UNIVERSITÀ DEGLI STUDI DI PADOVA

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

DEPARTMENT OF INFORMATION ENGINEERING
UNIVERSITY OF PADOVA

Design and Implementation of an On-Board Controller-based Odometer for the Estimation of the Lifetime of Power Modules in Hybrid Vehicles

Università Degli Studi di Padova
Corso di Laurea Magistrale
in Ingegneria Elettronica

Laureando: Alessandro Blascovich

Relatore: Prof. Dr. Gaudenzio Meneghesso

Supervisor presso ETH: Dr. Mauro Ciappa

Anno Accademico 2014-2015

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für Integrierte Systeme
Integrated Systems Laboratory

Master Thesis Project at the
Department of Information Technology an
Electrical Engineering of the ETH-Zurich
Fall 2014

Alessandro BLASCOVICH
Erasmus student (University of Padova, Italy)

Design and Implementation of an On-Board Controller-based Odometer for the Estimation of the Lifetime of Power Modules in Hybrid Vehicles

Supervisor ETHZ:	Dr. Mauro Ciappa
Supervisor UNIPD:	Prof. Dr. Gaudenzio Meneghesso, University of Padova

Start:	September 1, 2014
End:	February 28, 2015

I. Introduction

The end-of-life period of complex multi-chip modules is often defined by thermo-mechanics related failure mechanisms. The time-to-the-failure (lifetime) for these wear-out mechanisms is normally estimated on the base of deterministic models, which are calibrated with data extracted from accelerated power cycling experiments. Furthermore, these estimates are referred to a given application profile, which is specific for the technical system under consideration (hybrid car, locomotive, etc.). Simple lifetime prediction models, based on the Coffin-Manson law for the low-cycle fatigue of the joints and based on principle of the linear accumulation of the damage have been proposed and investigated in the past. Since the main issue associated with such an approach is the procedure to define and extract the number, the amplitude and the duration of the thermal cycles, dedicated prediction models based on the calculation of the deformation work by integration of the constitutive equations have been proposed as an alternative.

At present, no real-time on-board system is available that is able to acquire experimentally the thermal stress experienced by the power modules, to perform the thermal cycle statistics, and to calculate the residual lifetime. This thesis is aimed to explore the feasibility of such a reliability odometer by developing a working prototype that implements the lifetime calculation according the Coffin-Manson approach, as well as the lifetime calculation from the integration of the constitutive equations for the most common materials used in present power modules.

II. Purpose

The scope of this thesis is to develop a prototype of a reliability odometer with following characteristics

- Stand alone system based on a minicomputer
- Autonomy of 12 weeks
- Remote operation (WIFI, ethernet)
- Low cost hardware (< 80Euro)
- Based on open source software
-

and performing following functions (specifications)

- Sampling of the temperature (differential mode, sampling frequency at least 20Hz)
- Sampling of the power dissipated (sampling frequency at least 20Hz)
- Real time statistics of the thermal cycles (amplitude, duration)
- Calculation in real time of the residual lifetime according to Coffin-Manson
- Integration of the constitutive equations with calculation of the deformation work
- Graphical representation in real time of the results
- Logging of the sampled data

III. Tasks

The task breakdown of present Master Thesis Project is as follows:

1. *Failure Mechanisms and Models*
   a. Investigate the main thermo-mechanical Failure Mechanisms
   b. Investigate the damage accumulation principle (Miner's rule)

2. *Coffin-Manson Related Approach*
   a. Define the Basic Principles
   b. Search for experimental parameters
   c. Investigate the cycle counting algorithms
   d. Develop and implement a real time Rainflow algorithm

3. Constitutive equations

a. Define the constitutive equations for a one-dimensional bimetallic system
b. Compare previous approach with the Anand model
c. Solve numerically both models for stress and strain, given the instantaneous temperature
d. Optimize the numerics of the algorithm for implementation

4. Thermal Management

a. Define the compact thermal model of a typical module
b. Simulate by Comsol Multiphyiscs the transient behavior
c. Extract the thermal impedance Z_{th} by FEM analysis in analytical and compact form
d. Calculate the instantaneous temperature by convolution with the instantaneous power function
e. Optimize the numerics of the convolution algorithm for implementation

5. Hardware/Software

a. Benchmark Arduino, Galileo, PCDuino for the implementation of previous algorithms
b. Implement the code where possible in the Arduino environment
c. Implement data logging
d. Implement data processing and visualization in real time by open source codes (Octave)
e. Implement the code on a laptop (mac)

6. Applications

a. Apply the developed system to a typical example
b. Develop a digital-to-analog converter to simulate arbitrary mission profiles
c. Characterize the bandwidth of the system

IV. Report and Presentation

The research activity and its results will be documented in a final report (2 copies for IIS), including a short user manual. The contents of this report will be the topic of an oral presentation to be held at ETHZ. The receipt of the Thesis is acknowledged once the laboratory and the building keys are returned to the ETZ housekeeping manager.

Zurich, September 2014

[illegible]

Dr. Mauro Ciappa

Acknowledgments

First, I want to thank my Supervisor Dr. Mauro Ciappa, in Integrated System Laboratory at ETH, for the effort that he has always demonstrated to me in this work and for his helpfulness in these six months. I want to thank my Supervisor at University of Padua, Prof. Gaudenzio Meneghesso, and the Prof. Alessandro Paccagnella to be the most prepared and dedicated teachers that I have met in my academic studies. It is a pleasure for me to had the possibility to spend one year in Zürich, for that reason, I should like to thank the Eidgenössesche Techniche Hochschule (ETH) and University of Padua for the cooperation.

I want to thank my parents, my grandparents and all my family. I'm proud to have them on my side since I was born because they trust me and they are my fortune to realize my desire to become an engineer. Thank also to all those people that helped and supported me in this period of my life and growth.

Voglio ringraziare per primo il mio Supervisor Dr. Mauro Ciappa, del Integrated System Laboratory presso ETH, per il supporto che mi ha sempre dimostrato nello svolgimento di questo lavoro e per la sua disponibilità in questi sei mesi. Voglio ringraziare il mio Relatore, Prof. Gaudenzio Meneghesso, e il Prof. Alessandro Paccagnella per essere gli insegnati più preparati e attenti che ho incontrato durante i miei studi universitari. È stato un piacere aver avuto la possibilità di trascorrere un anno a Zurigo, perciò mi sento di ringraziare il Politecnico Federale di Zurigo e l'Università degli Studi di Padova per la collaborazione.

Voglio ringraziare i miei genitori, i miei nonni e tutta la mia famiglia. Sono orgoglioso di averli al mio fianco fin dalla nascita perché credono in me e sono la mia fortuna per realizzare il mio desiderio di diventare un ingegnere. Ringrazio anche tutte le persone che mi hanno aiutato e supportato in questo periodo della mia crescita e vita.

Alessandro Blascovich

Index

1 Abstract

This thesis presents a software implementation and hardware investigation to monitor the wear-out condition of power modules used in traction applications. A new and original stand-alone tool for a real-time analysis and lifetime estimation has been developed dealing with two optimized algorithms, the Rainflow and the thermo-mechanical Stress-Strain analysis. Finally, different solutions and microcontrollers have been analyzed and benchmarked to define the tool with the best performance. The implemented system fulfills the main requirements: high speed and real-time analysis, low cost implementation and high flexibility.

In dieser Studie wird eine integrierte Software und Hardware Lösung vorgestellt, die zur Quantifizierung des Verschleisszustandes von Leistungsmodulen, die in der elektrischen Traktion eingesetzt werden. Entwickelt wurde eine neue Methode zur Echtzeit Berechnung der restlichen Lebensdauer, sowie optimierte Algorithmen zur Bestimmung der Zyklenstatistik basierend auf die Rainflow-Methode und der geleisteten Verformungsarbeit. Unterschiedliche Hardware-Lösungen sind um eine optimale Implementation des Systems untersucht worden. Alle Anforderung konnten somit erfüllt werden: echtzeit Analyse der Daten, niedrige Kosten und hohe Flexibilität.

Il lavoro di tesi svolto è un'implementazione software e uno studio hardware per monitorare il degradamento dei transistor di potenza usati nei veicoli a trazione elettrica. È stato sviluppato un nuovo e originale strumento stand-alone che esegue in tempo reale la stima del periodo di vita sfruttando due algoritmi riadattati, il Rainflow e l'analisi di sforzo termomeccanico. In fine, sono state analizzate e testate differenti soluzioni software e hardware per ottenere il miglior sistema ad alte prestazioni. I risultati osservati sono in accordo con le specifiche principali: analisi veloce e in tempo reale, costo contenuto e alta flessibilità.

2 Introduction

The increasing of power electronics devices is becoming more important with years. Many application require devices the can manage high quantity of power and while the efficiency and the packaging density increase, the reliability is an issue that sill remains and demands accurate studies. The aim of this work is to present a tool to inquire and keep trace of health status of the Inverter module used in all the applications where is required the conversion, modulation and management of high quantity of power. It is worth to mention all the electric-traction applications where the power transistors are widely installed and stressed. Even though Hybrid Electric Vehicles are on the market since years and can be one of the most promising application, this is not the only one in which this work can be applied. Many issues still affect the reliability of the power electronic components that has been show to be one of the weakest part in the power uses [1][2][3].

The work conducted in this thesis has the main aim to develop a stand-alone tool which can be combined in every system where IGBT module are present to keep trace of the health condition, understand the degradation process and, most important, plan with good accuracy the device's maintenance and replacement.

In *Chapter 3* the main failure mechanism that affect the typical power device modules are presented. The basics structure of the IGBT module is described and the thermo-mechanical introduces damage is analyze for wire-bonds and solder-joint.

In *Chapter 4* are described the known method to estimate the lifetime, basing the calculation with two approaches. The Rainflow algorithm is a well-known method which measures the cumulated damage on the device based on the mission temperature profile. This method is readapted to obtain a real-time implementation. Some tests have been conducted to evaluate the accuracy of this method and the computational effort keeping in mind also the other possibilities.

In *Chapter 5* the second approach suitable to evaluate the end-of-life in IGBT modules is analyzed; it is the pure mechanical study of the material's behavior under thermal expansion and compression due to different thermal expansion coefficient (CTE). The constitutive equations under the mechanical fatigue are solved and computed, fending also alternative solutions. The strategies adopted in this part of the thesis are done with the goal to implement the code on a microcontroller.

The *Chapter 6* is a summary of the thermal management and dissipation of heat generated by power modules. A basic structure is taken into account, but that is good enough to describe the multilayered thermal behavior. The FEM simulation conducted helps to understand the heat propagation on the device and the different response of the material. In this chapter it is explained how it is possible to calculate the average and approximate junction temperature, which is the main input of the reliability odometer designed, from the simulation and the analytical model.

In *Chapter 7* the requirements of this tool and the available microcontrollers and platform to develop the software are specified. The devices taken into consideration are Arduino, Intel Galileo and pcDuino. Some benchmark has been necessary to define the right board that satisfies the given specification of the reliability odometer.

The final implementation is conducted and summarized in *Chapter 8*. In this chapter the software written is explained in detail and the main program's commands are commented. The overall performance, the problems encountered, the test and the further extension of the code are here reviewed.

3 Failure Mechanism

In this chapter the main reliability issues that are common in IGBT module device are discussed. The lifetime of those devices is strongly related to the failure mechanisms that are thermo-mechanical related. The common methods to estimate the lifetime expectation have been reviewed and summarized, underlining the novelties introduced and the critical aspects.

3.1 General aspects

The end-of-life period of a complex structure, like IGBT modules, is often describe through thermo-mechanical fatigue phenomena that are necessary present on those devices. Understanding the reliability calculation and the failure mechanism is the first step that has to be done before formulating the time-to-failure relationship and deal with algorithms which predict the devices' health status. The mechanisms presented here are just the most common failure mechanism that are widely studied [4] in the past and are nowadays still one of the most often cause of failure also in modern devices.

3.1.1 Basics Of Reliability Calculations

The definition of reliability is outlined by IEEE, and in the engineering field accepted, as *the ability of a system or component to function under stated conditions for a specified period of time* [5]. This statement can be translated into a mathematical context as a time-dependent function that defines the device's failure probability. The main parameter to characterize the reliability is the *failure rate* $\lambda(t)$. Given a working device at time *t*, the probability that it fails in the time interval $[t, \Delta t+t]$, with $\Delta t \rightarrow 0$, is described by the failure rate. The unit of measure of $\lambda(t)$ is the failure in time (FIT):

$$\lambda(t) = \frac{N_f}{10^9\ hours} \qquad (1)$$

where N_f is the number of failures recorded in 10^9 hours. The *mean-time-to-failure* (MTTF) is another useful parameter that explain the expected mean lifetime of the device and is related to the failure rate as reported in following formula:

$$\mathrm{MTTF} = \frac{1}{\lambda(\mathrm{t})} \qquad (2)$$

[6] Is important to underline that both, $\lambda(t)$ and *MTTF*, are time-dependent parameters that are very useful to describe the lifetime expectation and working conditions of the device. In particular, for most of the electronic device, the lifetime of a device can be divided in 3 different sections: the *early failures*, the *random failures* and the *wear out failures*. As reported in Figure 1, in IGBT devices, and in particular in this work, only the failure mechanisms that occur due to the

progressive degradation of the device are considered, so only the wear out part of the bathtub curve. In other words all those failure caused e.g., by defect in production, wrong handling and or mounting that occur in the first period of application are not taken into account.

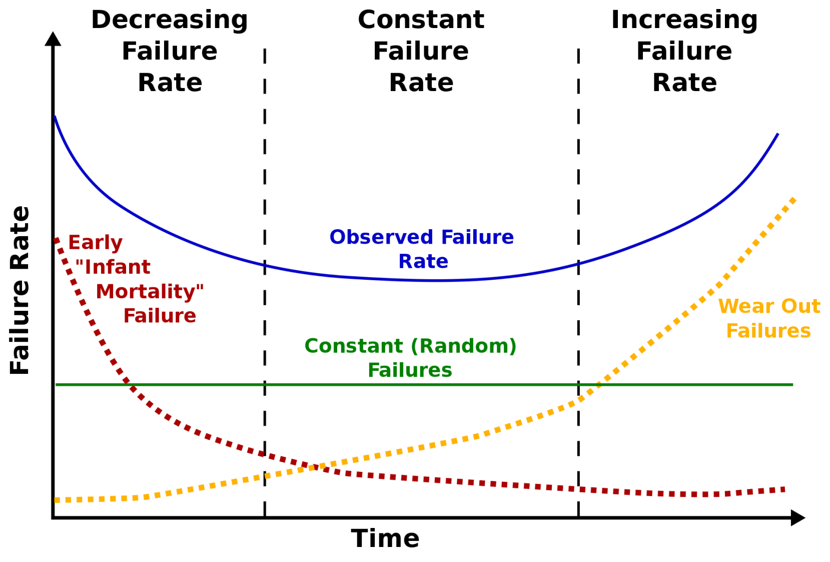


Figure 1. Bathtub curve representing in general the failure rate of a device. IGBT transistor consider in this work show only the failure due to the wear out of the device, the other failure mechanism are not taken into consideration [7]

3.1.2 Basic IGBT structure

In many applications that deal with a huge amount of power, IGBT (Insulated Gate Bipolar Transistor) are required. The main advantage of this kind of active devices is the very high input impedance and the low output resistance, making those devices suitable for power application, typically in inverter systems. Inverters are widely present in industrial application, photovoltaic plants, wind turbine, electric traction vehicles and in general in most of the systems where a voltage conversion, modulation or control to drive electric motors or heaters is required.

Figure 2 reports the circuit diagram of a 3-phase inverter composed by 6 IGBT and 6 diodes that are usually enclosed in the same package, called IGBT module. In Figure 4 an internal view of a IGBT module is reported; it is possible to identify the IGBT and the diodes.

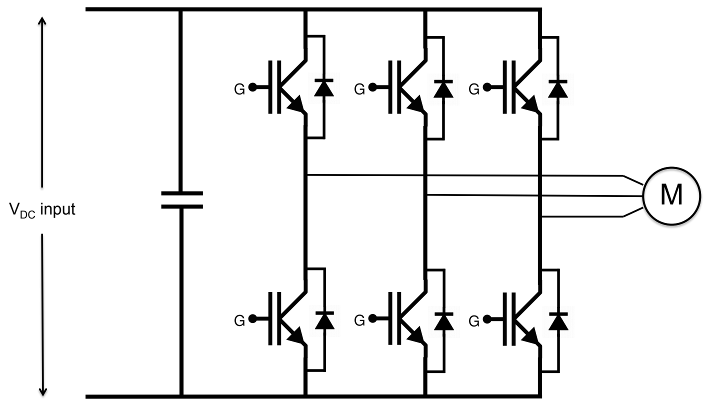


Figure 2. 3-phase inverter schematic. All the transistors are IGBT soldered together in the same module.

For analysing more in detail a module for high power applications it is opportune to describe the vertical structure layer by layer. Even though the structure of an IGBT module is complex and multi-layered, generally at least 3 layers are present in every system. Starting from the top, it is possible to recognize the *silicon chip*, the *substrate* (ceramic insulator) and the *base plate*. Each layer is soldered to the next layer with solder that in general is a tin alloy [8][9]. The electrical connections are done with the wire-bonds in the upper interface linking the IGBT and the Diode to the external power bus and through the solder in the copper path on the substrate (see Figure 3 and Figure 4).

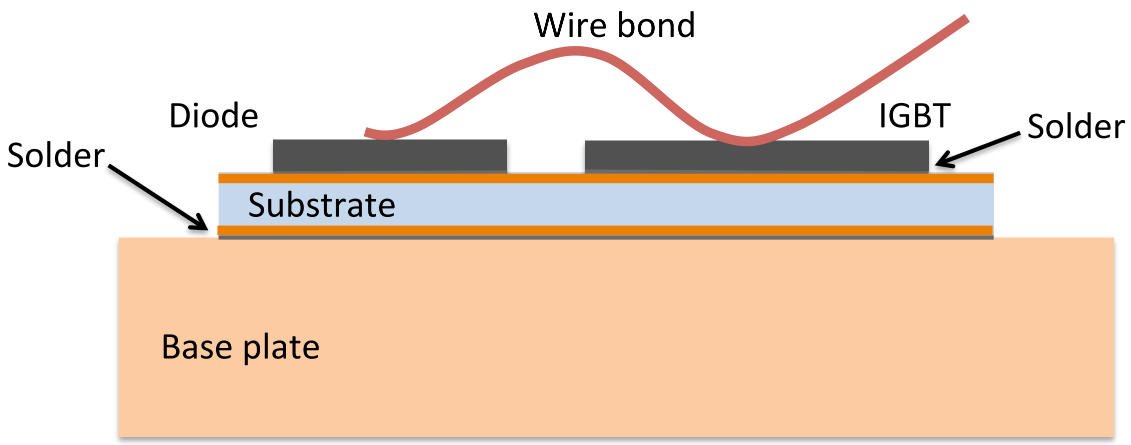


Figure 3. Schematic structure of a standard power module and the fundamental components. Note that the substrate, typically a ceramic insulator, has on top and bottom a thin copper layer. The base plate is then installed on the heat sink.

When the current flows on the silicon chip, the IGBT heats up and dissipates the power through the substrate and the base plate.

The modules taken into consideration in this thesis are multichip structures that are rated for collector-emitter voltages (V_{ce}) that vary form 650 V to 4 kV and current up to 1.2 kA. The management of this high power produces a dissipated power that goes from few hundreds of Watts up to some kilowatts, and for this reason an efficient cooling system, mounted on the baseplate, is required.

3.2 Thermo-mechanical induced failure mechanism on IGBT

In the previous section, the high-power IGBT modules structure has been described. Even though the multilayer structure is complex and many failures can occur, it has been proved that the majority of the failure mechanisms are thermo-mechanical fatigue related [4]. The ASTM International define the fatigue as: *the process of progressive localized permanent structural change occurring in material subjected to conditions which produce fluctuating stresses and strains at some point or points and which may culminate in crack or complete fracture after a sufficient number of fluctuations* [10].

As reported in Table 1, the *thermal expansion coefficient (CTE)*, the *temperature* and the *characteristic length or size* of each material of the layer are different; thus due to the mismatch, when a thermal and power cycling is applied to the chip the structure is exposed to a mechanical deformation. The weakest parts of module are the interface of between two next layers, in particular the wire bonds attachment and the solder joint [4].

	Material	Thickness [µm]	CTE [PPM/°C]
Wire bonds	Alluminium	300	22
Chip	Silicon	500	3
Solder	Tin alloy	Not relevant	See Table 2
Substrate	Al_2O_3 or AlN	1000	4 ~ 7
Base plate	Copper or AlSiC	4000-5000	17 ~8

Table 1. General parameters of material used in IGBT modules. More detailed parameters and take in account structure in Chapter 6

The expansion and compression process due to the thermal variation is called *thermal fatigue*, which leads the crack initiation and propagation. In a pure mechanical system, the damage accumulated due to the cyclic stress continues until the final fracture. In the case of IGBT modules, the final fracture would never reach because other breakdowns affect the application. Indeed, when a crack propagates on the wire bonds or the solder joint, the electrical effect observed are the V_{ce} and R_{series} increasing.

3.2.1 Wire-bonds

The wire bonds are responsible to carry the current from the power bus line, which are connected to the external contacts, to the active area of the device. For example, in EconoDUAL^{TM} 3, manufactured by Infineon, eight wires connect the IGBT and the freewheeling diode and in HiPak by ABB are present 12 wire bonds [11][12]. The typical values of the maximum DC current density per wire are about 30 kA/cm^2, meaning that they melt at 25A for 300 µm diameter and under normal operation condition they drive no more than 10A [4]. Since they are bonded onto the chip surface by ultrasonic wedge, the heat generated on the silicon is directly transferred to the wires. This causes a temperature gradient on them and, due to the thermal expansion mismatch, the result is a fatigue of the wire bonds caused either by sheer stress generate at the attaching point and by the continue flexures of the wire. First the ohmic resistance of the contact increase and, when the accumulated damage exceed the stress capability, the wire breaks or lifts off from the pad or melts; as a consequence, the density current in the other wires increase and a new distribution of the current is observed. Monitoring the V_{CE} as function of the thermal cycles, the degradation of the wire or the contact shows a continuous increasing of the ΔV_{CE} until the failure criteria threshold is reached (typically the pre-defined value is about 5-20% ΔV_{CE}) [13].

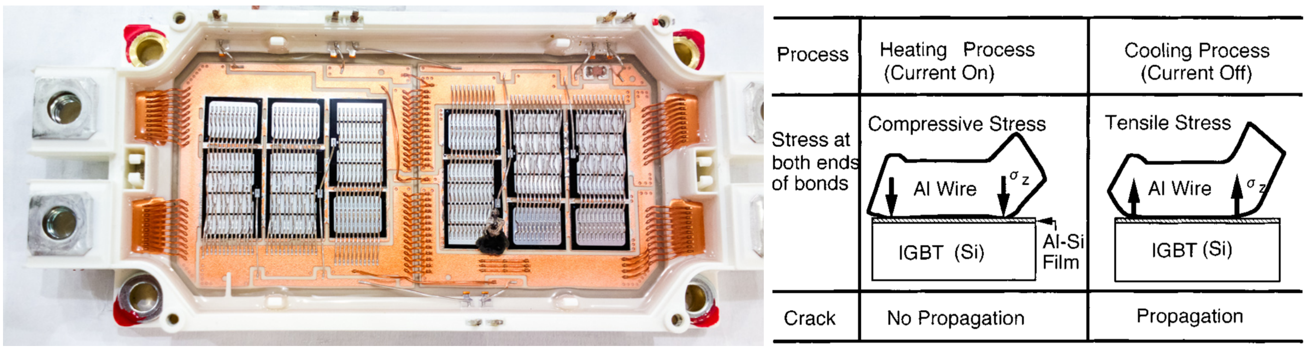


Figure 4. Left: inside view of EconoDUAL^{TM} 3 by Infineon. Right: explanation of expansion and compression process of the Aluminium wire bonds during the heating and cooling [2].

For any kind of device's failure, the mechanism that occurs in the wire bonds is due to a thermal-mechanical effect. As reported in Figure 4 (right), the thermal cycles cause an expansion of the wire during the heating process and compression during the cooling period. Heating and cooling instants are related to the conduction or cut-off mode of the device. The melting of the wire bond is seen when the current density on the wire overcomes the maximum rated current; this occurs when more then one wire has been lifted off form the pad or when the dissipated power on the chip is not well dissipated out of the device causing a overheating of the surface of the silicon chip. Typically, under normal operation conditions, the self-heating generated by the Joule effect due to the current flow in the wires, does not significantly contribute to the degradation.

The statistical distribution of the wire bonds lift off shows that the firsts connection affected are those in the center of the device where the temperature swing and variation reaches the highest values [4].

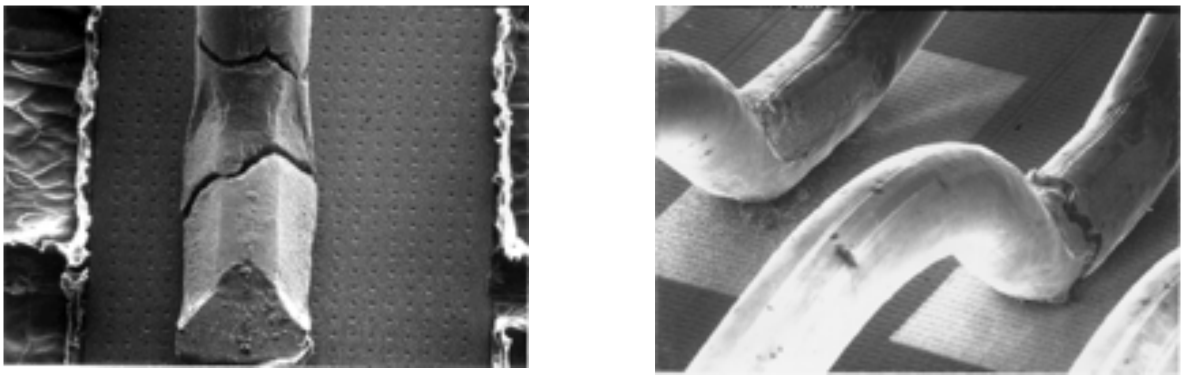


Figure 5. Wire bonds crack initiation and propagation. (SEM image 25x form [4])

3.2.2 Solder joint

The solder joint on IGBT module is responsible of the adhesion of each layer and the thermal conduction of the dissipated power by the silicon chip through the substrate and base plate. The most popular solder used in electrical application is a tin-lead alloy, in particular the eutectic Pb/Sn solder. [14] Since 2006 the European legislations imposes new strictly rules for industries in the lead use; for this reason new metal alloy without lead have been tested and investigated in the last 10 year. For example the new solder chemical composition is based on tin, silver and copper. Even though lead free material are coming out on the market, the reliability issue and degradation process are the same, but with different lifetime expectations. For more availability of information and technical data about Pb/Sn alloy, this work uses the parameter and coefficients of the classical type of solder. It is worth to note that since the same failure mechanism occurs for lead free solders, the study presented is still valid.

Solder	63Sn-37Pb	Sn3.5Ag	Sn-3.5Ag-0.7Cu
Melting Point [˚C]	183	221	218
Tensile strength [MPa]	34.7	55	44.0
Yield Point [MPa]	28.4	32.1	35.1
Young's Modulus [GPa]	40	56	51
Poission's Ratio	0.37	0.4	0.36
Specific Heat [J/g K]	0.18	0.24	0.23
Thermal Conductivity	52.8	50.1	62.1
CTE	21.6	21.9	21.2

Table 2. Material parameters of most common solders, either with and without lead [15].

Comparing the CTE of solders and all the material in the layer is possible to understand that the worst thermal parameters' mismatch is between the ceramic substrate and the baseplate. The solder joint, in combination with the temperature difference of the multi-layered structure, is one of the weakest parts of the device. The failure mechanism under power cycling for modules using lead-based solder between silicon chips and the substrate is described by the internal shear stress due to the layer's strain. Indeed, the internal force during the expansion or compression causes the cracking of the solder at the interface or the border. The cracks propagate from the external side of the device in lead-based solder, while in the new generation without lead, the crack starts in the center of the chip [16].

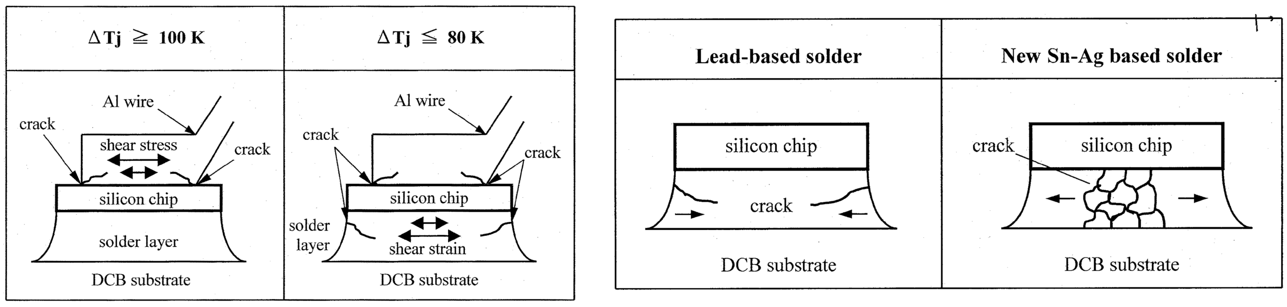


Figure 6. Representation of crack initiation and propagation on the wire bonds and solder layer. During a thermal cycle the substrate and the silicon chip have different elongation causing a strain at the interface (Figure from [16]).

The fatigue-induced damage of the solder layer has as main consequence the increase of thermal impedance between the silicon chip and the base plate. Given that the heat flux is basically one-dimensional, from the silicon chip to the base plate, when the solder thermal resistance increases, the dissipated power significantly produces a higher junction temperature and therefore accelerates the degradation process of the module and the wire bond lift off.

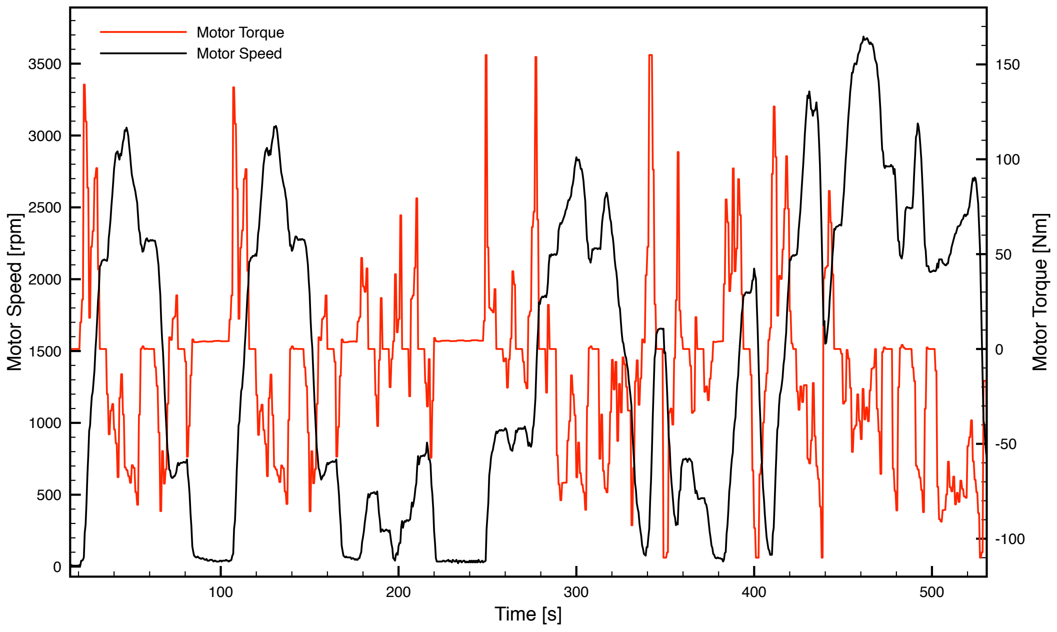
4 Life prediction for low-cycle fatigue

Lifetime estimation is usually based on reliability tests and failure probability described as mathematical function as explain in § 3.1.1. A deterministic model, based on available data and calibrated on several accelerated lifetime tests, generally explains the time-to-fail of any device. These tests give a qualitative results of the health condition of the device during time; for example, the Military Handbook [17] provides a common basis for reliability predicting, based on analysis of the best available data at the time of issue.

Other approaches and lifetime estimations are more appropriate for power application and in particular IGBT modules. *Power cycling* (PC), *thermal cycling* (TC) and *thermal shock* (TST) are failure measurements that usually are conducted to have a data set of multiple characteristic stress tests. During real application environment the overall load/stress is usually a combination of these tests with varied durations [18]. Even though the lifetime prediction is just an estimation value, the models that will be presented in the following chapter are two way to better predict the cumulate damage given as input the numbers of thermal cycles or the junction temperature.

4.1 Life-time prediction on Mission Profile

The mission profile describes the electrical, thermal, and in general all the stress data distribution of the device in its environment. In the applications with high lifetime requirements, e.g. railways or hybrid electrical vehicles, the lifetime estimation of power IGBT module is based on realistic mission profile. The mission profile results e.g. in the phase currents, collector-emitter voltages, motor speed, switching frequency and temperature in the inverter. In combination with the electrical properties of the power module a loss profile can be calculated. Further, the thermal behaviour of the power module and the cooling system allow to translate these losses into temperature profilers of IGBTs and Diodes [19]. In the case of electric-traction and hybrid vehicles, the acceleration, speed and the road slope define the inverter load condition and, thus, the thermal profile of IGBT as reported in Figure 7.



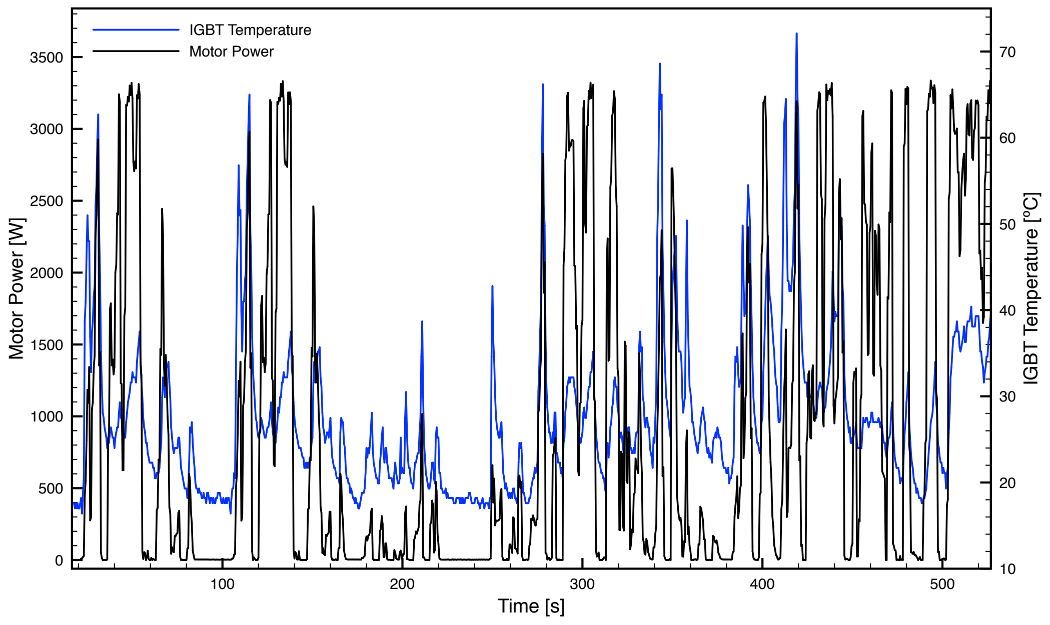


Figure 7. Example of mission profile for hybrid vehicles given by ARTEMIS project [20]**. The graphs above reports a real-world sampling of driving cycles. These data define the operational condition of the devices, in particular are fundamental to characterize the IGBT and diode junction temperature swing.**

Most of the models for lifetime prediction of IGBTs require the knowledge of the temperature mission profile to estimate the cumulated damage in function of the thermal cycles. Even though is possible to evaluate the health condition of the device through ΔV_{CE} or R_{series}, this way has been never used in power application; indeed, measuring these electrical parameter in real-time require additional circuits that most of the time are complex or expensive [21]. Indeed, the V_{CE} variation in a damaged device is about 50-100 mV in a system where in off condition V_{CE} is up to kV. The method based on the temperature cycling is simpler because it doesn't require a precise measurement system, but just a thermocouple or the dissipated power and the thermal impedance, which are usually information available.

4.2 Linear damage accumulation: Miner's rule

The algorithms that estimate the lifetime via thermal cycling analysis are established on the concept of linear damage accumulation. The linear damage accumulation model is based on a cycle-by-cycle computation of independent and progressive load cycles. It is an integration of calculated damage growth using the assumption that the damage per cycles is a time independent event, which doesn't depend from the cycle's history. The advantage of the linear damage accumulation rule is computational efficiency, whereas the disadvantage is non-consideration of non-linear degradation mechanics effects. The Miner's rule describes mathematically the damage as the ratio of the number of cycles in a certain operational condition (e.g. a temperature difference) to the number of cycles to failure at any given stress level by:

$$D = \sum_{i=1}^{k} \frac{n_i}{N_f} \tag{3}$$

where n_i is the number of cycles at *i-th* loading condition, N_f is the number of cycle to failure *i-th* condition and D is the total cumulated damage. If the value of the damage reaches $D = 1$, it is likely that a failure occurs in the material. Indeed, when $n_i = N_f$ immediately is $D = 1$.

In IGBT power modules the *i-th* loading condition, n_i , represents a temperature cycle with a temperature difference ΔT and N_f is the number of cycle to failure at ΔT. Usually manufacturing gives this characterization in term of N_f as function of ΔT. For example, ABB releases an application note for HiPak power modules in which the load-cycling capability is analysed for different mission profiles [9]. Figure 8 reports an example of the lifetime tests conducted for solder joint. Others analyses are available for the wire bonds.

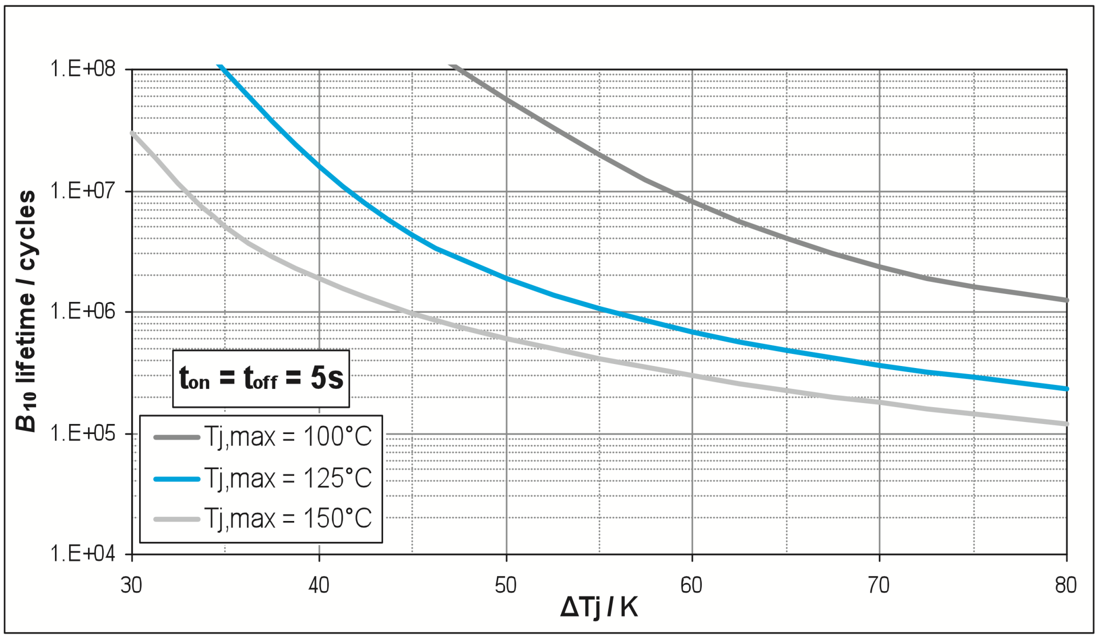


Figure 8. Typical lifetime estimation of the solder joint as function of ΔT. In x-axis ΔT and in y-axis the number of cycles to failure of the solder. ABB define a failure in a device when 10% of the modules of a population fail [9].

The experiments conducted by ABB show a strong dependence on the time-to-failure from the temperature cycles shape. Indeed, taking in account that the modules' layers are heated by the chip and cooled by the base plate, le length of the heating and cooling period sets the thermo-mechanical stress of each component.

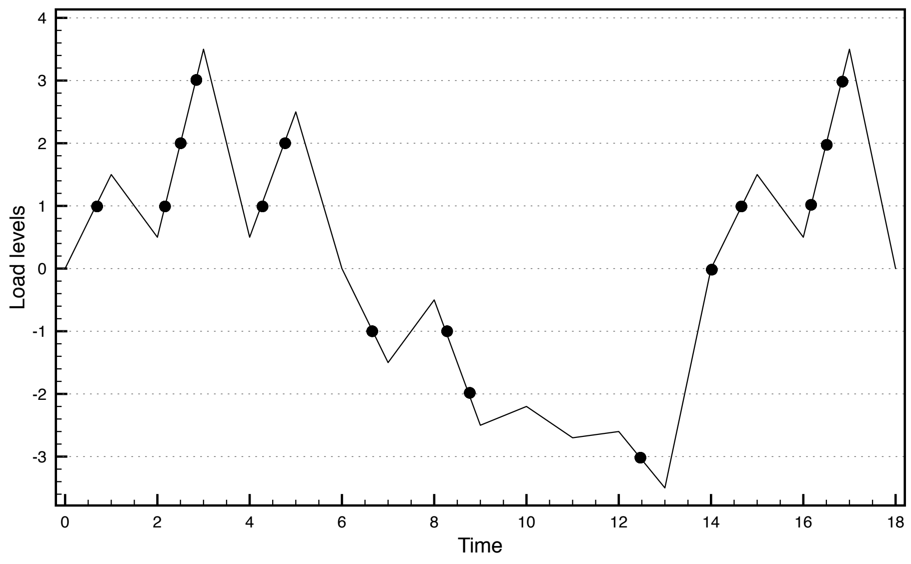
Generic mission profiles present temperature cycles that are complex and statistically distributed as a stochastic process. For this reason, the cycle's length and amplitude, have many interpretations and it aren't uniquely define.

4.2.1 Cycle counting

A compilation of acceptable procedures for cycle-counting methods employed in fatigue analysis are presented by ASTM International [22]. A comparison of those method has been conducted in many papers [23][24]. Here are present a summary of those methods and a revisited version for a real-time implementation.

4.2.1.1 Level crossing counting

Cycles calculation with level crossing counting algorithm is shown in Figure 9. One count is recorded each time the load crosses a pre-defined level load. It is a positive counted when the slope is positive and exceeds a positive level; it is counted negative when a negative level is crossed with a negative slope. This method is sensitive to stationary loads around a load reference. In practice, restrictions on the level-crossing counts are often specified to eliminate small amplitude variations, which can give rise to a large number of counts.

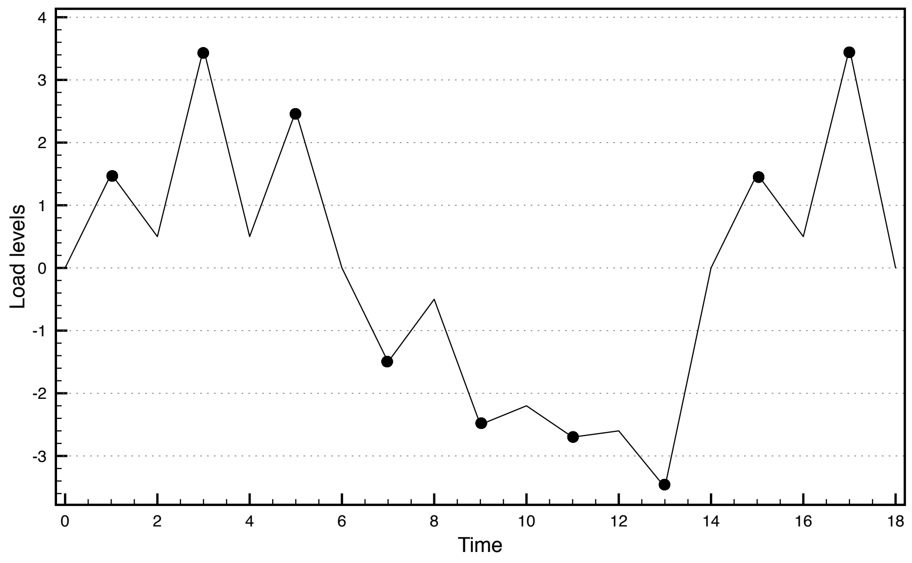


Level	Counts
+3	2
+2	3
+1	5
0	1
-1	2
-2	1
-3	1

Figure 9. Level crossing counting example. A count is recorded each time the load cross the positive reference with a positive slope or cross the negative reference with a negative slope. 0-reference is usually the average of the load or other pre-defined definitions are possible [22].

4.2.1.2 Peak counting

Peak counting identifies the occurrence of a relative maximum or minimum in load value profile. Peaks are counted above the reference load level, while valleys are counted when below the reference load level, as shown in Figure 10. Results for peaks and valleys are usually reported separately. A variation of this method is to count all peaks and valleys without regard to the reference load. To eliminate small amplitude loadings, mean-crossing peak counting is often used; in this way only the largest peak or valley between two successive mean crossings is counted

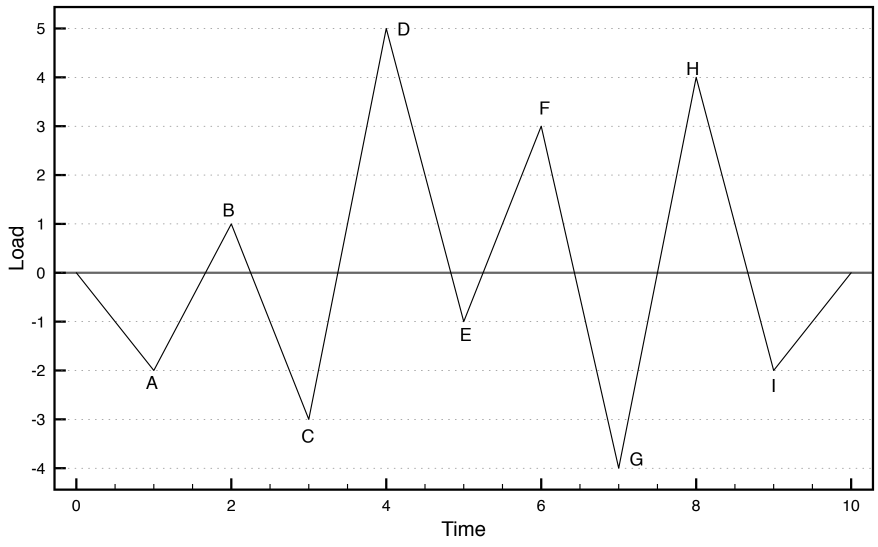


Level	Counts
+3.5	2
+2.5	1
+1.5	2
-1.5	1
-2.2	1
-2.7	1
-3.3	1

Figure 10. Peak counting example. A count is recorded each time the load presents a maximum in the positive load reference levels and when a valley is recorded in the negative load reference levels 0-reference is usually the average of the load or other pre-defined definitions are possible [22]**.**

4.2.1.3 Simple-Range counting

For this method, a range is defined as the difference between two successive maxima or minima; the difference being positive when a valley is followed by a peak and negative when a valley follows a peak. The method is illustrated in Figure 11. Positive ranges, negative ranges, or both, may be counted with this method. If only positive or only negative ranges are counted, then each is counted as one cycle. If both positive and negative ranges are counted, then each is counted as one-half cycle. Ranges smaller than a chosen value are usually eliminated before counting.



Range	Counts	Event
3	0.5	A-B
4	1	B-C, E-F
6	1	D-E, H-I
7	0.5	F-G
8	1	C-D, G-H

Figure 11 Peak counting example. A count is recorded each time the load presents a maximum in the positive load reference levels and when a valley is recorded in the negative load reference levels 0-reference is usually the average of the load or other pre-defined definitions are possible [22].

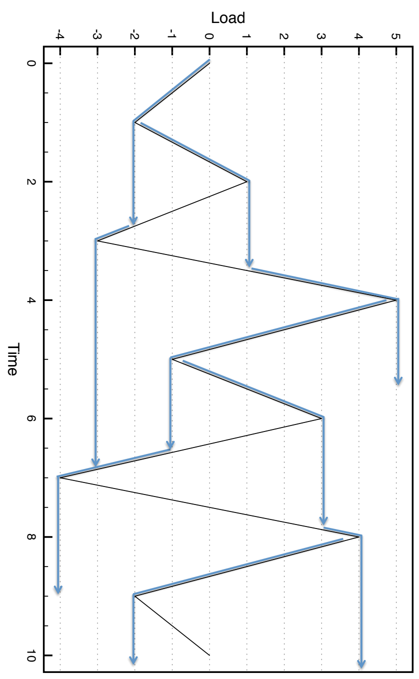
4.2.1.4 Rainflow counting

The Rainflow counting is the most accurate method for counting cycles. Many interpretation of this counting method have been proposed and investigated. Most of them are an off-line solution with different grade of accuracy, execution time, complexity and memory usage [25].

The ASTM version of the Rainflow algorithm is the older and more diffused [22]. This approach is is based on the following steps:

1. *Let X denote range under consideration; Y, previous range adjacent to X; and S, starting point in the history.*
2. *Read next peak or valley. If out of data, go to Step 6.*
3. *If there are less than three points, go to Step 1. Form ranges X and Y using the three most recent peaks and valleys that have not been discarded.*
4. *Compare the absolute values of ranges X and Y.*
    a. *If $X < Y$, go to Step 1.*
    b. *If $X \geq Y$, go to Step 4.*
5. *If range Y contains the starting point S, go to Step 5; otherwise, count range Y as one cycle; discard the peak and valley of Y; and go to Step 2.*
6. *Count range Y as one-half cycle; discard the first point (peak or valley) in range Y; move the starting point to the second point in range Y; and go to Step 2.*
7. *Count each range that has not been previously counted as one-half cycle.*

To understand Rainflow algorithm the “pagoda roof” example is a good analogy. Imagine to turn the signal plot by 90°. Now, let a water flow drops from their upper tops on each of the “pagoda roofs” so defined, until either a roof extends opposite beyond the vertical of the starting point, or the flow reaches a point that is already wet. A new cycle is found when a drop touches a portion of the roof already wet.



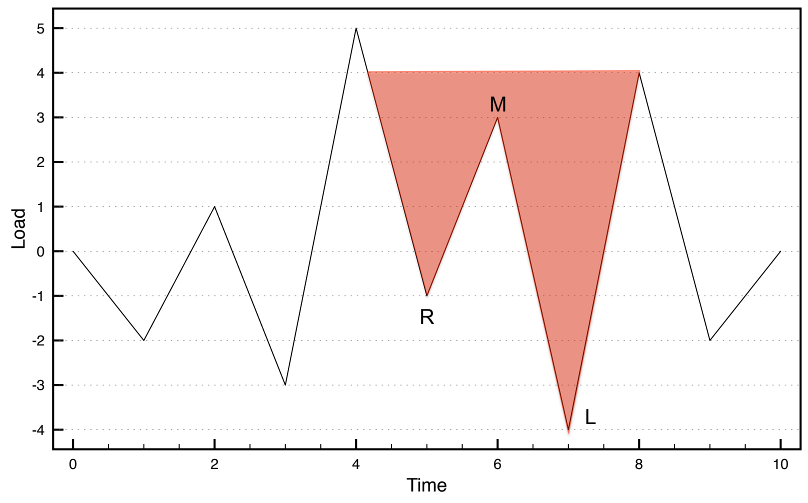


Figure 12. Left, the "pagoda roof" example of Rainflow algorithm. Right, mathematical definition of the Rainflow based on 3-point evaluation. Consider a local maximum M in the interval L, R. is max(L, R) where L and R are the overall minima on the intervals to the left and to the right until the signal reaches once again the level of M.

One more possibility to calculate the Rainflow cycles is the rearrangement of all the maxima and minima ordering by absolute value from the biggest range to the smallest one, as reported in Figure 13.

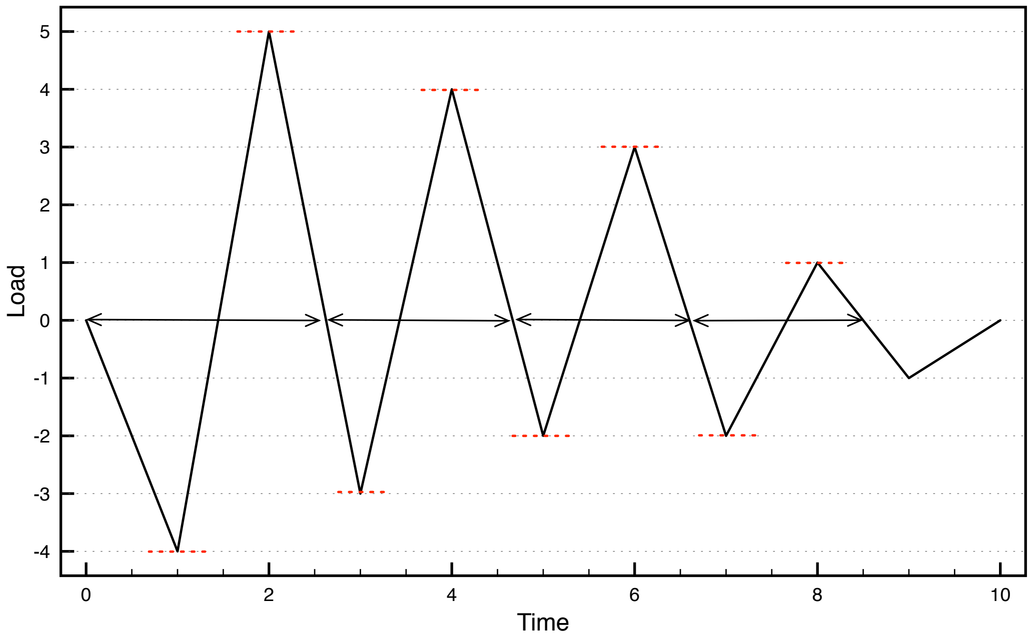


Figure 13. Rainflow ordered cycles. The cycles here presented are the same form Figure 11, but they are rearranged by amplitude range, from the biggest to the lowest.

4.2.2 Cycle Definition for thermal cycles: time and amplitude

The definitions presented above are general descriptions for any fatigue process. In the case of thermal cycles is convenient to introduce further explanations about the thermal excursion and the period of the cycle. [24] Ciappa et. al. proposed five definition of amplitude and duration of thermal cycles summarized in Table 3. The study evidences that the definition adopted has a strongly influence on the statistical distribution of the cycles.

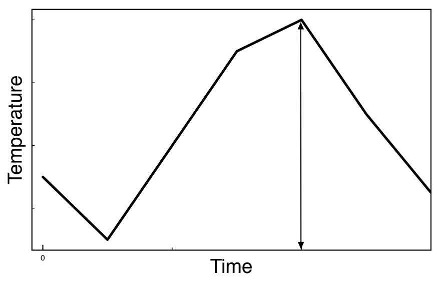
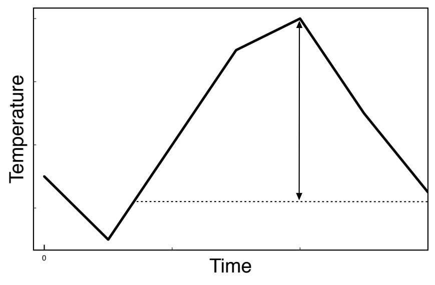
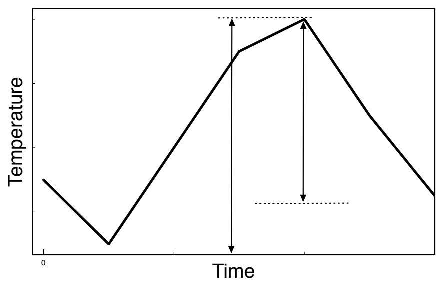
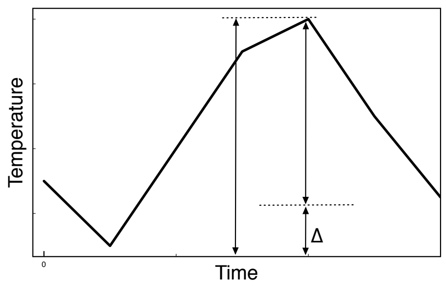
	**Definition A** *Duration*: Time span between two subsequent relative minima. *Thermal Excursion*: Difference between the maximum and the minimum temperature within a cycle duration.
	**Definition B** *Duration*: Time span between two subsequent relative minima. *Thermal Excursion*: Difference between the maximum temperature and the temperature at the highest relative minimum within a cycle duration
	**Definition C** *Duration*: Time span between two subsequent relative extremes. *Thermal Excursion*: Difference between the maximum and the minimum temperature within a cycle duration.
	**Definition D** *Duration*: is defined as the interval between two subsequence relative minima/maxima *Thermal Excursion*: as in definition B, with an additional corrective factor Δ.

Table 3. Possible definitions of thermal cycles: different characterization can be adopted in terms of time duration and thermal excursions [24].

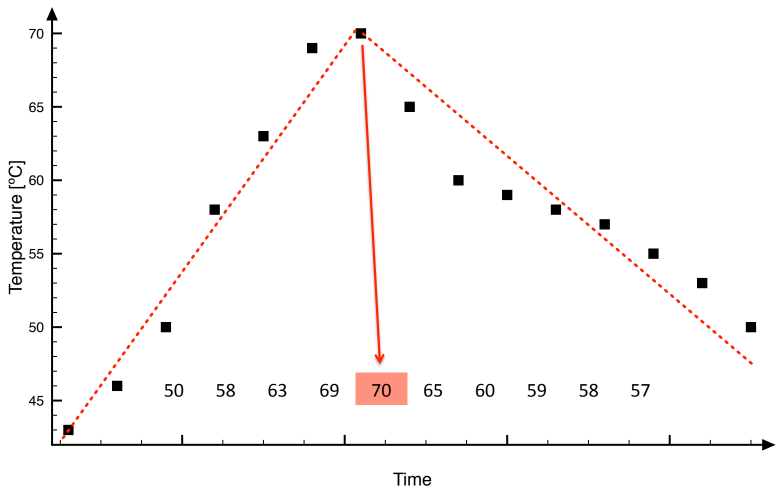
4.2.3 Real-time Rainflow

Most of the Ranflow algorithms are computed on a data set after the data acquisition and pre-elaboration; indeed the off-line implementation requires a data ordering and rearrangement of global minima and maxima. The off-line method is usually adopted in the studies where it is possible to stop the data sampling (and also the load applied to the device) to evaluate the fatigue introduced. Those tests are mostly used to characterize the device's lifetime and give an excellent approximation about the relation between the number of the thermal cycles and the damage introduced; they are thus suitable for datasheet. This way is not convenient for a real-time application where the IGBTs are continuously working and is important keep track of the damage accumulated. The real-time version is the case, e.g., of all those application where the planned preventative maintenance is mandatory and a failure is an economic loss.

The method here presented allows a real-time estimation of the health status of the device given as input the temperature trend. The output of the computation is the remaining lifetime and the description of the thermal cycles occurred. The cumulated damage is evaluated using Miner's rule and Coffin-Manson assumption, as described in § 4.2.4. The advantages introduced here are an extension and adaptation of the algorithm developed by Musallam et. al.[26]. Indeed, the time elapsed between two maximum and minimum are recorded, and moreover the data sampled are filtered to remove the cycle with the amplitude smaller than a set value.

The following procedure is developed to work with the continuous sampling of data and it is based on 3-point counting rule to identify relative maxima and minima (see Figure 12) [27]. Given 3 consecutive values where $d/dt = 0$ (maximum or minimum), the available scenarios are 2: one maximum and 2 minimums (one of them is absolute) or one minimum and 2 maximums (one of them is absolute); since the algorithm deals with real data is impossible that the two maximum or minimum are at the same value; anyway data are filtered to eliminate small oscillation and only well observable cycles are considered. The values that remain in the stack are the absolute maximum or minimum. The other 2 points of the triangle form a full or half cycle and they are removed from the stack after being processed. The procedure continues recursively.

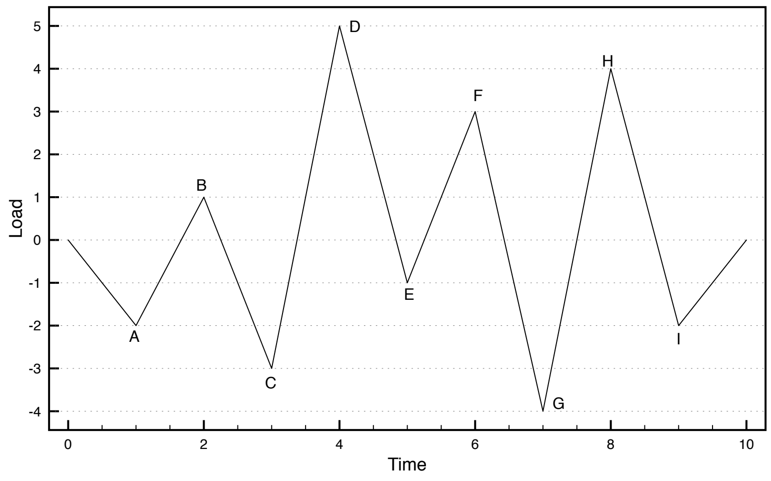
The further steps describe the algorithm in detail.



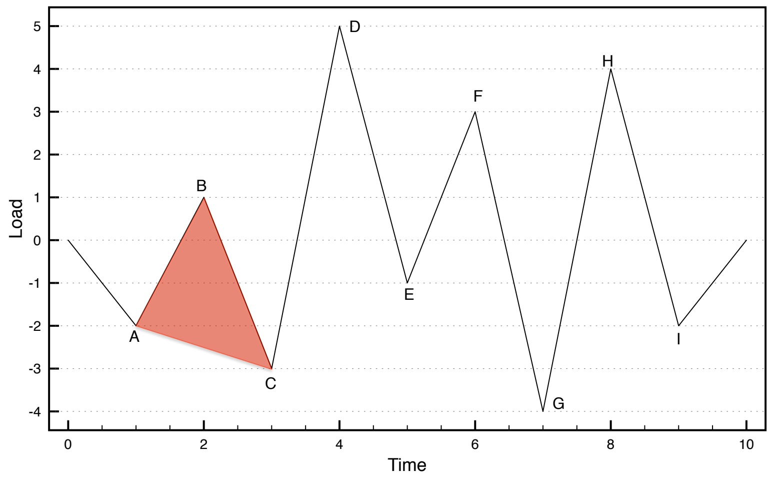
1. The temperature is acquired with a sampling time T_s, and the last few seconds of T are storage. This vector is used to filter the input data and localize a significative slope change. A new maximum o minimum is recorded when:

$$\frac{\mathrm{d}}{\mathrm{dt}}\mathrm{T(t)} = 0 \qquad (4)$$

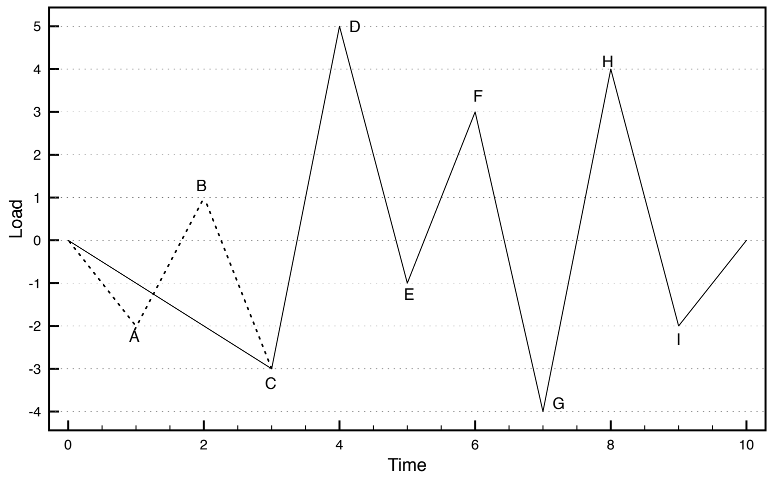
2. Both Temperature and time are saved.



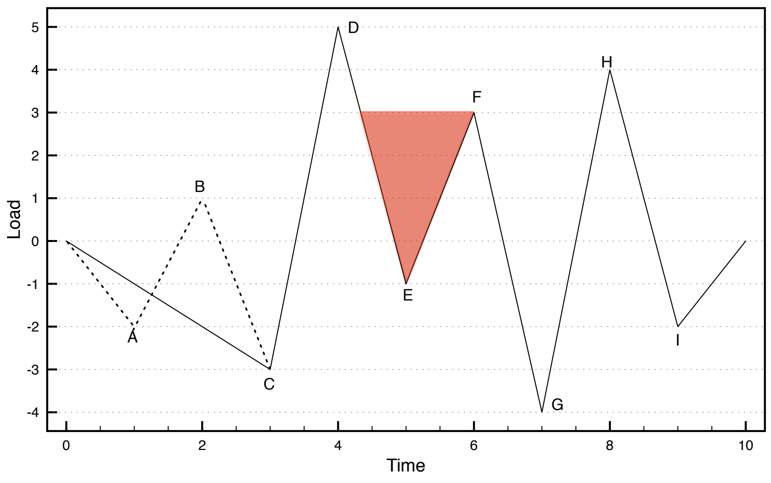
3. When a new max or min is found in 1., it is saved in a stack containing max or min respectively.
4. Each time a new max or min is found, it is compared with the previous saved values on the stack, using the 3 point rule.



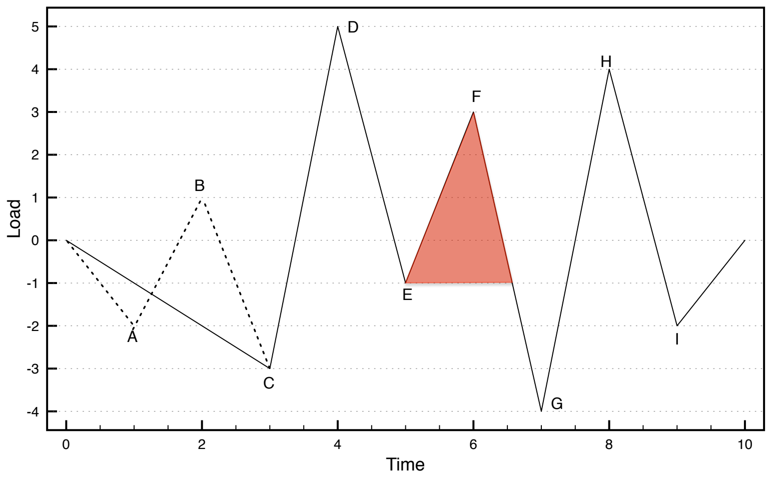
5. A-B-C is a triangle given by 2 min (A,C) and 1 max (B).
6. The two min are compared: │A│<│C│, so C is the new min saved and A and B are discard.
7. The difference $B - C$ gives the amplitude of the cycle and it is count as a half cycles. The duration of the cycles is t_C - t_B and the mean is $(B+C)/2$. The damage is $1/2N_f(\Delta T)$.



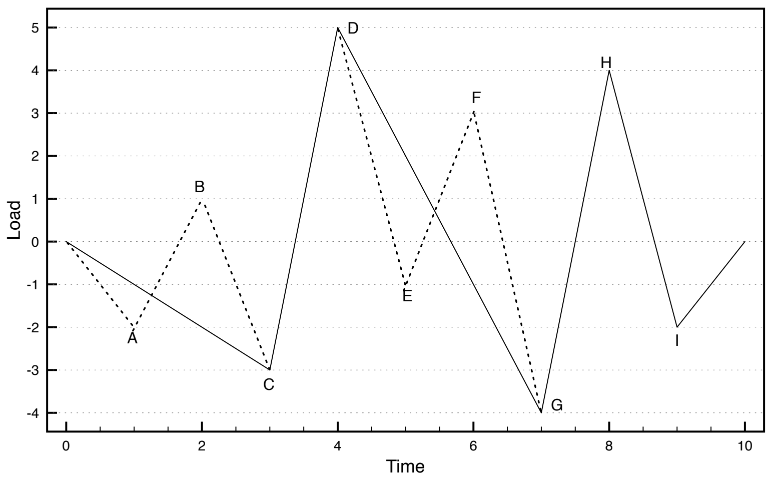
8. The new situation is represented in the figure on left. A and B are removed from the maxima and minima stack.
9. The algorithm proceeds until at least 3 values are available on the maxima or minima stack.



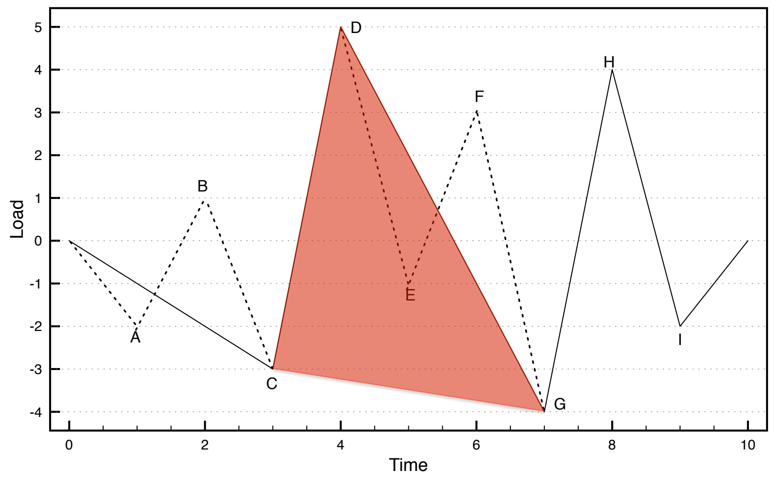
10. E is a min and the triangle C-D-E is analysed as in 5. and 6., in this case |C|>|E| so both C and E remains in the stack.
11. F is a max and the triangle D-E-F is analyse as in 5. and 6. |F|<|D|, so both D and F remains in the stack



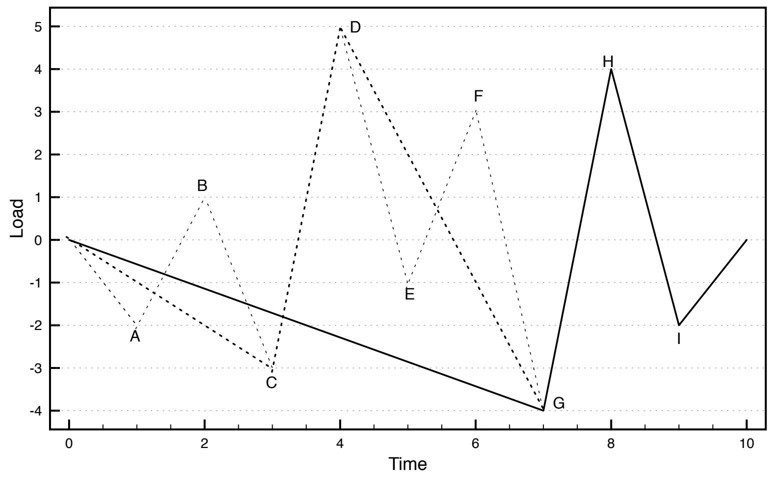
12. G is a min and the triangle E-F-G is analysed. |E|<|G|, so E and F can be discarded.
13. E and F form a full cycle; the amplitude is $F - E$, the period is $2(t_F - t_E)$ and the mean is $(F + E)/2$.
The damage is $1/N_f(\Delta T)$.



14. While 3 elements are still present on the stack (C,D,G) the algorithm continue running and doesn't wait for a new value to add on the stack.



15. The triangle C-D-G is processed as in 6. and 7. G is the new min in the stack because |G|>|C| so C and D are removed from the stack.



16. The final situation is represented in the figure on left.

This on-line version of the Rainflow algorithm is efficient and can be easily integrated in real-time application because it doesn't require high computational resources. Further, compared to off-line versions, the accuracy can be chosen varying the filter features and all the information about the cycles (amplitude, mean, duration) are preserved.

Is important to observe that at discrete time, the evolution of the temperature and the location of the maxima and minima can be done with the calculation of the Ordinary Least Squares and solving equation (4). The formula to evaluate at discrete time the slope of the vector of the last samples is:

$$s = \frac{\sum_{i=1}^{n}(t_i-\bar{t})(T_i-\bar{T})}{\sum_{i=1}^{n}(t_i-\bar{t})^2} \qquad (5)$$

where t represents the time and T the temperature.

4.2.4 Linear Damage accumulation and Coffin-Manson law

The Coffin-Manson law relates the thermal cycles, calculated with one of the method above mentioned, with the number of cycles to failure. As will be describe more in detail in § 5, a bimetallic assumption can be adopted to describe the thermal expansion and the mechanical stress. The elongation ε of the material is related to the temperature swing by:

$$\varepsilon = L(\alpha_1\text{-}\alpha_2)\Delta T \tag{6}$$

where L is the length of the metal junction, α_1 and α_2 are the two thermal expansion coefficient and ΔT is the temperature excursion. It is also possible to consider the relative extension, when α and L are expressed in relative quantities. In general, when an elongation occurs in a metal, it is the sum of the elastic elongation and the plastic elongation; however, taking into account that the elastic elongation is a reversible process while the plastic is dissipative, for ΔT not too small, yield

$$\varepsilon = \varepsilon_{\text{elastic}} + \varepsilon_{\text{plastic}} \approx \varepsilon_{\text{plastic}} \tag{7}$$

The Coffin-Manson law relate the plastic strain with the number of cycles to failure in the following formula:

$$N_f = \varepsilon_{\text{plastic}}^{-q} \tag{8}$$

where q is a positive exponent given by data fitting. Combining equations (6),(7) and (8), the number of cycles to failure are related with the thermal cycles in following equation:

$$\begin{aligned} N_f &= (L(\alpha_1\text{-}\alpha_2)\Delta T)^{-q} \\ &= (L\Delta\alpha)^{-q}(\Delta T)^{-q} \\ &= a(\Delta T)^{-q} \end{aligned} \tag{9}$$

The power function found shows the dependence of N_f on ΔT, expressed by the two parameters a and q. Those one usually are known or is possible to extract them via linear regression of the log-log plot of N_f vs. ΔT available in the component datasheet. Manufacturing, generally provide this information as result of accelerated lifetime tests.

For solder joint an accurate approximation has been proposed by Ciappa [4][28]. Compared to the previous equation (9), the following formula takes in account the thickness and the ductility factor of the solder:

$$N_f = \frac{1}{2}\left(\frac{L\Delta\alpha\Delta T}{\gamma x}\right)^{1/c} \tag{10}$$

Moreover, when the Arrenius corrector factor is important on the evaluation of the numbers of cycles to failure the equation (9) can be rewrite in the following manner:

$$N_f = a(\Delta T)^{-q} \exp\left(-\frac{E_a}{k_B T_m}\right) \tag{11}$$

where the first part of formula is the same terms of (9), and the second term is an exponential dependence given by the Arrenius equation: E_a is the activation energy, k_B is the Boltzman's constant and T_m is the mean temperature of the cycle.

5 Lifetime prediction on constitutive equations

The lifetime prediction of IGBT modules can be preformed in more than one technique. The previously presented methods don't analyse in detail the mechanical evolution of the expansion and compression process, rather they just confine in the estimation on temperature range and duration.

In general, the thermal expansion of a solder can be modelled by a bimetallic approximation of the thermo-mechanical stress at the interface of two different materials. Simplifying the problem in a uniaxial system, the elongation of a material is linear dependent function of the temperature variation; since the temperature in which the elongation is zero exists, is possible to reformulate equation (6) is this way:

$$\varepsilon = \mathrm{L}\Delta\alpha(\mathrm{T}\text{-}\mathrm{T}_0) \tag{12}$$

where T_0 is the temperature of null elongation.

It is worth to note that until now there are no universal descriptions of phenomena that involve the stress-stain relationship, in particular, in complex material like the solder joint or the wire bonds. Most of the works found in literature solve the mechanical behaviour using simulation tools like ANSYS. Other times the approach used is a simplification of the equations to valuate the stress-strain characteristic, which evaluate only some principal aspects [29].

The material's elongation causes different phenomena in terms of applied stress. There are at least three distinct aspect of strain-stress response important in the case of cyclic thermo-mechanical mechanisms. These are elastic, plastic and creep behaviour. The equation, assumption and law describing these mechanical aspects are expressed by time and temperature dependent equations which are valid in low-cycle fatigue regime [30].

The further chapters describe the deformation behaviour introduced by temperature cycling through the separation of the constitutive equations.

5.1.1 Elasticity, plasticity, creep

The physical deformation process operates at atomic level modifying the microstructural domains in a 3-D system [31]. At macroscopic level and considering just a uniaxial description, the material's behaviour has two main stress-strain responses, as depicted in Figure 14.

The first behaviour in the elongation process is the elastic strain. In Figure 14 it is represented by the first portion of the curve. The mathematical description the elastic response is given by:

$$\varepsilon_{\mathrm{elastic}} = \frac{1}{\mathrm{E}}\sigma_{\mathrm{elastic}} \tag{13}$$

where E represent the Young's module of the material.

The modulus of elasticity is a non-linear temperature dependent parameter [32]; the first order approximation yield:

$$\mathrm{E} = \mathrm{E}_0\text{-}\mathrm{E}_1\mathrm{T} \tag{14}$$

where E_0, E_1 are material's parameters and T is the temperature in ºC.

The elastic process, both in elongation and compression, is energy reversible and linear. When a thermal cycle is applied in the elastic behaviour the initial and final conditions are the same; this means that no deformation occurs to the material and any damage is accumulated.

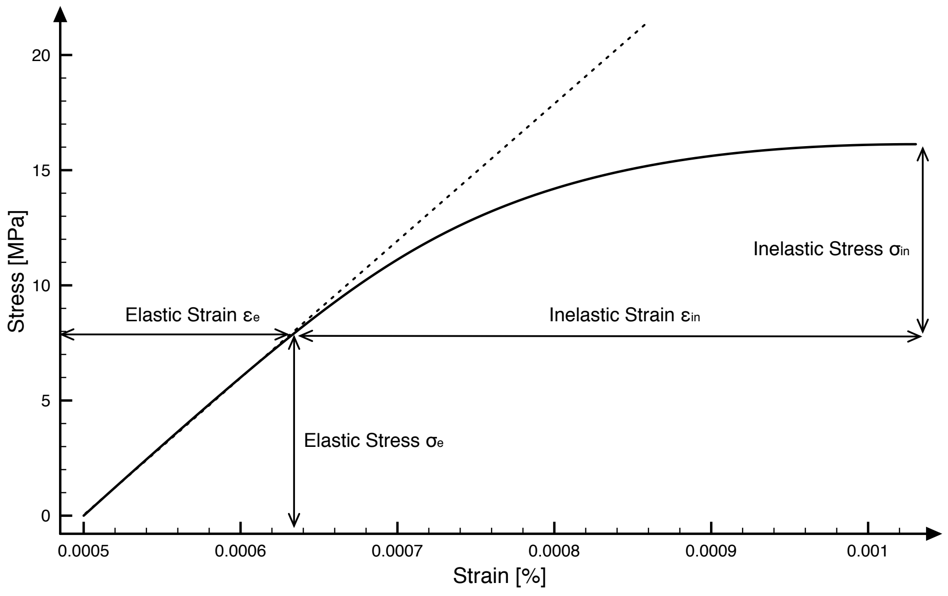


Figure 14. Representation of Elastic-plastic stress–strain curve plot. Dash line represents the ideal elastic behaviour.

When the strain exceeds the elastic range, the inelastic deformation begins; this is represented by the second portion of the curve in Figure 14. [32] The limit of elastic-inelastic domain is not univocally defined, and this limit goes under the name tensile yield. The cyclic loading and unloading process in this region of the strain is an irreversible process. The strain elongation and relaxation of the same quantity doesn't return to the initial stress condition. This statement is explained in Figure 15, where the strain difference between the beginning and the end is represent in red.

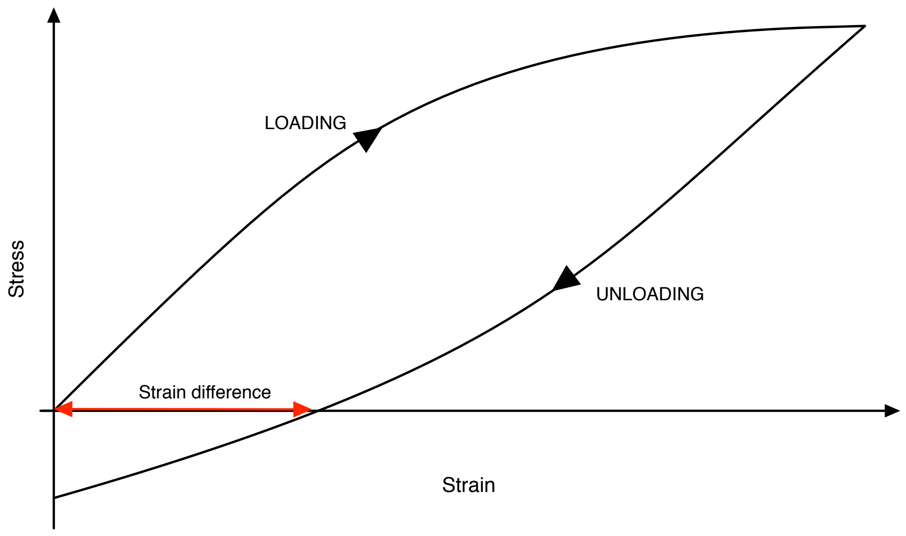


Figure 15. Loading and unloading process; effect of irreversible process due to plastic deformation.

The inelastic behaviour is more complex to describe in terms of equation. It is possible to compound the inelastic strain the sum of time-independent plastic strain and the time and temperature dependent creep strain.

The time-independent plasticity describes the large and fast thermo-mechanical strain process. The relationship strain-stress is a power law empirical formulation:

$$\varepsilon_p = C_1 \left(\frac{\sigma}{G}\right)^{m_p} \tag{15}$$

where σ is the applied stress, G is the shear modulus, C_1 and m_p are the plastic strain coefficient and the stress sensitivity of the plastic strain respectively. The main damage cumulated occurs in this material's behaviour.

The creep strain is a time-dependent process that has high influence in slow dilatation or compressive strain at maximum temperature; it can be split in the steady-state creep and in transient or primary creep. The following equation describes the creep behaviour.

The steady-state relation is:

$$\varepsilon_s = \dot{\varepsilon}_s t \tag{16}$$

where t is the time and the steady-state strain rate is given by:

$$\dot{\varepsilon}_s = C_1 \frac{G}{T}\left[\sinh\left(\alpha \frac{\sigma}{G}\right)\right]^{n_s} \exp\left(-\frac{E_a}{k_B T}\right) \tag{17}$$

where C_1, α, n_s are material parameter, G is the shear modulus, T is the temperature in Kelvin, σ is the stress, E_a is the activation energy and k_B is the Boltzman's constant.

The transient creep is has the following equation

$$\varepsilon_T = C_3[1\text{-}\exp(\text{-}B\dot{\varepsilon}_S t)] \tag{18}$$

where C_3 and B are material's parameters, t is the time and $\dot{\varepsilon}_s$ is the steady-state creep rate previously found.

5.2 Constitutive equations

The equations presented above describe the scenario in the expansion and compression mechanism due to thermo-mechanical events. The mechanical fatigue in solder joints can be described with this response taking into account that the deformation energy introduced per cycle is proportional to the area inside the stress and strain hysteresis curve [31]. Indeed it is possible to observe that a loading cycle inside the elastic behaviour doesn't introduce any damage, since the movement is on a straight line; when the strain reaches the inelastic region (in plastic or creep regime) it contributes to wrap the structure.

An important assumption that has to be done is that the stress is the same in all the 3 behaviour, while the strain is the sum of all the contribution above explained. Thus, yields:

$$\varepsilon_{tot} = \varepsilon_{elastic} + \varepsilon_{inelastic} = \varepsilon_e + \varepsilon_p + \varepsilon_S + \varepsilon_T \tag{19}$$

Substituting equations (13), (15), (16) and (18) into (19) the full constitutive model is achieved:

$$\varepsilon = \frac{\sigma}{E_0\text{-}E_1 T} + C_1\left(\frac{\sigma}{G}\right)^{m_p} + C_1\frac{G}{T}\left[\sinh\left(\alpha\frac{\sigma}{G}\right)\right]^{n_s}\exp\left(\text{-}\frac{E_a}{k_B T}\right)t + C_3[1\text{-}\exp(\text{-}B\dot{\varepsilon}_S t)] \tag{20}$$

Equation (20) describes the relation of strain as a function of applied stress, which is assumed constant in each material's behaviour. The mechanical system of solder joints here considered requires as input the strain given by thermal expansion in equation (12) to compute the corresponding stress level. It is not trivial to obtain a formulation of equation (20) like $\sigma = f(\varepsilon)$ that satisfies the situation explained.

For this reason other methods have been developed during years to evaluate the stress given the strain [32][33]. For example the Anand's model.

Solder/ Parameter	G_0 (10^4 MPa)	G_1 (10^4 MPa)	C_1 (K/s/MPa)	α	n_s	Ea (ev)	B	C_3	C_2	m_p
60Sn40Pb	1.31	0.559	28.720	1300	3.3	0.548	440	0.02	$2.3 \cdot 10^{13}$	5.6
62Sn36Pb2Ag	1.31	0.559	14.344	1300	3.3	0.548	137	0.037	$6.6 \cdot 10^{7}$	4.5
96.5Sn3.5Ag	1.931	0.698	$7.09 \cdot 10^{-2}$	1300	5.5	0.4	147	0.086	$2.1 \cdot 10^{11}$	4.4
97.5Pb2.5Sn	0.896	0.103	$7.31 \cdot 10^{9}$	1200	7.0	1.2	138	0.115	$2.8 \cdot 10^{10}$	4.0

Table 4. Material parameters of separated constitutive equation for solders [34].

5.3 Anand model

A method that has been tested in this work is the Anand's viscoplastic model [35][36]. This model describes the inelastic deformation of the system through empirical equations. The set of constitutive equations proposed by Anand explains the physical phenomena in terms of temperature sensitivity, strain rate history effects, strain hardening and the restoration process of the dynamic recovering. The study carried out by Wang et. al. [32] shows the equivalence of Anand method with separated constitutive equations.

As described in many articles, the equation of Anand's model are here reported and they use an internal variable s to take track of the deformation resistance.

$$\sigma = c \cdot s \tag{21}$$

$$c = \frac{1}{\xi}\sinh^{-1}\left[\left(\frac{\dot{\varepsilon}_P}{A}e^{Q/RT}\right)^m\right] \quad \Rightarrow \quad \dot{\varepsilon}_P = A\exp\left(-\frac{Q}{RT}\right)\left[\sinh\left(\xi\frac{\sigma}{s}\right)\right]^{1/m} \tag{22}$$

$$\dot{s} = \left\{h_0\left|1-\frac{s}{s^*}\right|^a \cdot \text{sign}\left(1-\frac{s}{s^*}\right)\right\} \cdot \dot{\varepsilon}_P \quad \text{where} \quad s^* = \hat{s}\left[\frac{\dot{\varepsilon}_P}{A}\exp\left(\frac{Q}{RT}\right)\right]^n \tag{23}$$

Note in equation (22) the dependence of temperature is included through Arrenius term. Moreover the internal variable evolution is given by the differential equation $\dot{s} = f(\sigma, s, T)\dot{\varepsilon}_P$ as described in equation (23).

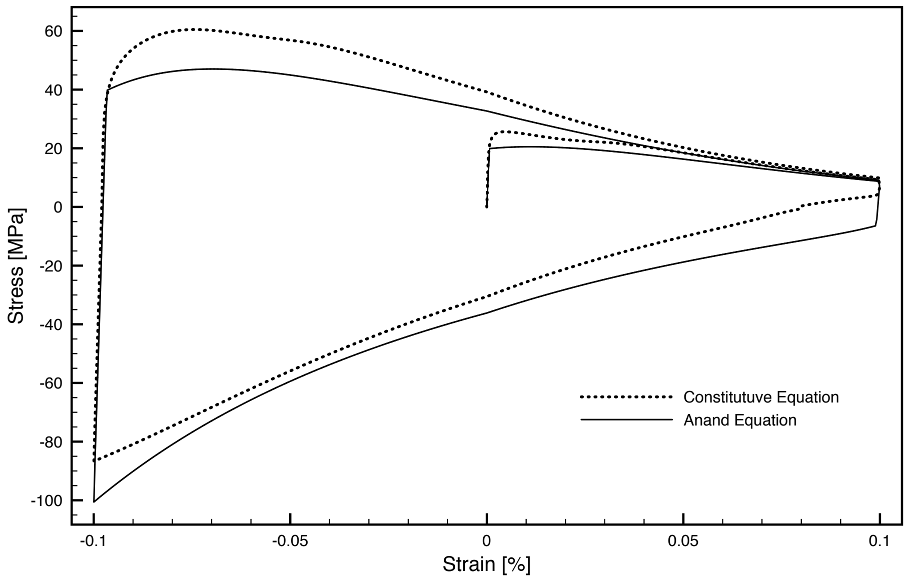


Figure 16. Anand model and separated constitutive equations in comparison. The approximation is good enough for deep relative strain and elongation. The area inside the cycle is in both the models is with good approximation the same.

In Figure 16 a typical cycle is show, where the constitutive equations are solved with Anand's model. The approximation introduced is good especially for relative strain of 0.1% or more.

The main limitation discovered in this model is the mechanical fatigue description only of the inelastic strain behaviour. The viscoplastic approach adopted in this work shows that the crucial point in the strain evaluation is the tensile yield region, where the elastic-inelastic transition is not defined precisely (see Figure 17).

Solder/ Parameter	A (s^{-1})	Q/R (K)	m	ξ	$\hat{s}$ (MPa)	n	h_0 (MPa)	a	s_0 (MPa)
60Sn40Pb	$1.49 \cdot 10^7$	10830	0.303	11	80.42	0.0231	2640	1.34	56.33
62Sn36Pb2Ag	$2.31 \cdot 10^7$	11262	0.303	11	80.79	0.0212	4121	1.38	42.32
96.5Sn3.5Ag	$2.23 \cdot 10^7$	8900	0.182	6	73.81	0.018	3321	1.82	39.09
97.5Pb2.5Sn	$3.25 \cdot 10^7$	15583	0.143	7	72.73	0.0043	1787	3.73	15.09

Table 5. Material parameters of viscoplastic Anand model for solders

In the tests conducted, based on Anand's model, the elastic and inelastic behaviour have been developed separately, and keeping the strain yield a fixed transition point between the two conditions; this limit value is given is s_0 in Table 5. This approximation is too inaccurate to estimate with good precision the damage accumulated in solder joint, because most of the thermal cycles in IGBT modules have typical range at the boundary of elastic strain.

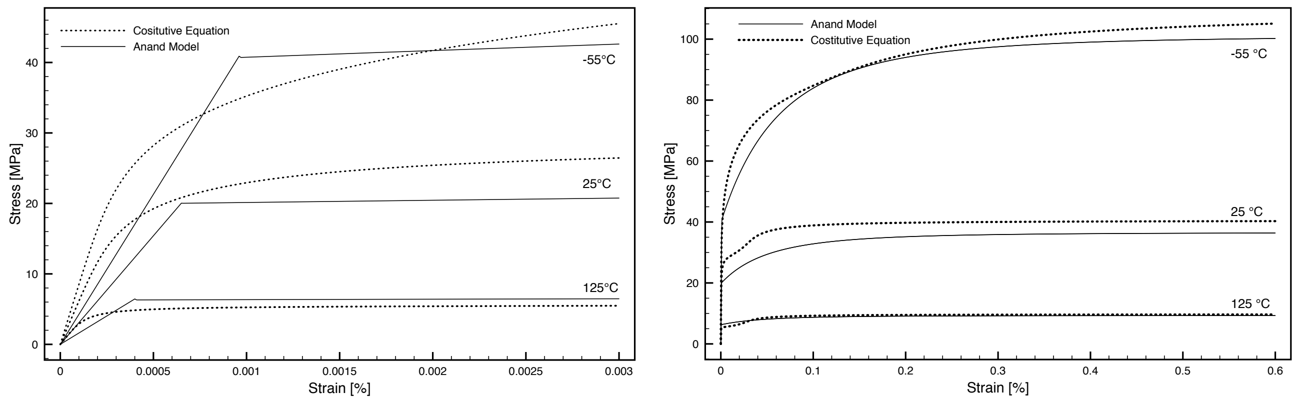


Figure 17. Comparison between Anand's model and constitutive equation of separated equation. Is possible to observe that Anand's model is an optimal approximation for high strain (figure right), while for small strain (figure left) the approximation is poor especially in the range of interest for IGBT module.

The problem met in this model cannot be ignored because the accuracy of the measurement shows high sensitivity to the transient elastic-inelastic region. For this reason it has been chosen to maintain the complex model and solve it in through numerical approximation.

5.4 Newton solution of constitutive equation

The explicit solution of equation (20) in form $\sigma = \mathrm{f}(\varepsilon)$ is impossible to find. For this reason, the only method available to know the stress σ given the strain ε is the numerical solution of equation $\mathrm{g}(\sigma) = \varepsilon - \mathrm{f}(\sigma) = 0$ using Newton-Raphson method.

The equation recursively implemented to compute the stress are the following:

$$\sigma_{n+1} = \sigma_n - \frac{\mathrm{g}(\sigma_n)}{\mathrm{g}'(\sigma_n)} \tag{24}$$

The algorithm computes, given the initial point σ_0, a convergence sequence $\sigma_1, \sigma_2 \ldots \sigma_n$ until the difference $|\sigma_n - \sigma_{n-1}| < \epsilon$, where ϵ is the approximation tolerance.

5.4.1 Discretization of constitutive equations

The mathematical formulation of Newton-Raphson explained in § 5.4 is solved using Matlab with the following discretization:

$$\sigma_{n+1} = \sigma_n - \frac{\mathrm{g}(\sigma_n)}{\mathrm{g}(\sigma_n + h_0)/h_0} \tag{25}$$

where h_0 is the incremental step. The numerical solution of the Newton method here implemented, takes also into consideration the possibility to change the parameters ϵ, h_0 and σ_0 to force the convergence of the sequence.

5.4.2 Numerical integration of constitutive equation

The full equation and the deformation work have been computed in deferential way, taking into account that it is possible to derive equation (19) at both side and write:

$$d\,\varepsilon = d\,\mathrm{f}(\sigma) \tag{26}$$

Thus, the integration of differential equation (26), using newton method, yields to compute the stress value at discrete time. In this way is possible to write equation (26) in following discrete-time:

$$\frac{\Delta\varepsilon}{\Delta Ts} = \frac{\Delta\mathrm{f}(\sigma)}{\Delta Ts} \tag{27}$$

where T_s is the sampling time. Integration of this equation gives:

$$\sigma(i) = \sum \Delta\mathrm{f}(\sigma) = \sum \mathrm{f}(\sigma_i) - \mathrm{f}(\sigma_{i-1}) = \sigma(i-1) + \Delta\mathrm{f}(\sigma) \tag{28}$$

where $\sigma(i)$ is the stress at i-th time step.

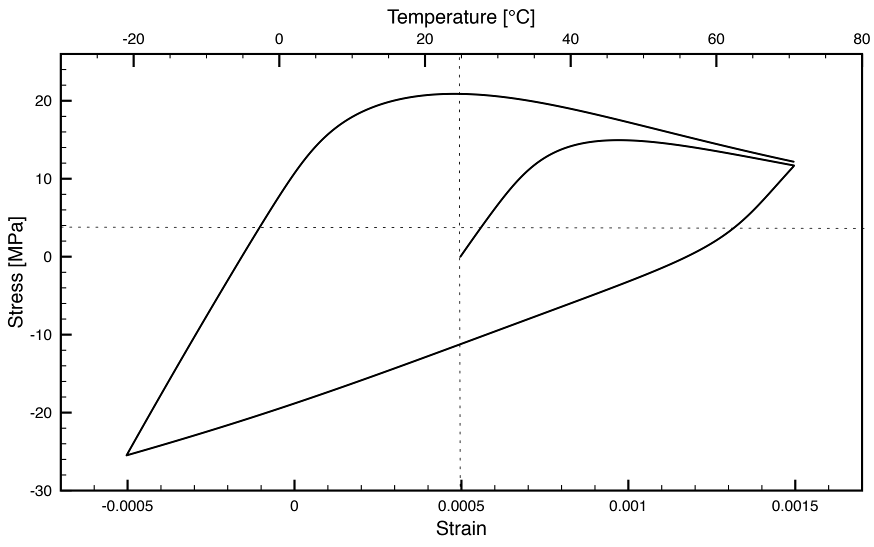


Figure 18. Representation of a typical cycle implemented solving the separated constitutive equation. The initial condition, at 0 stress level, considered in this case is at 25°C.

At the same time the dissipated energy due to the fatigue in the solder joint is calculated as the area contained inside the hysteresis loop described in Figure 15 and Figure 18. The method used for integration is the trapezoidal rule, as reported in following equation (29) and in Figure 19.

$$W(i) = \int \sigma \, d\varepsilon = W(i\text{-}1) + \frac{(\sigma(i) + \sigma(i\text{-}1)) \cdot \Delta\varepsilon}{2} \tag{29}$$

where W(i) is the cumulated energy dissipated to deform the solder layer at i-th time step.

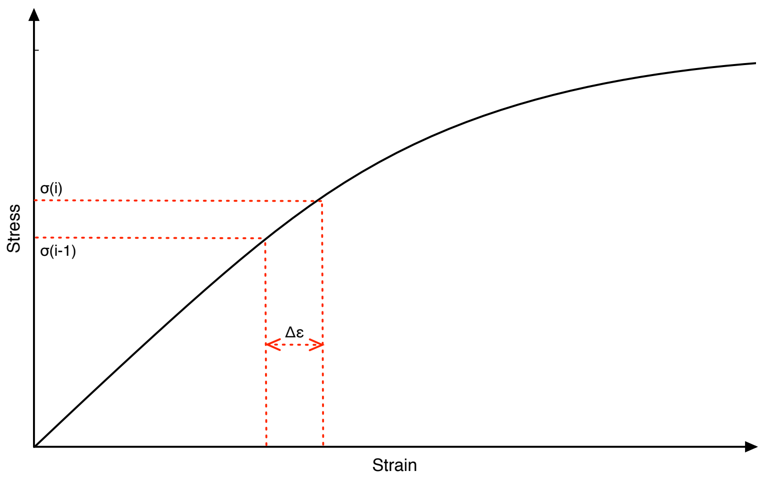


Figure 19. Trapezoidal rule for integration of mechanical stress.

The described method and solution via Newton method is working in most of situation, but it is necessary to underline that the algorithm's convergence is not guaranteed. Even though in the Newton solution it is possible to change the approximation during the execution, it has been observed that many times the convergence takes many iteration to find the solution and others it never converge due to numerical issue related to newton algorithm or the initial convergence point

σ_0. Furthermore, since the procedure has to be implemented on a microcontroller and real-time evaluation is required, this implementation is used only to generate a look-up-table to evaluate the stress given the strain and temperature values. The evaluation of the look-up table is a procedure that is done off-line and is allowed a slow convergence of the algorithm.

5.5 Look-up table implementation

The newton method previously descripted presents possible convergence issues, which in a real-time environment are not acceptable. A convergence issue means a wrong evaluation of the stress value or, in the worst case, the slowing down or the stuck of the program execution.

The real-time execution requires fast performance and parameter evaluation. Given that the newton method is an inadequate implementation, a look-up-table realization has been carried out. Even though a look-up table requires a lot of memory compared to the algorithmic version, the execution speed is many times faster, thus optimal for real-time analysis. Even in thermal and electric circuit simulations this method is widely used [37][38][39].

The look-up table is a map that takes as input values the temperature, the strain and the strain rate, and it returns the corresponding approximated stress value, $\hat{\sigma} = \mathrm{m}(T, \varepsilon, \dot{\varepsilon})$, by inspection of a 3D-matrix containing all the possible values. The table so implemented is a multidimensional array that replaces runtime computation with a simpler indexing operation, as explained in Figure 20.

The realized table is a discretization of all possible stress values in the input ranges of this problem. The temperature range considered goes from -50 to 150 ˚C, the strain range is 0 to 0.003% of the length of the silicon chip (e.g. 12 mm silicon chip elongate up to 36µm at 150 ˚C), and the strain rate from 0.01 ˚C/s to 100 ˚C/s in logarithmic scale.

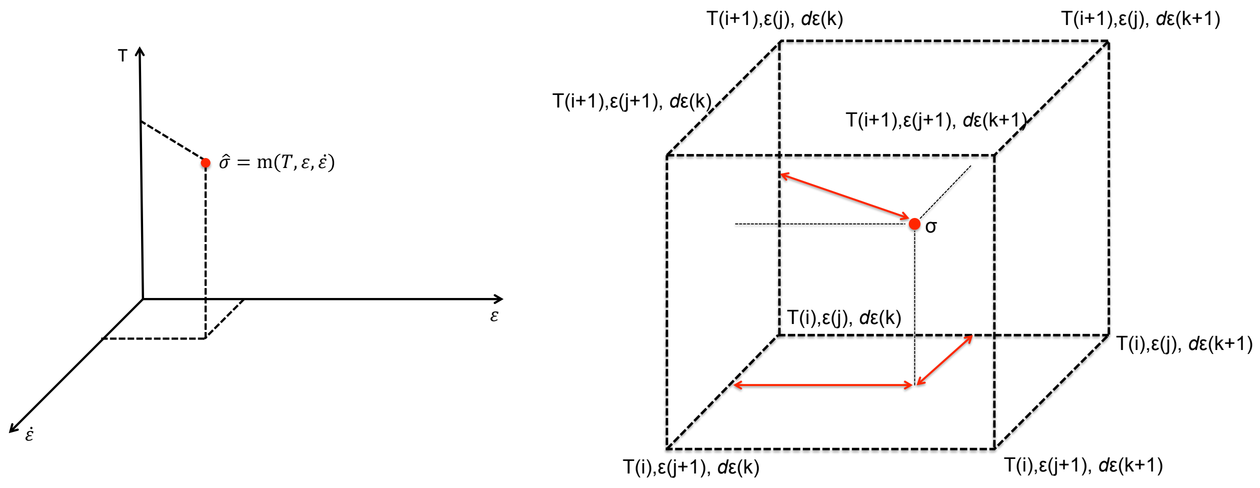


Figure 20. Look-up table working principle and interpolation.

The map works locating on a 3D matrix the corresponding value of the stress given $T, \varepsilon, \dot{\varepsilon}$. Since the input value are real number and the index of matrix are positive integer a second map is necessary. The matrix coordinates that identify the corresponding stress level have the form $\hat{\sigma} = \mathrm{m}(i, j, k)$, where i, j and k are computed as following:

$$\mathrm{i} = \lfloor (\mathrm{T}\text{-}\mathrm{T}_{\min})/\Delta\mathrm{T} \rfloor \tag{30}$$

$$\mathrm{j} = \lfloor \varepsilon/\Delta\varepsilon \rfloor \tag{31}$$

$$\mathrm{k} = \lfloor \log(\dot{\varepsilon}) \cdot \Delta\dot{\varepsilon} + \mathrm{q}_{\dot{\varepsilon}} \rfloor \tag{32}$$

where $\lfloor\cdot\rfloor$ operator is the floor rounding function, T_{min} is the minimum temperature available in the array (T_{min}=-50 °C), ΔT is the discretization step of the temperature, $\Delta\varepsilon$ is the discretization on the strain, is the discretization in the strain rate and $q_{\dot{\varepsilon}}$ is a linearization parameter. Note that the map realized with equations (30),(31) and (32), calculates an approximated value that corresponds to the rounded input values:

$$(\mathrm{T}\text{-}\epsilon_{\mathrm{T}}, \varepsilon\text{-}\epsilon_{\varepsilon}, \dot{\varepsilon}\text{-}\epsilon_{\dot{\varepsilon}}) \longrightarrow (\mathrm{i},\mathrm{j},\mathrm{k}) \longrightarrow \hat{\sigma} \tag{33}$$

where $\epsilon_T,\ \epsilon_\varepsilon,\ \epsilon_{\dot{\varepsilon}}$ are the difference obtained from the real input values and the look-up table. To increase the accuracy of this approximation the linear interpolation has been applied. The value returned by the look-up table is a rough approximation; the real stress value can be pinpointed inside the cube identified by the adjacent cells of the matrix. Consider the cube in Figure 20, the value returned by the matrix is $\hat{\sigma} = \mathrm{m}(i_T, j_\varepsilon, k_\varepsilon)$, which is the vertex given by rounding the inputs; since the real stress is the red point inside the cube, the eight vertex of the cube are calculated and tri-linear interpolation of the cube has been carried out. The cube have the vertex (i, j, k) in bottom/left/back corner, and all the adjacent vertex are (i, j, k + 1), (i, j + 1, k), (i + 1, j, k), (i + 1, j + 1, k), (i + 1, j, k + 1), (i, j + 1, k + 1), (i + 1, j + 1, k + 1). The searched value is the red point within the cube, and assume it have x, y, z coordinates calculated from $\epsilon_T,\ \epsilon_\varepsilon,\ \epsilon_{\dot{\varepsilon}}$. The linear interpolation of value at x, y, z is:

$$\begin{aligned}\sigma(\mathrm{T},\varepsilon,\dot{\varepsilon}) = {} & \sigma(\mathrm{i},\mathrm{j},\mathrm{k}) \cdot (1\text{-x}) \cdot (1\text{-y}) \cdot (1\text{-z}) + \\ & \sigma(\mathrm{i}+1,\mathrm{j},\mathrm{k}) \cdot \mathrm{x} \cdot (1\text{-y}) \cdot (1\text{-z}) + \\ & \sigma(\mathrm{i},\mathrm{j}+1,\mathrm{k}) \cdot (1\text{-x}) \cdot \mathrm{y} \cdot (1\text{-z}) + \\ & \sigma(\mathrm{i},\mathrm{j},\mathrm{k}+1) \cdot (1\text{-x}) \cdot (1\text{-y}) \cdot \mathrm{z} + \\ & \sigma(\mathrm{i}+1,\mathrm{j},\mathrm{k}+1) \cdot \mathrm{x} \cdot (1\text{-y}) \cdot \mathrm{z} + \\ & \sigma(\mathrm{i},\mathrm{j}+1,\mathrm{k}+1) \cdot (1\text{-x}) \cdot \mathrm{y} \cdot \mathrm{z} + \\ & \sigma(\mathrm{i}+1,\mathrm{j}+1,\mathrm{k}) \cdot \mathrm{x} \cdot \mathrm{y} \cdot (1\text{-z}) + \\ & \sigma(\mathrm{i}+1,\mathrm{j}+1,\mathrm{k}+1) \cdot \mathrm{x} \cdot \mathrm{y} \cdot \mathrm{z}\end{aligned} \tag{34}$$

Note that the z coordinates corresponding to the strain rate $\dot{\varepsilon}$, is linearized from logarithmic scale use same equation as in (32).

Some tests have been developed to find the best trade-off between the table discretization and the memory usage. The matrix counts 4.8 million of stress value and the memory usage in binary format is 35 Mbyte (floating point value of MATLAB).

6 Thermal management

The thermal analysis of IGBT modules is necessary to understand the behaviour of the device under operation and to characterize the fatigue cycles due to thermal variations. The heat generated by the device influence each layer of the module. In this chapter the thermal analysis is carried out through *compact thermal modelling* and is explain how the heat generated by the silicon propagate across the device.

The typical temperature profile of junction under operation is plotted in Figure 21; the temperature rising and falling edges are likely approximated with the exponential function.

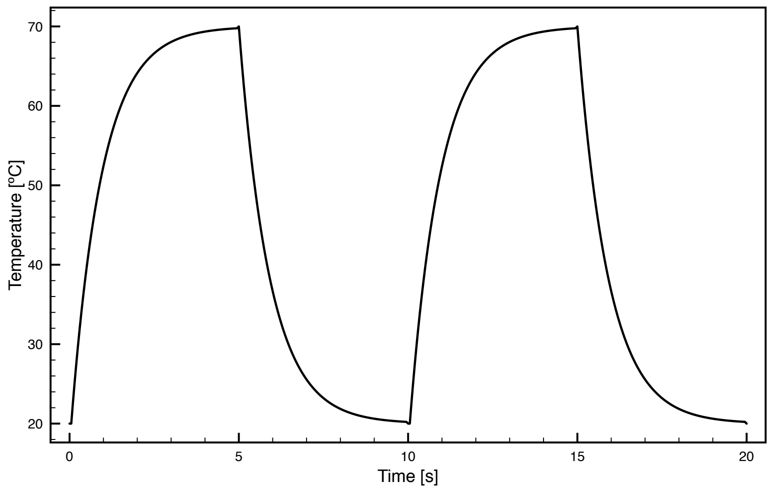


Figure 21. Heating and cooling process of the IGBT junction on silicon chip; the temperature profile is in good approximation an exponential function.

6.1 Compact modelling

The heat distribution in the device can be studied with complex tools (like ANSYS or COMSOL MultiPhysics) to have a full thermal analysis. This software can indeed simulate the dynamic response of the system due to a heating propagation in each point of the device. The graphical representation of the heat distribution can be observed in Figure 24. The simulation of the IGBT module in Figure 24 takes about 20 minutes to analyse a transient period of 15 seconds (considering also that a quarter of the system is simulate to take advantage of the structure symmetry). Of course, the accuracy and the number of point generated by the simulation are overmuch for the lifetime estimation, since it is not the single temperature in one point that contributes to the whole damage. Under the hypothesis of uniform temperature at layer's interface it is possible to evaluate only some relevant temperature in damage analysis.

To accurately predict the temperature of the IGBT module only in few critical points, (e.g., junction, substrate and baseplate) the compact thermal model (CTM) is adopted. It is a behavioural model that aims to evaluate the temperature evolution with some assumption but without computational effort. The compact model is an abstraction, based on geometry, material properties and the environment, of the thermal response of a multi-layered structure. The simplifications introduced by this method are the uniaxial heat propagation and the isothermal boundary condition. It is moreover assumed that, at each interface plane, the temperature and the heat propagation are uniform and the border effect can be neglected.

Most CTM approaches use a thermal resistor and capacitor network to evaluate an equivalent electrical model that follows Ohm's law. Thus it is possible to have an equivalence between the thermal parameters and the electrical concepts. In Table 6 the thermal-electrical similarity is summarized; in this analogy, if the thermal properties of material are known, interpreting the current as power and voltage as temperature difference, the thermal network can be handled as an electrical network.

Type	Resistance	Potential	Energy	Capacitance
Electrical $R=V/I$	R= Resistance Ohms [Ω]	V=Potential difference Volt [V]	I=Current Amperes [A]	C=Capacitance Farad [C/V]
Thermal $R_{th}=K/W$	R_{th}= Thermal resistance [K/W]	K=Temperature difference [Kelvin]	W=Dissipated power Watt [W]	C_{th}=Thermal capacitance [J/K]

Table 6. Fundamental parameters analogy between thermal and electrical model. Using these analogies is possible to convert a thermal model into an equivalent electrical network that can be solved using Ohm's law.

The thermal impedance, $Z_{th}(t)$, is the time-dependent parameter that describes the temperature evolution in each layer. It can be computed using the equivalent electrical network or via FEA (*Finite Element Analysis* or *Finite Elements Method*).

6.1.1 Foster and Cauer thermal network

There are two techniques to model each layer of the IGBT module with the electrical equivalence, the Foster's representation and the Cauer's [40].

The Foster electrical representation is an empirical formulation of the thermal impedance. The series of RC-parallel elements describe the transport effect of the heat in each layer; ideally it is possible to find the resistance and capacitance value that fit the thermal impedance of junction to case path. In Figure 22 a schematic of the electrical equivalence of an IGBT module is described. For each layer, a couple of RC terms identify the contribution of the layer to the total thermal impedance Z_{th}.

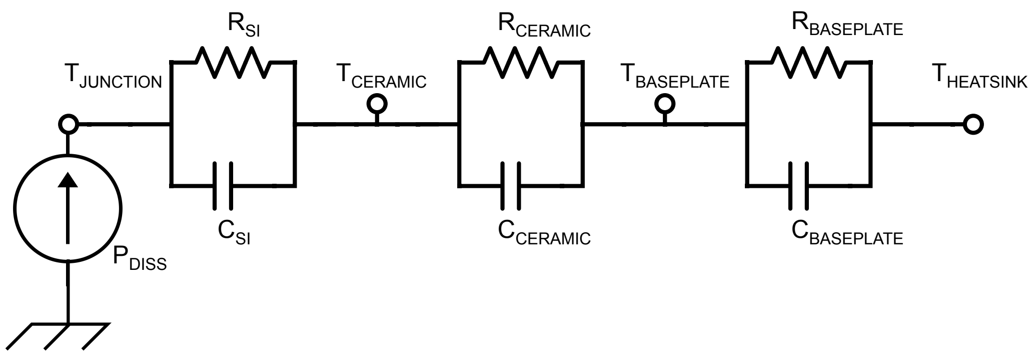


Figure 22. Foster electrical representation of thermal impedance of IGBT power module. Each RC block approximate the layer thermal behaviour. The RC value are obtained by fitting the thermal response of the system.

It is worth to underline that the RC terms do not have any physical meaning, they are obtain by fitting the thermal response of the structure to a heat pulse. The number of terms are irrelevant because is possible to obtain a better fitting of the curve using more RC element. Assuming that only a linear combination of exponential terms gives the total thermal impedance, the sum of all the RC, yields

$$Z_{TH(j\text{-}c)} = \sum_{i}^{n} R_i (1\text{-}e^{-t/\tau_i}) \tag{35}$$

where n is the number of RC-terms considered, τ_i , is the time constant of i-th RC-term and is $\tau_i = R_i C_i$.

With the Cauer's representation a physical analogy is possible, in particular it describes the heat propagation through the layers and the introduced time delays. The equivalent electrical network of resistor and capacitor is indicated in Figure 23. This model is more suitable when the thermal material's characteristics of the layers are known.

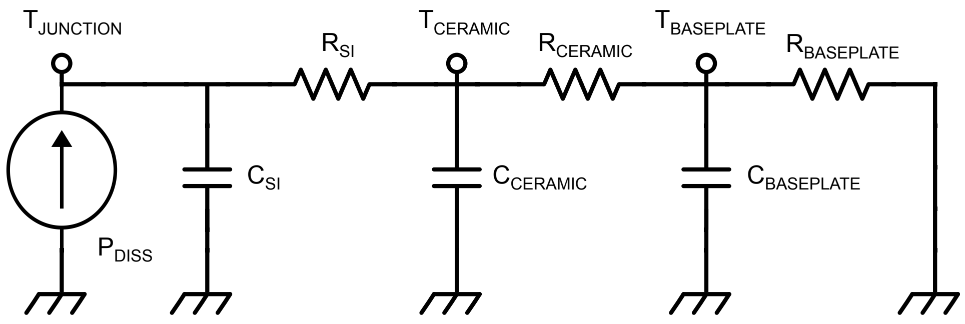


Figure 23. Cauer representation of the IGBT module. Each layer is connected through a capacitance to ground; this allows to consider the delay in the heat propagation.

Note that the Ri and Ci values of the two models are not the same.

6.1.2 Power losses

To evaluate the heat generated in power module it is necessary to understand the power dissipation mechanism. The heat is dissipated from the silicon junction of both IGBT and freewheeling diode, where it is assumed that a uniform volume heat distribution occurs. The power losses are divided in the following main contributes: conduction losses (*P_{cond}*) due to series resistance, switching losses (*P_{sw}*), and locking (leakage) losses (*P_b*), which normally being neglected [41][42]. Thus, is possible to write:

$$P_{diss} = P_{cond} + P_{sw} + P_l = P_{cond} + P_{sw} \tag{36}$$

6.1.2.1 Conduction Losses

The instantaneous value of the IGBT conduction losses is:

$$p(t) = v_{CE}(t) \cdot i_C(t) \tag{37}$$

where v_{ce} is the collector-emitter voltage an i_c is the collector current. Assuming that the collector emitter voltage can be approximated with a DC constant source and a current dependent source, is possible to write V_{ce} as:

$$v_{CE}(t) = V_{CE0} + R_s i_C(t) \tag{38}$$

Substituting $v_{CE}(t)$ found in (38) into (37), yields:

$$p(t) = v_{CE}(t) \cdot i_C(t) = V_{CE0} \cdot i_C(t) + R_s {i_C}^2(t) \tag{39}$$

The same result is obtained for the diode. The average losses in IGBT and diode can be computed as follows:

$$P_{IGBT} = \frac{1}{T_{sw}} \int_0^{T_{sw}} p_{IGBT}(t)dt = \frac{1}{T_{sw}} \int_0^{T_{sw}} V_{CE0} \cdot i_C(t) + R_s {i_C}^2(t)\,dt = V_{CE0} \cdot \hat{I} + R_s \cdot {I_{RMS}}^2 \tag{40}$$

where $\hat{I}$ is the average current value and ${I_{RMS}}^2$ is the root-mean-square of IGBT current. In the same way is calculated the dissipated power on freewheeling diode:

$$P_D = \frac{1}{T_{sw}} \int_0^{T_{sw}} p_D(t)dt = \frac{1}{T_{sw}} \int_0^{T_{sw}} V_{D0} \cdot i_D(t) + R_D {i_D}^2(t)\,dt = V_{D0} \cdot \widehat{I_D} + R_D \cdot {I_{RMS}}^2 \tag{41}$$

The electrical parameters R_s and R_D can be extract from the device's datasheet, while V_{CE0}, $\hat{I}$, V_{D0}, and $\hat{I}_D$, can be measured in real-time or depends from inverter topology.

6.1.2.2 Switching losses

The switching losses are due to charging and discharging the overall capacitance of IGBT and Diode. The switching losses in the IGBT and the diode are the product of switching energies and the switching frequency (f_{sw}):

$$P_{swIGBT} = (E_{on\,IGBT} + E_{off\,IGBT}) \cdot f_{sw} \tag{42}$$

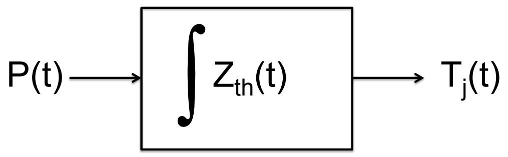
$$P_{swD} = (E_{on\,D} + E_{off\,D}) \cdot f_{sw} \approx E_{on\,D} \cdot f_{sw} \tag{43}$$

where $E_{on\,IGBT}$, $E_{off\,IGBT}$, $E_{on\,D}$ and $E_{off\,D}$ are the energy losses in each commutation and are duty-cycle dependent and the value can be calculated according to the topology of inverter used [42] [41].

6.1.3 Thermal impedance, heat transfer and convolution

The junction temperature of IGBT and diode are generally hard to measure directly. In particular the instrumentation to measure the temperature of silicon chip is too much sophisticated to obtain good accuracy and is never included in this module. Sometimes Infineon in EconoDUAL module installs a NTC (Negative Thermo Couple) to estimate the internal temperature of the device [43]. The temperature measured by this sensor is just an average of the module's temperature. The cross-talking effect of all the IGBT and diode inside cannot be neglected and, moreover, take into account the location of the sensor compared to the chip position is impossible to know the instantaneous junction temperature, due to the slow time constant and heat propagation delay.

For this reason the most accurate alternative to know the temperature of IGBT chip is to calculate it through the dissipated power and the thermal impedance. Indeed, the dissipated power can be calculated as described in § 6.1.2 by using the thermal impedance that can be obtained from FEM simulation and/or by fitting the data from datasheet of device. [44] Since the relationship between the input power and the temperature is given by time invariant linear system, the junction temperature can be thus obtained calculating the convolution of the power losses and the thermal impedance:



$$T_j(t) = P(t) * \dot{Z}_{th}(t) = T_{heatsink} + \int P(t) \cdot \dot{Z}_{th}(t\text{-}\tau)d\tau \tag{44}$$

In this way the junction temperature can be known given as input the average power dissipated on the module, where $\dot{Z}_{th}$ is the thermal impedance derivate.

Equation (44) can be rewrite in the discrete form that will be implemented in the computation of the algorithm:

$$T_j(i) = P(i) * \dot{Z}_{th}(i) = \sum_{n=0}^{N\text{-}1} P(n)\, \dot{Z}_{thN}(n\text{-}i) \tag{45}$$

where N is the number of samples of the thermal impedance. The term $\dot{Z}_{thN}$ is the thermal impedance derivate calculated taking the difference of two sequence values from the FEM simulation.

The compact module is determined form a FEM simulation of a basic structure of the module, as reported in Figure 24 From the simulation, the thermal impedance Z_{th} is calculated and the than the equivalent electrical model is derived.

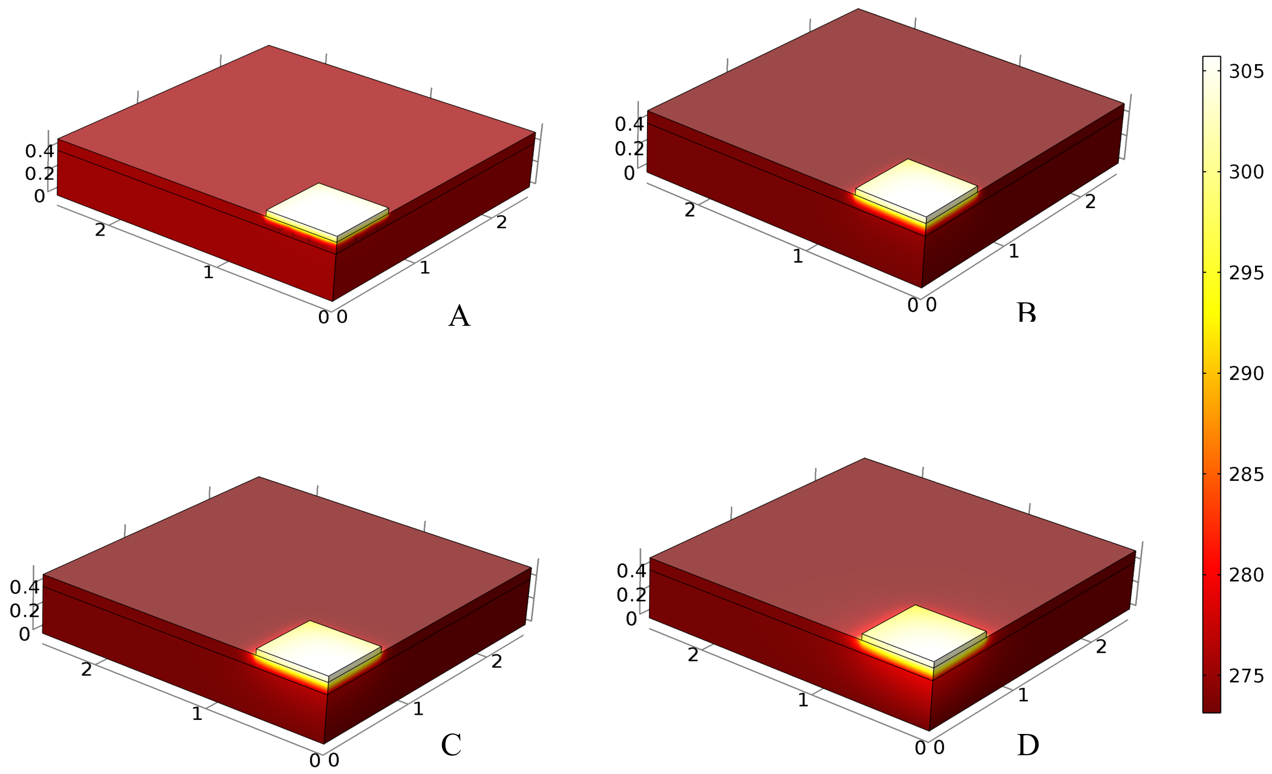


Figure 24. Thermal simulation of IGBT module through FEM analysis. A) time 0.01s after the applied power step; B) after 0.05s; C) after 0.1s and D) after 5s.

6.2 Extraction of the thermal impedance

As mentioned above, there are two ways to evaluate the thermal impedance of IGBT module; first option is to fit the datasheet data about the thermal impedance or, sometimes, a set of time constant τ_i is given to compute equation (35). The second possibility is to obtain the thermal impedance from the FEM simulation; in this manner it is also possible to know the temperature evolution in the other interface of the module structure. In this paragraph is explained how it is possible to acquire the thermal parameter from the thermal simulation. Otherwise, the FEM-simulated thermal impedance of the module can be used directly in the discrete version of equation (44).

The multilayer structure is represented in Figure 24, where the temperature distribution is analysed. Taking into account the symmetry of the module, just a quarter of the system has been simulated, gaining in computational time. Since many material layer combinations are available on the IGBT's market, some tests have been performed. In detail, different dimensions, materials and layer's thickness have been considered; in Table 7 the typical combination usually adopted by the manufacture are summarized. Common materials are Allumina, and AlN as insulator layer while AlSiC or copper is used in the baseplate.

Thickness Insulator layer	Allumina + Copper R_{th} (K/W)	Allumina + AlSiC R_{th} (K/W)	AlN +Copper R_{th} (K/W)	AlN + AlSiC R_{th} (K/W)
1 mm	**0.265**	0.325	0.0917	0.147
0.7 mm	0.1938	0.2554	0.0877	0.1446
0.5 mm	0.1446	0.21	0.05758	0.1013

Table 7. Thermal resistance value obtained from FEM simulation for different combination of insulator thickness and materials. The chip dimension is in all combination 1.2X1.2 cm, the power dissipated is 100W.

The combination chosen for the analysis is that marked in Table 7. It is observed that the thermal resistance value, obtained in the Allumina and Copper combination, is that one much closer to the Infineon EconoDUAL [45] that is 0.26 K/W.

In Figure 25 the transient evolution of the layers' interface temperatures is reported in logarithmic scale. It can be observed that the temperature response of the system is about 10ms after the heat step. The input power is 100 W and it is assumed that the heat is uniformly distributed in the volume of the silicon chip. The chip size is 12mm X 12mm and the thickness is 0.5 mm. (In the simulation just a quarter of the chip is considered, thus 25W are dissipated in 6mm X 6mm chip). To evaluate the thermal impedance of the system, an isothermal condition at the bottom side of the baseplate is assumed. It is important to keep in mind that this assumption is a strong approximation. Usually the heatsink connected to the baseplate bottom face is better modelled using a convective heat flux. Typical value of forced water cooled heatsink is 0.6 W/m^2K with a flow rate of 400 l/h [46].

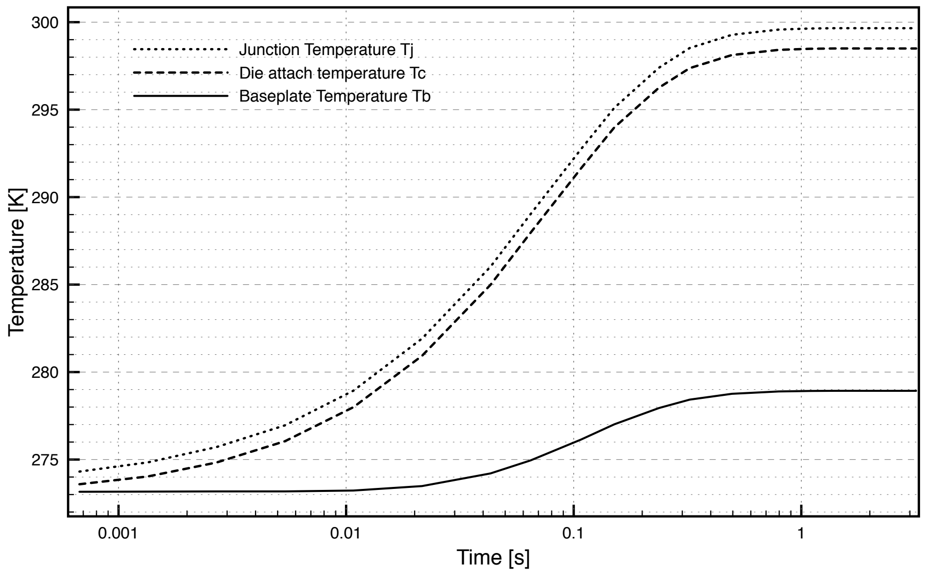


Figure 25. Thermal response of each structure layer. Tj is the junction temperature measured via FEM simulation at the top of the silicon chip; Tc is the temperature at the top of ceramic layer (at silicon-ceramic interface); Tb is the temperature at the top of the baseplate (ceramic-baseplate interface). An isothermal boundary condition is assumed at bottom side of baseplate.

The equivalent compact model developed is composed by three RC block in Foster representation. To evaluate the overall thermal impedance trough the FEM simulation, the temperature evolution of silicon chip top face, die attach and baseplate, are recorded. In general, the thermal resistance, R_i, of each layer is calculated using equation:

$$R_i{}^{th} = \left. \frac{T_i(t)\text{-}T_{i+1}(t)}{P} \right|_{t\to\infty} \tag{46}$$

where, T_i is the temperature at layer i-th, T_{i+1} is the temperature in the next layer below, P is the dissipated power and $t \to \infty$ means the steady-state condition. Assuming that only a linear combination of exponential terms as in equation (35) can describe the thermal evolution of the system, the single time constant τ_i and than the C_i can be evaluated fitting the temperature curve. In Table 8 the values of the RC terms that better fit the simulation are reported.

	1st RC-term [Si chip]	2nd RC-term [Insulator]	3rd RC-term [Baseplate]
R_{th} [K/W]	0.0229	0.1735	0.0587
C_{th} [J/W]	0.0902	0.1947	1.1014
τ [sec]	0.0021	0.0338	0.0646

Table 8. Extracted values of thermal impedance in Foster model. The RC terms are evaluated with fitting tool of Matlab.

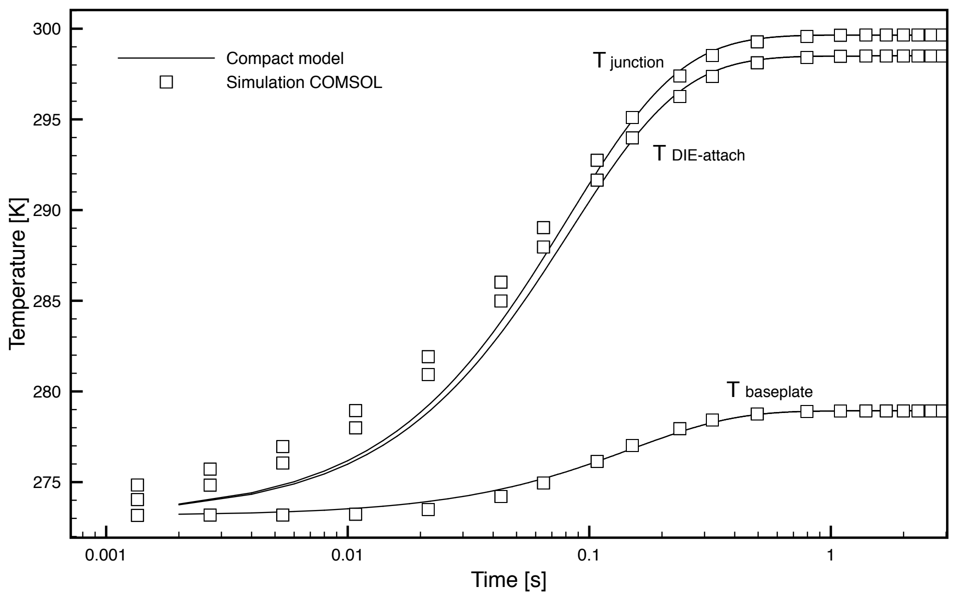


Figure 26. Comparison of COMSOL simulation and extracted Foster model. Can be observed that the heat propagation delay cannot be precisely approximate.

6.2.1 Compact model improving with Elmor delay

The model presented above uses the hypothesis that the thermal impedance can be model just with a sum of exponential terms. As previously underlined, this approach does not take into account the propagation delay that occur especially at the ceramic-baseplate interface; as reported in Figure 25 and Figure 26, the temperature rise of 5 ºC only after 100ms the heat pulse.

The Elmore delay can be included in the calculation of Cauer model to introduce the propagation heat flux at each component layer. The following steps explain how to describe the thermal impedance though Cauer network and Elmore delay [47].

1. The thermal resistance is calculated per each layer as in equation (46), and the total thermal impedance can be rewritten as:

$$Z_{th}(t) = \sum_{i=1}^{n} \frac{T_i(t)\text{-}T_{i+1}(t)}{P} = \sum_{i=1}^{n} Z_{th}^{i}(t) = \sum_{i=1}^{n} R_{th}^{i}\, f_{th}^{i}(t) \tag{47}$$

where $f_{th}^{i}(t)$ is the dynamic component response function and it represents the i-th layer in the structure.

2. $f_{th}^{i}(t)$ terms represent the function that describe the propagation of the heat flux in each layer. They can be extract from the FEM simulation using:

$$f_{th}^{i}(t) = \frac{Z_{th}^{i}(t)}{R_{th}^{i}} = \frac{1}{R_{th}^{i}} \cdot \frac{T_i(t)\text{-}T_{i+1}(t)}{P} \tag{48}$$

3. The delay is compute with Elmore formula

$$d_i = \int_0^{\infty} (1\text{-}f_{th}^{i}(t))\ dt \tag{49}$$

where d_i represent the propagation delay of the heat flux from the heat source (in this case the silicon chip) to the i-th layer.

4. Given this result, the time constants can be obtained though:

$$\tau_i = d_i\text{-}d_{i\text{-}1} \tag{50}$$

where $d_0 = 0$.

5. The thermal capacitance of each layer can be calculated from the time constant as follows:

$$C_{th}^{i} = \frac{\tau_i}{R_{th}^{\Sigma\, i}} \tag{51}$$

where it is important to underline that $R_{th}^{\Sigma\, i}$ is the sum of the thermal resistance R_{th}^{i} from the layer 0 (the silicon chip) to layer i-th.

With this procedure is possible to take into account the propagation delay in each layer and the results are resumed in Table 9.

	1st RC-term [Si chip]	2nd RC-term [Insulator]	3rd RC-term [Baseplate]
R_{th} [K/W]	0.0229	0.1735	0.0587
C_{th} [J/W]	4.707	0.4113	2.986

Table 9. Cauer model parameters extracted with Elmor delay method, and simulated as electrical network.

Is worth to note that more complex thermal models have been developed during years and most of them have better approximations of the thermal behaviour and the secondary effects are taken into consideration [48]. The aim of this work is just to use the models that have enough accuracy to characterize the solder joint temperature in the IGBT module. More complex models are not taken into account.

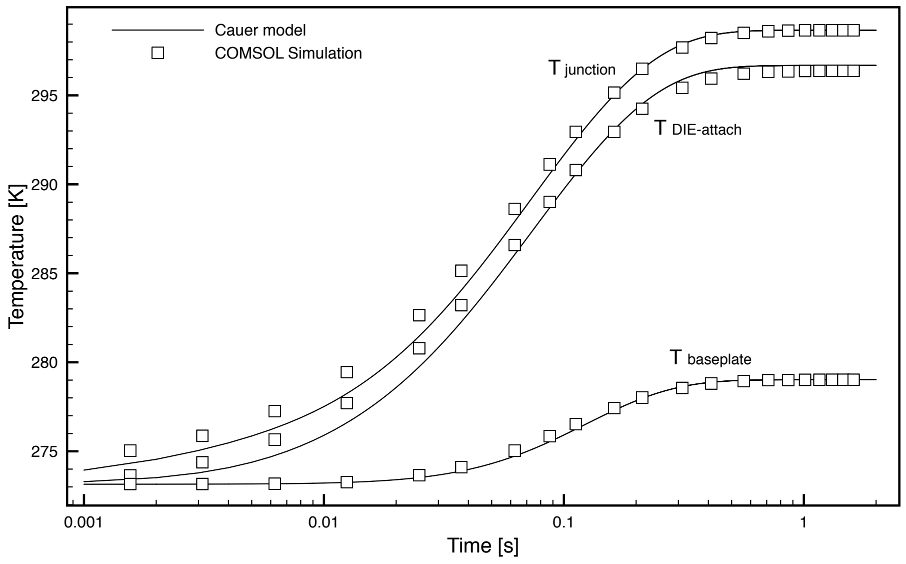


Figure 27. Cauer approximation

Even though Forster's and Cauer's models are good enough to calculate the thermal impedance of each layer, it is important to have the possibility of estimate the junction temperature through the basics heat transfer equations.

6.3 Analytical model

Another method to evaluate the junction temperature is through the base plate surface (or the temperature in another interface) and the knowledge of the thermal behaviour of the module. The Foster model gives the possibility to estimate the temperature evolution solving the RC electrical network.

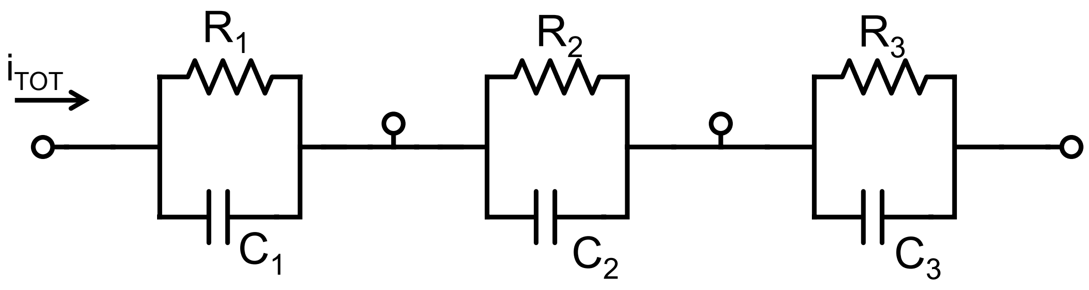


Figure 28. Generic RC network to represent the IGBT module layer.

Consider the generic RC network in Figure 28 and suppose that all the value R_i and C_i are known from FEM simulation, and is possible to measure the voltages V_0 and V_1 (corresponding to heatsink temperature and baseplate temperature). Thus, assuming that the heat flux in the same in each layer, and keeping in mind the analogies of Table 6, it is possible to write:

$$i_{TOT} = i_{R_1} + i_{C_1} = i_{R_2} + i_{C_2} = i_{R_3} + i_{C_3} \quad (52)$$

for each i_{R_i} and i_{C_i} yields,

$$\Delta v(t) = v_2(t)\text{-}v_1(t) = R_2 i_{R_2} = \frac{1}{C_2}\int i_{C_2}(t)dt \qquad (53)$$

where, from equation (52), it is possible to find out:

$$\frac{v_1(t)}{R_1} + C_1 d\frac{v_1(t)}{dt} = \frac{v_2(t)}{R_2} + C_2 d\frac{v_2(t)}{dt} \qquad (54)$$

where $v_1(t)$ and $v_2(t)$ are the voltage evolution form ground reference. Rewriting equation (54) in discrete time, yields:

$$\frac{v_1(i)}{R_1} + C_1\frac{v_1(i)\text{-}v_1(i\text{-}1)}{Ts} = \frac{v_2(i)}{R_2} + C_2\frac{v_2(i)\text{-}v_2(i\text{-}1)}{Ts} \qquad (55)$$

Solving the equation to explicit $v_2(i)$ as function of the other variables, yields:

$$v_2(i) = \frac{v_1(i)\left[\frac{Ts}{R_1} + C_1\right]\text{-}C_1 v_1(i\text{-}1) + C_2 v_2(i\text{-}1)}{\frac{Ts}{R_2} + C_2} \qquad (56)$$

Equation (56) shows how it is possible to calculate the temperature in the upper layers given the temperature in other interfaces. Note that the same calculation can be done to know the junction temperature and, by reversing the formulas, it is also possible to know the temperature in the above layers when the junction temperature in known.

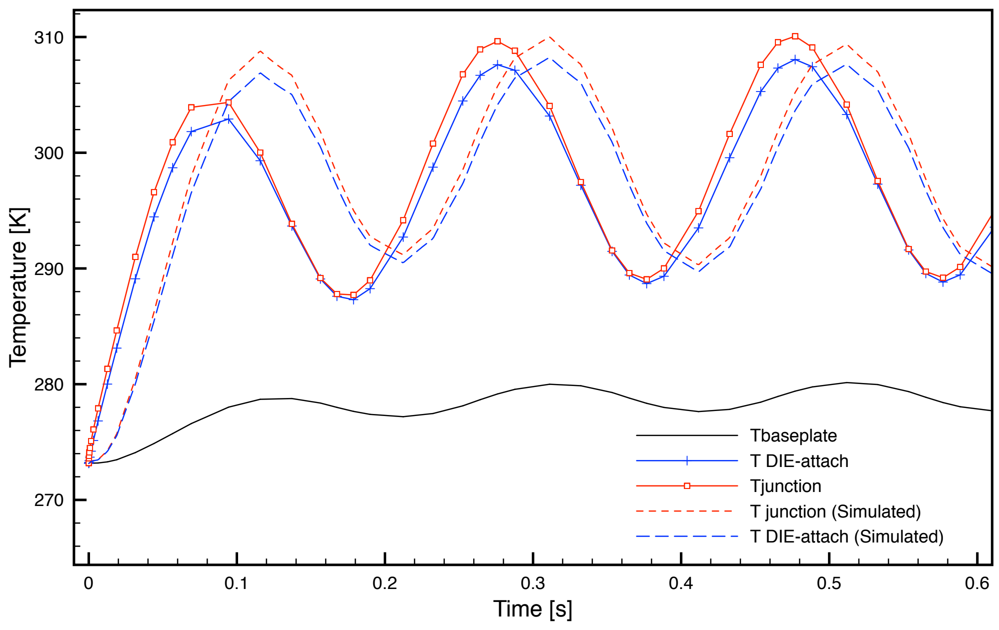


Figure 29. Analytical model solved with RC Foster network equivalent. Continuous line represents the temperature obtained from FEM simulation in COMSOL. Dashed lines are the temperature extracted form Foster RC Network. The dissipated

power has a sinusoidal form with frequency 10Hz. The simulated results approximate well the real evolution of temperatures even though a small delay is introduced. The sampling time in the simulation is 20ms

In Figure 29 the comparison between the results of FEM analysis and the temperature values calculated with this analytical RC Foster model is represented. When the power dissipated is a sinusoidal wave with frequency of 10Hz and amplitude 100W, and the time step is 20ms, it is absolutely possible to reconstruct the temperature profile at the junction layer. Just a little delay of about 40ms is observed, but the amplitude and the frequency are unchanged.

Since the equations (55) and (56) describe a discrete time high-pass filter (anyway not infinite pass-band), when the power applied has a faster frequency it is possible to estimate only the mean value at the junction; in Figure 30 the same analysis has been performed, but in this case the power evolution frequency is 50Hz, while the sampling frequency is still 20ms.

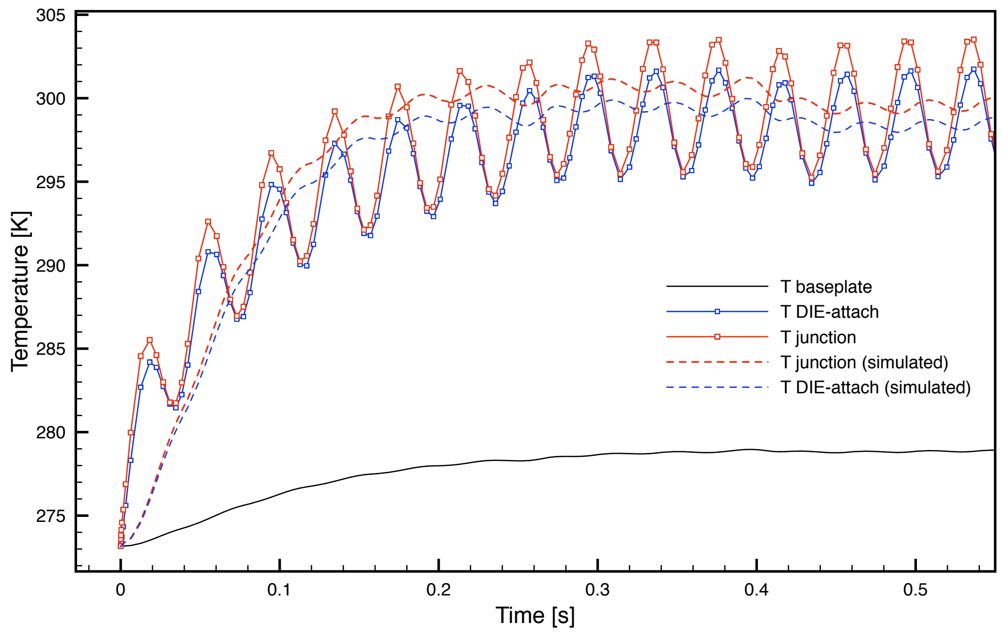


Figure 30. The same analysis plotted in Figure 29 has been carried out applying a signal of 50Hz.

Since it is quite complicated to sample faster and the thermal behaviour is not too fast to use a higher sampling frequency, this approach can be used by taking into account these limitations.

In the case of junction temperature is known and the bottom layers have to be estimated, this limitation doesn't occur, since the system acts as a low-pass filter.

7 Reliability odometer

In this chapter the reliability odometer implemented will be explained. The aim of this tool is to evaluate the device's health condition in a real-time environment. The reliability measurement of power modules takes into accounts the assumption, analysis and results found in previous chapters. In particular, the lifetime prediction is estimated with Rainflow algorithm and, at the same time, the mechanical fatigue stress analysis is performed. The thermal behaviour is developed using the FEM thermal simulation, and extracting the thermal impedance to compute the temperature evolution as function of dissipated power, or as function of other inputs.

7.1 Specifications and Design

The feature of a reliability meter can be reassumed in flexibility and accuracy in a real-time context. To implement this tool with a good precision, it requires a sampling time variable between 10 to 50 ms, equivalent to the sampling frequency of 20Hz to 100Hz. Many inputs can be used to give the possibility to choose which measurement is more suitable in the custom applications; the input can be the power losses of the device, the junction temperature, or another temperature of the device. The thermal impedance of the device is required to know the thermal behaviour and estimate the temperature in the solder joint.

The two algorithms to estimate the lifetime, the Rainflow and the thermo-mechanical fatigue, run simultaneously, and for each sample the health status is calculated.

The output data are given in graphical format and are the histogram that describes the cycles distribution of Rainflow algorithm, the mechanical stress-strain graph and the lifetime estimation; moreover a log of data acquired is continuously saved.

The main goal of the tool is a portable device that can perform a real time analysis and give an output visualization of the computation.

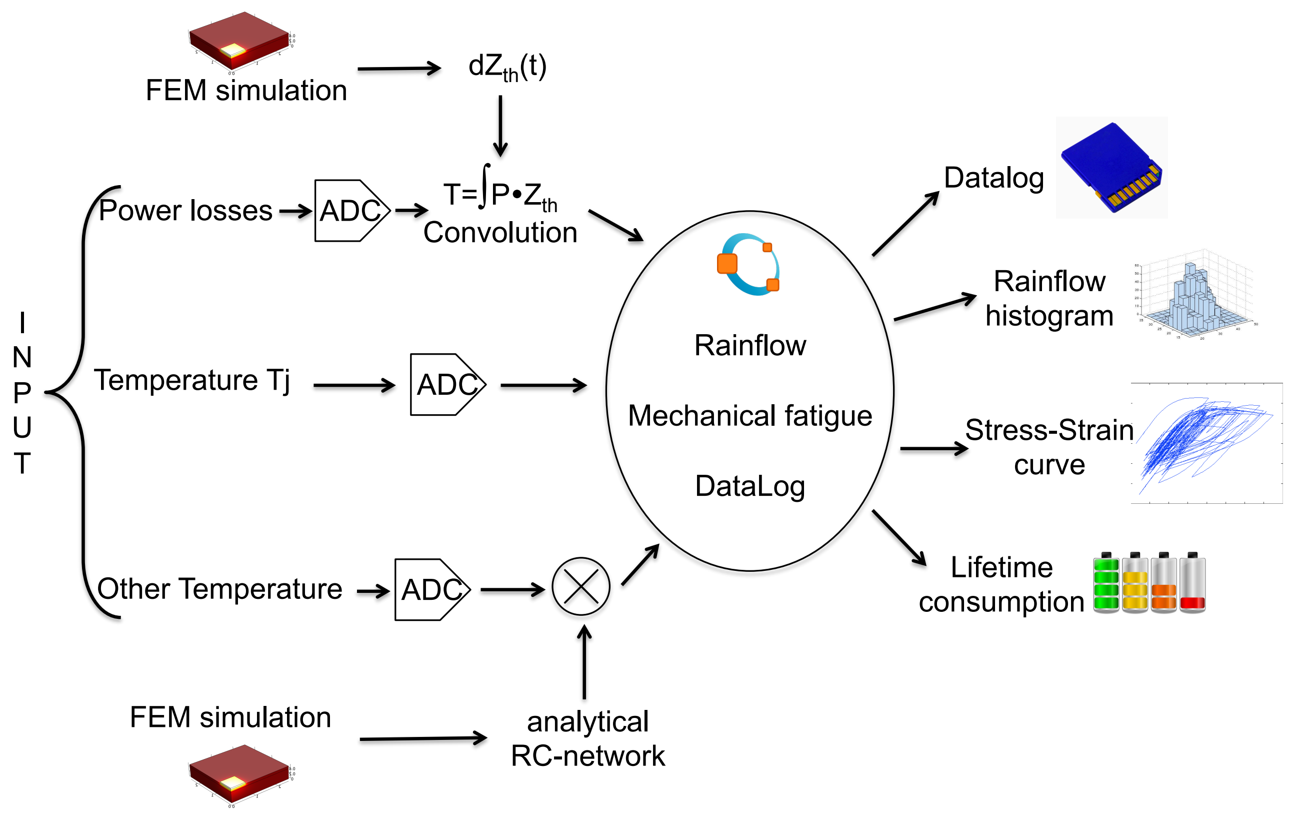


Figure 31. Schematic block representation of developed tool. The system input can be the power dissipated, the temperature at the junction or other temperature. FEM simulations are required to evaluate the thermal impedance of the system. The algorithm evaluate the thermal-mechanical fatigue through Rainflow and constitutive equation. The output are the log of the data acquired, the histogram to represent the cycle's distribution of Rainflow algorithm, the stress-strain evolution and finally, the health status of the device.

7.2 Hardware implementation

The tool proposed has been tested in different platform. The main aim of the device is a real time analysis of the IGBT module.

The system requires a sampling interface to convert the analog input signal (temperature or power), into a digital signal to be processed. An analog-to-digital converter is the first block that is required, where it is assumed that the input signal is already filtered in the proper signal band. The ADC resolution has to be good enough to have a signal to noise ratio suitable for this application, but are not required at a very high performance. If a 10-bit resolution ADC with a sampling frequency of 1kHz, which is a general AD converter, the specification are achieved.

The features required for this tool can be implemented in a microcontroller. For simplicity of implementation, stand-alone solution and cheap price the following board has been used: Arduino Uno, Intel Galileo, pcDuino. Since different platforms and CPUs are available, the implementation of the software, that has to be tasted and develop in the specific case in each board to know if the specification and performance are met.

In Table 10 are resumed the characteristic of the microprocessor and the skills of the board.

Board/Features	Arduino Uno	Galileo	pcDuino
CPU	ATmega328p, single core	Intel Quark SoC X1000, single core	A20 SoC ARM Cortex A7 Dual Core and GPU **Dual Core**
CPU Freq.	32MHz	400MHz	1GHz
RAM	Internal SRAM, 2KB	Internal SRAM, 512 KB, External DRAM, 256MB	1GB DRAM
Storage memory	Internal Flash, 32KB	8 MByte Legacy SPI Flash, sdCard expansion up to 32GB	2GB Flash, microSD up to 32GB
ADC	Internal ADC (on the µC), 10-bit SAR ADC	External (on the board) 12-bit SAR ADC	Internal (on CPU) 12-bit SAR ADC
Communication	Serial port	Serial port, Ethernet	Serial, Ethernet, Monitor…
Output	12 digital pin	12 digital pin	Digital pins, HDMI,
Cost	Only CPU ~3.5$ Arduino Board ~20$	Only CPU ~15$ Galileo Board ~65$	Only CPU ~20$ pdDuino Board ~70$

Table 10. Board performance comparison

7.2.1 Arduino

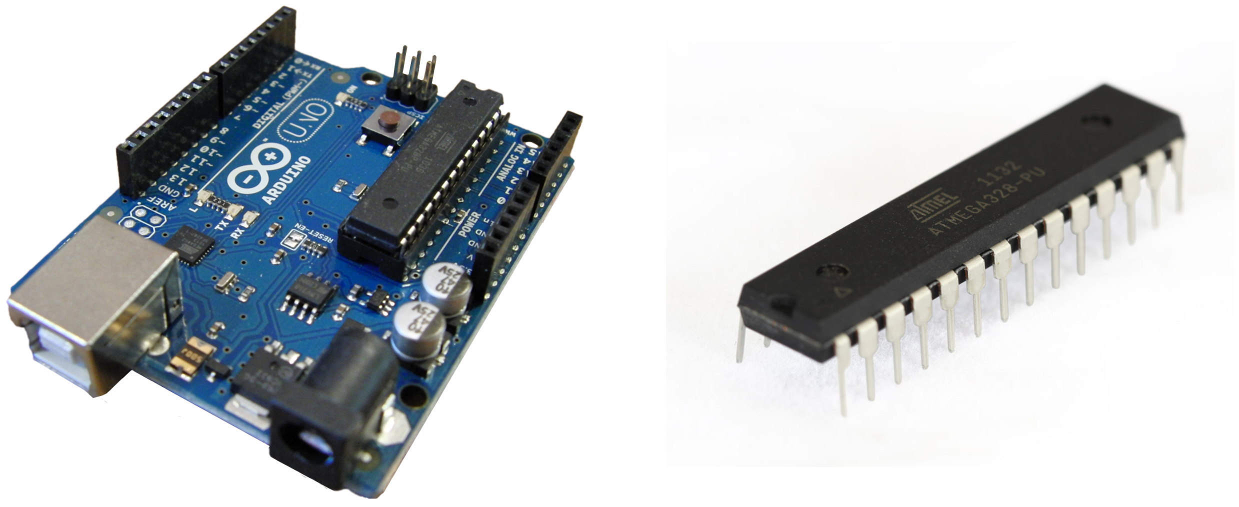


Figure 32. Arduino board and ATmega328 microcontroller in DIP package.

[49] Arduino board can be easily programmed with a c-like programming language or directly in c. This microcontroller is for general-purpose application and does not have particular feature in arithmetical operation. Indeed, it has been observed that the convolution time that is proportional to the kernel size of the thermal impedance, is slow and influences the overall performance. Moreover, the internal SDRAM memory is limited, so the thermal impedance length (number of samples and vector dimension, 1 float number is 4 byte) must be taken into account. E.g., load into RAM memory 100 float variables requires 800 bytes: 4*100 byte for the array containing the kernel of thermal impedance and 4*100 byte to storage the last temperature values to perform the convolution. Anyway it is possible to save the data on the flash memory, but it slows down the reading procedure.

The CPU's clock frequency is not very high, and especially in mathematical operation, this has to be taken in consideration. Finally, the data communication can be done only through serial port, that weights a lot on system performance.

The convolution and sampling procedure are tested, and the results are reported in following table.

Arduino Uno, ATmega328p, 32MHz	
Operation complexity	Execution time and note
Convolution: 100 multiplications, corresponding to 1 second saved sampled at 10ms.	Average execution 2.9 ms, some times execution up to 3.2 ms.
Convolution: 300 multiplication, corresponding to 15 seconds saved and sampled at 50ms	Average execution 8.6 ms, some times execution up to 9.4 ms.
Convolution: 400 multiplication, corresponding to 10 seconds saved and sampled at 25ms	Average execution 11.8 ms, some times execution up to 13.9 ms. NOTE: Most of the SRAM memory is used by the kernel array and last temperature values array. The other operation has been not implemented.
Rainflow: just if-else comparison, and data filtering (moving-average filter on 20 values)	The execution of the procedure occurs only when a new maximum or minimum is found. The execution in this case depends on the numbers of maxima and minima saved in the stack. For 50 values in the stack (max and min), the execution takes 12.5 ms.

Table 11. Summary of Arduino performance in the test performed.

The main limitations found in this platform are recourses related; it has been observed that calculation errors occur especially when the SRAM memory is almost full, with some stuck of the execution of the program. Moreover the CPU frequency is not high enough to sample and compute the analysis at 100Hz. For this reason a faster CPU is preferred.

It has to be taken into account that the price of ATmega328p is very cheap and it can be a solution if only Rainflow algorithm has to be implemented with low sampling frequency of about 50Hz. This µC requires a very optimized code to be useful in this application, in this work has been preferred to use it with other aims.

7.2.2 Galileo

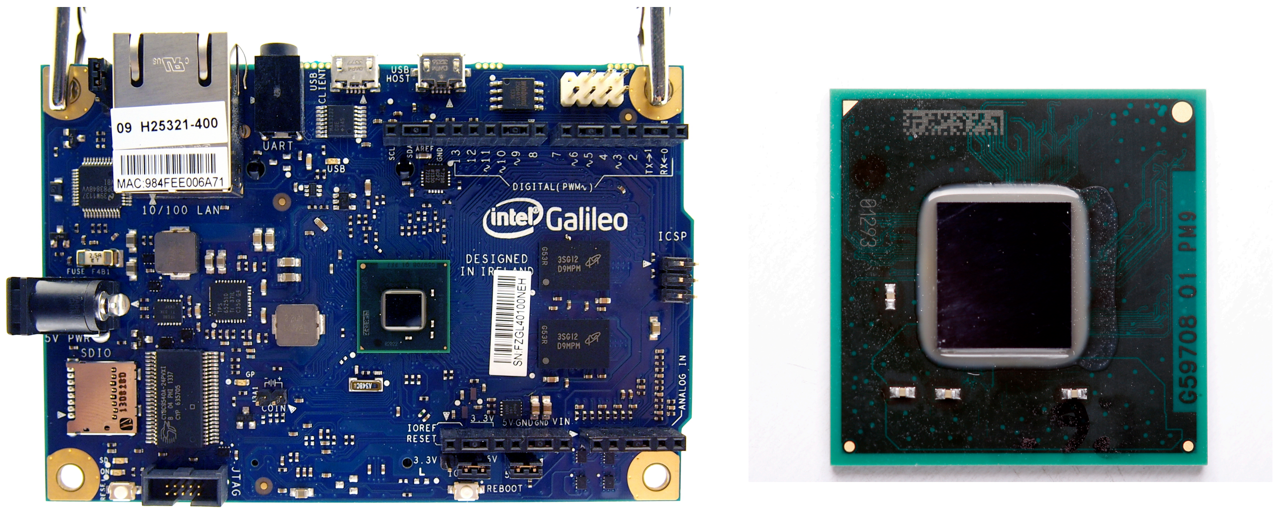


Figure 33. Intel Galileo and Intel Quark SoC X1000 processor

Intel Galileo board is a faster platform with Intel Quark SoC X1000 processor at 400MHz [50]. The CPU speed and the RAM memory of this device are perfect for the specification of this software. The results reported in Table 12 show the benchmark of Intel Galileo with some operation used in the algorithm.

Intel Galileo, Intel Quark SoC X1000, 400MHz	
Operation complexity	Execution time and note
Convolution: 15k multiplications, corresponding to 15 seconds saved sampled at 1ms.	Average execution 1.9 ms
Convolution: 3k multiplication, corresponding to 15 seconds saved and sampled at 5ms	Average execution 400 µs
Rainflow: just if-else comparison, and data filtering (moving-average filter)	The execution of the procedure occurs only when a new maximum or minimum is found. The execution in this case depends to the numbers of maxima and minima saved in the stack. For 100 values in the stack (max and min), the execution takes 4.1 ms.

Table 12. Summary of Intel Galileo performance in the test performed.

The system here considered shows optimal performance in the arithmetical calculation. The disadvantages in this platform are the absence of a graphical output (to plot the output of the algorithms only the serial port od Ethernet communication can be used, no GPU is present on the device) and it is require to write a c-class to access the external SD card where the look-up table values of the stress-strain mechanical analysis are stored.

7.2.3 PcDuino

The idea to have also a visual output of the lifetime estimation, with the possibility of further implementation or analysis, pushes the developing process to a new device with the opportunity to run a Linux operating system, but without the features and issues of a full computer. Two main board are available on the market at cheap price and stand-alone, Raspberry-PI [51] and pcDuino [52]. It has been chosen to use the pcDuino as it has the analog-to-digital converter already mounted on the board, whereas raspberry require an external ADC.

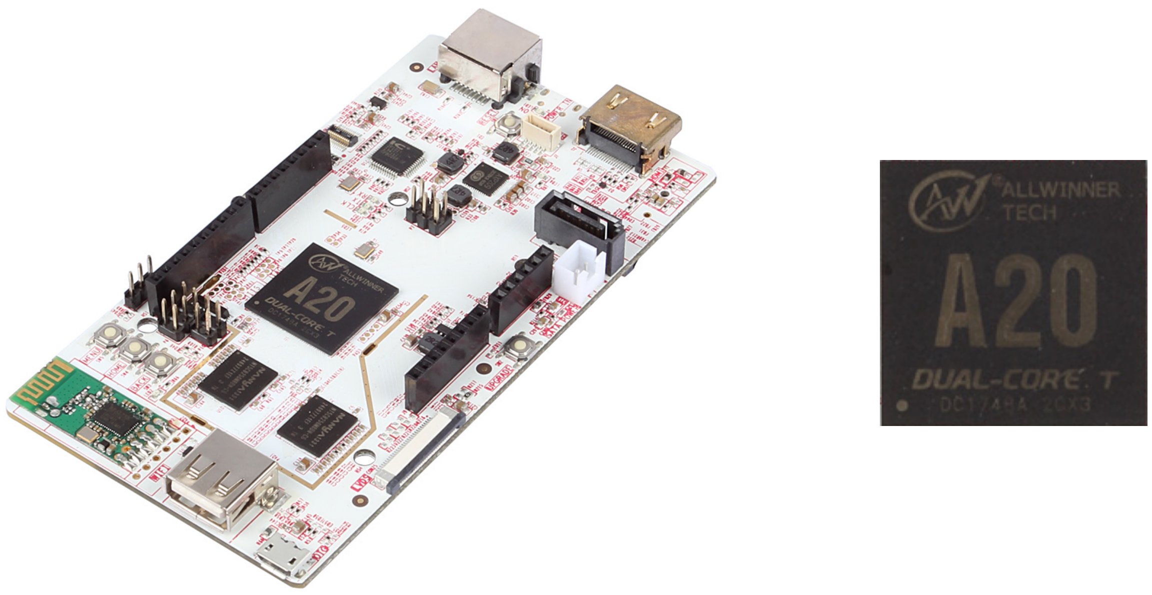


Figure 34. pcDuino board and A20 ARM Cortex A7 CPU.

pcDuino is basically a computer on a single board. The CPU mounted is a A20 SoC ARM Cortex A7 Dual Core and graphics GPU; the CPU clock is 1 GHz. With 1GB RAM memory and a SDcard slot, a basic version of Linux can easily runs on this device. Since the kernel Linux is installed in this board, it is possible to use Octave/Matlab to compute the algorithm. This is useful also to plot the results and have a real-time visualization of the computation underway. The choose to use Octave has been done mainly for the following reasons:

- to use the code already written for the simulation and the analysis described above
- to have the opportunity to visualize the results in easy way, that other programming languages, such C/C++, require more code to write.
- Octave, compared to Matlab, is open source code and doesn't require buying use licence.

Other solution, for example a full C/C++ code in this environment could, be faster and more optimized, but it requires more effort to write the code. This solution can be considered in a future version of the tool.

Since it has been decided to develop the software with Octave, a procedure in C language is required to acquire the samples from the analog-to-digital converter and transfer them to Octave. Different way to communicate has been tested and considered because the execution times issue or synchronization problem occurred many times. Finally using pipe communication between the two processes has been solved.

7.2.4 PC/MAC

The same code implemented on pcDuino can run also in every PC/MAC with Octave installed without many extension of the program. The interface and ADC that can be used is Arduino that has enough resources to sample and transfer data to Octave. The limitation in this version is the not-stand alone of the system and the transferring speed of the serial port.

7.3 Benchmark of the hardware

The tests performed have been done for each operation that the lifetime odometer requires. Some parts of the code cannot be performed because during the development it has been observed that the device’s performances are inadequate to complete the tasks. Since the board can be programmed with different programming language, the codes have been readapted to the microcontroller used. The operations tested are the sampling process, the convolution, the mechanical analysis, the Rainflow algorithm and the datalog of samples. In Table 9 the best average times of execution of each parts of the algorithm that are discussed before or after are resumed.

While developing the tool some important performance issues have been noted. As it has probably been known, Octave is well optimized to calculate multiplication and use efficiently the matrix and array. The convolution, calculated as a vector multiplication, is fast, while the Octave code `conv(x,y)` to calculate the convolution is two times slower, probably due to further calculation and accuracy. Another important observation done related to the slow execution during the development of the tool, is the evaluation of some built-in function in Octave; for example the first version of Rainflow algorithm uses the relation `var(x,y)/cov(x,y)` to know the temperature slope and change and evaluate equation (4). This solution is really slow compared to a simple evaluation using the *Ordinary Least Squares* as reported in equation (5).

Board	Arduino Uno	Intel Galileo	pcDuino	PC/MAC
Sampling	Mean:1090μs	Mean: 660μs	Mean: 715μs Max: 2.3ms	N/A
Convolution	See table above	See table above	Mean: 1.2ms Max: 5.2ms	Mean: 17μs
Rainflow	Mean: 6.3ms Max: 6.9ms	Mean: 4.1ms Max: 4.4ms	Mean: 1.7ms Max: 3.8ms	Mean: 26μs
Mecchanical analysis	N/A	N/A	Mean: 6.1ms Max: 17ms	Mean: 96μs
Saving data	N/A	N/A	Mean: 3ms Max: 3.7ms For 5 minutes at Ts=10ms	Mean: 34μs

Table 13. Performance comparison of the boards tested. Some tests are not performed (N/A) in some platform because it wasn't possible to compute the algorithm due to memory or RAM insufficient. The simulation done on the PC has been conducted using off-line data and testing the procedure separately. The pc CPU is an Intel Core i7 at 2.5Ghz frequency. Further consideration about the sampling performances and the execution time distribution are reported in § 8.5.

Other time-spending process is the conversion of time from integer format (an integer number that count the milliseconds from 1970) to a string containing the data and time in characters. To avoid the slowing down of the software, it has been decided to introduce a post-elaboration procedure to export data and avoid the conversion in real-time. Also number-to-character conversion is slow using the procedure `num2str`, while `fprint` function (inherited C function) is many times faster. These tips in the development has to be taken into account, because most of the solutions adopted have been tested and modified many times to find the right performance and guarantee the specifications requirements.

8 Final design and performance

The final software completed and carried out has been written in Octave and C to work under pcDuino platform. The system chosen is that one showing the best performance and flexibility to future extension of the software and to quickly perform further analysis. Moreover, keeping in mind the that tool has to be a stand-alone instrument, the pcDuino solution is the best solution to have at the same time a computer and the sampling interface all together in the same board. Further it is possible to connect a monitor and have a visual output without interfacing with a computer and use other software; related to this point it is worth to observe that it is possible to connect also a screen via LVDS interface available on the board (should be possible to have a touch-screen input).

8.1 Implemented algorithm

The main block of the software can be reassumed in the following key-block:

- the sampling C procedure lunched via pipeline of Octave. It is responsible to communicate with the low-level ADC-interface and acquire data to transfer in Octave. The main program in Octave asks for a new sample to the C-class and it answer when a new sample is available.
- when the power input is selected, the convolution is performed on Octave. It has been done shifting the array's values one-by-one to keep only the last value that are multiplied with the kernel of the thermal impedance. This operation with Octave in pcDuino is fast event for 15000 float values, because it is optimized to deal with vectorized numbers.
- the Rainflow lifetime evaluation is continuously running; if no new maximum or minimum are found noting is performed. When a new max or min is found, the algorithm proceeds as described in § 4.2.3.
- the mechanical fatigue is computed using the strain expansion given the temperature at the junction; this step is done evaluating equation (6). The current stress value is found in the look-up table and interpolating the adjacent values as describe in § 5.5.
- the value of dissipated power, temperature, strain, stress and deformation fatigue are saved and dumped into a file with regular timing (user selectable)

All the software parameter and the configuration of sampling time, damping time, material constant are set before the running of the tool in a configuration file. This Octave file sets all the necessary internal variable and parameters that are used during the analysis, post-elaboration and visualization; moreover it contains the setting of the simulation that have to been performed and the output form or files.

8.2 Interfacing

The pcDuino board has many interfaces and is suitable for further expansion of the tools here presented. Here it is used with the Linux basic kernel, where Octave is installed, both running from the sdCard. Mouse and keyboard are connected in the USB port, while Ethernet connection is used to access to Internet.

The output is visualized though the screen connected to HDMI port, but note that a LVDS connection is available to use a different screen and so have a full stand-alone device.

The analog input pins of the 12-bit Analog-to-digital converters are used to acquire data from the sensors. Here the analog part to drive the measurement interfaces and to pre-filter the signal coming from the current, voltage, temperature or power sensors is not developed. It is important to note that the signal input for these signals have to be in the rage 0-3.3V. The analog interfacing can be easily done with a simple RC filter or/and an operational amplifier to filter the signal in the opportune frequency band and adjust it to the input rage required, but it has to be performed for the particular sensor in use. In the software is then necessary to set the conversion factor, arranging the conversion factor of the analog-to-digital converter. The conversion can be done keeping in mind that the analog input voltage in 0-3.3 Volts range corresponds in an integer value in the range 0 - 2^{12} = 0 – 4096, where 12 is the number of bits of the ADC.

The communication between the analog-to-digital converters and Octave has been the tough issue encountered. In the first developed software the acquired data are transferred to Octave using its built-in library which communicates using the serial port; this solution was problematic and worked marginally, in particular the performances were poor and many sample were lost or cumulated in a buffer slow down the data throughput. Once localized the problem, a new solution is adopted; the communication using pipes and redirecting the output of the c-program in the Octave input. The I/O redirection of sampling program and Octave has been performed dealing with `stdin` and `stdout` classes of the C code and operating system.

The tool is set up using an external function generator or tested with Arduino board to generate data and simulate the analog signal coming from the sensors; in Figure 35 the instrumentation used is schematized.

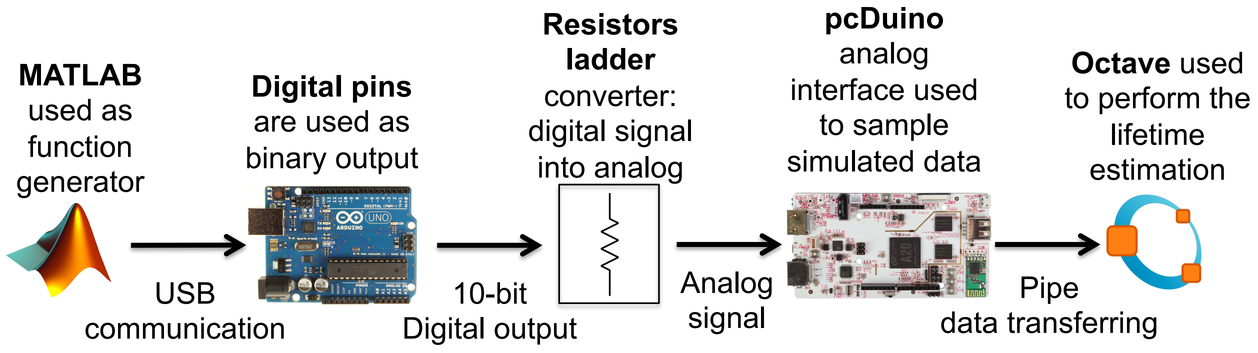

MATLAB used as function generator
USB communication
Digital pins are used as binary output
10-bit Digital output
Resistors ladder converter: digital signal into analog
Analog signal
pcDuino analog interface used to sample simulated data
Pipe data transferring
Octave used to perform the lifetime estimation

Figure 35. Interfacing block diagram used to test the software and simulation.

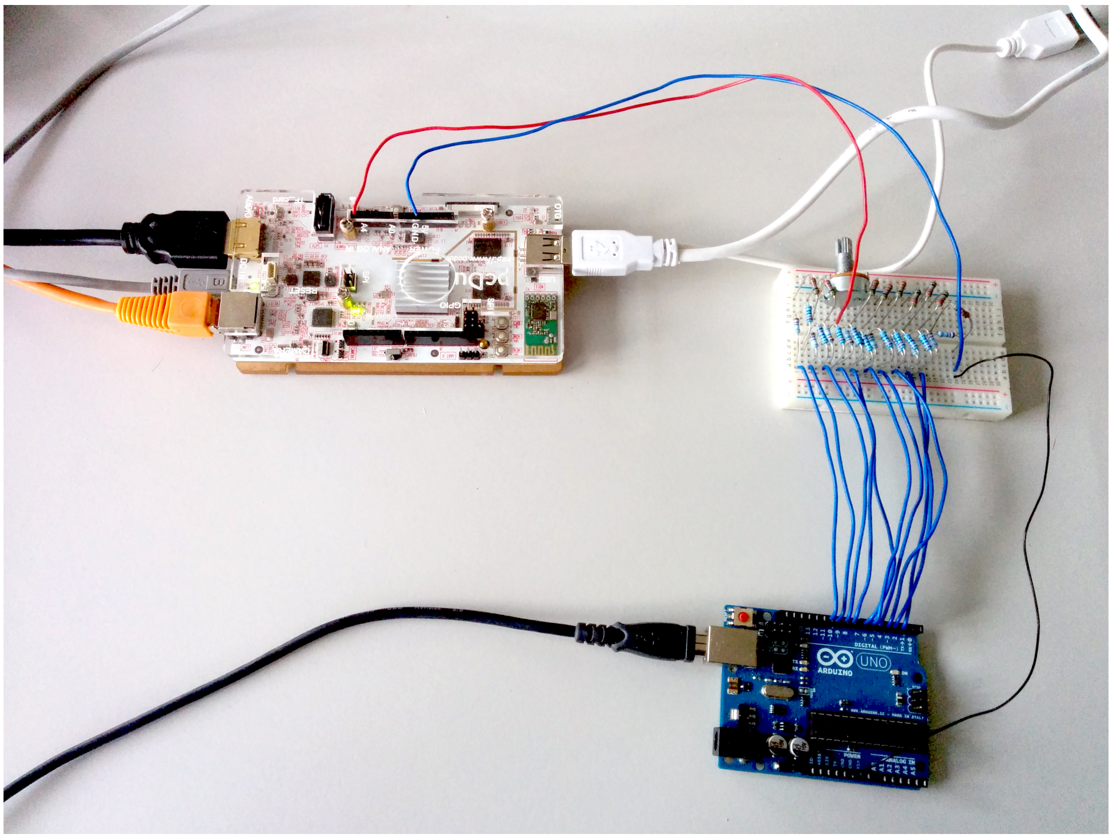


Figure 36. Testing equipment and sampling interface. Top-left: pcDuino. Top-right: Resistors Ladder. Bottom-right Arduino connected to USB at PC.

8.3 Code structure

`runLifetimeEstimation.m` is the main file. This file has to be run from the Octave console in the folder with the others software files, to start the execution of the software. Looking at the structure of this file, the key-points of the code commands are:

1. `config` the configuration file is lunched to load the parameters and the setting of the simulation.
2. `[in_oct, out_oct, pid_oct]=popen2("octave",{"--eval","plotData"})` This command launch another instance, via pipe, of Octave where are collected data that have to be plotted. In this way, since the plotting instance of Octave runs in the second core of the CPU the plotting procedure doesn't influence the sampling rate and analysis.
3. `[in, out, pid]=popen2("./getSample", num2str(inputPin))` this line which starts the sampling procedure. The `getSample` command is a compiled and executable program that is running to perform the sampling. It requires the input parameter `inputPin`, to select `the` corresponding pin in the device where the signal is connected.
4. `if(currentTime-prevTime>Ts)` is used to give the timing clock. When a sampling time step of `Ts` is over, the new sample is processed by the algorithm. The two variable `currenTime` and `prevTime` are used to measure the time elapsed.
5. the variables `in` and `out` are adopted to communicate with the sampling procedure; they represent two buffered stream of data; the commands `fputs(in, "1\n")`, `fseek(out,0,"bof")` and `a=fgets(out)` have the function to request a new sample and read the answer in the standard buffer stream. The system wait until a new sample is available.

6. `currentTemp=convolution(currentPower, zTH)` the convolution is performed when the dissipated power input is selected. The power value is converted from the analog value with an opportune scale factor depending to the ratio bit/power and the sensor specification. The variable `zTH` is the pointer to the vector containing thermal impedance derivative.
7. `rainflow(currentTemp,time())` performs the Rainflow analysis when it is selected in configuration file, the two inputs are the temperature and the time.
8. `[stress, strain, W_def] = evaluateMeccStress(currentTemp, time())` performs the thermo-mechanical fatigue evaluation. The two input in this function are the current temperature and the current time. It returns the stress-strain relationship and the deformation work cumulated at this point.
9. during the execution of the function explained above all the results are saved in a variable to carry out also the data loggin function. `fprintf(in_oct, plotString, [dataResults(1,1) - startTime, dataResults(1,2:end)])` this procedure communicate with the second instance of Octave open in 2. and sends the last data calculated in this step.
10. the data dumping is done periodically with command `save("-binary", fileName, "outputData", "dataLength")` in the folder `"tempData/"` and the file name contains the data and time of the analysis.
11. the main program loops until the user quit the application; when the program exit all the process are closed and clean.

`config.m` is the configuration file which contains all the setting for the execution of the program. The most relevant parameter to set are listed here:

1. the conversion factor representing the correspondence 1 bit ADC = power or temperature quantity: `tempScaleFactor, powerSacleFactor,` and the input type.
2. the file containing the thermal impedance derivate when the power input is selected `zTH_file="dZth"`.
3. the sampling time `Ts=0.01`
4. the setting for the Rainflow analysis, such as the histogram discretization and other auxiliary parameters
5. the filename of the look-up table used for the mechanical analysis `data_file='table_data.mat'`, the thermal expansion coefficient, some material parameters, and the initial condition.
6. other setting like the dumping time and the files name.

`rainflow.m` is the function that perform the Rainflow analysis; the two input requires are the current time and temperature. The key-points in the algorithm are:

1. `if (wait>numSample/2)`, at the beginning of execution of the code the function wait until there are enought value on the buffer to evaluate the temperature samples.
2. `slope1=…` and `slope2=…` calculate the derivative of the first half and the second one of the buffer vector using formula (5).
3. when a new max or min is found [`if (slope1*slope2<0), if (slope1>slope2)…else`] both time and temperature are saved:
   ```
   [new_maxT, iMax] = max(samplesArray(_TEMP,:))
   timeAtMax=samplesArray(_TIME, iMax);
   ```
4. now the algorithm proceed as described in § 4.2.3:
   ```
   if new_maxT>maxima_stack(_TEMP,Ptr_max)
   ```
   a new cycle is identified and the amplitude, duration, mean and damage are summed
   ```
   else
   ```

`maxima_stack(_TEMP,Ptr_max)=new_maxT;`
add the new relative max to the stack

The same computation are performed also when a new min is found.

5. the pointer in the maxima and minima stack are updated and removed the old value; the same procedure is compute when a min is found
6. the lifetime consumption is estimate and the cycles are counted. `addToHistogram.m` is the function that create the Rainflow histogram and add each cycle in the proper amplitude and mean range; the histogram is saved in a file that is used from `plotData.m` to display the results in real time.

`evaluateMeccStress.m` is the function file that execute the mechanical fatigue calculation. The main functions in this code are here summarized:

1. the only elastic behavior is calculated getting the Young's module in dependence of the temperature in Kelvin; the differential stress-strain relation is performed using equation (13) `d_stress=d_strain*getYoungModulus(newTemp + 273.15).`
2. the elastic and inelastic response of the all system is compute with the look-up table in the differential way; the command `searchStressInterplOct(table_data, newTemp, plastic_strain+abs(d_strain), abs(strain_rate)),` call the external function that return the stress value for the inputs temperature, strain increment and strain rate. The interpolation is evaluated in `searchStressInterplOct.m` file where the calculation explained in §5.5 is performed.
3. The differential equations are integrated and the damage in each expansion or compression is cumulated.

`plotData.m` is the file that runs in the second instance of Octave and plots in real-time the last minutes (the time can be chosen by the user) of the analysis. The plots displayed are the real-time temperature evolution, the stress vs. strain plot, the lifetime estimation calculated with both the Rainflow and the mechanical fatigue analysis, and, further, the histogram of temperature cycles.

8.3.1 Auxiliary function

Some other functions are necessary to this tool. In particular a pre-processing computation of the look-up table is mandatory to evaluate the stress-strain behavior for the material parameter in use. This operation is done in `constitutiveModel.m` and `getTable.m`. Next to the look-up table, the thermal impedance can be calculated from the FEM simulation or from the datasheet time constant using file `getThermalImpedence.m`. Further, the post-processing file is required to export data from the binary format of Octave to a ASCII text file. This last operation has been necessary to be executed off-line (on pcDuino or another PC), because the data conversion (number or time to sting) wasn't possible in real-time due to a sensible slowing down of the performance; to export data in a text file the command `exportData.m` has to be called, and all the file created by the main program in the `tempData` folder are converted.

8.4 Sampling performance

The specifications given at the beginning of the development have been hard to satisfy in all the requirements. During the testing it has been observed that some built-in functions in Octave are slower compered to other method writhe ad hoc. In the first version of the software it was observed that an execution of one cycle step took more than 100ms due to bad solution adopted; in particular the serial communication, the number to sting conversion and other built-in Octave function such as `var(x,y)/cov(x,y)`. Moreover, as already known, it is mandatory to pre-allocate the memory space for the variable to avoid that Octave moves data and allocate new space. To pre-allocate the necessary space on the RAM, given the dumping time and the analysis performed, a procedure to calculate the amount of variables and array size has been added at the end of configuration file.

In the final version of the code, the average execution time of one analysis is 8ms. But many times it is observed that the execution of the code has taken more than 18ms. It is unclear why the execution slows down, but it is possible that some others Octave or operating system threads require CPU resources.

Since the execution time of the code is not constant, it has been chosen to increase the time step from 10ms to 20ms (or 25ms) to be sure that the all the samples can be analysed in time. The second option is to make the system running at the maximum speed without a fixed timing, but taking in consideration that the time step is not constant. In Figure 37 the average time of the execution when the sampling is at the maximum speed are plotted in logarithmic scale, and any clock timing is present.

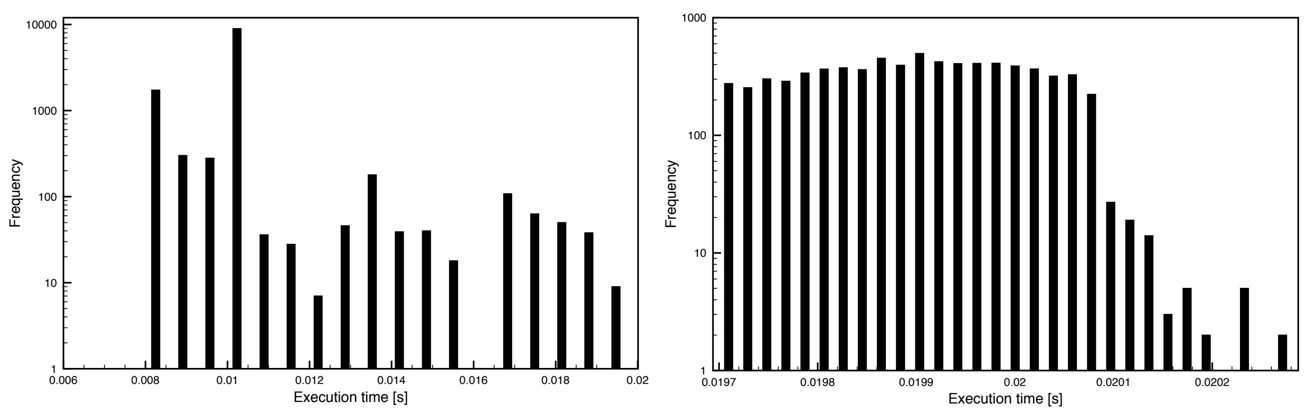


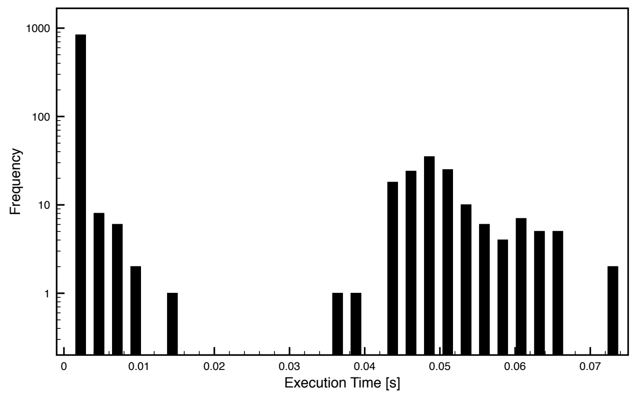
Figure 37. Comparison between the sampling performances when the timing clock is used and when the sampling occurs with a fix time of 20ms. Please note that the y-axis is in logarithmic scale. Moreover is worth to observe that execution average time is 8ms when no clock is present, while with a sampling time of 20ms the maximum error in the clock is about 200μs

The time necessary to create the look-up table, which is calculated separately and before the execution of the whole code, takes about 40 minutes (on PC/MAC) to built up the table with 4.8 million and about 35Mbyte. Since no particular memory limitations and table size issue related are present in pcDuino, the discretization is very fine: the temperature range considered goes from -50 to 150 ˚C with a step of 0.5 ˚C, the strain range is 0 to 0.003% with a $5 \cdot 10^{-5}$ % step, and the strain rate considered goes from 0.01 ˚C/s to 100 ˚C/s in logarithmic scale with 10 points/decade.

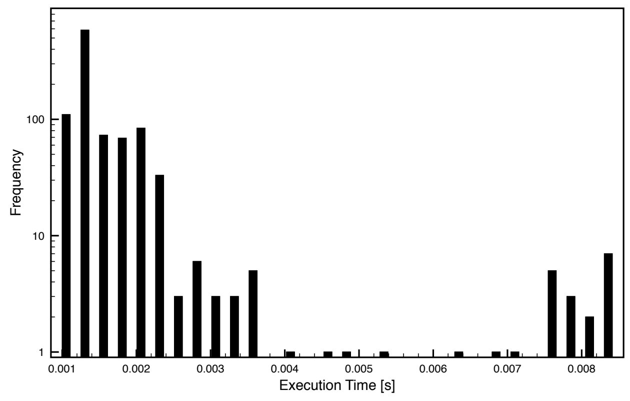
The time required to export data in a post-execution process is quite slow; the method to convert the number into strings are the main cause of this poor performance; to export one hour of data (sampling time 10ms) in binary format into ASCII format are required about 1:40 hour on pcDuino and 40 minute on PC.

8.5 Testing and results

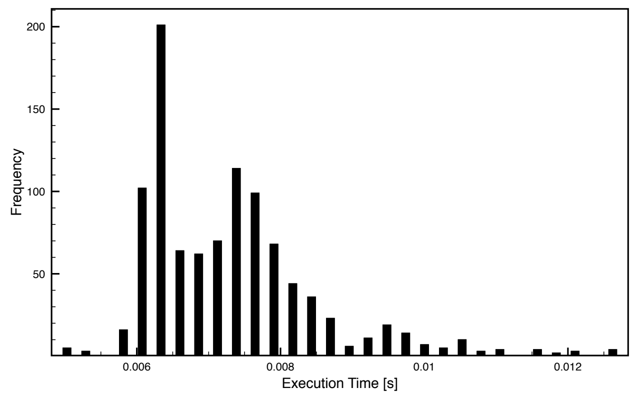
Each procedure of the whole system has been tested and optimized to speed up the computation and to try to reduce as much as possible the sampling time. The following plots (Figure 38) show the execution time of the most important procedures in the algorithm:



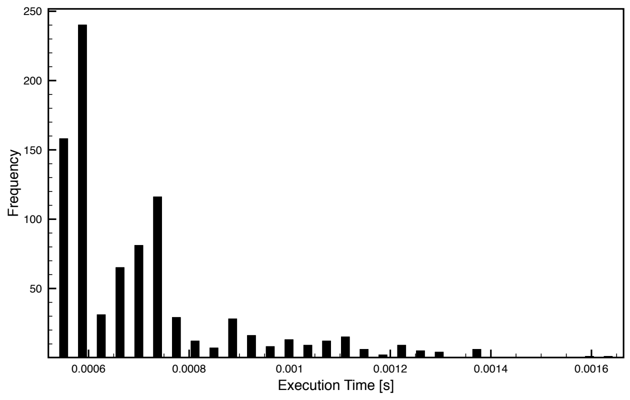
Rainflow not optimized
(note: logarithmic scale)



Rainflow optimized
(note: logarithmic scale)



Mechanical fatigue analysis

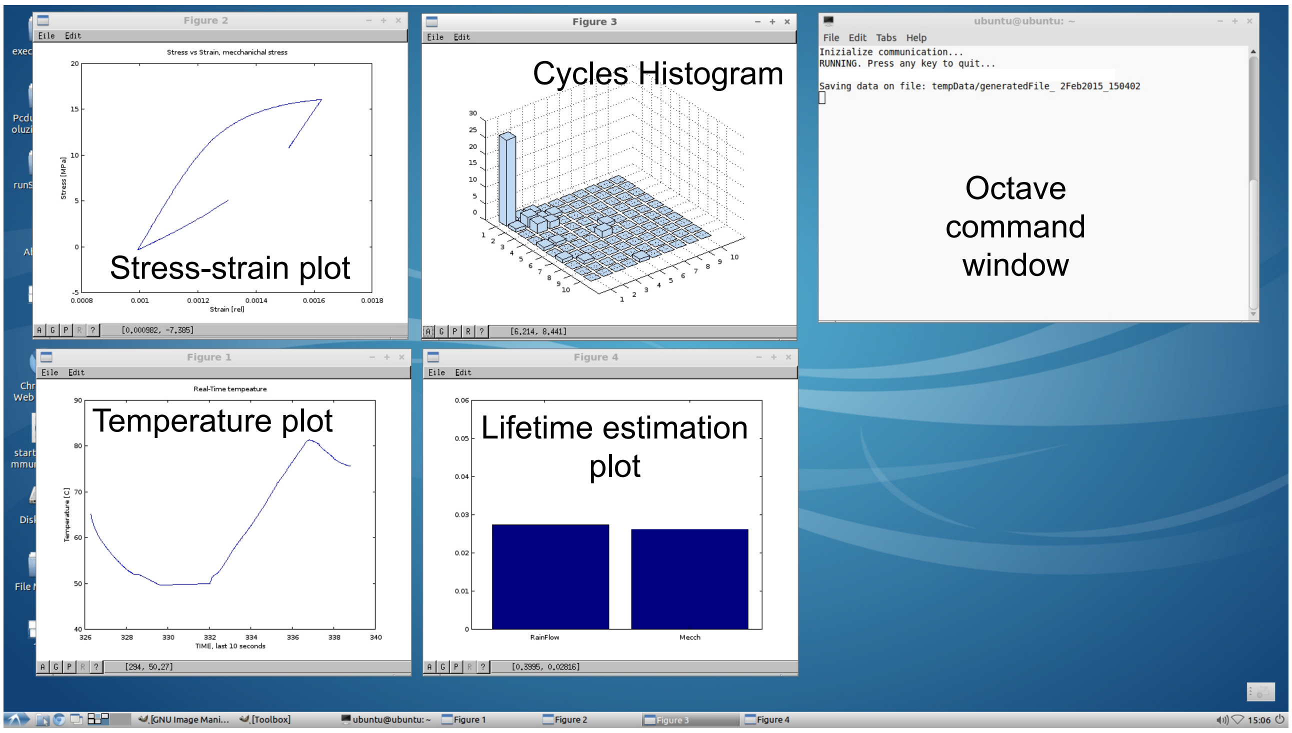


Sampling time

Figure 38. Running time execution of the principal process in the algorithm. Is important to observe the big difference in the Rainflow algorithm between the implementation not optimized, where the maximum and minimum are located using the variance and covariance Octave functions, and the faster version, where the slope change is individualized using the Least Mean Square method. The other two plots are the execution time of the mechanical analysis and the sampling performance and data transferring to Octave.

The datalog files, which save all the evolution of the system in real-time, takes up 10MByte/hour in Octave binary format, while, once the data are exported, 13Mb/hour in ASCII format.

The visualization of the output of data acquired is updated every second and only the last samples are displayed in the plots; in Figure 39 a screenshot of the system while running is reported.



```
Inizialize communication...
RUNNING. Press any key to quit...

Saving data on file: tempData/generatedFile_ 2Feb2015_150402
```

Figure 39. Screenshot of the software while running. All the plots are update in real time, the temperature and stress vs. strain plot every second, and the cycles histogram distribution and the lifetime estimation every 10 seconds. The histogram shows the cycles distribution sorted by amplitude range and mean temperature or duration (user selectable). On the terminal window are plotted the information about the system, like the files generated and error messages.

8.6 Examples

The tests conducted on this system are performed simulating the mission profile of a typical drive cycles. The data used as example are the samples taken by Artemis [20] which are plotted in Figure 7. In this case, the input considered is the dissipated power in the IGBT module that is estimated taking into account the motor speed and the current.

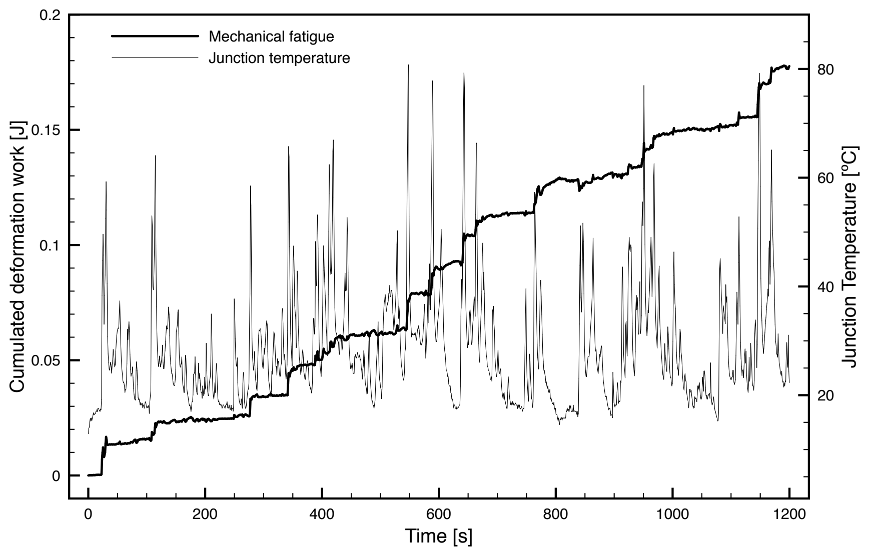


Figure 40. Example of thermo-mechanical fatigue analysis. The Junction temperature is estimated via convolution, and the mechanical fatigue is calculated by stress-strain algorithm.

In Figure 40 is compared the junction temperature in the IGBT module with the thermo-mechanical fatigue analysis. It is possible to observe that the damage introduced is higher when the temperature excursion is wide.

Note that there is no direct correlation of the two lifetime estimation algorithms: the Rainflow and the mechanical fatigue analysis. The relation that can be introduced is a correlation between the cumulated damage when the number of cycles to failure are occurred.

9 Conclusion

The work conducted propones an innovative application of the lifetime studies about the reliability of IGBT power modules. The system developed can be used in many contexts due to its high flexibility and portability. It is a very useful tool, which combined with available IGBT devices, is suitable to monitoring the health condition of power device and plan maintenance.

In the first part of the thesis it has been reviewed the typical failure mechanism and the wear-out damage that are the main cause of breakdown in IGBT modules. Further the most accurate method to estimate the end-of-life of IGBT modules has been explained, in particular the focus was on the Rainflow algorithm and the thermo-mechanical fatigue analysis. The Rianflow algorithm proposed here is a revisited version of the classical algorithm, where the innovative real-time feature is introduced. The mechanical damage introduced by the thermal expansion and compression is calculated through the solution of the constitutive equations and to speed up the computation of the mechanical behavior the look-up table is used.

Even though more the one issues occurred in the development of the software due to sampling performance or transferring data, those are been solved successfully, finding alternatives solution that have guaranteed the adequate sampling time and elaboration speed. Different microcontrollers and platforms have been taken into account in order to meet the elaboration speed specification.

This tool is open to further extensions and optimizations. Here Octave is used to reuse the code already written for the simulation and to be fast in the visualization of the output, but in a further developing of the code other solution has to be consider.

10 Bibliography

[1] M. Ciappa, “Some reliability aspects of IGBT modules for high-power applications,” 2000.

[2] J. Onuki, M. Koizumi, and M. Suwa, “Reliability of thick Al wire bonds in IGBT modules for traction motor drives,” *IEEE Trans. Adv. Packag.*, vol. 23, no. 1, pp. 108–112, 2000.

[3] D. Hirschmann, S. Member, D. Tissen, S. Schröder, and R. W. De Doncker, “Reliability Prediction for Inverters in Hybrid Electrical Vehicles,” vol. 22, no. 6, pp. 2511–2517, 2007.

[4] M. Ciappa, “Selected failure mechanisms of modern power modules,” *Microelectron. Reliab.*, vol. 42, no. 4–5, pp. 653–667, Apr. 2002.

[5] IEEE, “IEEE Standard Glossary of Software Engineering Terminology,” *Office*, vol. 121990, p. 1, 1990.

[6] B. Bertsche, *Reliability in Automotive and Mechanical Engineering: Determination of Component and System Reliability*. 2008, p. 500.

[7] Wikipedia, “Bathtub Curve.” [Online]. Available: http://en.wikipedia.org/wiki/Bathtub_curve.

[8] I. F. Kova, U. Drofenik, and J. W. Kolar, “New Physical Model for Lifetime Estimation of Power Modules,” pp. 2106–2114, 2010.

[9] ABB Semiconductors, “Application Note 5SYA 2043-04, Load-cycling capability of HiPak IGBT modules.” 2014.

[10] Astm, “ASTM Standards: G31-72,” 2012.

[11] I. T. Ag, “EconoPACK^{TM}4 IGBT Modules.” [Online]. Available: http://www.infineon.com/cms/en/product/promopages/econopack-4/. [Accessed: 08-Jan-2015].

[12] ABB, “HiPak IGBT.” [Online]. Available: http://www.abb.com/product/db0003db004291/c125739900722305c1256f18003a25ce.aspx. [Accessed: 07-Jan-2015].

[13] A. Hamidi, N. Beck, K. Thomas, and E. Herr, “Reliability and lifetime evaluation of different wire bonding technologies for high power IGBT modules,” *Microelectron. Reliab.*, vol. 39, no. 6–7, pp. 1153–1158, Jun. 1999.

[14] J. H. L. Pang, *Lead Free Solder: Mechanics and Reliability*. Springer, 2011.

[15] T. Siewert, “Properties of lead-free solders,” Colorado School of Mines, 2002.

[16] a. Morozumi, K. Yamada, T. Miyasaka, S. Sumi, and Y. Seki, “Reliability of power cycling for igbt power semiconductor modules,” *IEEE Trans. Ind. Appl.*, vol. 39, no. 3, pp. 665–671, May 2003.

[17] M. Handbook, “Reliability prediction of electronic equipment,” *MILHDBK-217F, Dep. defense, Washingt.*, p. 80, 1991.

[18] I. Paul, L. Beaurenaut, F. Sauerland, and M. Stoilkova, “Application based modified reliability tests and their physical correlation with lifetime assessment models,” in *Proc. PCIM 2013*, 2013, no. May, pp. 14–16.

[19] K. Mainka, M. Thoben, O. Schilling, and I. T. Ag, “Counting Methods for Lifetime Calculation of Power Modules,” no. 6, pp. 35–38, 2011.

[20] M. André, “The ARTEMIS European driving cycles for measuring car pollutant emissions.,” *Sci. Total Environ.*, vol. 334–335, pp. 73–84, Dec. 2004.

[21] P. Ghimire, S. Beczkowski, S. Munk-Nielsen, B. Rannestad, and P. B. Thogersen, “A review on real time physical measurement techniques and their attempt to predict wear-out status of IGBT,” in *2013 15th European Conference on Power Electronics and Applications (EPE)*, 2013, pp. 1–10.

[22] A. International, “ASTM E1049-85: Standard Practices for Cycle Counting in Fatigue Analysis,” 2011.

[23] O. S. Krzysztof Mainka, Markus Thoben, “Counting Methods for Lifetime Calculation of Power Modules,” *Infineon Technologies AG, Issue 6 2011 Power Electronics Europe*, Warstein, Germany, 2011.

[24] M. Ciappa, F. Carbognani, and W. Fichtner, “Lifetime prediction and design of reliability tests for high-power devices in automotive applications,” *IEEE Trans. Device Mater. Reliab.*, vol. 3, no. 4, pp. 191–196, 2003.

[25] L. GopiReddy, L. M. Tolbert, and B. Ozpineci, “Lifetime prediction of IGBT in a STATCOM using modified-graphical rainflow counting algorithm,” in *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, 2012, pp. 3425–3430.

[26] M. Musallam and C. M. Johnson, “An efficient implementation of the rainflow counting algorithm for life consumption estimation,” *IEEE Trans. Reliab.*, vol. 61, no. 4, pp. 978–986, 2012.

[27] C. H. McInnes and P. A. Meehan, “Equivalence of four-point and three-point rainflow cycle counting algorithms,” *Int. J. Fatigue*, vol. 30, no. 3, pp. 547–559, Mar. 2008.

[28] M. Ciappa, “Lifetime Modeling and Prediction of Power Devices,” in *5th International Conference on Integrated Power Electronics Systems*, 2008, pp. 27–36.

[29] M. N. Tamin and Y. B. Liew, “Numerical modeling of cyclic stress-strain behavior Of sn-pb solder joint during thermal fatigue,” in *National Seminar on Computational &amp; Experimental Mechanics (CEM) 2005*, 2005, pp. 351–360.

[30] G. R. Halford, “Low-cycle thermal fatigue,” Feb. 1986.

[31] D. Rubesa, *Lifetime prediction and constitutive modelling for creep?fatigue interaction*. Stuttgart, Germany: Schweizerbart Science Publishers, 1996.

[32] G. Z. Wang, Z. N. Cheng, K. Becker, and J. Wilde, “Applying Anand model to represent the viscoplastic deformation behavior of solder alloys,” *J. Electron. Packag.*, vol. 123, no. 3, pp. 247–253, 2001.

[33] X. Chen, G. Chen, and M. Sakane, “Prediction of stress-strain relationship with an improved Anand constitutive model for lead-free solder Sn-3.5 Ag,” *Components Packag. Technol. IEEE Trans.*, vol. 28, no. 1, pp. 111–3331, 2005.

[34] R. Darveaux and K. Banerji, “Constitutive relations for tin-based solder joints,” *Components, Hybrids, Manuf. Technol. IEEE Trans.*, vol. 15, no. 6, pp. 1013–1024, 1992.

[35] L. Anand, “Constitutive equations for hot-working of metals,” *Int. J. Plast.*, vol. 1, no. 3, pp. 213–231, 1985.

[36] S. BROWN, K. KIM, and L. ANAND, “An internal variable constitutive model for hot working of metals,” *International Journal of Plasticity*, vol. 5. pp. 95–130, 1989.

[37] C.-W. Pan, Y.-M. Lee, P.-Y. Huang, C.-P. Yang, C.-T. Lin, C.-H. Lee, Y.-F. Chou, and D.-M. Kwai, “I-LUTSim: An iterative look-up table based thermal simulator for 3-D ICs,” in *2013 18th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2013, pp. 151–156.

[38] H. Huang, A. T. Bryant, and P. A. Mawby, “Electro-thermal modelling of three phase inverter,” pp. 1–7.

[39] Z. Zhou, M. S. Khanniche, P. Igic, S. T. Kong, M. Towers, and P. A. Mawby, “A fast power loss calculation method for long real time thermal simulation of IGBT modules for a three-phase inverter system,” in *2005 European Conference on Power Electronics and Applications*, 2005, p. 9 pp.–P.10.

[40] J. W. Kolar, “Losses in PWM inverters using IGBTs,” *IEE Proceedings - Electric Power Applications*, vol. 142. p. 285, 1995.

[41] D. Graovac and M. Pürschel, “IGBT Power Losses Calculation Using the Data-Sheet Parameters Application Note,” 2009.

[42] J. Pou, D. Osorno, J. Zaragoza, S. Ceballos, and C. Jaen, “Power losses calculation methodology to evaluate inverter efficiency in electrical vehicles,” in *2011 7th International Conference-Workshop Compatibility and Power Electronics, CPE 2011 - Conference Proceedings*, 2011, pp. 404–409.

[43] Infineon Technologies, “AN2009-10: Using the NTC inside a power electronic module,” 2009.

[44] T. Hunger and O. Schiling, “Numerical investigation on thermal crosstalk of silicon dies in high voltage IGBT modules Model description,” in *PCIM Europe*, 2008.

[45] I. T. Ag, “TechnischeInformation / Technical Information FS200R07N3E4R IGBT-modules,” 2013.

[46] S. G. Kandlikar and C. N. Hayner, “Liquid Cooled Cold Plates for Industrial High-Power Electronic Devices—Thermal Design and Manufacturing Considerations,” *Heat Transfer Engineering*, vol. 30. pp. 918–930, 2009.

[47] C.-S. Yun, P. Malberti, M. Ciappa, and W. Fichtner, “Thermal component model for electrothermal analysis of IGBT module systems,” *Adv. Packag. IEEE Trans.*, vol. 24, no. 3, pp. 401–406, 2001.

[48] M. Ciappa, W. Fichtner, T. Kojima, Y. Yamada, and Y. Nishibe, “Extraction of accurate thermal compact models for fast electro-thermal simulation of IGBT modules in hybrid electric vehicles,” *Microelectron. Reliab.*, vol. 45, no. 9, pp. 1694–1699, 2005.

[49] Wikipedia, “Arduino,” *Wikipedia*. 2012.

[50] “Intel Galileo.” [Online]. Available: http://www.intel.com/content/www/us/en/do-it-yourself/galileo-maker-quark-board.html. [Accessed: 01-Dec-2014].

[51] “Raspberry-PI.” [Online]. Available: http://www.raspberrypi.org/. [Accessed: 01-Dec-2014].

[52] “pcDuino.” [Online]. Available: http://www.pcduino.com/. [Accessed: 01-Dec-2014].