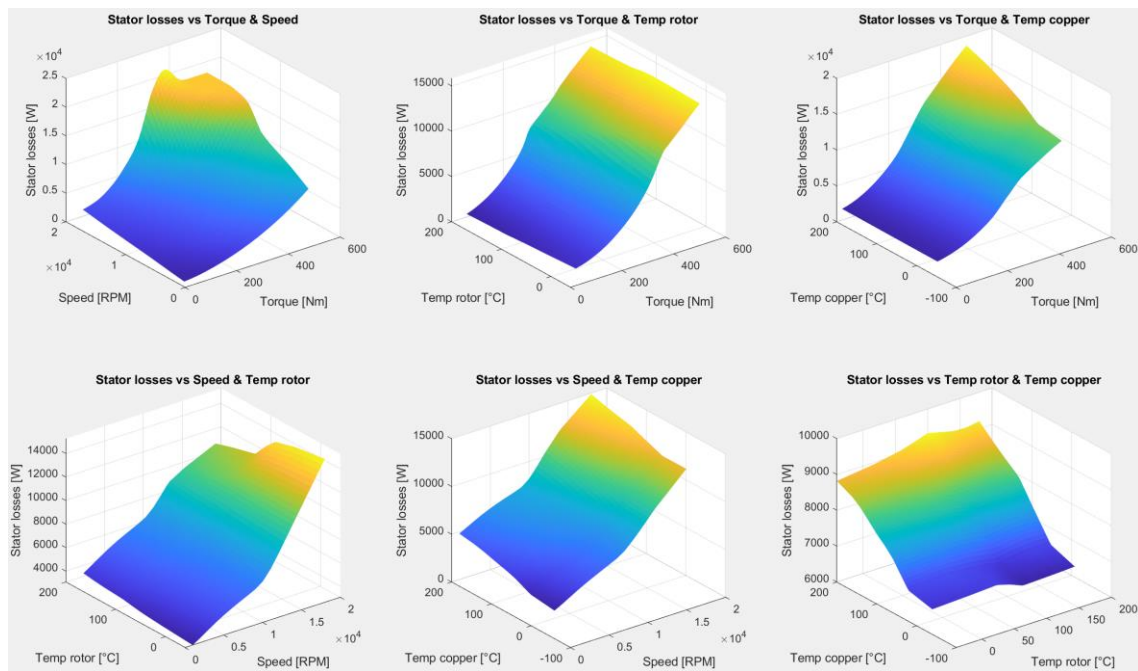


Figure 2.31: 3D Surface Visualization of the Influence of Input Parameter Interactions on Stator Losses

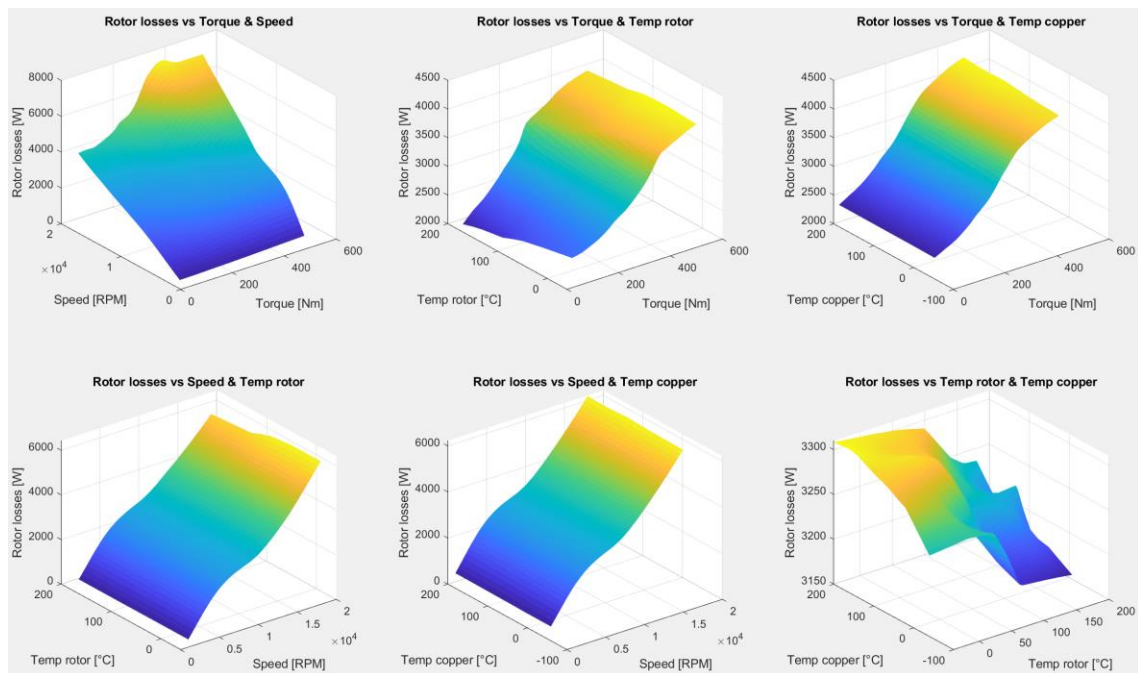


Figure 2.32: 3D Surface Visualization of the Influence of Input Parameter Interactions on Rotor Losses

With the dataset generated and analyzed, the next step was to develop a thermal equivalent circuit model for the motor. This model simplifies the heat transfer dynamics between the stator, rotor, and ambient temperature using thermal resistances and capacitances.

Simulation of a RHODaS e-Motor thermal

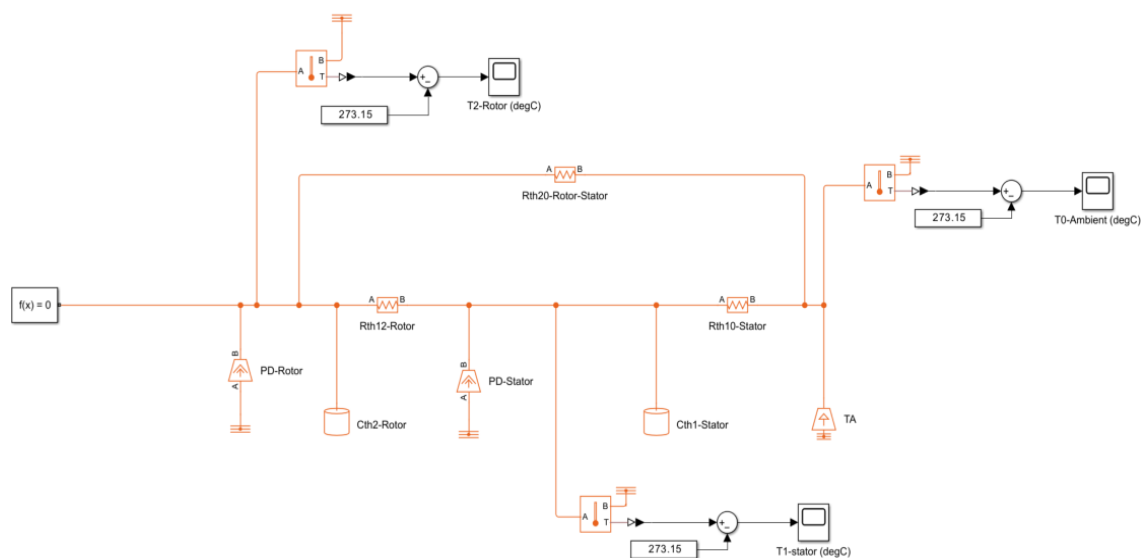


Figure 2.33: Thermal Equivalent Circuit Representing Heat Transfer Between Stator, Rotor, and Ambient Environment

Figure 2.33 shows the simplified thermal equivalent circuit, where power losses from the Simulink model serve as heat sources, and the changes in rotor and stator temperatures are observed over time.

The parameters that were used for the thermal conductances and capacitances in the thermal equivalent circuit can be seen below.

Thermal conductances

- $G_{th10} = 85 \frac{W}{K}$
- $G_{th20} = 9 \frac{W}{K}$
- $G_{th12} = 1.5 \frac{W}{K}$

Thermal capacitances

- $C_{th1} = 8500 \frac{J}{K}$
- $C_{th2} = 5300 \frac{J}{K}$

Figure 2.34 displays the output of the thermal simulation based on these parameters. It shows how temperature varies over time, verifying the physical consistency of the thermal network.

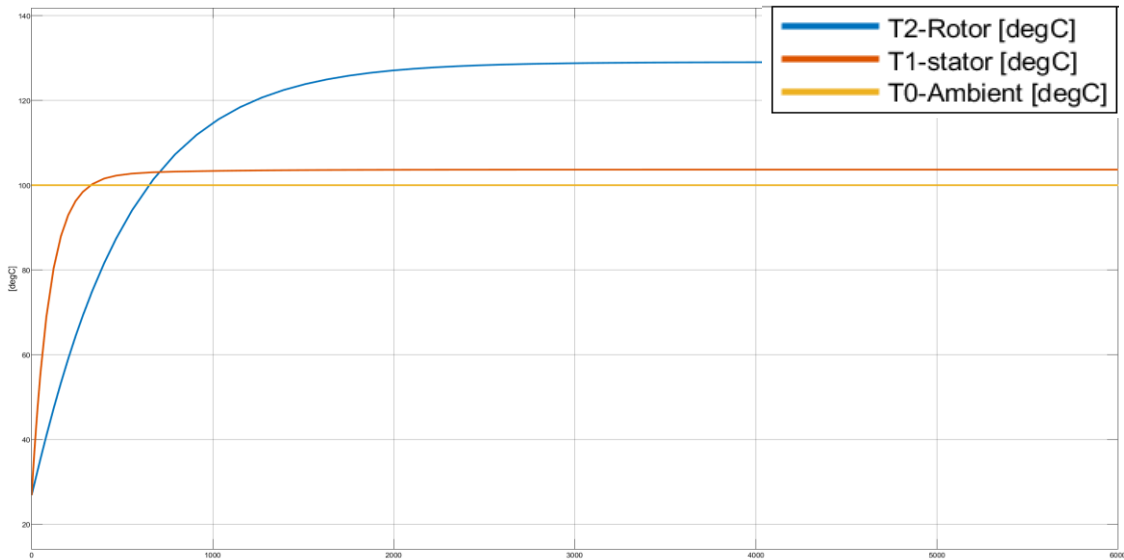


Figure 2.34: Temperature Evolution Over Time in the Thermal Equivalent Circuit Simulation

Based on the thermal equivalent model, the following differential equations were derived.

Differential equations for T1 (stator):

$$\frac{dT_1(t)}{dt} = \frac{1}{C_{th1}} \cdot \left(P_1 + \frac{T_2 - T_1}{R_{th12}} + \frac{T_{amb} - T_1}{R_{th10}} \right)$$

Differential equations for T2 (rotor):

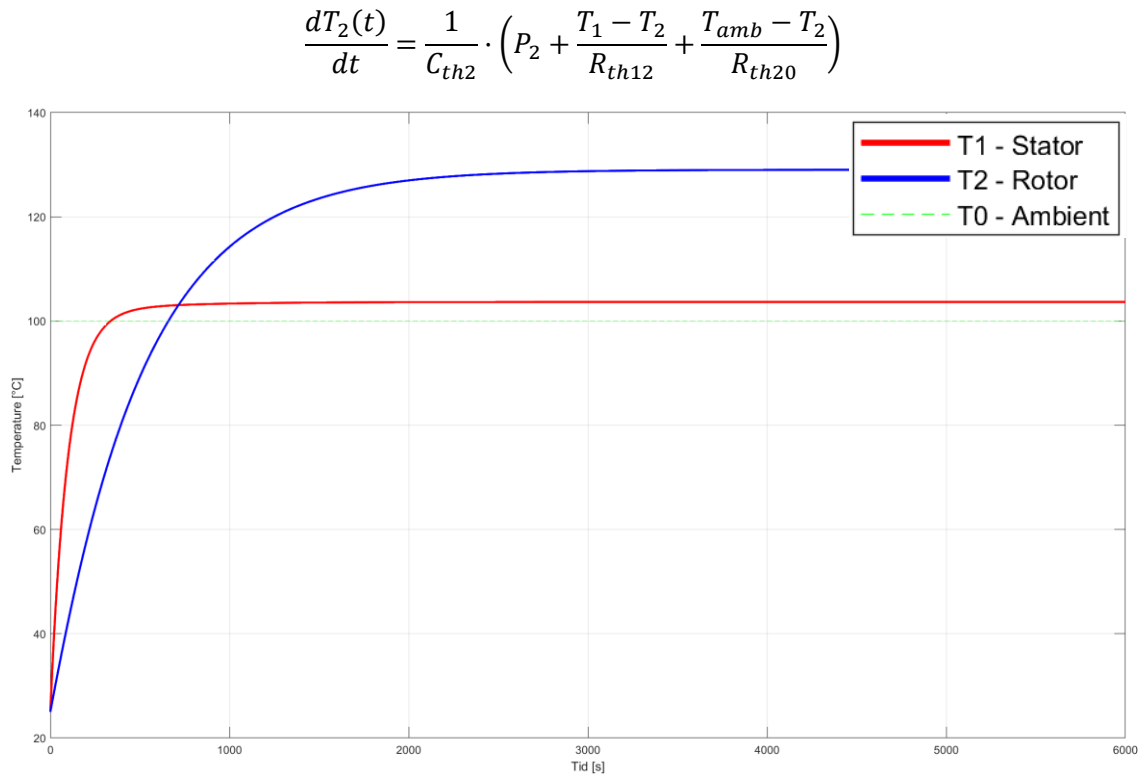


Figure 2.35: Simulated Rotor and Stator Temperature Progression Using Runge-Kutta Solution of the Thermal Model

These equations were initially implemented in MATLAB using the 4th-order Runge-Kutta method to confirm accuracy. The power losses used in these equations match the ones extracted from the Simulink model.

Figure 2.35 shows the temperature progression of both the rotor and stator over time, when compared with Figure 2.34 it validates the correct implementation of the mathematical model.

Once validated in MATLAB, the model was translated into a C++ program. The program is capable of interpolating power loss values for arbitrary input combinations using 4D linear interpolation and subsequently computing the thermal response of the system over time.

The program workflow is as follows:

1. Load Dataset: The data must be stored in a CSV file (default: data_v4.csv) in the following format: Torque; Speed; T_rotor; T_copper; P1_stator; P2_rotor.
2. User Input: The program prompts for four inputs—torque, speed, rotor temperature, and copper (stator) temperature.
3. Interpolation: The function Lin_interp4() performs interpolation within the input space.
4. Thermal Simulation: The interpolated losses are used to compute temperature changes over a 6000-second simulation, with a 1-second time step.
5. Output: Results are written to a CSV file (P_losses.csv) for further analysis.

The following parts of the code show how the program loads the dataset and adjusts the file separator if necessary:

```
int main() {  
    if (!load_csv_data("data_v4.csv")) return 1; // <--- Change file name here  
  
    while (std::getline(ss, cell, ',')) { // Change separator here if needed (e.g. ',')  
        cell = trim_non_printable(cell);  
    }
```

Should the input values fall outside the dataset range, the program will return an error message: *"Interpolation error: Input out of bounds!"*

When the program starts, it attempts to load the CSV file. If the file is missing, incorrectly formatted, or contains invalid data (e.g., non-numeric text in numeric fields or the wrong number of columns), the program will display an error and stop execution.

If the data loads successfully, each row is stored in a global table. All unique values of torque, speed, rotor temperature, and copper temperature are also stored to define the valid interpolation range.

After loading the file, the program enters a loop where the user is prompted to enter four inputs:

1. Torque
2. Speed
3. Rotor temperature
4. Copper temperature

If the interpolation is successful, the program uses the estimated losses to perform a thermal simulation over a period of 6000 seconds with a time step of 1 second. The simulation is based on the validated 4th-order Runge-Kutta solver and the thermal RC model previously described.

Figure 2.36 shows the initial portion of the `rk4_thermal_sim()` function, where all relevant thermal parameters such as capacitances, resistances, and initial conditions are defined in C++. These parameters directly reflect the values used in the Simulink thermal model.

```
// === RUNGE-KUTTA 4 ===  
void rk4_thermal_sim(float P1_stator, float P2_rotor, float T_amb = 100.0f) {  
    int t_end = 6000; // Simulation time  
    float dt = 1.0f; // Time step  
    float T_stator = 25.0f; // Initial temperatures, can be changed if needed  
    float T_rotor = 25.0f; // Initial temperatures, can be changed if needed  
  
    const float Rth_10 = 1.0f / 85.0f; // Thermal resistances  
    const float Rth_20 = 1.0f / 9.0f; // Thermal resistances  
    const float Rth_12 = 1.0f / 1.5f; // Thermal resistances  
    const float Cth_1 = 8500.0f; // Thermal capacitances  
    const float Cth_2 = 5300.0f; // Thermal capacitances
```

Figure 2.36: Initialization Section of the `rk4_thermal_sim()` Function Defining Thermal Parameters in C++

After the simulation, results are exported to a file named *"P_losses.csv"*, which stores the calculated rotor and stator temperatures at each time step.

Figure 2.37 shows the implementation of the full 4th-order Runge-Kutta method in C++. The function iteratively calculates the temperature of the rotor and stator based on the interpolated power losses and the predefined thermal network. The logic includes four intermediate slope evaluations (k_1 – k_4) to improve the accuracy of the simulation.

```
for (int i = 0; i <= t_end; ++i) {
    outputFile << i << ", " << T_rotor << ", " << T_stator << "\n";

    auto f1 = [&](float T1, float T2) {
        return (1.0f / Cth_1) * (P1_stator + (1.0f / Rth_12) * (T2 - T1) + (1.0f / Rth_10) * (T_amb - T1));
    };

    auto f2 = [&](float T1, float T2) {
        return (1.0f / Cth_2) * (P2_rotor + (1.0f / Rth_12) * (T1 - T2) + (1.0f / Rth_20) * (T_amb - T2));
    };

    float k1_T1 = dt * f1(T_stator, T_rotor);
    float k1_T2 = dt * f2(T_stator, T_rotor);
    float k2_T1 = dt * f1(T_stator + 0.5f * k1_T1, T_rotor + 0.5f * k1_T2);
    float k2_T2 = dt * f2(T_stator + 0.5f * k1_T1, T_rotor + 0.5f * k1_T2);
    float k3_T1 = dt * f1(T_stator + 0.5f * k2_T1, T_rotor + 0.5f * k2_T2);
    float k3_T2 = dt * f2(T_stator + 0.5f * k2_T1, T_rotor + 0.5f * k2_T2);
    float k4_T1 = dt * f1(T_stator + k3_T1, T_rotor + k3_T2);
    float k4_T2 = dt * f2(T_stator + k3_T1, T_rotor + k3_T2);

    T_stator += (1.0f / 6.0f) * (k1_T1 + 2.0f * k2_T1 + 2.0f * k3_T1 + k4_T1);
    T_rotor += (1.0f / 6.0f) * (k1_T2 + 2.0f * k2_T2 + 2.0f * k3_T2 + k4_T2);
}
```

Figure 2.37: Implementation of the 4th-Order Runge-Kutta Method for Thermal Simulation in C++

The following part of the code shows the console interface where the program prompts the user to input four variables: torque, speed, rotor temperature, and stator temperature. These inputs must lie within the interpolation dataset range:

```
Loaded CSV with 9000 rows.
Unique values: Torque [Nm] = 8 Speed [RPM] = 15 Temperature rotor [degC] = 15 Temperature copper [degC] = 5
Enter torque: |
```

Figure 2.38 shows the interpolated results printed to the console. These are the estimated stator and rotor power losses at the specified operating point. This confirms that the interpolation step was successful, and the data is ready to be used for thermal simulation.

```
Loaded CSV with 9000 rows.
Unique values: Torque [Nm] = 8 Speed [RPM] = 15 Temperature rotor [degC] = 15 Temperature copper [degC] = 5
Enter torque: 100
Enter speed: 3000
Enter temperature rotor: 100
Enter temperature copper: 100
Interpolated stator losses: 1140.3 W
Interpolated rotor losses: 922.9 W
Results written to P_losses.csv
Run again? (y/n): |
```

Figure 2.38: Interpolated Stator and Rotor Power Losses at the Specified Operating Point

Figure 2.39 shows the power loss values obtained via interpolation, compared to original Simulink data, verifying the 4D interpolation done by the digital twin.

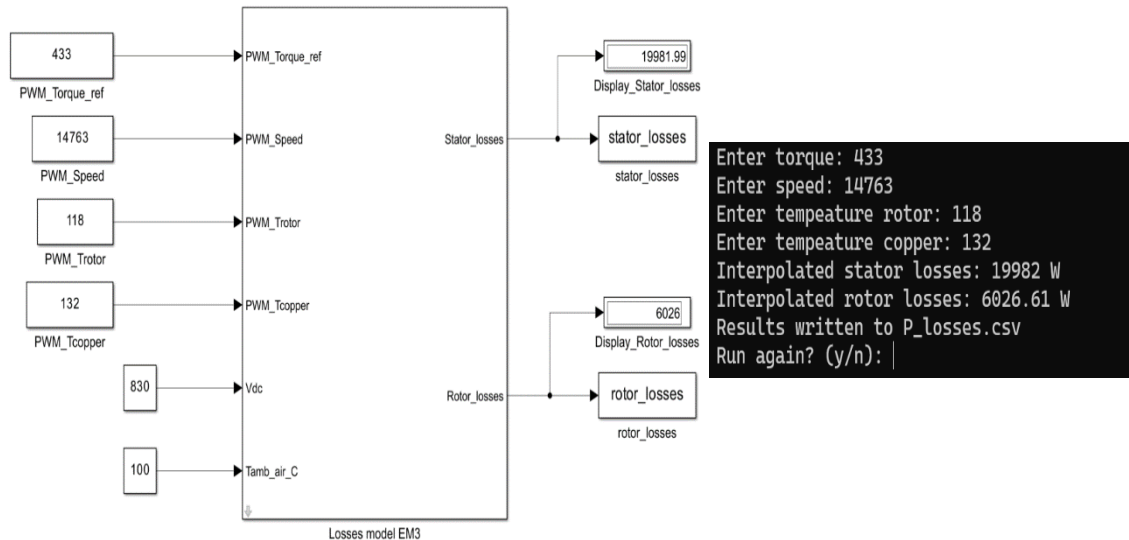


Figure 2.39: Comparison of Power Loss Values from Interpolation and Original Simulink Data

Figure 2.40 shows a comparison between the thermal simulation results obtained from the MATLAB implementation of the differential equations using the 4th-order Runge-Kutta method, and the C++ implementation based on interpolated power losses.

The results demonstrate that the temperature evolution of both the stator and rotor is nearly identical across both implementations, when using the same initial conditions and input values. This confirms the numerical consistency between the MATLAB and C++ models and verifies that both the interpolation logic and the Runge-Kutta solver in C++ are functioning as intended.

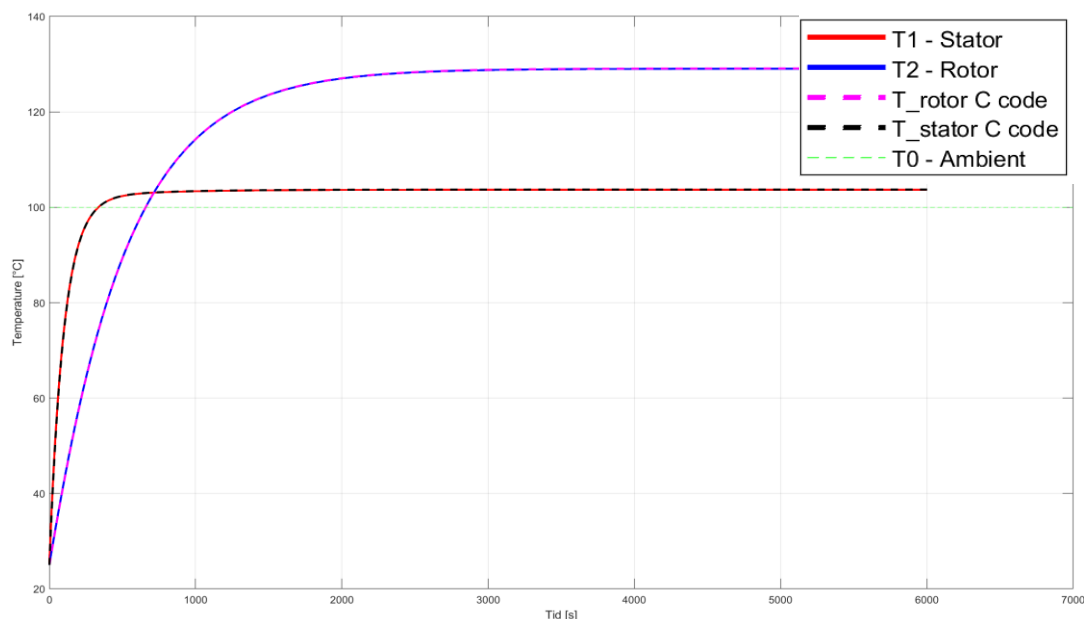


Figure 2.40: Comparison of Thermal Simulation Results from MATLAB and C++ Implementations

Figure 2.41 shows that both stator and rotor losses increase nonlinearly with torque. Stator losses grow much more sharply than rotor losses, indicating that the stator is more sensitive to increases in torque. Since torque is the dominant mechanical driver of heat generation, particularly for the stator, efficiency optimization should focus on torque smoothing or loss-compensated control strategies.

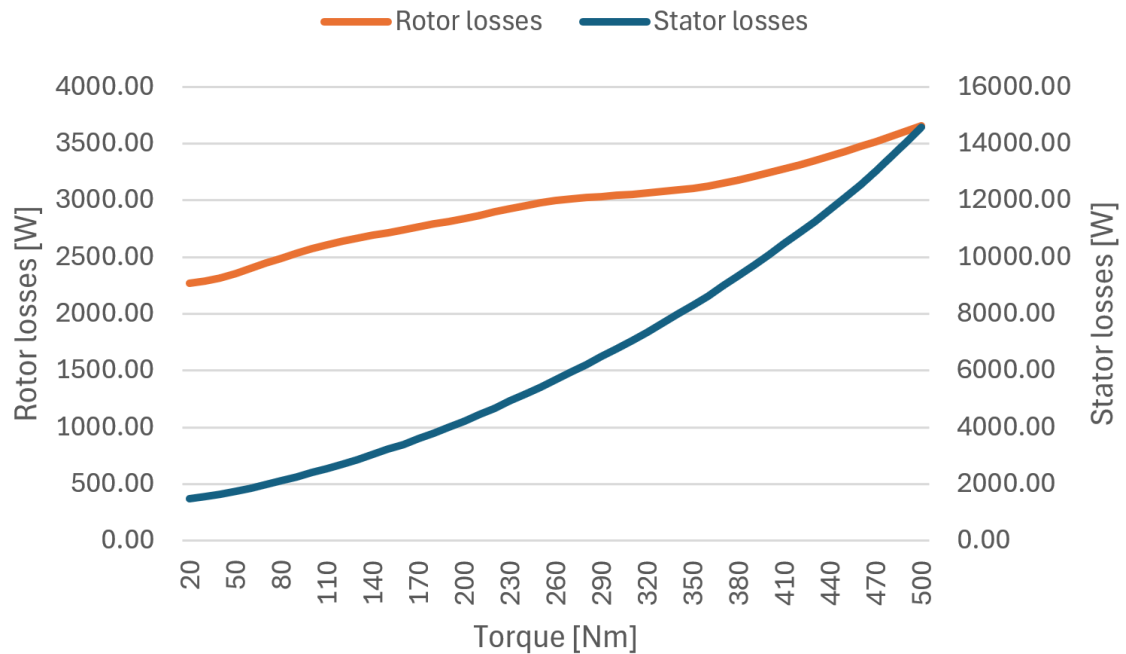


Figure 2.41: Nonlinear Increase of Stator and Rotor Losses with Torque

Figure 2.42 shows that T1 (stator temperature) rises faster and higher than T2 (rotor temperature), particularly under high torque. T2 remains smoother, as the rotor exhibits a slower thermal response due to its rotation and lower electrical stress.

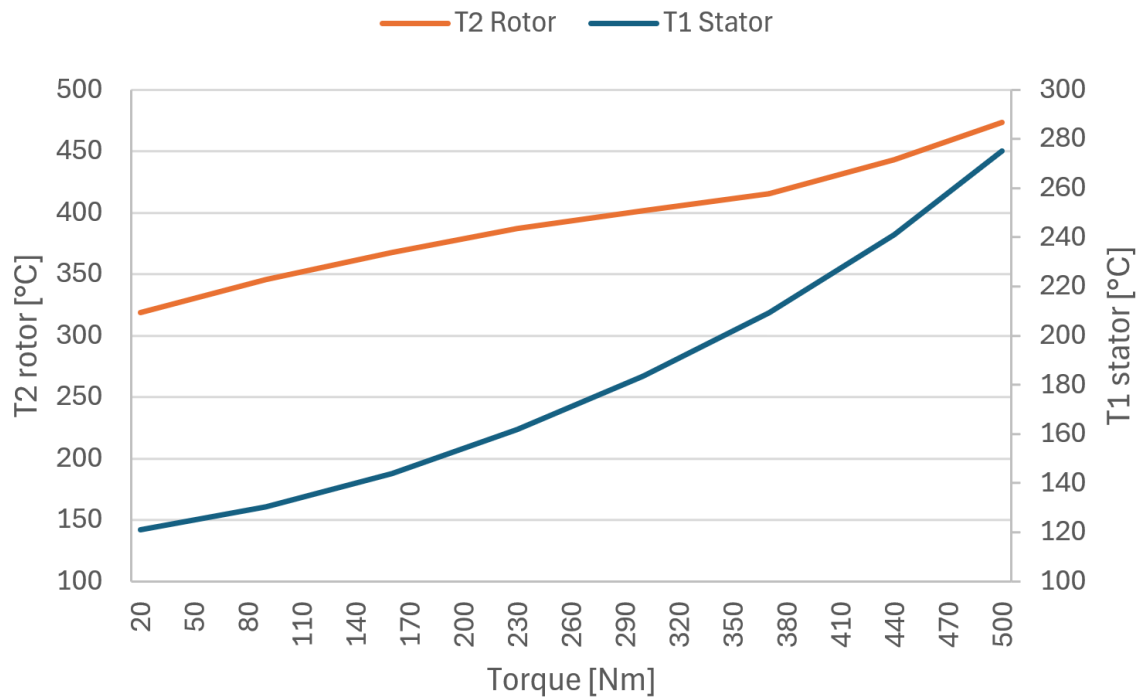


Figure 2.42: Comparison of Stator and Rotor Temperature Responses to Torque

Figure 2.43 shows that both stator and rotor losses increase with speed, though the stator exhibits stronger sensitivity to changes in speed.

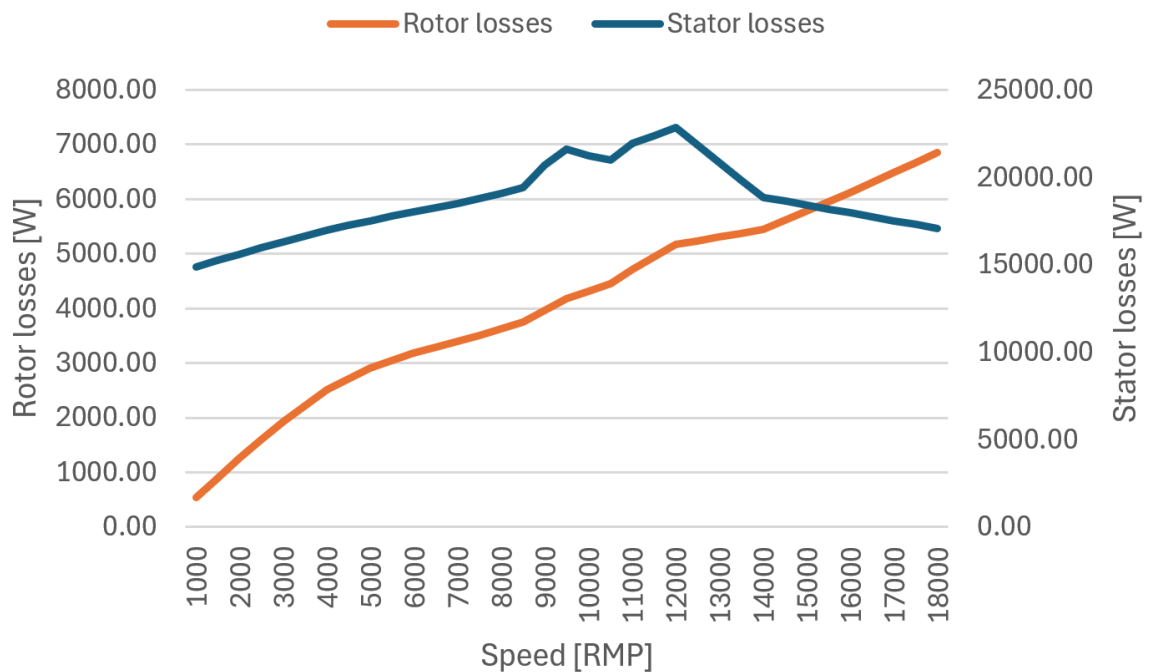


Figure 2.43: Increase of Stator and Rotor Losses with Speed

Figure 2.44 shows that both T1 (stator temperature) and T2 (rotor temperature) increase with speed, but T1 responds more sharply. This suggests that speed indirectly drives rotor heating via thermal conduction.

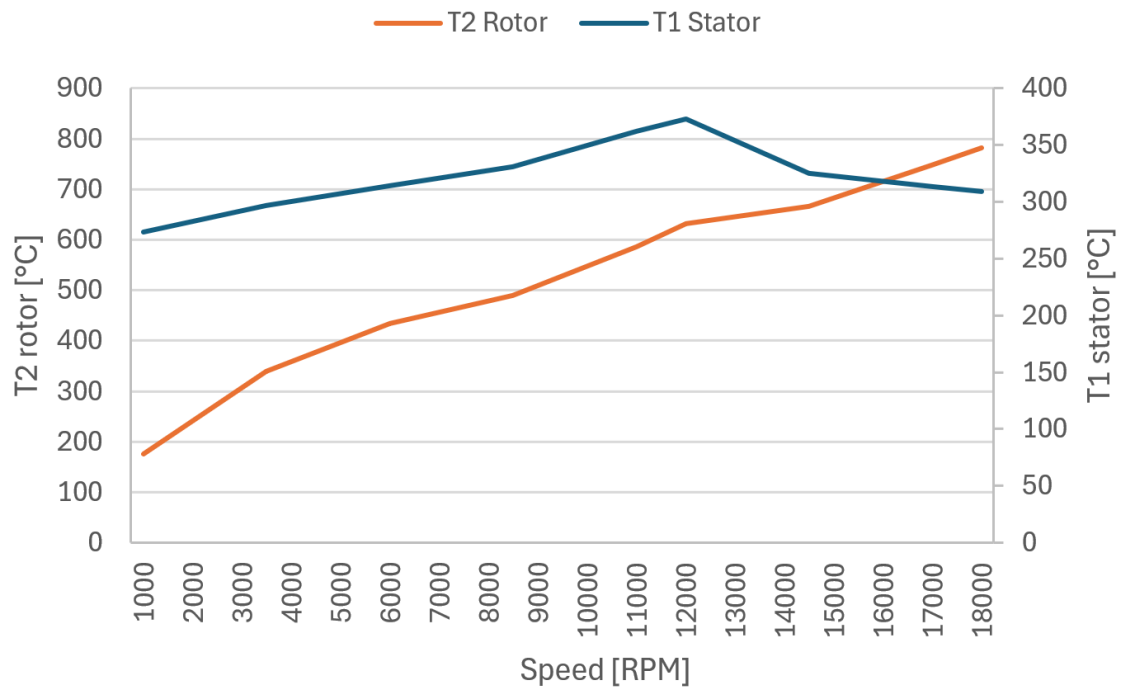


Figure 2.44: Nonlinear Increase of Stator and Rotor Losses with Torque

Figure 2.45 shows that stator losses increase significantly with rising copper temperature, while rotor losses remain relatively flat.

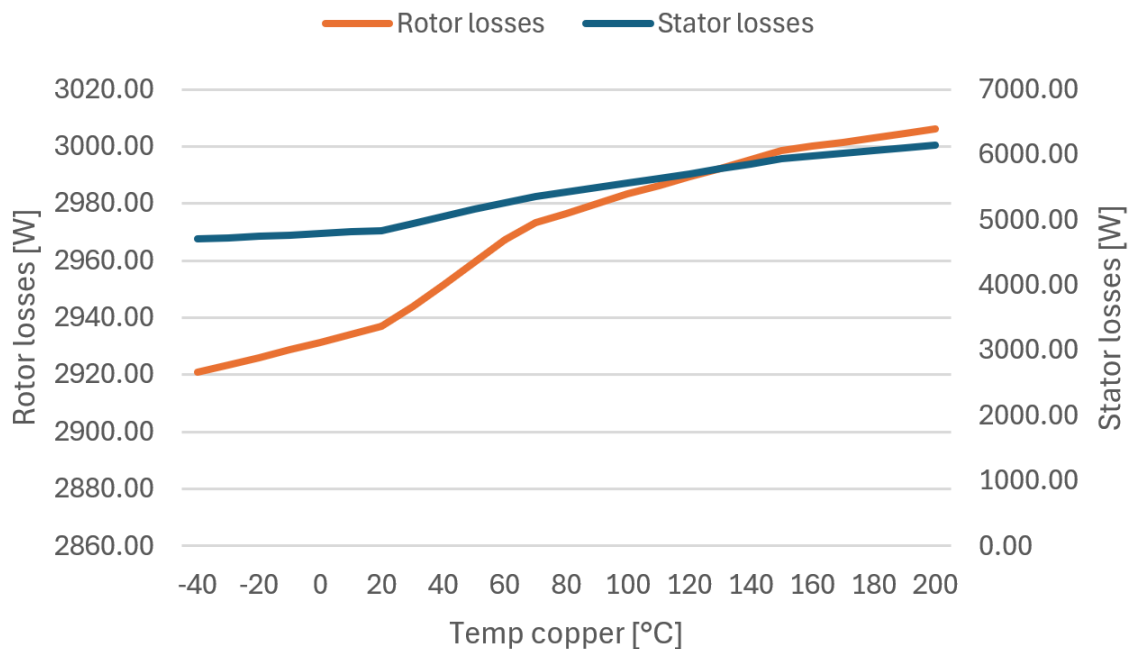


Figure 2.45: Effect of Copper Temperature on Stator and Rotor Losses

Figure 2.46 shows that T1 (stator temperature) increases with initial copper temperature, starting high and continuing to climb. In contrast, T2 (rotor temperature) remains low and stable, indicating weak conductive or radiative interaction between the stator and rotor.

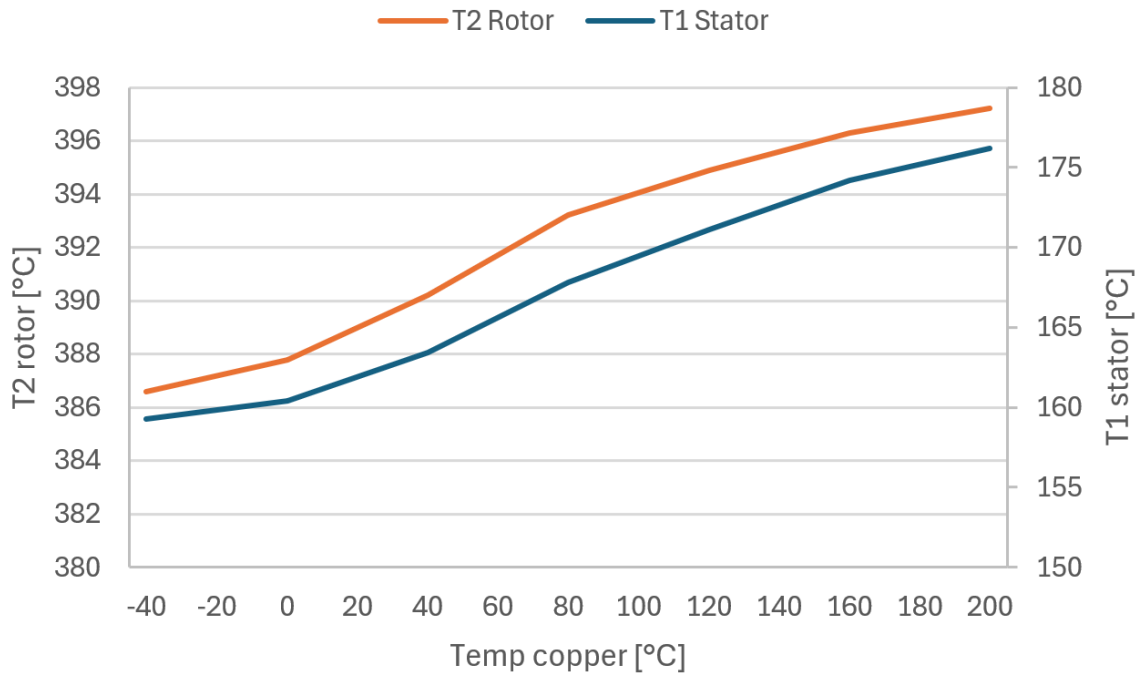


Figure 2.46: Stator and Rotor Temperature Response to Initial Copper Temperature

Figure 2.47 shows that rotor losses increase steadily with rotor temperature, while stator losses remain mostly unchanged. This indicates that rotor temperature has no meaningful influence on stator losses.

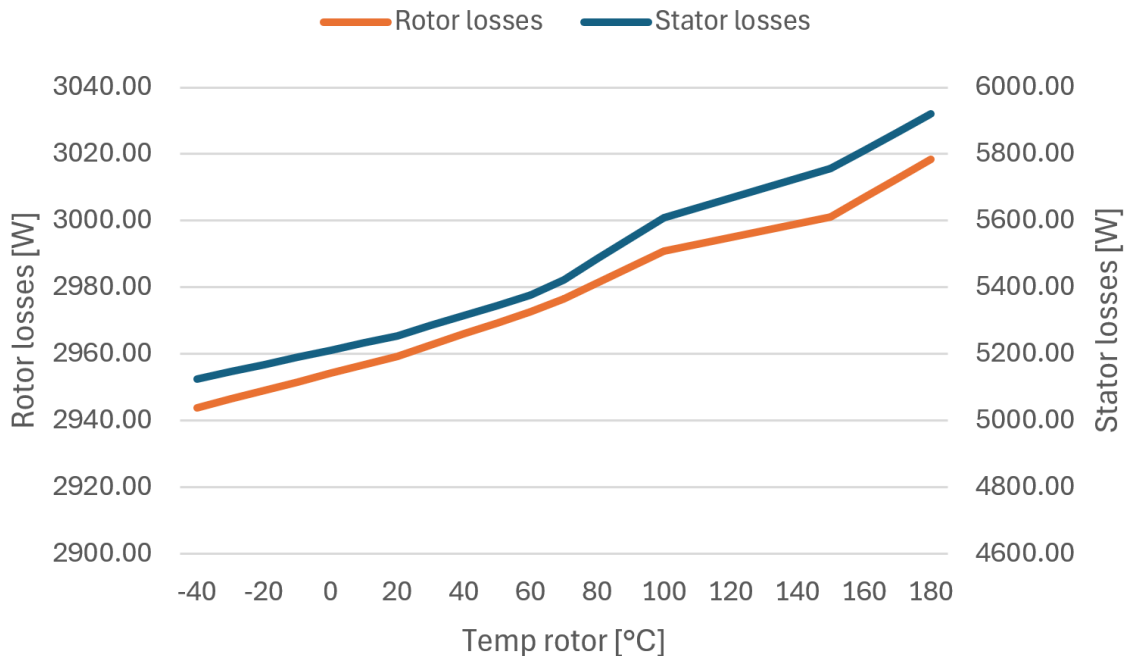


Figure 2.47: Effect of Rotor Temperature on Rotor and Stator Losses

Figure 2.48 shows that T2 (rotor temperature) begins at higher values based on the input and climbs gradually, while T1 (stator temperature) remains stable. This suggests that rotor heat does not significantly propagate to the stator.

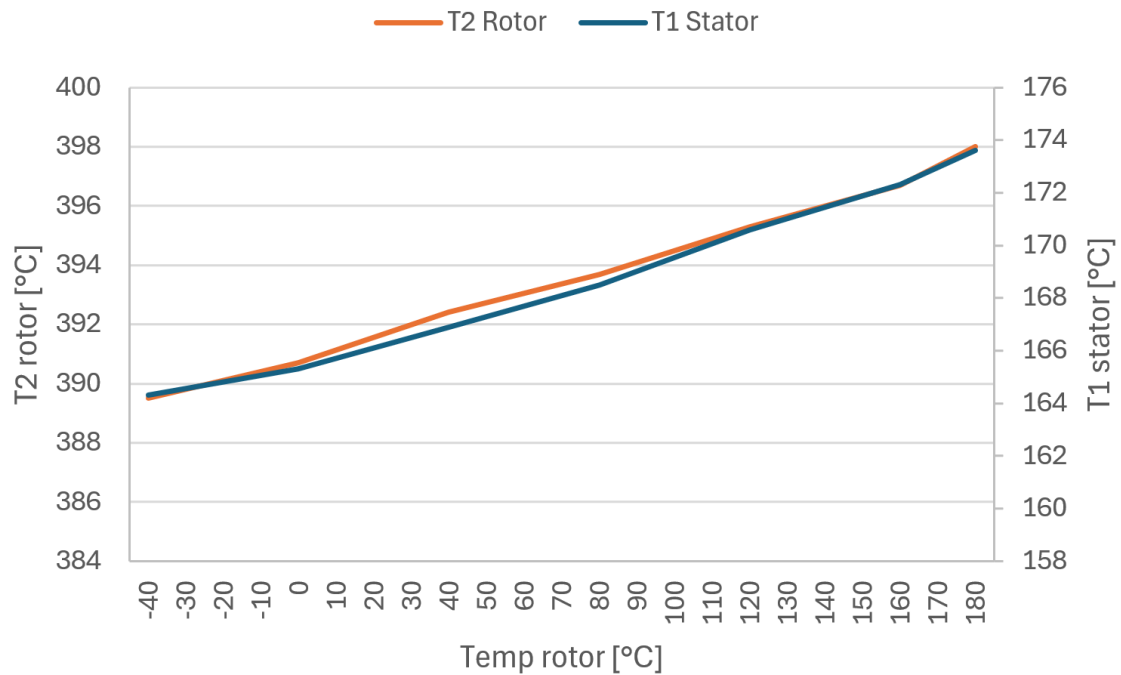


Figure 2.48: Nonlinear Increase of Stator and Rotor Losses with Torque

2.3 RHODAS DIGITAL TWIN FOR GEARBOX

This section describes the development, program and methods of the digital twin. The developed model is based around a dataset provided by Valeo. The model can simulate surface temperature of the gearbox over time. The system is implemented in C++, employing both 3D interpolation and the 4th order Runge-Kutta Method for accurate simulation of surface temperature over time.

The TECM model of the system has been developed around the dataset provided by Valeo. This dataset consists of 576 test cases, with variation to different parameters. The parameters that are described in the system are as follows:

- Load case
- Power Flow
- Input power
- Mechanical loss
- Bearing loss
- Parallel Gear loss
- Hydraulic pump loss
- Speed Cut
- Power Cut
- Temperature
- Input Shaft speed.
- Input Shaft Torque
- System total Loss
- System total efficiency

A screenshot of the dataset can be seen below:

Load Case	Powerflow	Input Power (W)	Simulation Mechanical Power Loss (W)	Bearing	Parallel Gear	Hydraulic Pump (W)	Speed Cut (W)	Power Cut (W)	Temp (°C)	Input shaft Speed (rpm)	Input shaft Torque (Nm)
load_case6	1Q	551538	9781	2449	7332	305	305	305	80	10500	-502
load_case144	1Q	551538	9305	1973	7332	305	305	305	80	10500	502
load_case150	1Q	551538	9820	2411	7409	305	305	305	90	10500	-502
load_case288	1Q	551538	9343	1934	7409	305	305	305	90	10500	502
load_case294	2Q	551538	8098	2161	5937	305	305	305	80	10500	-502
load_case432	2Q	551538	7716	1779	5937	305	305	305	80	10500	502
load_case438	2Q	551538	8116	2116	6000	305	305	305	90	10500	-502
load_case576	2Q	551538	7732	1733	6000	305	305	305	90	10500	502
load_case12	1Q	503578	8849	2273	6576	305	305	305	80	10500	-458

Figure 2.49: Sample View of the Valeo Dataset Used for Gearbox Temperature Modeling

A simplified thermal model of the system has been developed based on the available simulated data. According to the provided specifications, the internal components of the gearbox consist of 80 kg of steel, while the housing is made of 45 kg of aluminum. In the model, these two components are treated as solid bodies with no air gap between them, allowing for continuous thermal conduction.

The model, illustrated in Figure 2.50, is constructed based on a highest temperature scenario, where the temperature specified in the dataset reaches 90 °C. This provides a conservative basis for thermal analysis and ensures the model remains robust under high-load conditions.

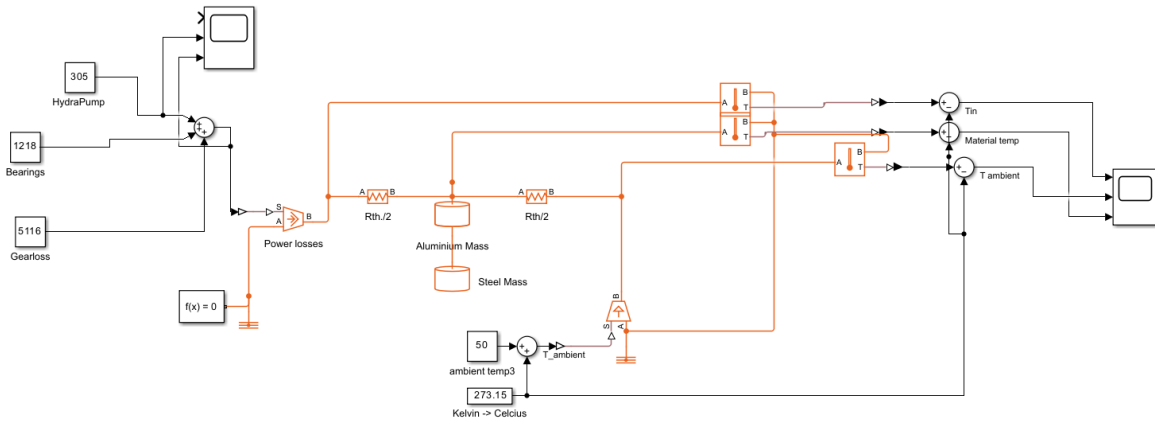


Figure 2.50: Simplified Gearbox Thermal Model Based on Worst-Case Temperature Scenario

The system accepts user input for various individual loss components, which are then aggregated into a total power loss value. This total loss is injected into the thermal model to simulate heat buildup within the gearbox. According to the provided dataset, the maximum recorded internal losses correspond to an internal temperature of 90 °C. Additionally, an ambient temperature of 50 °C is assumed to represent a worst-case environmental condition.

The model subsequently simulates the development of the external surface temperature of the gearbox casing over time. It should be noted that this is a simplified representation of the system, developed based on several assumptions due to the lack of detailed physical and structural information about the actual gearbox.

The model is built around the following parameters, summarized in Table 2.1:

Table 2.1: Different functionalities of the RHODAS IOTP

Component	Value	Unit
Aluminum weight	45	kg
Steel Weight	80	kg
Aluminum Thermal capacitance	$45\text{kg} \cdot 900 = 40500$	J/°K
Steel thermal capacitance (based on tool steel)	$80\text{kg} \cdot 460 = 36800$	J/°K
Thermal resistance	0.0039511	K/W

The model allows the user to input values for three key loss components that contribute to the total power loss in the system:

- *Hydraulic pump loss*
- *Bearing losses*
- *Gear losses*

Internally, all temperature calculations within the model are performed in Kelvin (K). However, appropriate conversions are applied to allow user input and output visualization in degrees Celsius (°C) for ease of interpretation.

To monitor and analyze the thermal behavior of the gearbox, the model includes scopes that display both input and output data. These scopes enable users to observe the temperature development of the system in real time throughout the simulation.

When testing the model with a total power loss input of 10,125 W, the resulting temperature response is shown in Figure 2.51. This scope plot illustrates the simulated development of the gearbox surface temperature over time under maximum load conditions.

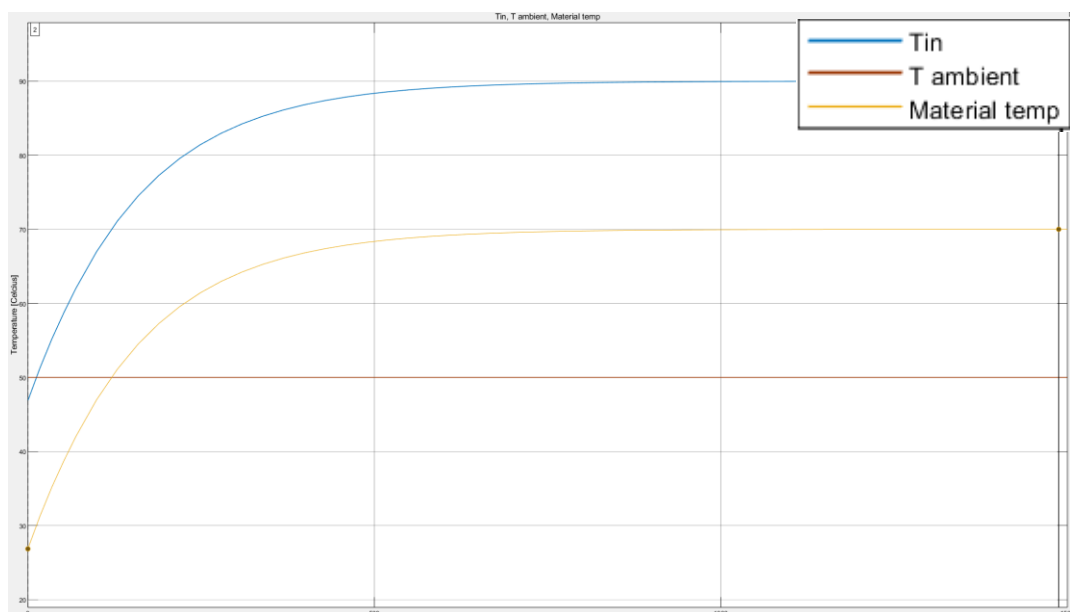


Figure 2.51: Simulated Gearbox Surface Temperature Response at 10,125 W Power Loss

The simulation results show that the internal temperature of the gearbox stabilizes at approximately 90 °C, while the surface temperature of the housing settles around 70 °C. These values appear to be within a reasonable range for high-load operating conditions (*n/a, u.d.*). It is important to note that the model developed is a simplified representation of a complex thermal system. As it is based solely on material mass and a limited dataset, it does not fully capture the real-world dynamics of a gearbox with intricate geometries and heterogeneous material interactions. Furthermore, the model relies on assumptions regarding material properties, such as the specific type of steel alloy used for internal components.

The interpolation functionality within the code is based on linear interpolation using three input parameters:

- Torque (Nm)
- Shaft speed (Rpm)
- Temperature (°C)

These parameters define a three-dimensional space over which the model interpolates to estimate power loss and thermal behavior under various operating conditions.

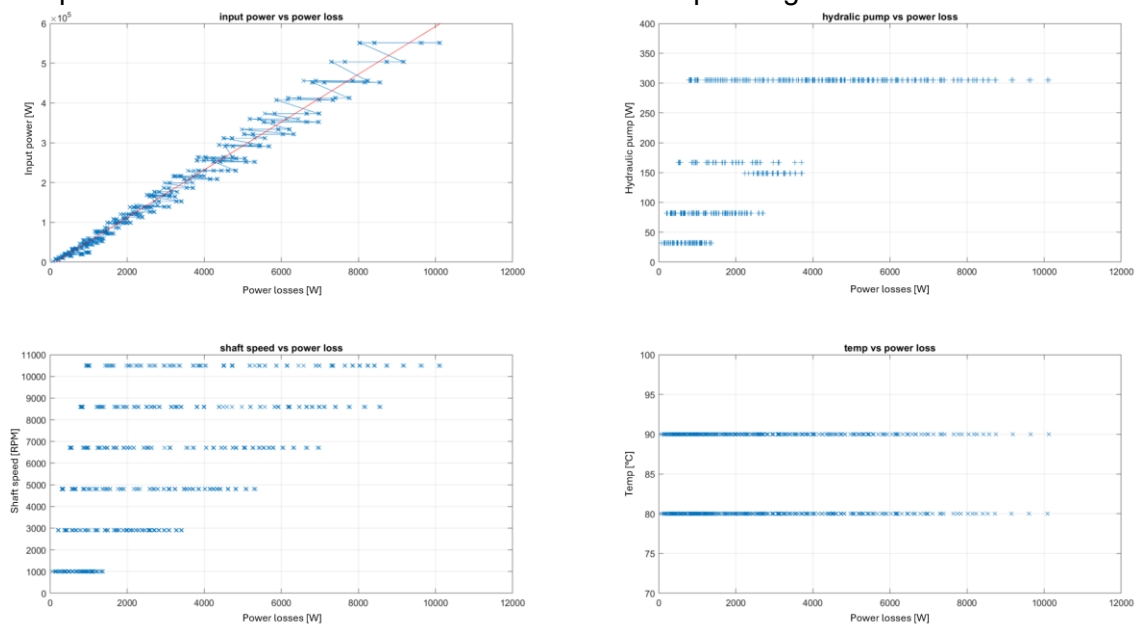


Figure 2.52: Visualization of Interpolation Parameters and Their Relationship to Total Power Loss

The interpolation parameters have been selected from the dataset provided by Valeo. Figure 2.52 presents 2D plots that illustrate the relationship between each selected parameter and the corresponding total power losses. These plots indicate that the relationships are predominantly linear.

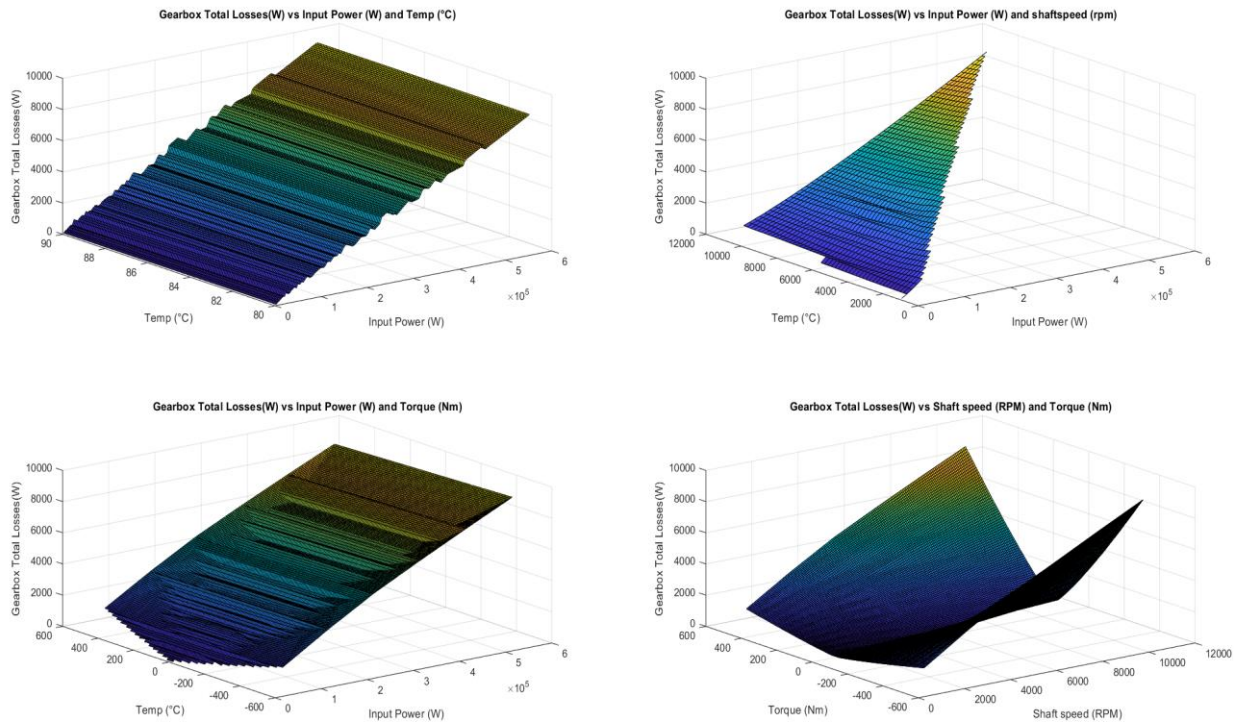


Figure 2.53: 3D Plot of Interpolation Variables Versus Total Power Loss

In Figure 2.53, a 3D plot visualizes the combined effect of torque, shaft speed, and temperature on total power losses. The overall trend also suggests linearity, with the exception of Figure 2.53- 2.2, which exhibits a parabolic shape. This deviation arises from the torque values ranging between -502 Nm and 502 Nm, indicating that the gearbox was operated in both rotational directions during testing.

The interpolation uses three columns from the dataset and the total losses to create a 3D linear interpolation. The values from the dataset are saved in a separate CSV file that is included in the C++ code. Each of the unique values in each column (except the total power losses) are saved in a lookup table. When the user inputs values for each of the three variables, the code interpolates between the closest points to the input values and calculates an estimated power loss, based on the 4 columns.

To evaluate the interpolation model, three test cases will be considered: low power loading condition input, medium power loading condition input, and high-power loading condition input. The input power in the test data ranges from $2,284$ W to $551,538$ W. Testing will involve selecting values for the three input parameters within the following power ranges: $2,284$ – $176,766$ W, $186,385$ – $373,173$ W, and $407,659$ – $551,538$ W. Table 2.2 summarizes the conditions for the first case, corresponding to the low power loading condition.

Table 2.2: Parameters and Conditions for Low Power Loading First Test Case

Case 1 - low power		
Parameter	Values	Unit
Torque	384	Nm
Shaft speed	3341	RPM

Tempera- ture	87	°C
Interpolated loss	2699.5	W
Enter the shaft torque (Nm): 384 Enter the shaft speed (rpm): 3341 Enter the temperature (degC): 87 Interpolated value = 2699.484375		

The interpolated value is calculated as 2,699 W. When comparing this to adjacent values from the dataset, we find a test case (load_case272) that matches the values shown in Table 2.3.

Table 2.3: Parameter Values for Test Case (load_case272) in the Dataset

Load_case272		
Parameter	Values	Unit
Torque	414	Nm
Shaft speed	2900	RPM
Tempera- ture	90	°C
Interpolated loss	2601.6	W

By examining another load case with parameters slightly above those we used, we find load_case411, which corresponds to the parameter values shown in Table 2.4.

Table 2.4: Parameter Values for Test Case (load_case411) in the Dataset

Load_case411		
Parameter	Values	Unit
Torque	371	Nm
Shaft speed	4800	RPM
Tempera- ture	80	°C
Interpolated loss	2949.1	W

As shown in Table 2.5, the interpolated value is correctly positioned between these two cases, closer to load_case272.

Table 2.5: Parameter Values for Test Case (load_case411) in the Dataset

Case 2 - medium power		
Parameter	Values	Unit
Torque	355	Nm
Shaft speed	7500	RPM
Tempera- ture	84	°C
Interpolated loss	5053.6	W

Adjacent cases, load_case130 and load_case108, have the parameter values shown in Table 2.6.

Table 2.6: Parameter Comparison of Adjacent Cases (load_case130 and load_case108)

Load_case130		
Parameter	Values	Unit
Torque	414	Nm
Shaft speed	6700	RPM
Tempera- ture	80	°C
Interpolated loss	5416.7	W
Load_case108		
Parameter	Values	Unit
Torque	240	Nm
Shaft speed	10500	RPM
Tempera- ture	80	°C
Interpolated loss	4509.3	W

Case 3, the high-power loading condition, is summarized in Table 2.7.

Table 2.7: Parameters for High Power Loading Condition (Case 3)

Case 3 - high power		
Parameter	Values	Unit
Torque	376	Nm
Shaft speed	8500	RPM
Tempera- ture	85	°C
Interpolated loss	5947.4	W

Adjacent cases, load_case130 and load_case108, have the values and parameters shown in Table 2.8.

Table 2.8: Parameter Values of Adjacent Cases (load_case130 and load_case108) for Case 3 Comparison

Load_case130		
Parameter	Values	Unit
Torque	414	Nm
Shaft speed	10500	RPM
Tempera- ture	80	°C
Interpolated loss	7841.5	W
Load_case 108		
Parameter	Values	Unit

Torque	414	Nm
Shaft speed	6700	RPM
Temperature	90	°C
Interpolated loss	5437.0	W

The differential equation is derived from the simplified Thermal Equivalent Circuit Model (TECM). It takes the following form:

$$T(t) = T_{amb} + (T_0 - T_{amb}) e^{-\frac{t}{R_{th}C_{th_tot}}} + \frac{P_{loss}R_{th}}{C_{th_tot}} \left(1 - e^{-\frac{t}{R_{th}C_{th_tot}}} \right)$$

By applying the Runge-Kutta method, the differential equation is transformed into the following numerical format:

```
k1 = h * temperature_change(T, T_amb, P_loss, C_th_tot, R_th);
k2 = h * temperature_change(T + 0.5 * k1, T_amb, P_loss, C_th_tot, R_th);
k3 = h * temperature_change(T + 0.5 * k2, T_amb, P_loss, C_th_tot, R_th);
k4 = h * temperature_change(T + k3, T_amb, P_loss, C_th_tot, R_th);
T = T + (k1 + 2 * k2 + 2 * k3 + k4) / 6;
```

To validate the numerical implementation against the results obtained from the developed Thermal Equivalent Circuit Model (TECM), the function is plotted using a power loss input of 10,125 W. The resulting temperature progression is presented in Figure 2.54.

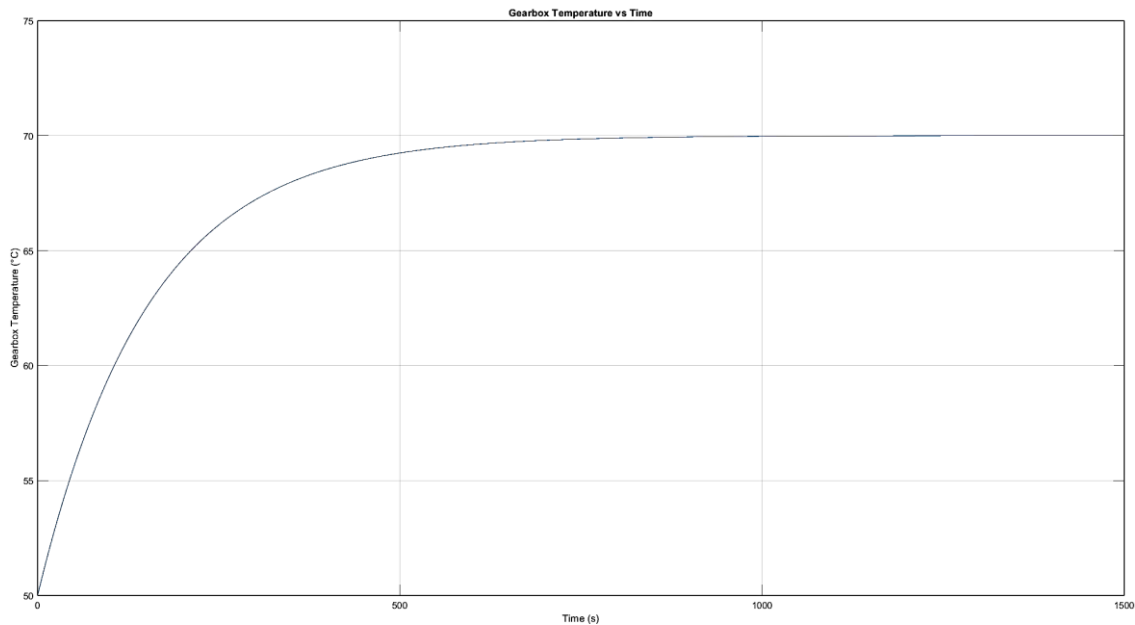


Figure 2.54: Temperature Progression Based on Numerical Implementation with Power Loss Input of 10,125 W

The temperature of the mathematical model stabilizes at approximately 70 °C, exhibiting the same dynamic behavior as observed in the TECM simulation. This confirms that the developed mathematical model accurately reflects the behavior of the simulated TECM.

The development of the C++ code includes two main components: the implementation of a standalone interpolation method and the translation of the MATLAB-based thermal model into C++. The complete code will not be detailed in this document. The program utilizes three-dimensional linear interpolation to compute power losses and applies the 4th-order Runge-Kutta method to simulate the temperature evolution over time.

The workflow of the C++ code is as follows:

1. Load Dataset
2. Wait for user input to Torque.
3. Wait for user input to Shaft Speed
4. Wait for user input to Temperature.
5. Interpolate the system power loss.
6. Compute the temperature development and output the temperature at each second.

If any of the input values fall outside the valid range, the program will display an error message indicating that the input is out of bounds, as shown below:

```
if (interp == 0.0) {  
    int k = 404;  
    printf("input(x, y, z) = (%f, %f, %f) which is outside look-up table --> therefore error %i\n", input[0], input[1], input[2], k);  
}
```

The boundary values are defined by the upper and lower limits of the lookup table, which are specified at the beginning of the code. These values can be modified by adjusting the boundary values and updating the CSV files used for the lookup table.

Upon first execution, the program will attempt to open the CSV file as follows:

```
int main(int argc, char** argv) {  
    FILE *file = fopen("Gearbox_interp.csv", "r");  
    if (file == NULL) {  
        fprintf(stderr, "Could not open file\n");  
        return 1;  
    }  
}
```

If the program fails to open the CSV file, it will report an error in the terminal. If the dataset (CSV file) is changed, the file name should also be updated accordingly in the code; otherwise, the program will continue to fail, as shown below:

```
float Lin_interp3(float input[3]) {  
    const float tableX[Nx] = {-502, -458, -414, -371, -327, -284, -240, -196, -153, -109, -65, -22, 22, 65, 109, 153, 196, 240, 284, 327, 371, 414, 458, 502}; // unique values of torque  
    const float tableY[Ny] = {1000, 2900, 4800, 6700, 8600, 10500}; // unique values of speed  
    const float tableZ[Nz] = {80, 90}; // unique values of temperature
```

In the initial lines of the interpolation code, the unique values for each of the aforementioned parameters must be defined. This means that if the dataset used for interpolation or the digital twin (DT) is modified, the values must be rechecked or adjusted to align with the new dataset, should they differ from the original.

The definitions for the interpolation parameters can be found at the beginning of the code (lines 10–20), as shown below:

```
10 int Nx = 24; // Amount of data with respect to torque
11 int Ny = 6; // Data with respect to Shaft Speed
12 int Nz = 2; // Data with respect to Shaft temp
13
14 int ROWS = 288; // Nx*Ny*Nz;
15 int COLS = 4; // Amount of columns in the look-up table
16
17 float Table[288][4];
18 float interp;
19 float input[3];
20 float Lin_interp3(float input[3]);
```

This includes the number of unique data points in each vector (Torque, Temperature, and Shaft Speed, in that order), the total number of unique combinations ($24 \times 6 \times 2 = 288$), and the number of columns in the lookup table. In this interpolation, an external lookup table is utilized, stored in a CSV file named "Gearbox_interp.csv."

In line 17, the dimensions of the lookup table are defined. Line 18 defines the float for the interpolation, while line 19 specifies the number of inputs. If the number of inputs changes, this value should be updated accordingly. Similarly, the value in line 20 should also be adjusted if necessary.

Line 48 checks whether the input values fall outside the lower and upper bounds of the lookup table and returns an error if this is the case. If the input values are within bounds, the code proceeds to search the lookup table for the closest points to the input values and interpolates the system losses based on the table (as shown in the subsequent part of the code):

```
184 int Diff() {
185
186
187     // Time step for the equation to run
188     float h = 1; // Time step in seconds
189     int num_steps = 1500; // Number of steps, increase to extend the simulation time.
190
191     // Run the simulation
192     vector<float> temperatures = runge_kutta(T0, T_amb, interp, C_th_tot, R_th, h, num_steps);
193
194     // Print the interpolated value
195     if (interp == 0.0) {
196         int k = 404;
197         printf("input(x, y, z) = (%f, %f, %f) which is outside look-up table --> therefore error %i\n", input[0], input[1], input[2], k);
198     } else {
199         printf("Interpolated value = %f\n", interp);
200     }
201
202     // Print the temperature values//// TO DISABLE THE CONSOLE OUTPUT, REMOVE THE FOLLOWING LOOP
203     for (int i = 0; i == num_steps; ++i) {
204         printf("Temperature: %f\n", temperatures[i]);
205     }
206
207     return 0;
208 }
```

The differential equation is used to calculate the temperature development within the system. By modifying the integer value in line 189, the simulation runtime can be adjusted accordingly. The code is configured in debug mode, where the terminal outputs the interpolated value and the temperature at each second of the specified runtime (in this case, 1500 seconds). This can be disabled by commenting out lines 203-205. To suppress the printed interpolation values, lines 195-200 should be disabled.

Initial parameters of the system are defined in lines 22-30 in the following section of the code:


```
22 //Parameters for the system
23 float T_amb = 50; // Ambient temperature (°C)
24 float T0 = 50; // Initial temperature condition (°C)
25 // Thermal capacitance constants
26 float C_th_Al = 40500; // Thermal capacitance of aluminium casing (J/°K) consisting of 45 kg of aluminium
27 float C_th_St = 36800; // Thermal capacitance of steel internals (J/°K) consisting of 80 kg of tool steel
28 float C_th_tot = C_th_St + C_th_Al; // Combined capacitance of gearbox materials (J/°K)
29 // Estimated Thermal resistance in the system (K/W)
30 float R_th = 0.00197555;
```

As specific values for these parameters have not been provided, assumptions were made in their determination. Despite this, the model demonstrates a reasonable level of accuracy when compared to the developed thermal equivalent model.

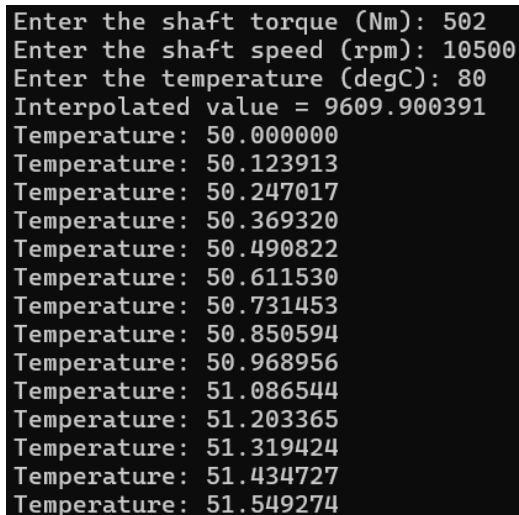
Upon execution, the program will open a terminal window and prompt the user for three input values: torque, shaft speed, and temperature.



```
C:\Users\jespe\OneDrive - Aa... x + v
Enter the shaft torque (Nm): 502
Enter the shaft speed (rpm): 10500
Enter the temperature (degC): 80
```

In future developments, with increased connectivity to sensors, this input method should be disabled and replaced with direct sensor connections.

Once the inputs are provided, the system will display the interpolated values, followed by the calculated temperatures, as shown below:



```
Enter the shaft torque (Nm): 502
Enter the shaft speed (rpm): 10500
Enter the temperature (degC): 80
Interpolated value = 9609.900391
Temperature: 50.000000
Temperature: 50.123913
Temperature: 50.247017
Temperature: 50.369320
Temperature: 50.490822
Temperature: 50.611530
Temperature: 50.731453
Temperature: 50.850594
Temperature: 50.968956
Temperature: 51.086544
Temperature: 51.203365
Temperature: 51.319424
Temperature: 51.434727
Temperature: 51.549274
```

The calculated temperatures will be displayed for each time step until the specified runtime duration is reached.

A sensitivity analysis of the gearbox digital twin was conducted to evaluate the system's behavior and investigate its boundaries. The tests involved varying one of the three input variables (torque, shaft speed, or temperature) along with the ambient temperature, while keeping the other parameters constant. After each test, the final temperature reading and the interpolated power loss were recorded. These results were then plotted in a combined plot. The analysis led to the following four tests:

Table 2.9: Data Sets for Power Losses and gearbox Temperature Response to Torque Variations

Test no.	Torque [Nm]	Temperature [°C]	Shaftspped [RPM]	Ambient temp [°C]	Power losses [W]	Temperature [°C]
1	62	80	8600	50	1308.93	52.59
2	124	80	8600	50	2118.33	54.18
3	187	80	8600	50	3021.56	55.97
4	249	80	8600	50	3955.36	57.81
5	312	80	8600	50	4953.51	59.79
6	374	80	8600	50	5964.78	61.78
7	437	80	8600	50	7033.57	63.89
8	499	80	8600	50	8093.81	65.99

Table 2.10: Data Sets for Power Losses and gearbox Temperature Response to internal Temperature Variations

Test no.	Torque [Nm]	Temperature [°C]	Shaftspped [RPM]	Ambient temp [°C]	Power losses [W]	Temperature [°C]
1	205	80	8600	50	3290.02	56.50
2	205	81	8600	50	3288.59	56.50
3	205	82	8600	50	3287.15	56.49
4	205	84	8600	50	3284.27	56.49
5	205	85	8600	50	3282.83	56.49
6	205	86	8600	50	3281.39	56.48
7	205	88	8600	50	3278.51	56.48
8	205	89	8600	50	3277.08	56.47

Table 2.11: Data Sets for Power Losses and gearbox Temperature Response to shaft speed Variations

Test no.	Torque [Nm]	Temperature [°C]	Shaftspped [RPM]	Ambient temp [°C]	Power losses [W]	Temperature [°C]
1	205	80	1000	50	488.79	50.97
2	205	80	2500	50	1063.02	52.10
3	205	80	4000	50	1581.91	53.12
4	205	80	5500	50	2105.48	54.16
5	205	80	7000	50	2664.33	55.26
6	205	80	8500	50	3250.92	56.42
7	205	80	10000	50	3732.14	57.37
8	205	80	10500	50	3890.04	57.68

Table 2.12: Data Sets for Power Losses and gearbox Temperature Response to ambient Temperature Variations

Test no.	Torque [Nm]	Temperature [°C]	Shaftspped [RPM]	Ambient temp [°C]	Power losses [W]	Temperature [°C]
1	205	80	8600	-30	3290.02	-23.50
2	205	80	8600	-15	3290.02	-8.50

3	205	80	8600	0	3290.02	6.50
4	205	80	8600	15	3290.02	21.50
5	205	80	8600	30	3290.02	36.50
6	205	80	8600	45	3290.02	51.50
7	205	80	8600	60	3290.02	66.50
8	205	80	8600	75	3290.02	81.50

Figures 2.55 to 2.58 depict the results of the sensitivity analysis conducted to evaluate the gearbox digital twin's behavior and investigate system boundaries. Figure 2.55 shows the results of the sensitivity analysis for power losses and gearbox temperature in response to torque variations. Figure 2.56 illustrates the power losses and gearbox temperature response to internal temperature variations. Figure 2.57 highlights the power losses and gearbox temperature response to shaft speed variations. Finally, Figure 2.58 demonstrates the power losses and gearbox temperature response to ambient temperature variations. These plots provide insights into the linearity of the system's response under varying conditions.

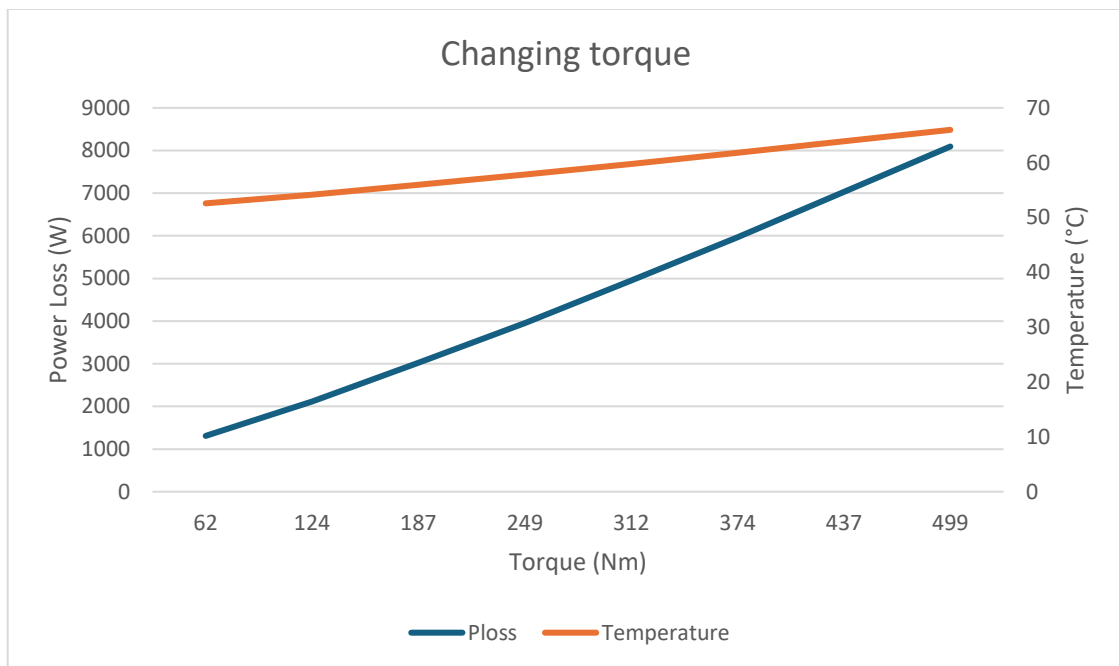


Figure 2.55: Power Losses and gearbox Temperature Response to Torque Variations

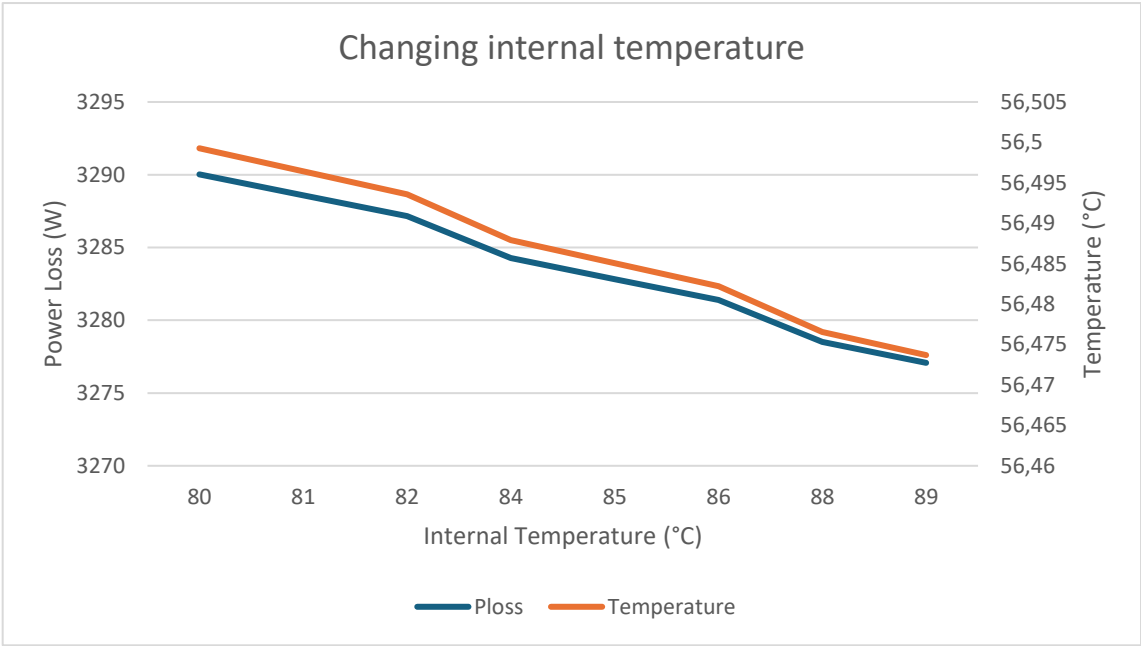


Figure 2.56: Power Losses and gearbox Temperature Response to internal Temperature Variations

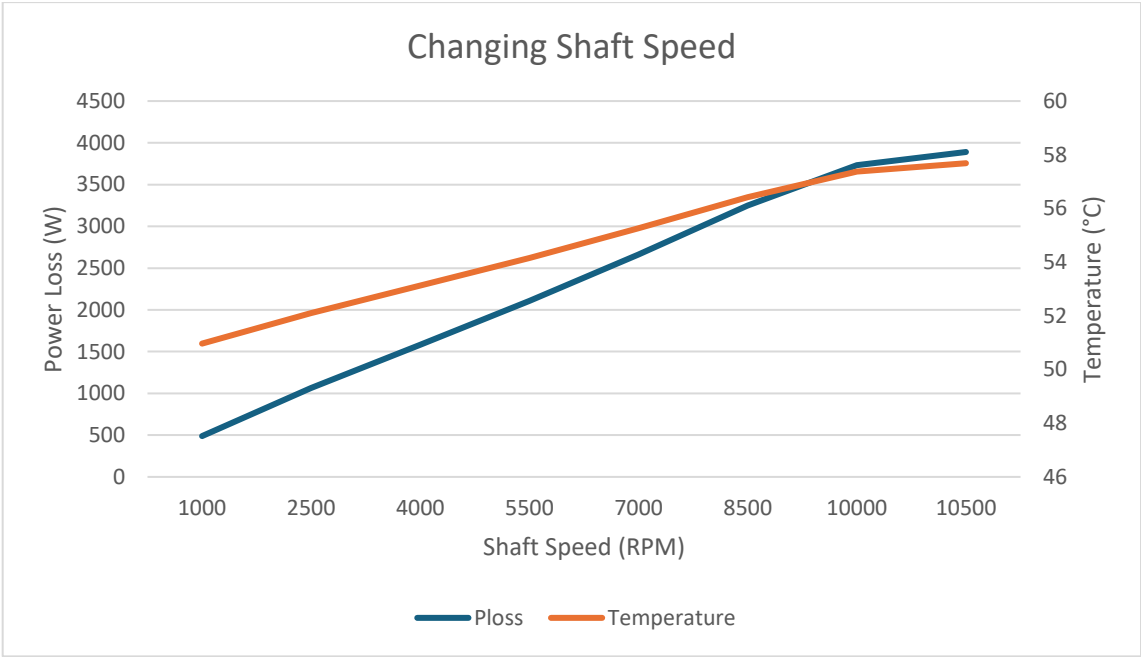


Figure 2.57: Power Losses and gearbox Temperature Response to shaft speed Variations

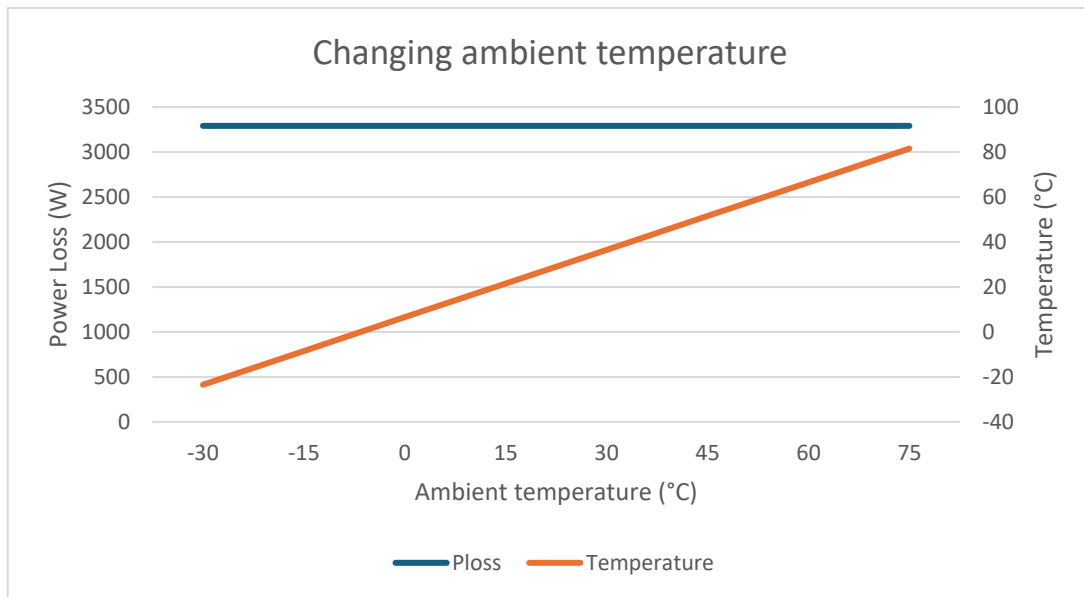


Figure 2.58: Power Losses and gearbox Temperature Response to ambient Temperature Variations

From all the tests, we observe a good linear relationship between both power loss and temperature as the input variables change. However, when adjusting the internal temperature of the gearbox, we find that this leads to a decrease in both power loss and temperature. Despite this, the linearity remains. When changing the ambient temperature, we notice that it does not affect the power loss, but it does influence the final temperature. While the resulting plots appear perfectly linear, this is a consequence of the varying step sizes used in the tests.

As discussed, the three digital twins developed in Task 4.3 of WP4 in the RHODAS project are summarized in Figure 2.59. This figure illustrates the structured inputs and outputs for each of the digital twins, which include the inverter, e-motor, and gearbox models.

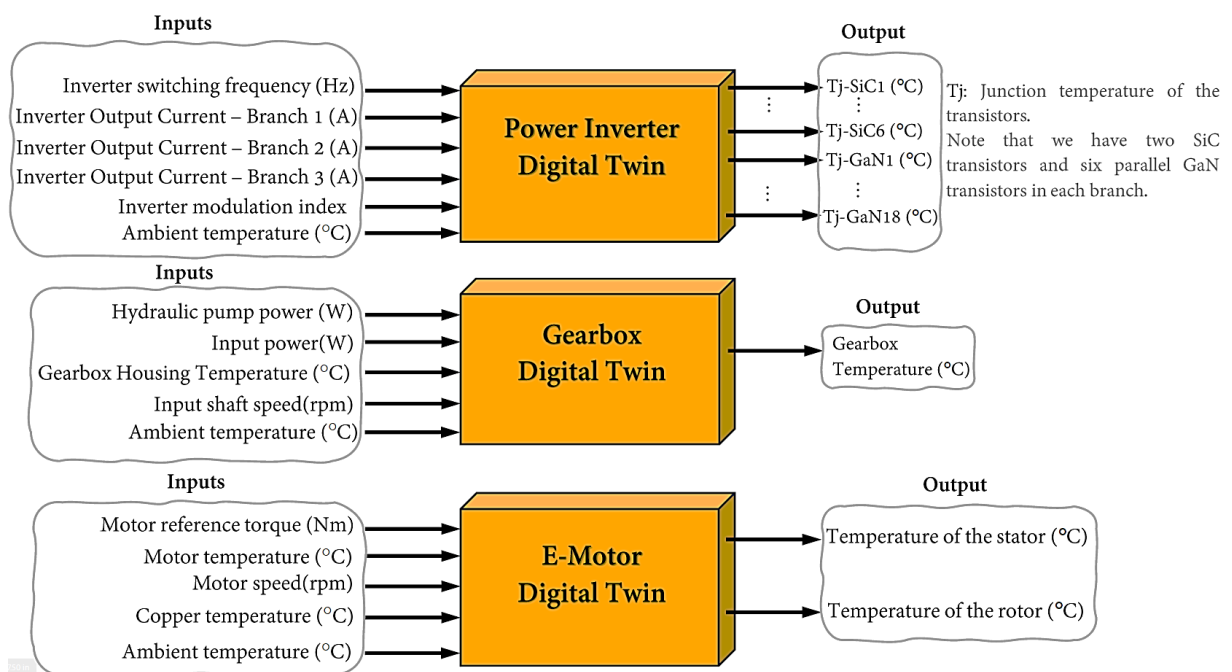


Figure 2.59: Overview of Inputs and Outputs for the Inverter, E-Motor, and Gearbox Digital Twins

3 CONCLUSION

The digital twin developments presented in this document mark a step forward in enhancing the design, analysis and operational efficiency of the RHODAS electromechanical powertrain. By focusing on three key subsystems—the T-type inverter, e-Motor, and gearbox—Task 4.3 has delivered a modular and scalable digital twin framework capable of simulating thermal behavior, predicting performance issues, and supporting virtual testing under realistic operating conditions. Through the integration of power loss modeling, thermal equivalent circuit approaches and numerical methods, each digital twin provides a virtual representation of its respective physical counterpart. These models not only support real-time condition monitoring and predictive maintenance but also lay the foundation for future system-level co-simulation and control strategies.

The results achieved within this task can contribute to the broader RHODAS vision of creating a more efficient, reliable and intelligent powertrain platform. The methodologies and tools developed here will support the ongoing integration, validation and optimization of the complete RHODAS system in subsequent project phases.

REFERENCES

1. Al Sakka, Monzer, et al. "Thermal modeling and heat management of supercapacitor modules for vehicle applications." *Journal of Power Sources* 194.2 (2009): 581-587.

Appendix A – SiC-CAB450M12XM3 from Wolfspeed Cree

CAB450M12XM3 1200V, 450A All-Silicon Carbide Conduction Optimized, Half-Bridge Module

V_{DS}	1200 V
I_{DS}	450 A

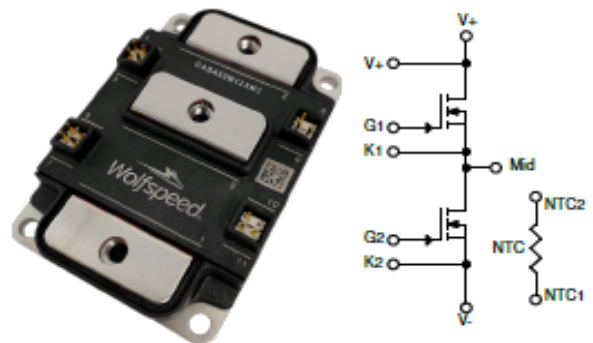
Technical Features

- High Power Density Footprint
- High Junction Temperature (175 °C) Operation
- Low Inductance (6.7 nH) Design
- Implements Conduction Optimized Third Generation SiC MOSFET Technology
- Silicon Nitride Insulator and Copper Baseplate

Applications

- Motor & Traction Drives
- Vehicle Fast Chargers
- Uninterruptable Power Supplies
- Smart-Grid / Grid-Tied Distributed Generation

Package 80 x 53 x 19 mm



System Benefits

- Terminal layout allows for direct bus bar connection without bends or bushings enabling a simple, low inductance design.
- Isolated integrated temperature sensing enables high-level temperature protection.
- Dedicated drain Kelvin pin enables direct voltage sensing for gate driver overcurrent protection.

Key Parameters ($T_c = 25^\circ\text{C}$ unless otherwise specified)

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions	Note
$V_{DS\max}$	Drain-Source Voltage			1200			
$V_{GS\max}$	Gate-Source Voltage, Maximum Value	-4		+19	V	AC frequency $\geq 1\text{Hz}$	Note 1
$V_{GS\text{op}}$	Gate-Source Voltage, Recommended Op. Value	-4		+15		Static	
I_{DS}	DC Continuous Drain Current			450		$V_{GS} = 15\text{ V}$, $T_c = 25^\circ\text{C}$, $T_{vj} \leq 175^\circ\text{C}$	Fig. 20
			409			$V_{GS} = 15\text{ V}$, $T_c = 90^\circ\text{C}$, $T_{vj} \leq 175^\circ\text{C}$	Note 2
I_{SD}	DC Source-Drain Current			450	A	$V_{GS} = 15\text{ V}$, $T_c = 25^\circ\text{C}$, $T_{vj} \leq 175^\circ\text{C}$	
$I_{SD\text{BD}}$	DC Source-Drain Current (Body Diode)		225			$V_{GS} = -4\text{ V}$, $T_c = 25^\circ\text{C}$, $T_{vj} \leq 175^\circ\text{C}$	
$I_{DS(\text{pulsed})}$	Maximum Pulsed Drain-Source Current			900		t_{pmax} limited by $T_{j\text{max}}$	
$I_{SD(\text{pulsed})}$	Maximum Pulsed Source-Drain Current			900		$V_{GS} = 15\text{ V}$, $T_c = 25^\circ\text{C}$	
$T_{vj\text{op}}$	Maximum Virtual Junction Temperature under Switching Conditions	-40		175	$^\circ\text{C}$		

Note 1 If MOSFET body diode is not used, $V_{GS\max} = -8/+19\text{ V}$

Note 2 Assumes $R_{thJC} = 0.11^\circ\text{C/W}$ and $R_{DS(on)} = 4.6\text{ m}\Omega$. Calculate $P_D = (T_{vj} - T_c) / R_{thJC}$. Calculate $I_{D,\text{MAX}} = \sqrt{P_D / R_{DS(on)}}$

MOSFET Characteristics (Per Position) ($T_c = 25^\circ\text{C}$ unless otherwise specified)

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions	Note
$V_{DS(BOSS)}$	Drain-Source Breakdown Voltage	1200				$V_{GS} = 0\text{ V}, I_D = 200\text{ }\mu\text{A}$	
$V_{GS(th)}$	Gate Threshold Voltage	1.8	2.5	3.6	V	$V_{DS} = V_{GS}, I_D = 132\text{ mA}$	
			2.0			$V_{DS} = V_{GS}, I_D = 132\text{ mA}, T_J = 175^\circ\text{C}$	
I_{GSS}	Zero Gate Voltage Drain Current		5	200	μA	$V_{GS} = 0\text{ V}, V_{DS} = 1200\text{ V}$	
I_{GSS}	Gate-Source Leakage Current		0.05	1.3		$V_{GS} = 15\text{ V}, V_{DS} = 0\text{ V}$	
$R_{DS(on)}$	Drain-Source On-State Resistance (Devices Only)		2.6	3.7	m Ω	$V_{GS} = 15\text{ V}, I_D = 450\text{ A}$	Fig. 2 Fig. 3
			4.6			$V_{GS} = 15\text{ V}, I_D = 450\text{ A}, T_J = 175^\circ\text{C}$	
g_{fs}	Transconductance		355		S	$V_{DS} = 20\text{ V}, I_{DS} = 450\text{ A}$	Fig. 4
			360			$V_{DS} = 20\text{ V}, I_{DS} = 450\text{ A}, T_J = 175^\circ\text{C}$	
E_{on}	Turn-On Switching Energy, $T_J = 25^\circ\text{C}$ $T_J = 125^\circ\text{C}$ $T_J = 175^\circ\text{C}$		11.0 11.7 13.0		mJ	$V_{DS} = 600\text{ V},$ $I_D = 450\text{ A},$ $V_{GS} = -4\text{ V}/15\text{ V},$ $R_{G(int)} = 0.0\text{ }\Omega,$ $L = 13.6\text{ }\mu\text{H}$	Fig. 11 Fig. 13
E_{off}	Turn-Off Switching Energy, $T_J = 25^\circ\text{C}$ $T_J = 125^\circ\text{C}$ $T_J = 175^\circ\text{C}$		10.1 11.3 12.1				
$R_{G(int)}$	Internal Gate Resistance		2.5		Ω		
C_{iss}	Input Capacitance		38.0		nF	$V_{GS} = 0\text{ V}, V_{DS} = 800\text{ V},$ $V_{AC} = 25\text{ mV}, f = 100\text{ kHz}$	Fig. 9
C_{oss}	Output Capacitance		1.5				
C_{rss}	Reverse Transfer Capacitance		90		pF		
Q_{GS}	Gate to Source Charge		355		nC	$V_{DS} = 800\text{ V}, V_{GS} = -4\text{ V}/15\text{ V}$ $I_D = 450\text{ A}$ Per IEC60747-8-4 pg 21	
Q_{GD}	Gate to Drain Charge		500				
Q_G	Total Gate Charge		1330				
R_{thJC}	FET Thermal Resistance, Junction to Case		0.11	0.13	$^\circ\text{C/W}$		Fig. 17

Body Diode Characteristics (Per Position) ($T_c = 25^\circ\text{C}$ unless otherwise specified)

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions	Note
V_{SD}	Body Diode Forward Voltage		4.7		V	$V_{GS} = -4\text{ V}, I_{SD} = 450\text{ A}$	Fig. 7
			4.2			$V_{GS} = -4\text{ V}, I_{SD} = 450\text{ A}, T_J = 175^\circ\text{C}$	
t_r	Reverse Recovery Time		52		ns	$V_{GS} = -4\text{ V}, I_{SD} = 450\text{ A}, V_R = 600\text{ V}$ $di/dt = 8\text{ A/ns}, T_J = 175^\circ\text{C}$	
Q_{rr}	Reverse Recovery Charge		6.6		μC		
I_{rr}	Peak Reverse Recovery Current		195		A		
E_{rr}	Reverse Recovery Energy $T_J = 25^\circ\text{C}$ $T_J = 125^\circ\text{C}$ $T_J = 175^\circ\text{C}$		0.2 1.1 1.9		mJ	$V_{DS} = 600\text{ V}, I_D = 450\text{ A},$ $V_{GS} = -4\text{ V}/15\text{ V}, R_{G(int)} = 0.0\text{ }\Omega,$ $L = 13.6\text{ }\mu\text{H}$	Fig. 14

Temperature Sensor (NTC) Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
R_{25}	Rated Resistance		4.7		k Ω	$T_{NTC} = 25^{\circ}\text{C}$
$\Delta R/R$	Tolerance of R_{25}			± 1	%	
P_{25}	Maximum Power Dissipation			50	mW	

Steinhart-Hart Modified Coefficients for R/T Computation: $\frac{1}{T} = A + B \times \ln\left(\frac{R}{R_{25}}\right) + C \times \ln^2\left(\frac{R}{R_{25}}\right) + D \times \ln^3\left(\frac{R}{R_{25}}\right)$

	A	B	C	D
$T_{NTC} < 25^{\circ}\text{C}$	3.3540E-03	3.0013E-04	5.0852E-06	2.1877E-07
$T_{NTC} \geq 25^{\circ}\text{C}$	3.3540E-03	3.0013E-04	5.0852E-06	2.1877E-07

Module Physical Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
R_{S1}	Package Resistance, M1		0.72		m Ω	$T_c = 125^{\circ}\text{C}$, Note 3
R_{S2}	Package Resistance, M2		0.63			$T_c = 125^{\circ}\text{C}$, Note 3
L_{stray}	Stray Inductance		6.7		nH	Between Terminals 2 and 3
T_c	Case Temperature	-40		125	$^{\circ}\text{C}$	
W	Weight		175		g	
M_s	Mounting Torque	2.0	3.0	4.0	N-m	Baseplate, M4 bolts
		2.0	4.0	5.0		Power Terminals, M5 bolts
V_{insul}	Case Isolation Voltage	4.0			kV	AC, 50 Hz, 1 min
CTI	Comparative Tracking Index	600				
	Clearance Distance	12.5			mm	From 2 to 3, Note 4
		11.5				From 1 to Baseplate, Note 4
		5.7				From 2 to 5, Note 4
		13.7				From 5 to Baseplate, Note 4
	Creepage Distance	14.7				From 2 to 3, Note 4
		14.0				From 1 to Baseplate, Note 4
		14.7				From 2 to 5, Note 4
		14.3				From 5 to Baseplate, Note 4

Note 3 Total Effective Resistance (Per Switch Position) = MOSFET $R_{\text{DS(on)}}$ + Switch Position Package Resistance.

Note 4 Numbers reference the connections from the Schematic and Package Dimensions sections of this document.

Typical Performance

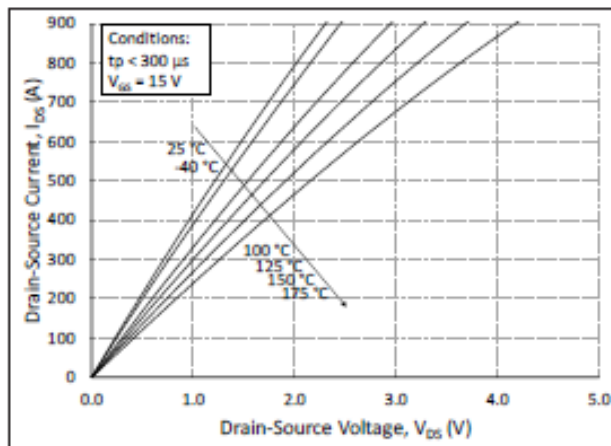


Figure 1. Output Characteristics for Various Junction Temperatures

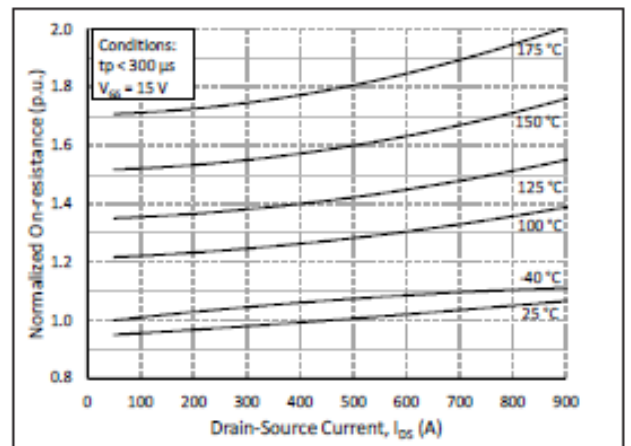


Figure 2. Normalized On-State Resistance vs. Drain Current for Various Junction Temperatures

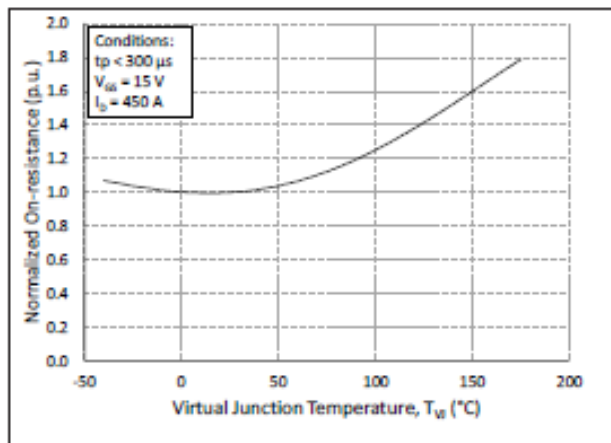


Figure 3. Normalized On-State Resistance vs. Junction Temperature

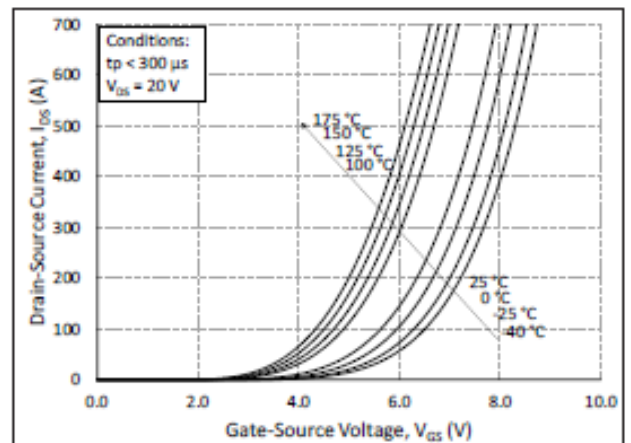


Figure 4. Transfer Characteristic for Various Junction Temperatures

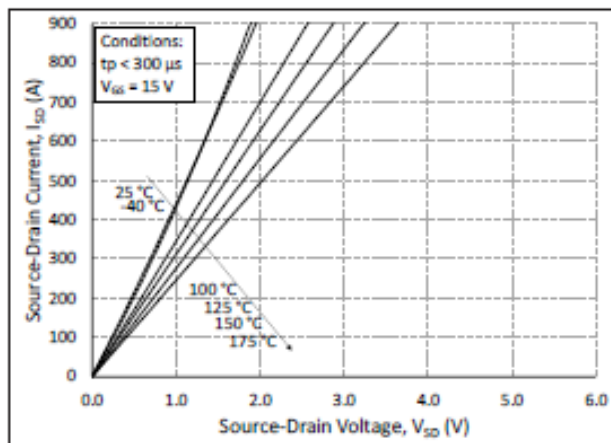


Figure 5. 3rd Quadrant Characteristic vs. Junction Temperatures at $V_{GS} = 15$ V

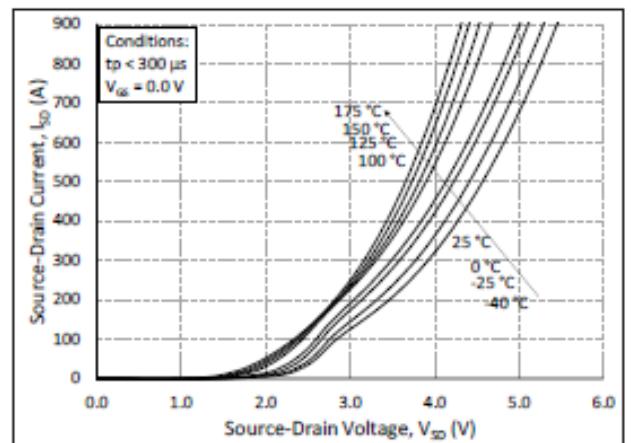


Figure 6. 3rd Quadrant Characteristic vs. Junction Temperatures at $V_{GS} = 0$ V (Body Diode)

Typical Performance

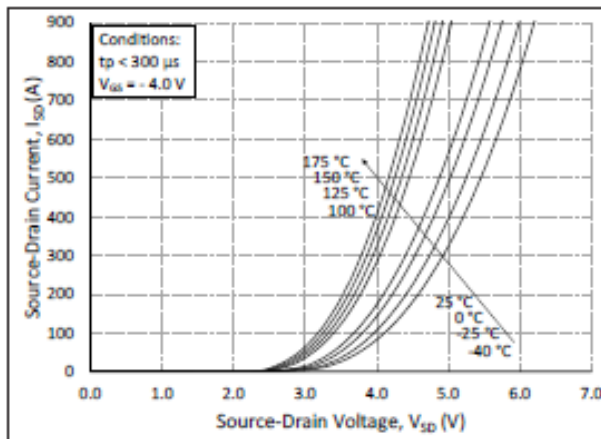


Figure 7. 3rd Quadrant Characteristic vs. Junction Temperatures at $V_{GS} = -4$ V (Body Diode)

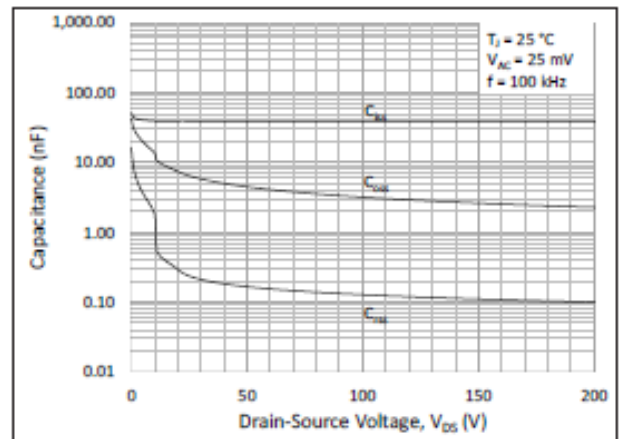


Figure 8. Typical Capacitances vs. Drain to Source Voltage (0 - 200V)

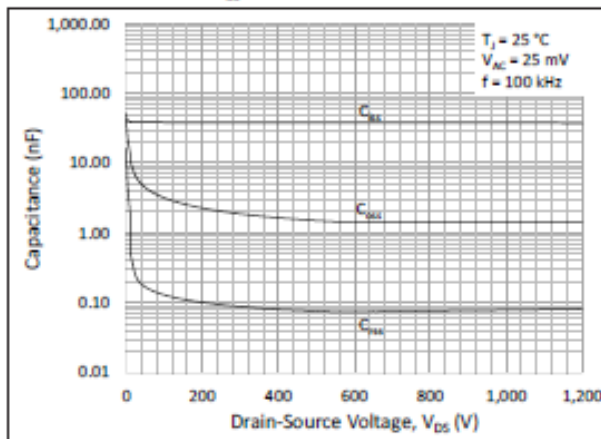


Figure 9. Typical Capacitances vs. Drain to Source Voltage (0 - 1200V)

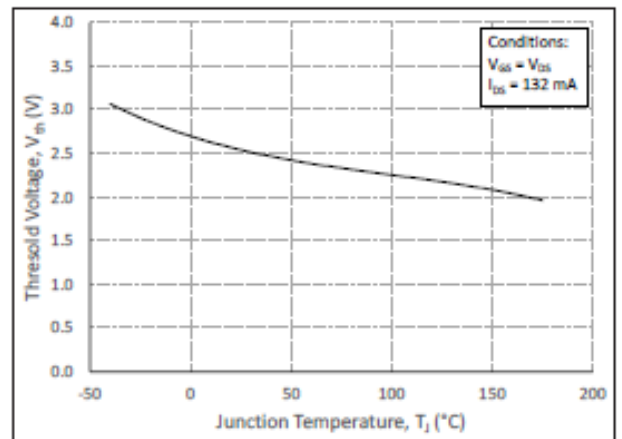


Figure 10. Threshold Voltage vs. Junction Temperature

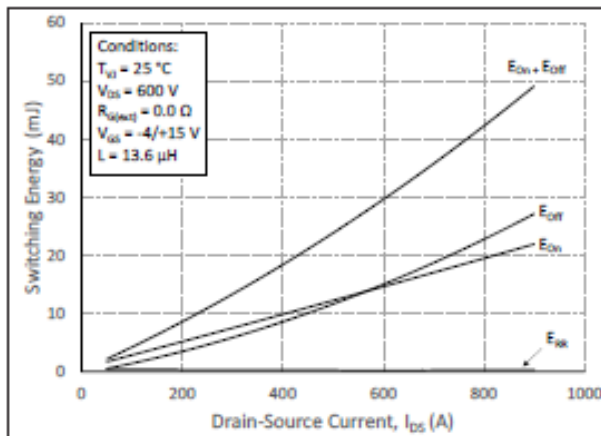


Figure 11. Switching Energy vs. Drain Current ($V_{DS} = 600$ V)

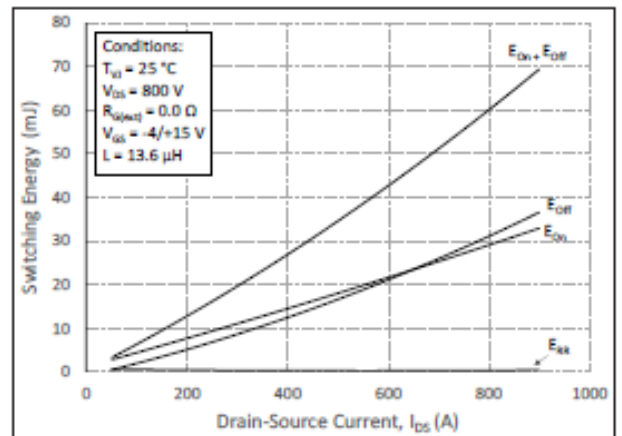


Figure 12. Switching Energy vs. Drain Current ($V_{DS} = 800$ V)

Typical Performance

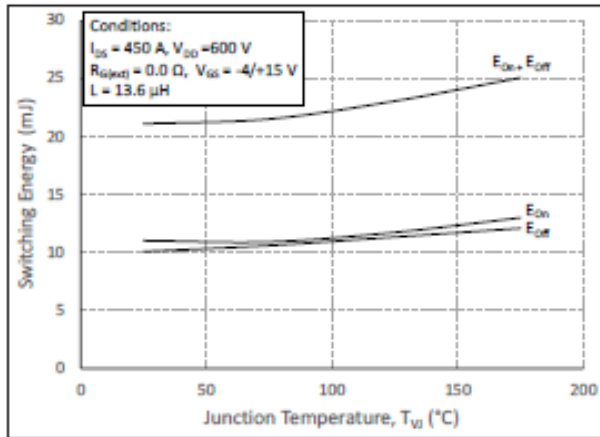


Figure 13. MOSFET Switching Energy vs. Junction Temperature

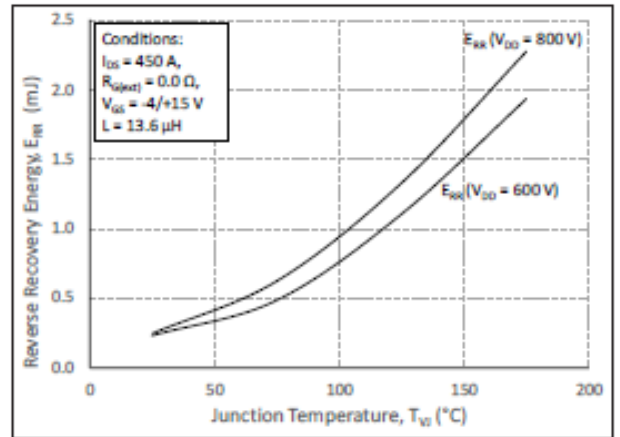


Figure 14. Reverse Recovery Energy vs. Junction Temperature

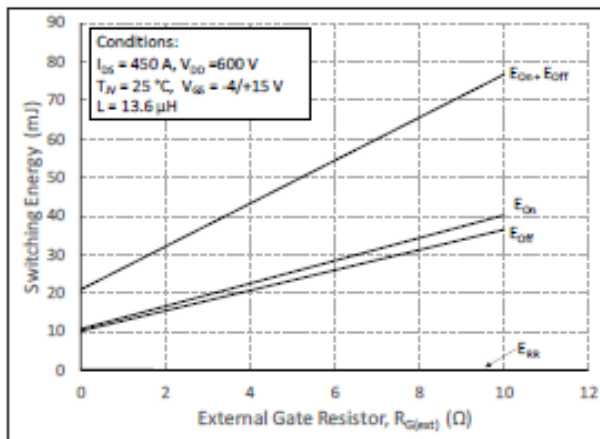


Figure 15. MOSFET Switching Energy vs. External Gate Resistance

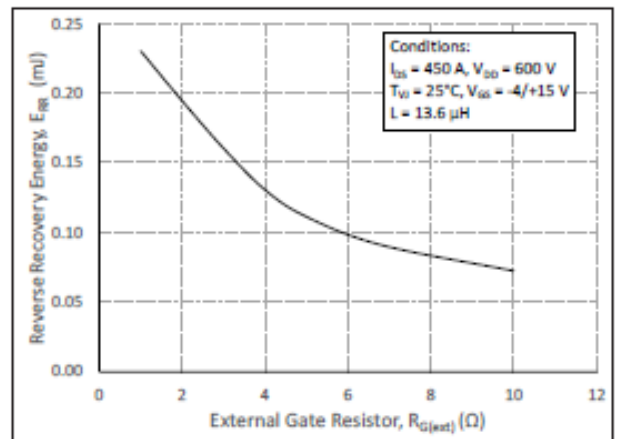


Figure 16. Reverse Recovery Energy vs. External Gate Resistance

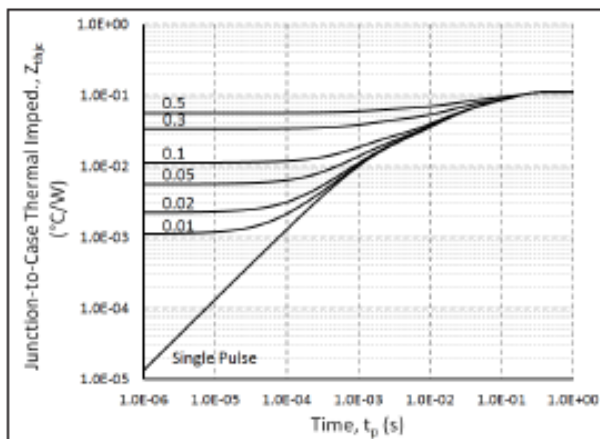


Figure 17. MOSFET Junction to Case Transient Thermal Impedance, $Z_{\theta JA}$ (°C/W)

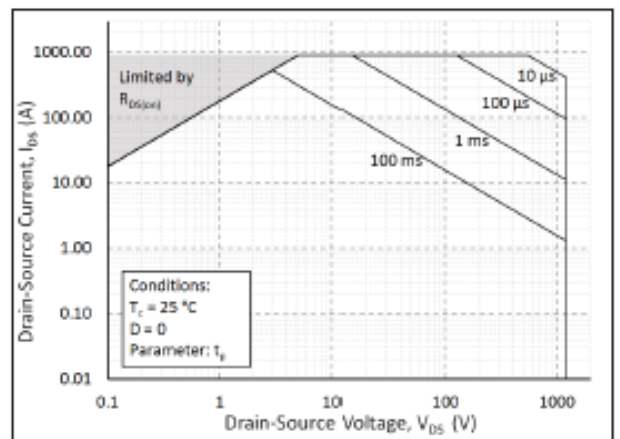


Figure 18. Forward Bias Safe Operating Area (FBSOA)

Typical Performance

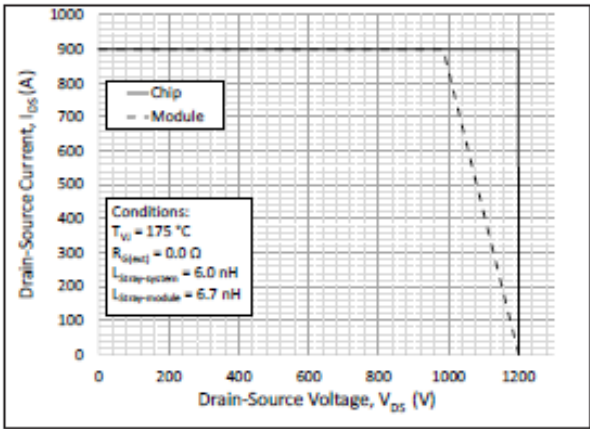


Figure 19. Reverse Bias Safe Operating Area (RBSOA)

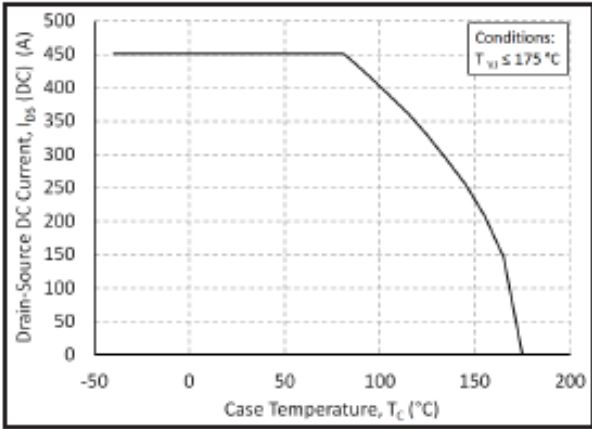


Figure 20. Continuous Drain Current Derating vs. Case Temperature

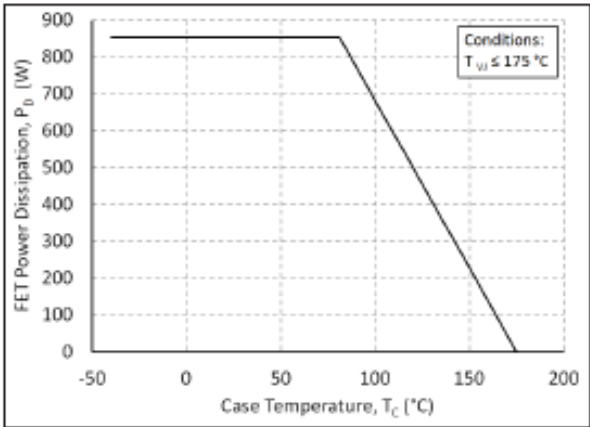


Figure 21. Maximum Power Dissipation Derating vs. Case Temperature

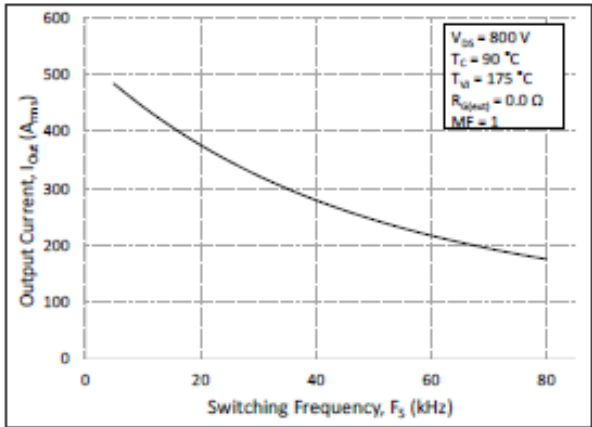
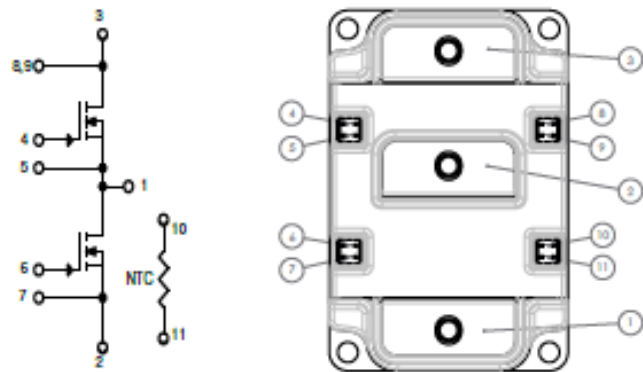
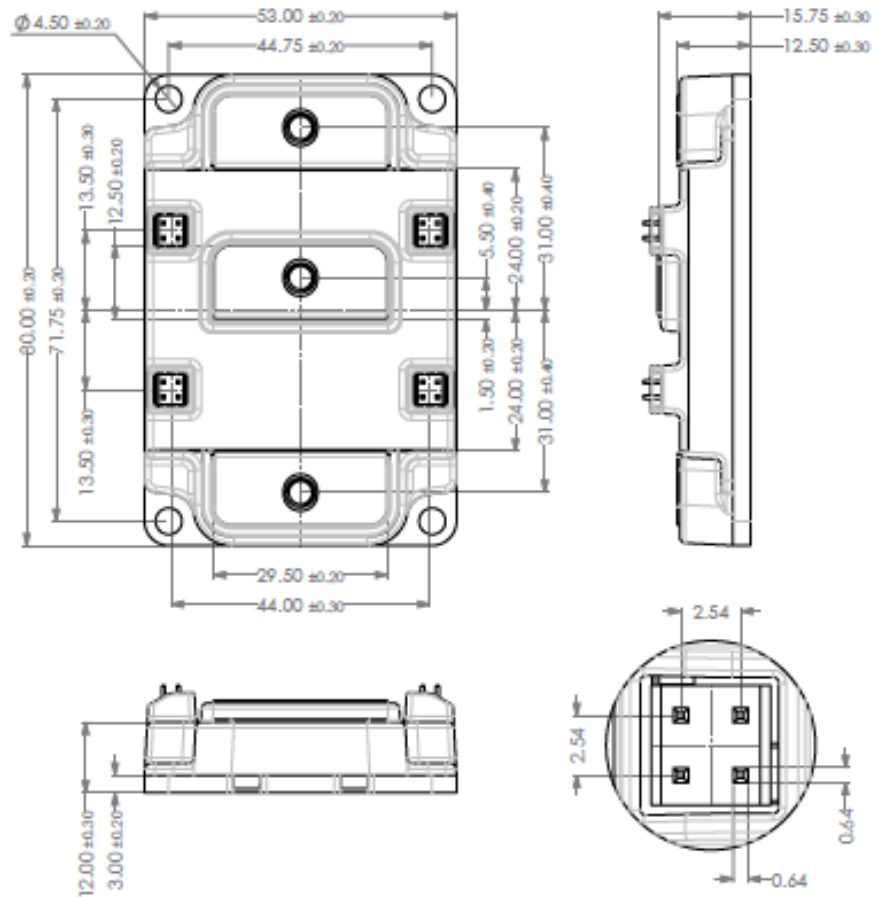


Figure 22. Typical Output Current Capability vs. Switching Frequency (Inverter Application)

Schematic and Pin Out

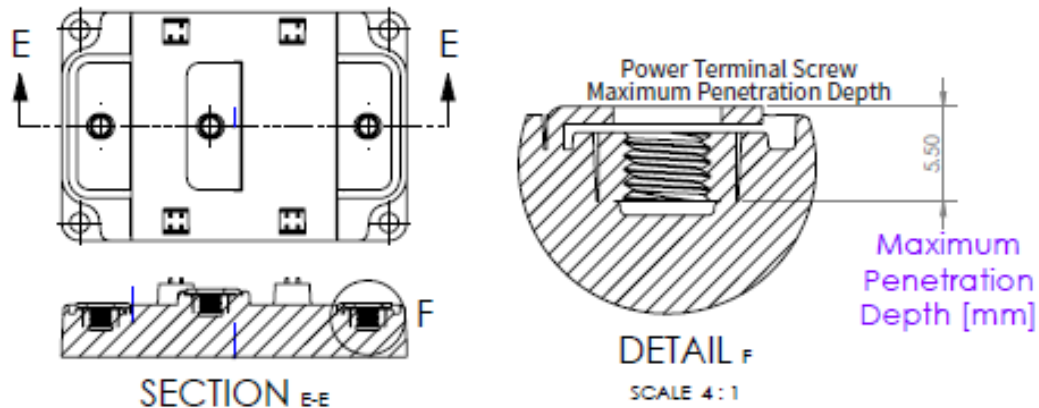


Package Dimmension (mm)





Package Dimension (mm)



Supporting Links & Tools

- [CGD12HBXMP: XM3 Evaluation Gate Driver](#)
- [CGD12HB00D: Differential Transceiver Board for CGD12HBXMP](#)
- [CRD300DA12E-XM3: 300 kW Inverter Kit for Conduction-Optimized XM3 \(CPWR-AN30\)](#)
- [KIT-CRD-CIL12N-XM3: Dynamic Performance Evaluation Board for the XM3 Module \(CPWR-AN31\)](#)
- [CPWR-AN28: Module Mounting Application Note](#)
- [CPWR-AN29: Thermal Interface Material Application Note](#)

Notes

- This product has not been designed or tested for use in, and is not intended for use in, applications implanted into the human body nor in applications in which failure of the product could lead to death, personal injury or property damage, including but not limited to equipment used in the operation of nuclear facilities, life-support machines, cardiac defibrillators or similar emergency medical equipment, aircraft navigation or communication or control systems, or air traffic control systems.
- The SiC MOSFET module switches at speeds beyond what is customarily associated with IGBT-based modules. Therefore, special precautions are required to realize optimal performance. The interconnection between the gate driver and module housing needs to be as short as possible. This will afford optimal switching time and avoid the potential for device oscillation. Also, great care is required to insure minimum inductance between the module and DC link capacitors to avoid excessive VDS overshoot.

Rev. A, 2019-06-01 CAB450M12XM3 4600 Silicon Dr., Durham, NC 27703

Copyright ©2019 Cree, Inc. All rights reserved. The information in this document is subject to change without notice. Cree®, the Cree logo, Wolfspeed®, and the Wolfspeed logo are registered trademarks of Cree, Inc.

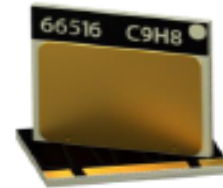
Appendix B – GaN-GS66516T from GAN systems



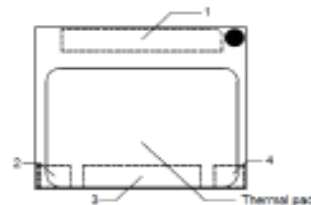
GS66516T Top-side cooled 650 V E-mode GaN transistor Datasheet

Features

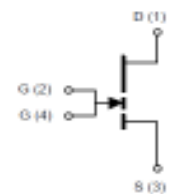
- 650 V enhancement mode power transistor
- Top-side cooled configuration
- $R_{DS(on)} = 25 \text{ m}\Omega$
- $I_{DS(max)} = 60 \text{ A}$
- Ultra-low FOM die
- Low inductance GaNPX® package
- Simple drive requirements (0 V to 6 V)
- Transient tolerant gate drive (-20 / +10 V)
- Very high switching frequency (> 10 MHz)
- Fast and controllable fall and rise times
- Reverse current capability
- Zero reverse recovery loss
- Small 9.0 x 7.6 mm² PCB footprint
- Dual gate pads for optimal board layout
- RoHS 3 (6+4) compliant



Package Outline



Circuit Symbol



The thermal pad is internally connected to Source (S pin 3) and substrate

Applications

- AC-DC Converters
- DC-DC Converters
- Bridgeless Totem Pole PFC
- Inverters
- Energy Storage Systems
- On Board Battery Chargers
- Uninterruptable Power Supplies
- Solar Energy
- Industrial Motor Drives
- Laser Drivers
- Traction Drive
- Wireless Power Transfer

Description

The GS66516T is an enhancement mode GaN-on-silicon power transistor. The properties of GaN allow for high current, high voltage breakdown and high switching frequency. GaN Systems innovates with industry leading advancements such as patented **Island Technology®** and **GaNPX®** packaging. **Island Technology®** cell layout realizes high-current die and high yield. **GaNPX®** packaging enables low inductance & low thermal resistance in a small package. The GS66516T is a top-side cooled transistor that offers very low junction-to-case thermal resistance for demanding high power applications. These features combine to provide very high efficiency power switching.



GS66516T
Top-side cooled 650 V E-mode GaN transistor
Datasheet

Absolute Maximum Ratings ($T_{case} = 25\text{ }^{\circ}\text{C}$ except as noted)

Parameter	Symbol	Value	Unit
Operating Junction Temperature	T_J	-55 to +150	$^{\circ}\text{C}$
Storage Temperature Range	T_S	-55 to +150	$^{\circ}\text{C}$
Drain-to-Source Voltage	V_{DS}	650	V
Transient Drain-to-Source Voltage (Note 1)	$V_{DS(transient)}$	750	V
Gate-to-Source Voltage	V_{GS}	-10 to +7	V
Gate-to-Source Voltage - transient (Note 1)	$V_{GS(transient)}$	-20 to +10	V
Continuous Drain Current ($T_{case}=25\text{ }^{\circ}\text{C}$)	I_{DS}	60	A
Continuous Drain Current ($T_{case}=100\text{ }^{\circ}\text{C}$)	I_{DS}	47	A
Pulse Drain Current (Pulse width 50 μs , $V_{GS} = 6\text{ V}$) (Note 2)	$I_{DS\text{ Pulse}}$	120	A

(1) For $\leq 1\text{ }\mu\text{s}$

(2) Defined by product design and characterization. Value is not tested to full current in production.

Thermal Characteristics (Typical values unless otherwise noted)

Parameter	Symbol	Value	Units
Thermal Resistance (junction-to-case) – top side	$R_{\theta JC}$	0.27	$^{\circ}\text{C}/\text{W}$
Maximum Soldering Temperature (MSL3 rated)	T_{SOLD}	260	$^{\circ}\text{C}$

Ordering Information

Ordering code	Package type	Packing method	Qty	Reel Diameter	Reel Width
GS66516T-TR	GaNPX® Top-Side Cooled	Tape-and-Reel	3000	13" (330mm)	16mm
GS66516T-MR	GaNPX® Top-Side Cooled	Mini-Reel	250	7" (180 mm)	16mm



GS66516T
Top-side cooled 650 V E-mode GaN transistor
Datasheet

Electrical Characteristics (Typical values at $T_J = 25\text{ }^{\circ}\text{C}$, $V_{GS} = 6\text{ V}$ unless otherwise noted)

Parameters	Sym.	Min.	Typ.	Max.	Units	Conditions
Drain-to-Source Blocking Voltage	$V_{(B)DSS}$	650			V	$V_{GS} = 0\text{ V}$, $I_{DSS} = 100\text{ }\mu\text{A}$
Drain-to-Source On Resistance	$R_{DS(on)}$		25	32	m Ω	$V_{GS} = 6\text{ V}$, $T_J = 25\text{ }^{\circ}\text{C}$ $I_{DS} = 18\text{ A}$
Drain-to-Source On Resistance	$R_{DS(on)}$		65		m Ω	$V_{GS} = 6\text{ V}$, $T_J = 150\text{ }^{\circ}\text{C}$ $I_{DS} = 18\text{ A}$
Gate-to-Source Threshold	$V_{GS(th)}$	1.1	1.7	2.6	V	$V_{DS} = V_{GS}$ $I_{DS} = 14\text{ mA}$
Gate-to-Source Current	I_{GS}		320		μA	$V_{GS} = 6\text{ V}$, $V_{DS} = 0\text{ V}$
Gate Plateau Voltage	V_{plat}		3.0		V	$V_{DS} = 400\text{ V}$, $I_{DS} = 60\text{ A}$
Drain-to-Source Leakage Current	I_{DSS}		4	100	μA	$V_{DS} = 650\text{ V}$ $V_{GS} = 0\text{ V}$, $T_J = 25\text{ }^{\circ}\text{C}$
Drain-to-Source Leakage Current	I_{DSS}		800		μA	$V_{DS} = 650\text{ V}$ $V_{GS} = 0\text{ V}$, $T_J = 150\text{ }^{\circ}\text{C}$
Internal Gate Resistance	R_G		0.3		Ω	$f = 5\text{ MHz}$, open drain
Input Capacitance	C_{iss}		518		pF	$V_{DS} = 400\text{ V}$ $V_{GS} = 0\text{ V}$ $f = 100\text{ kHz}$
Output Capacitance	C_{oss}		126		pF	
Reverse Transfer Capacitance	C_{rss}		5.9		pF	
Effective Output Capacitance Energy Related (Note 3)	$C_{O(ER)}$		207		pF	$V_{GS} = 0\text{ V}$ $V_{DS} = 0\text{ to }400\text{ V}$
Effective Output Capacitance Time Related (Note 4)	$C_{O(TR)}$		335		pF	
Total Gate Charge	Q_G		14.2		nC	$V_{GS} = 0\text{ to }6\text{ V}$ $V_{DS} = 400\text{ V}$
Gate-to-Source Charge	Q_{GS}		3.8		nC	
Gate-to-Drain Charge	Q_{GD}		5.4		nC	
Output Charge	Q_{OSS}		134		nC	$V_{GS} = 0\text{ V}$, $V_{DS} = 400\text{ V}$
Reverse Recovery Charge	Q_{RR}		0		nC	

(3) $C_{O(ER)}$ is the fixed capacitance that would give the same stored energy as C_{OSS} while V_{DS} is rising from 0 V to the stated V_{DS}

(4) $C_{O(TR)}$ is the fixed capacitance that would give the same charging time as C_{OSS} while V_{DS} is rising from 0 V to the stated V_{DS} .



GS66516T
Top-side cooled 650 V E-mode GaN transistor
Datasheet

Electrical Characteristics continued (Typical values at $T_J = 25^\circ\text{C}$, $V_{GS} = 6\text{ V}$ unless otherwise noted)

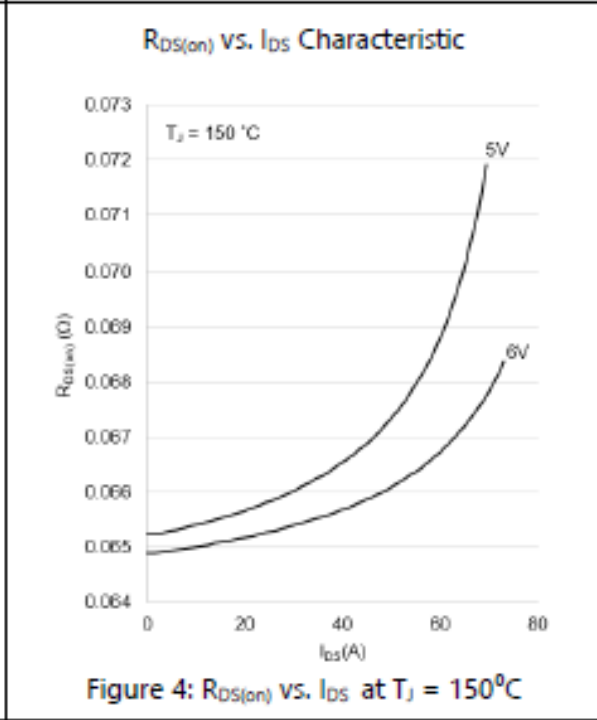
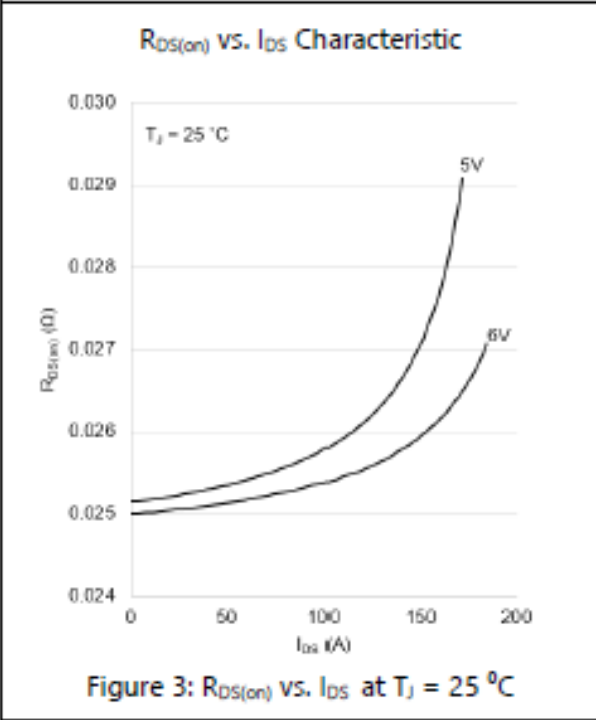
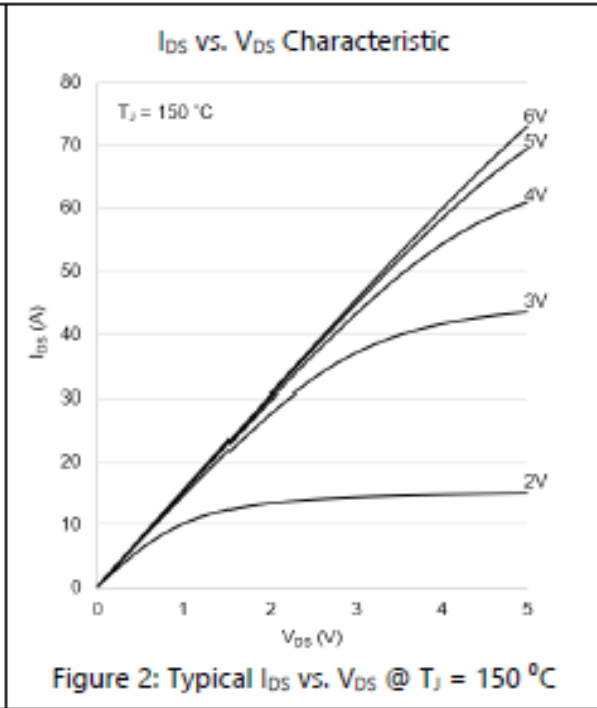
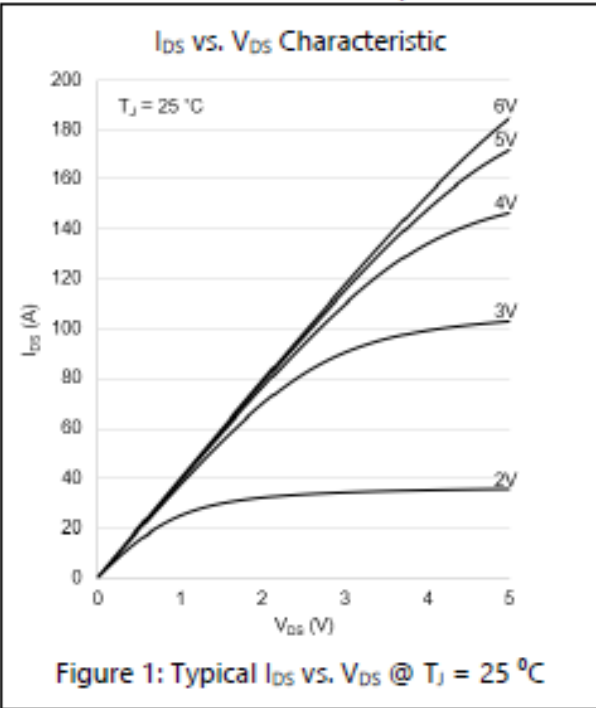
Parameters	Sym.	Min.	Typ.	Max.	Units	Conditions
Turn-On Delay	$t_{D(on)}$		4.6		ns	$V_{DD} = 400\text{ V}$ $V_{GS} = 0 - 6\text{ V}$ $I_D = 16\text{ A}$, $R_{G(ext)} = 5\ \Omega$ $T_J = 25^\circ\text{C}$ (Note 5)
Rise Time	t_R		12.4		ns	
Turn-Off Delay	$t_{D(off)}$		14.9		ns	
Fall Time	t_F		22		ns	
Output Capacitance Stored Energy	E_{OSS}		17		μ	$V_{DS} = 400\text{ V}$ $V_{GS} = 0\text{ V}$ $f = 100\text{ kHz}$
Switching Energy during turn-on	E_{on}		134.1		μ	$V_{DS} = 400\text{ V}$, $I_{DS} = 20\text{ A}$ $V_{GS} = 0 - 6\text{ V}$ $R_{G(on)} = 10\ \Omega$, $R_{G(off)} = 1\ \Omega$ $L = 120\ \mu\text{H}$ $L_P = 2\text{ nH}$ (Notes 6, 7)
Switching Energy during turn-off	E_{off}		17		μ	

(5) See Figure 16 for timing test circuit diagram and definition waveforms

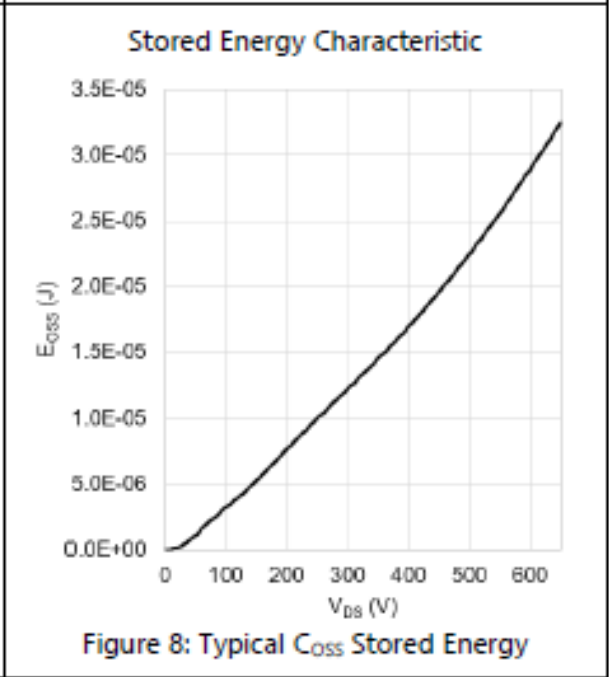
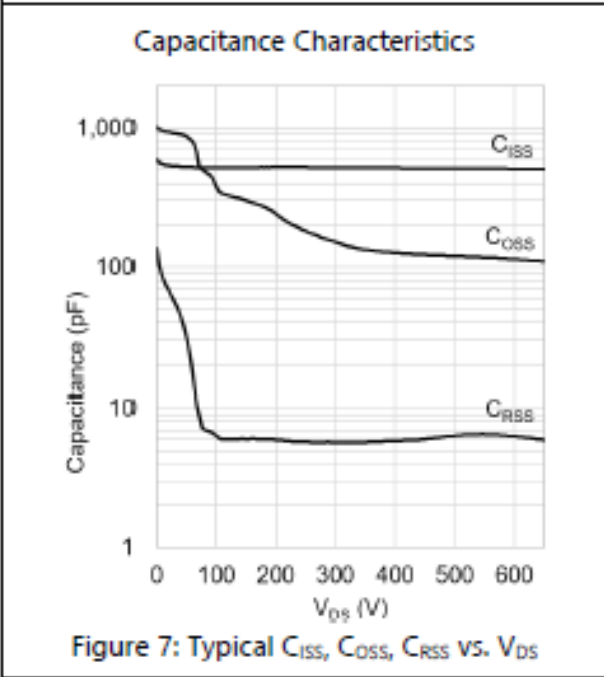
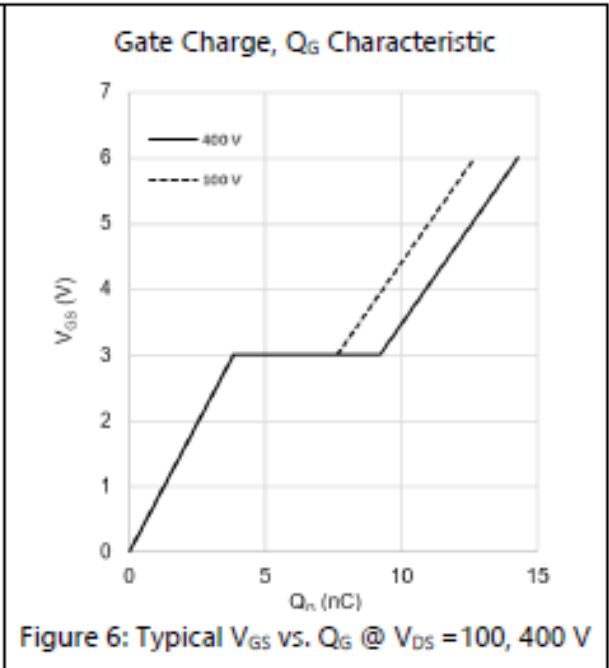
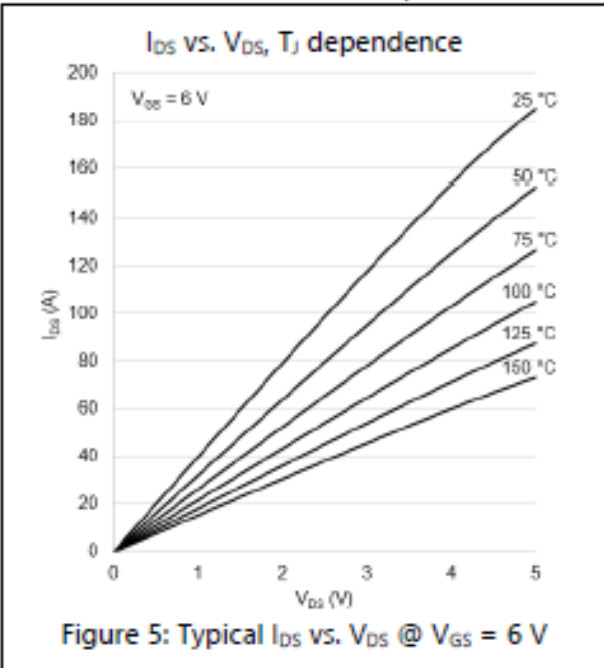
(6) L_P = parasitic inductance

(7) See Figure 17 for switching loss test circuit

Electrical Performance Graphs



Electrical Performance Graphs



Electrical Performance Graphs

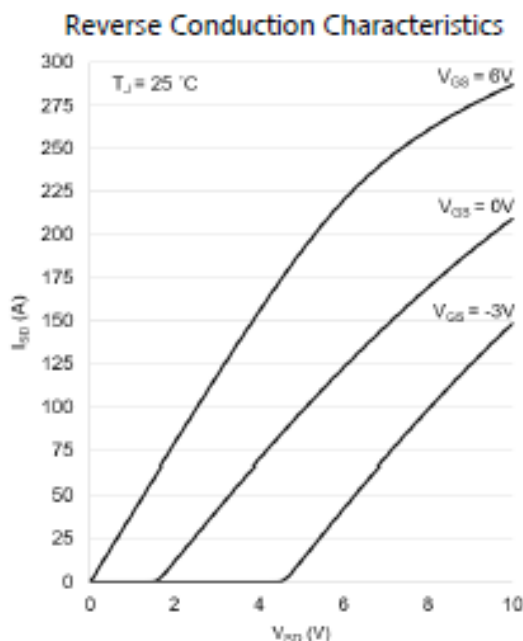


Figure 9: Typical I_{SD} vs. V_{SD} ($T_J = 25\text{ }^{\circ}\text{C}$)

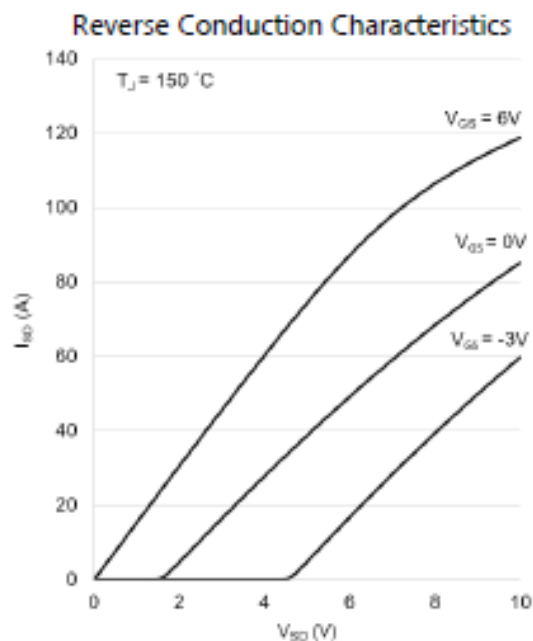


Figure 10: Typical I_{SD} vs. V_{SD} ($T_J = 150\text{ }^{\circ}\text{C}$)

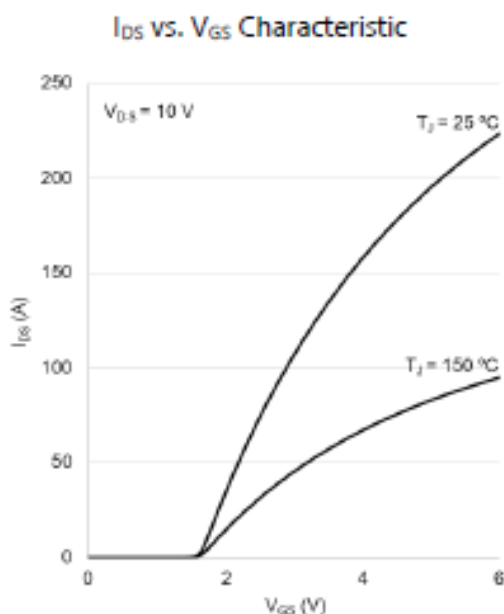


Figure 11: Typical I_{DS} vs. V_{GS}

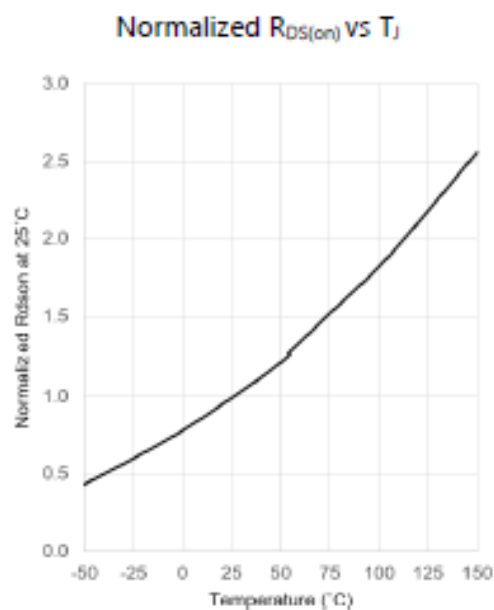
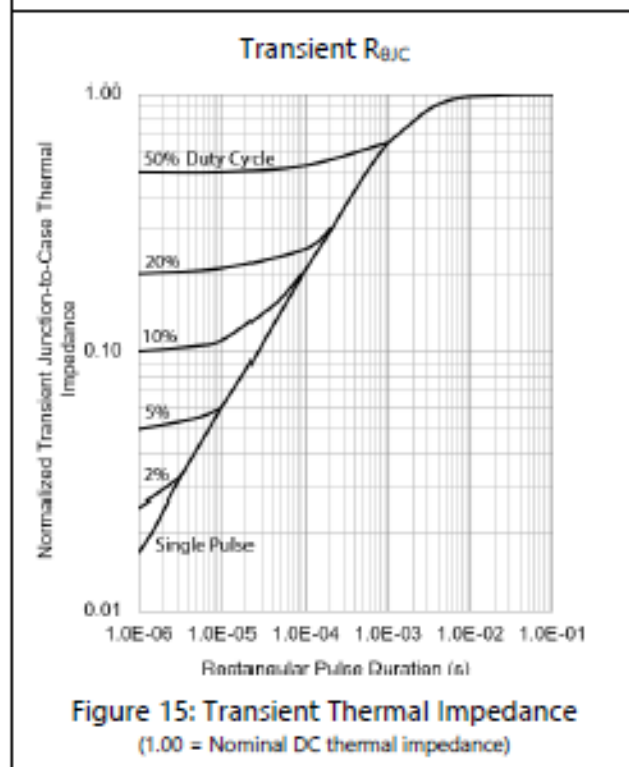
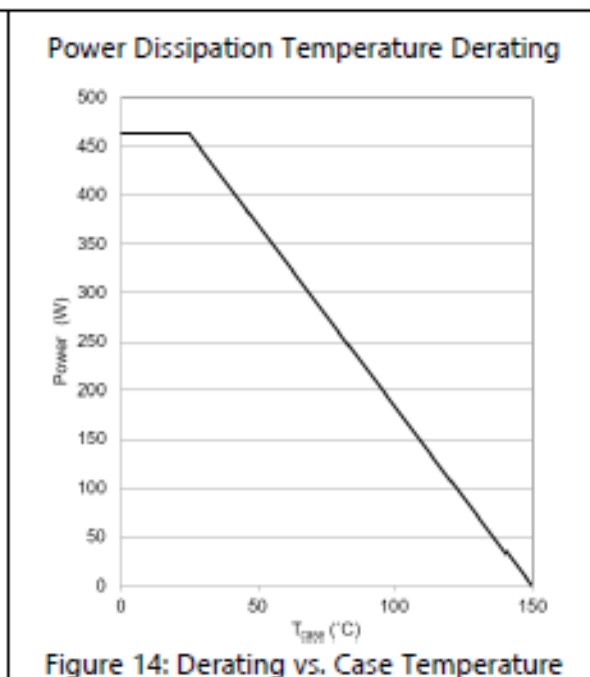
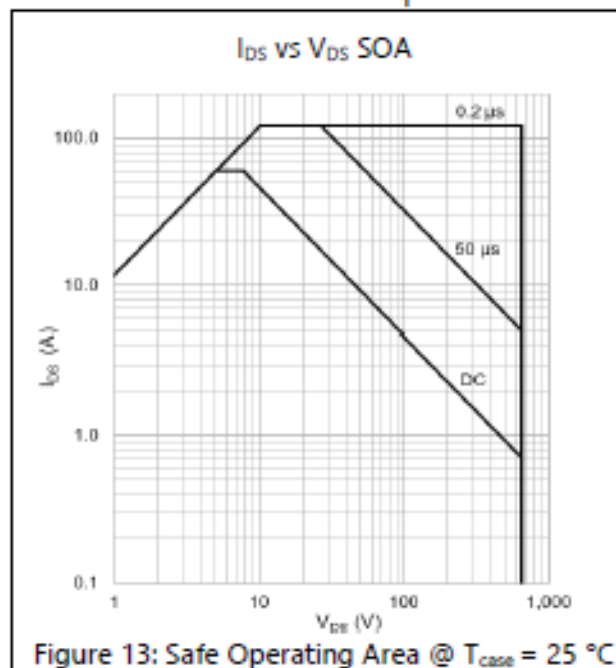


Figure 12: Normalized $R_{DS(on)}$ as a function of T_J

Thermal Performance Graphs



Test Circuits

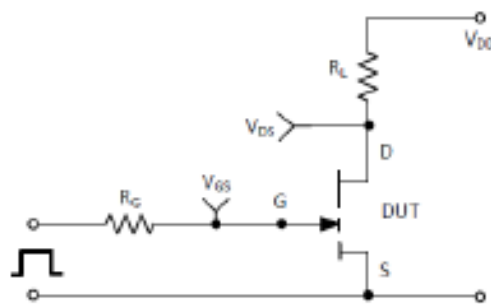


Figure 16: switching time test circuit

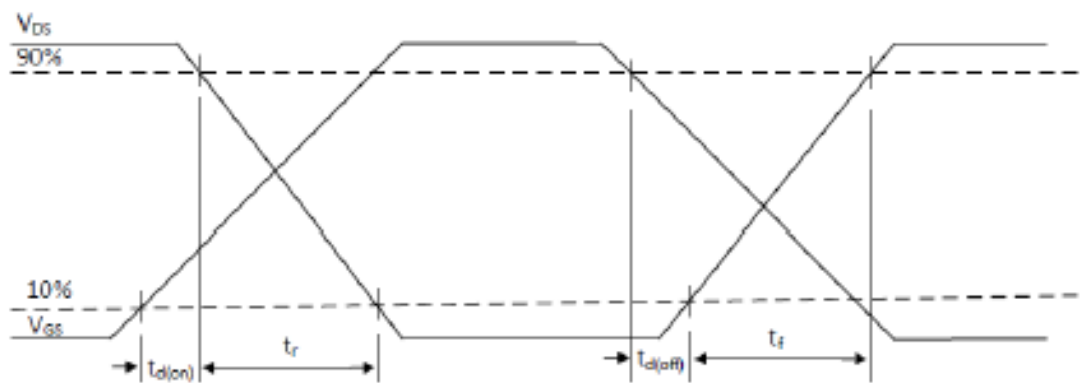


Figure 17: switching time waveforms

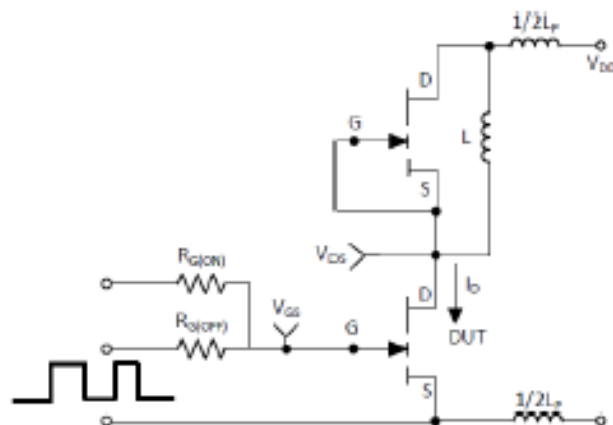


Figure 18: Switching Loss Test Circuit

