Challenges and Strategies for a Real-Time Implementation of a Rainflow-Counting Algorithm for Fatigue Assessment of Power Modules

Antonios Antonopoulos
Department of Electric Power Engineering
Norwegian University of Science and Technology
Trondheim, Norway
antonios.antonopoulos@ntnu.no

Salvatore D'Arco
Department of Energy Systems
SINTEF Energy Research
Trondheim, Norway
salvatore.darco@sintef.no

Magnar Hernes
Department of Energy Systems
SINTEF Energy Research
Trondheim, Norway
Magnar.Hernes@sintef.no

Dimosthenis Peftitsis
Department of Electric Power Engineering
Norwegian University of Science and Technology
Trondheim, Norway
dimosthenis.peftitsis@ntnu.no

***Abstract*—The aim of this work is to obtain a reliable estimation of the remaining lifetime of power-electronic modules. Lifetime models provide information on the fatigue durability of power modules under repetitive loading circumstances. However, the thermo-mechanical stresses that a device is exposed to during operation are different than the ones evaluated to create the lifetime models. Rainflow counting provides a tool to analyze and evaluate the stress content of a randomly varying stress waveform, but counting stress cycles while the device is operating is challenging. In this paper, implementation challenges for an online rainflow-counting method are presented, and solutions to overcome them are discussed. An algorithm for online rainflow counting is implemented, and numerical results from tests on experimental device-temperature data are presented.**

***Index Terms*—Reliability, Power Modules, Rainflow Counting, Lifetime Models, Real-Time Simulation**

I. Introduction

An important cause for end-of-life type of failures in power devices is linked with cracks formed between the different conductive and insulating surfaces in the packaging structure of the semiconductor chip [1]. These cracks are the result of accumulated fatigue on the materials, due to repeated stress and strain forces caused by differences in expansion and contraction coefficients of the materials when exposed to temperature variations. Usually, fatigue-based cracks lead to bond-wire lift-offs, or solder-layer failures [2]. From an end-user's perspective, it is important to identify indicators in order to know the remaining lifetime of a power-electronic converter or module, and be able to replace it or plan for the appropriate maintenance, before the device results in a catastrophic failure.

One way to acquire information about the condition of a power device is through the well-known device-lifetime indicators, i.e., the conducting voltage, and the thermal resistance of the module [3]. Methods to measure these quantities are available in literature [4], [5], but face obstacles, especially when they are implemented online, such as lack of accuracy, or need for recalibration with device ageing. An alternative indicator is to look into the historical temperature profile that the semiconductor surfaces have been exposed to, assuming that the mechanical stresses in the packaging structure are a function of the junction-temperature variation. Typically, this method is applied for offline lifetime predictions in the design phase of a product [6], [7]. In this case, the availability of data about the ambient temperature variation is required, which can be combined with a (predicted) mission profile for the targeted application over a certain period of time (e.g. one year). Alternatively, the option is to obtain reliable information about the actual chip temperature during operation, which is not trivial, combined with the challenges imposed by the online processing of the acquired temperature data.

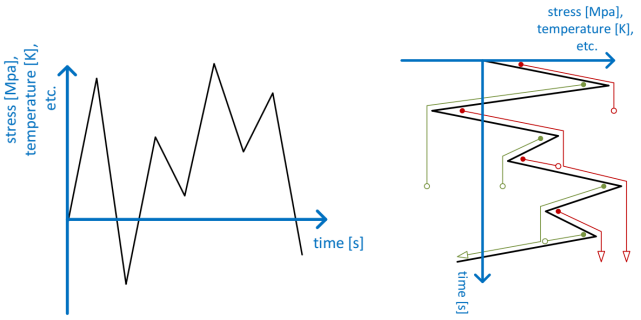
Ongoing research activities show that the temperature profile can be acquired either by direct measurements (using temperature sensors embedded in the module), or via indirect methods. Indirect approaches may include sensing of temperature-sensitive electrical parameters (TSEPs) [5], but can also be based on estimations using an accurate thermal model of the device in combination with the actual device losses, which are the main source of heat on the chip. Studying these methods falls outside the scope of this work; in this paper it is assumed that a reliable estimation of the temperature profile is available, and is generated simultaneously to the operation of the power converter. This work focuses on the challenge to analyze the temperature data online, and get the appropriate information to apply lifetime models and estimate the remaining lifetime of a device.

The main challenge is that the temperature variations applied on a power device in operation, in general, do not follow any specific pattern, and are difficult to predict. These

This is the accepted version of a paper published in 2019 IEEE Applied Power Electronics Conference and Exposition - APEC
DOI: 10.1109/APEC.2019.8722284

variations depend on many different factors, including the load variations, the ambient temperature, the device ageing etc. Their random character makes it hard to analyze them in simple components or identify if they include any repetitive patterns. Simple components or patterns could be reproduced in offline power-cycling experiments, and could be related to a certain stress content. A tool for analyzing randomly-varying stress waveforms into individual simple stress components is rainflow counting [8], [9], which has been historically developed for studying fatigue-based failures in mechanical-engineering applications. In order to apply the original rainflow-counting algorithm though, the whole time history of the stress waveform needs to be available. This is not possible when the algorithm should be implemented to calculate stress while the device is in operation (online), as the counting process has to take place simultaneously with the generation of the device-temperature waveform. Another problem with the online application is that very long time may be necessary for the complete evaluation of some stress components. This is a disadvantage of the rainflow-counting algorithm itself, as the stress components are determined by the future behaviour of the processed waveform. As a result, this method requires sufficient physical memory availability in the processing system. The long (actually undefined) evaluation time, along with the random nature of the temperature profile makes the amount of stress components to be simultaneously evaluated unknown during the development of the algorithm. It is inefficient then to reserve a defined amount of memory for the needs of the rainflow counting in the development phase; alternatively, dynamic memory-allocation techniques have to be utilized, creating further challenges for the software architecture. If these obstacles are dealt with, the rainflow-counting algorithm can become a powerful and efficient tool to analyze stress profiles into simple components in real time. Lifetime models can then be applied on these components, providing a reliable indication of the actual consumed lifetime, while the power device is in operation.

Unlike previous efforts, mainly applying rainflow-counting techniques in offline measurements [6], [7], this paper focuses on the efficient implementation of a rainflow counting algorithm, dealing with the constraints an online implementation imposes. A first effort for a similar approach has been presented in [10], [11]. In the current approach, the runtime performance of the algorithm is enhanced, and the possibility for running it in a real-time control system is demonstrated. Several modifications of the algorithm are also discussed, which may improve the runtime performance even further, but may compromise the accuracy of the damage calculation. This paper starts with a short overview of a rainflow-counting algorithm, and gives a detailed description of the online implementation for power-device lifetime-estimation applications. Techniques that enhance the performance of the algorithm and enable a more resource-efficient implementation for real-time applications are evaluated. Then, the lifetime model used to translate the results of the rainflow counting into accumulated damage is discussed, and finally the algorithm is tested in simulations, in order to evaluate its performance.



(a) (b)

Fig. 1. Random stress signal to be evaluated (a). Waterflows according to the original rainflow-counting algorithm (b).

II. Rainflow-counting algorithm

The rainflow-counting method was first introduced by Matsuishi and Endo in 1968 [8], in order to obtain the stress/strain cycles under randomly fluctuating mechanical-loading conditions. The algorithm is nowadays described in international standard governing practices for cycle counting in fatigue analysis of mechanical components [9].

A. Original algorithm

A simple presentation of the steps the original rainflow-counting algorithm follows is given in the following:

1) The algorithm identifies the signal-reversal points, i.e., peaks and valleys from the stress-history (or temperature-history) waveform.
2) The amplitudes and the sequence of signal-reversal points are stored in memory. The algorithm does not keep any intermediate points from the original signal, so the stored information from the input-signal waveform is simplified to a form as shown in Fig. 1a.
3) The waveform is turned clockwise 90°, as shown in Fig. 1b.
4) Each stress-reversal point (peak and valley) can be imagined representing a source of water flowing down the "pagoda"-shaped rotated waveform signal.
5) Each waterflow is terminated in any of the following cases:
   - When it meets a new water source of the same kind as its sourcing point, but with a "deeper" source.
   - When it merges with a flow dripping from an earlier (and deeper) source.
6) Each terminated flow is assigned a "range" value, equal to the length it has travelled on the horizontal axis. The range value is referring to the stress (or temperature) difference between its sourcing point and termination point. A terminated flow represents a half cycle of stress with the assigned magnitude.

The disadvantage of the rainflow counting method is that the resulting range of each stress cycle depends not only on the history, but also on the future behaviour of the stress signal. In order to make a proper evaluation of the signal, the whole time-history of the waveform needs to be available.

This is the accepted version of a paper published in 2019 IEEE Applied Power Electronics Conference and Exposition - APEC
DOI: 10.1109/APEC.2019.8722284

A rainflow-counting implementation aiming to process the signal in real time (while the signal is generated) is obviously unable to determine directly the damage contribution of all the apparent waterflows at each time step. A solution is to calculate the damage from all terminated waterflows until the actual time point, and store the available information for the remaining (open) waterflows in memory, until they become "terminated" and can be evaluated. However, the maximum time that this information needs to remain stored in memory cannot be defined, as it is related to the future behaviour of the input signal. Furthermore, in real application conditions, the amount of "open" waterflows can theoretically be unlimited. In practice, it can be restricted if the expected maximum and minimum temperature is known, and the accuracy of the temperature measurement (or estimation) is given. In all cases, the problem with a real-time implementation is the possibility to end up in a situation with a very large number of open waterflows. This amount may overflow the reserved amount memory in the processing system, but, most importantly, the "open" waterflows do not give (yet) any indication about the damage the device has suffered due to them. In such a situation, the indicated damage underestimates the real damage that the monitored device has suffered. In order to evaluate how severe this problem can be in practice, a strict implementation of the original rainflow-counting algorithm in a real-time processing system is evaluated in the first place, without caring about potential memory limitations. Then, two modifications in the algorithm have been evaluated as potential solutions to the aforementioned challenges.

Fig. 2. Flowchart of the online rainflow counting algorithm implementation.

B. Real-time implementation

The basic data structure utilized in this algorithm is a list of records. Each record contains data fields referring to one waterflow sourcing from a peak reversal. The relevant data values in this record are: the absolute time when the sourcing signal reversal appeared, the "altitude" (y-axis value) of the source, the accumulated on-time until the current time, the actual water height of the open flow, and finally, the actual status of the waterflow (open/closed). A list is selected as the basic data structure in this architecture, even if it has a more abstract character compared to a stack or a queue. A stack or a queue would be inappropriate in this case, as their definition allows access to their content only in a predetermined sequence. A queue allows removal of only the older list element (first in, first out), while a stack allows removal of only the latest list element (last in, first out), which is not useful in this application. The advantage of the list is that each time a flow is closed, its damage contribution can be evaluated immediately, and then the relevant record can be removed from the list, irrespectively of its actual placement in this list. In the rainflow-counting algorithm there is no guarantee that the closing waterflow always appears at a certain place in this list; on the contrary, it is most often neither the oldest nor the latest waterflow that closes when the algorithm is processing a signal in real time. In this way, a list has the benefit of reserving the least amount of memory among other data structures, as the records can be removed immediately as soon as their respective waterflows are closed, so the memory space is released to be reused by new flows. A flowchart explaining the steps of this implementation is shown in Fig. 2. The minimum amount of necessary flows to keep in this list includes certainly the ones that are still open, as well as the two most recent ones (sourcing from the latest two signal reversals of each kind), irrespectively of their status.

C. Additional criteria for closing the flows

Following the implementation of the original rainflow-counting algorithm described above, it becomes obvious that there is no other mechanism that secures the closing action of a waterflow, except for meeting a strictly higher peak or lower valley than the sourcing point. Supposing a randomly-varying temperature waveform, however, there is no guarantee for the maximum time that this event will happen, so each flow can remain open for (theoretically) infinite amount of time. The problem with a large number of open flows is twofold; they both reserve memory for a significant amount of time, and also their damage contribution cannot be evaluated before each flow is closed. This fact leads to a continuous underestimation of the cumulative damage, especially when these (open) flows have a significant amplitude and/or on-time, which are both indicators of significant damage contributions. In order then to secure that waterflows will close, and also that

This is the accepted version of a paper published in 2019 IEEE Applied Power Electronics Conference and Exposition - APEC
DOI: 10.1109/APEC.2019.8722284

the memory requirements will be limited, special strategies can be implemented to enhance the performance of the algorithm in runtime. Two potential strategies are:

1) **Close flows within a predefined amplitude range**. In this strategy, the requirement of meeting a strictly higher peak or lower valley for a flow to be closed is relaxed. According to this rule, two signal reversals can be regarded as equal in amplitude even if the most recent one is lower, as long as it appears within a given amplitude range below the older one. This convention is used strictly as a closing criterion. The actual signal-reversal magnitudes should not be rounded up/down, as this can disturb the stress-signal data significantly. For a temperature signal, it can be assumed, for instance, that two peaks that differ below 1°C can be treated as equal. The result will be to close the flow sourcing from the older peak, even when meeting a new peak that is 1°C below the sourcing peak (or higher for a valley). Given that the temperature estimation/measurement has limited accuracy, this situation may anyway appear within the measurement accuracy range; however, it can be expanded to ranges significantly higher than the accuracy of the temperature signal.
2) **Close flows in predefined maximum-time windows**. As already mentioned, the damage contribution of a flow cannot be evaluated before the flow is formally closed. However, due to the actual temperature or load variations, a number of flows may remain open for a very large amount of time, often without any additional contribution to its data values at the closing moment. In this case, both the damage contribution of a flow will be evaluated much later than it could, and memory space is reserved for a useless purpose. An additional criterion to close the flows can then be to look into the absolute time a flow was generated, and force the flow to close after a certain time limit. This will result in a maximum allowable time to keep an open flow in the memory, and at the same time give an up-to-date damage estimation.

Implementation of any of these strategies results in some inaccuracy in the calculation of the stress levels compared to the original rainflow-counting algorithm, as some flows are forced to close prematurely. A prematurely closed flow results in a smaller damage contribution, but it may also have the opposite effect: it will be unable to stop other (more recent) flows at the appropriate level, allowing them to appear as larger stress contributors than they actually are. It is not a priori clear if this effect will cause an underestimation, or even an overestimation of the accumulated damage, but will be shown in the following simulations. The level of inaccuracy depends on the shape of the actual stress waveform in consideration as well. On the other hand, these strategies are expected to improve the runtime performance of the online rainflow-counting algorithm. The significance of this inaccuracy and the performance trade-off in each case will be evaluated. In order to have a fair comparison, these modified rainflow-counting strategies have to process the exact same waveform data as the original rainflow-counting processes in the real-time implementation. For this purpose, the temperature-waveform data are saved, and the modified strategies are implemented on the same input data offline.

III. Damage calculation

The rainflow-counting algorithm analyzes the random junction-temperature waveform into simple contributions of amplitude variations (ΔT), at a given junction temperature (T_j), applied for a certain duration (t_{on}). The estimated lifetime in load cycles under a (repetitive) stress profile can be evaluated using a lifetime model, as the one presented in [12], expressed by

$$N_f = K\Delta T^{\beta_1} e^{\frac{\beta_2}{T_j+273}} t_{\text{on}}^{\beta_3} I^{\beta_4} V^{\beta_5} D^{\beta_6}. \tag{1}$$

In this model, the β-exponents and the K-factor are determined from offline power-cycling experiments performed on the power modules planned to be used in the targeted application. Damage contributions from different stress components can be combined using Miner's rule, as in

$$\text{Damage} = \sum_{f=1\ldots k} \frac{n_f}{N_f}. \tag{2}$$

These equations are implemented in runtime, and are applied to every closing waterflow, on the resulting $\{\Delta T,\ T_j,\ t_{\text{on}}\}$ data packages.

IV. Implementation in a real-time digital simulation (RTDS) system

The rainflow counting, along with the damage-calculation function are implemented in a real-time system, in order to evaluate the feasibility of running these algorithms in real time. The real-time system operates as a control and measurement platform for a downscaled modular multilevel converter (MMC) designed to simulate a high-voltage direct current (HVDC) application. A load pattern is simulated for 40 s, as in Fig. 3. The real-time processing system is collecting the actual current waveform together with the switching pattern of a converter submodule, and derives the voltage and current stress of an equivalent submodule in a high-voltage (full-scale) application.

The semiconductor device studied in this paper has been the Mitsubishi CM1000HC-66R power module. It has been subject to power-cycling tests under different conditions, in order to derive its dedicated parameters for the lifetime model in (1). Significant efforts have been made to obtain a reliable thermal model of the module as well, and derive the parameters of a Foster model [13], [14]. Using this thermal model and assuming full-scale HVDC application conditions, it is possible to derive both the device losses under the given load conditions and the junction-temperature variation. The process for calculating this junction temperature is running in the same real-time system, creating temperature waveforms for IGBT and diode modules as in Fig. 3.

Test runs have shown that the real-time system is definitely capable of adding the lifetime estimator functionalities to its existing processes, and also that the memory requirements of the original algorithm are extremely low. However, as the

This is the accepted version of a paper published in 2019 IEEE Applied Power Electronics Conference and Exposition - APEC
DOI: 10.1109/APEC.2019.8722284

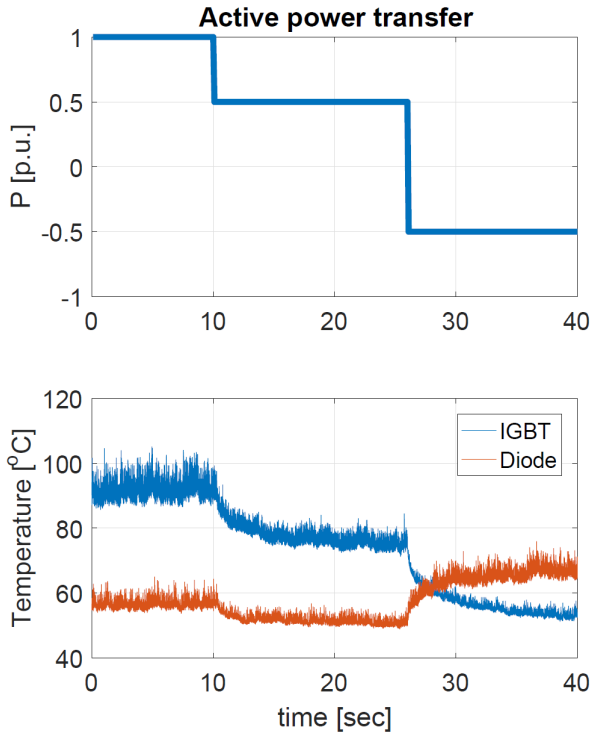


Fig. 3. Active power transfer and resulting junction-temperature variation on an IGBT and a diode chip within a converter submodule.

temperature waveform cannot be regenerated in every single detail on two different testing occasions, the evaluation of the aforementioned strategies needs to be performed offline, in order to compare the results on exactly the same temperature data.

V. Simulation Results

In the first place, the original rainflow-counting algorithm is implemented in the real-time control platform, and operates on the IGBT-temperature waveform shown in Fig. 3, along with the waveform generation. Then, the temperature data are saved, and the modified algorithms are applied offline on the same waveform. A selection of amplitude ranges and maximum-time windows is tested as criteria for closing the flows. The selected values are evaluated based on the maximum number of simultaneously open flows that appear when processing the temperature signal (i.e., equivalent to the memory usage), and also on the error they cause on the accumulated-damage calculation compared to the original rainflow-counting algorithm. The results of these simulations are shown in Table I.

In case the modified amplitude range is used in the closing criteria, the trend for reduced memory usage is clear, as the closing range grows to larger values. However, already at 1°C away from the actual value, the error in damage calculation becomes significant, which results in an underestimated value for the accumulated fatigue in the device.

In case of the maximum-time windows, on the other hand, the results are very different than expected. Even if the flows are forced to close in narrow time windows (e.g. 5 s), there

TABLE I
Simulation Results

Close flows within a predefined amplitude range		
Range [°C]	Max # of open flows simultaneously	Error in damage [%]
(Original) 0.0	11	0.0
0.1	11	-2.8
0.5	10	-9.4
1.0	8	-32.2
Close flows in predefined maximum-time windows		
Max time window [s]	Max # of open flows simultaneously	Error in damage [%]
(Original) Infinite	11	0.0
20	11	+11.1
10	11	+11.1
5	11	+7.9

seems to be no memory benefit when processing the waveform shown in Fig. 3. At the same time, the accumulated fatigue in the device becomes overestimated. The reason behind this effect is that the "older" flows (which are forced to close prematurely) would have stopped flows generated later in the time history from growing to higher stress levels. These stress levels now appear as more significant than they should compared to the original rainflow-counting algorithm application, and increase the estimate of the total accumulated damage.

It has to be noted that the experimental results presented here are not sufficient to draw safe conclusions on the general behaviour of these methods. A limited number of load variations is tested, and the full length of the recording is quite short (40 s). As a result, the maximum number of simultaneously observed open flows is small, which makes the improvement obtained by the strategies perceived as insignificant. In order to quantify this improvement in a more solid way, a load variation with higher memory requirements has to be evaluated. The results must also be demonstrated in recordings with longer duration, as these algorithms are supposed to run throughout the whole lifetime of the power modules.

It is remarkable though, that the damage-estimation error appears in different (but fixed) direction for each method. Once again, the short duration of the recorded window does not allow for testing of any other maximum-time values than the ones presented in Table I, so the actual error trend cannot lead to safe conclusions. Similarly, in case of the amplitude-range method, the actual error seems to grow significantly, even for quite small values of the range (e.g. 1°C). It can be claimed that this is the result of the exact circumstances described by this IGBT-temperature waveform, in this window; however, that high error in damage calculation is difficult to ignore. If this error is confirmed for such low temperature-measurement differences (which is close to the actual measurement-accuracy level), significant hesitation can be justified regarding the applicability of this whole lifetime-estimation methodology.

This is the accepted version of a paper published in 2019 IEEE Applied Power Electronics Conference and Exposition - APEC
DOI: 10.1109/APEC.2019.8722284

VI. Conclusion

This paper has investigated the implementation of a rainflow-counting algorithm in a real-time system, aiming for online assessment of the fatigue accumulated in power modules during operation. Applying the method described here, it becomes possible to realize such an implementation within a modern real-time control application. It has been also shown that this implementation does not require a very large amount of memory in realistic-application conditions, even if a number of flows has to remain open and available in memory for an undefined amount of time.

However, two methods aiming to reduce the memory requirements have been developed and evaluated as well. Such methods could be included in the algorithm, but applied conditionally, only if the amount of system memory becomes critically low and can compromise the numerical results. It has been shown that relaxing the closing criteria from the strict "lower or equal" to "lower, equal or higher but within a given range" has a positive impact on the memory usage, and presents a clear trend in the underestimation of the calculated damage. If this relaxation is kept within a reasonable range, this method can be used to reduce the memory requirements, with a predictable impact on the fatigue assessment. On the other hand, closing the flows based on maximum time-window criteria does not present significant improvement on the memory usage, and also disturbs the accumulated damage in a way that is not easy to predict. As a result, this type of criteria for closing the flows should be avoided when implementing a rainflow-counting algorithm.

References

[1] J. Lutz, "Packaging and reliability of power modules," in *Proc. CIPS 2014; 8th Int. Conf. Integrated Power Electronics Systems*, Feb. 2014, pp. 1–8.

[2] ——, "IGBT-modules: Design for reliability," in *Proc. 13th European Conf. Power Electronics and Applications*, Sep. 2009, pp. 1–3.

[3] U. M. Choi, F. Blaabjerg, and S. Jørgensen, "Power cycling test methods for reliability assessment of power device modules in respect to temperature stress," *IEEE Trans. Power Electron.*, vol. 33, no. 3, pp. 2531–2551, Mar. 2018.

[4] C. Herold, J. Franke, R. Bhojani, A. Schleicher, and J. Lutz, "Methods for virtual junction temperature measurement respecting internal semiconductor processes," in *2015 IEEE 27th International Symposium on Power Semiconductor Devices IC's (ISPSD)*, May 2015, pp. 325–328.

[5] H. Luo, W. Li, F. Iannuzzo, X. He, and F. Blaabjerg, "Enabling junction temperature estimation via collector-side thermo-sensitive electrical parameters through emitter stray inductance in high-power IGBT modules," *IEEE Trans. Ind. Electron.*, vol. 65, no. 6, pp. 4724–4738, Jun. 2018.

[6] F. Hahn, M. Andresen, G. Buticchi, and M. Liserre, "Thermal analysis and balancing for modular multilevel converters in HVDC applications," *IEEE Trans. Power Electron.*, vol. 33, no. 3, pp. 1985–1996, Mar. 2018.

[7] L. Wang, J. Xu, G. Wang, and Z. Zhang, "Lifetime estimation of IGBT modules for MMC-HVDC application," *Microelectronics Reliability*, vol. 82, pp. 90 – 99, 2018.

[8] M. Matsuishi and T. Endo, "Fatigue of metals subjected to varying stress," *Japan Society of Mechanical Engineers*, vol. 68, no. 2, pp. 37–40, 1968.

[9] *Standard practices for cycle counting in fatigue analysis*, ASTM E 1049-85 (Reapproved 1997) Std.

[10] M. Musallam, C. M. Johnson, C. Yin, C. Bailey, and M. Mermet-Guyennet, "Real-time life consumption power modules prognosis using on-line rainflow algorithm in metro applications," in *2010 IEEE Energy Conversion Congress and Exposition*, Sept 2010, pp. 970–977.

[11] M. Musallam and C. M. Johnson, "An efficient implementation of the rainflow counting algorithm for life consumption estimation," *IEEE Transactions on Reliability*, vol. 61, no. 4, pp. 978–986, Dec 2012.

[12] R. Bayerer, T. Herrmann, T. Licht, J. Lutz, and M. Feller, "Model for power cycling lifetime of IGBT modules - various factors influencing lifetime," in *Proc. 5th Int. Conf. Integrated Power Electronics Systems*, Mar. 2008, pp. 1–6.

[13] G. Bergna, S. D. Arco, J. A. Suul, and M. Hernes, "Analysis of power cycling for semiconductor devices in modular multilevel converters," in *2016 IEEE 17th Workshop on Control and Modeling for Power Electronics (COMPEL)*, June 2016, pp. 1–7.

[14] L. Tinscher, M. Hernes, and J. Lutz, "Improving the short circuit ruggedness of igbts," *Microelectronics Reliability*, vol. 64, pp. 519–523, 2016.

This is the accepted version of a paper published in 2019 IEEE Applied Power Electronics Conference and Exposition - APEC
DOI: 10.1109/APEC.2019.8722284