Data-Driven Approach for Feature Drift Detection in Embedded Electronic Devices

Oussama Djedidi, Mohand Djeziri, Nacer M'Sirdi

Data-Driven Approach for Feature Drift Detection in Embedded Electronic Devices ⋆

Oussama Djedidi ** **Mohand A. Djeziri** ***
Nacer K. M'Sirdi ****

Aix Marseille University, Université de Toulon, CNRS, LIS, SASV, Marseille, France.
** *e-mail: oussama.djedidi@lis-lab.fr*
*** *mohand.djeziri@lis-lab.fr*
**** *nacer.msirdi@lis-lab.fr*

Abstract: This paper is a part of a project aiming to develop supervisor and monitoring devices for embedded systems in airplanes and vehicles. It focuses on the reliability of these systems and establishes a monitoring framework to detect drifts and faults in the behavior of the heterogeneous central processing units (CPU) and graphics processing units (GPU) chips powering them. In this work, we use a previously developed incremental model of these chips and associate it with a fault detection algorithm. Estimations from the model constitute inputs to the diagnosis module. The latter generates alarms in the presence of faults or drifts in the characteristics and features of the System-on-Chip (SoC). The obtained results validate the proposed monitoring algorithm and demonstrate the effectiveness of the fault detection algorithm.

Keywords: Analytical redundancy, Embedded systems, Modeling, Monitoring, Microprocessors, System identification, System-on-chip

1. INTRODUCTION

The study of the reliability of airborne or wheeled transportation machinery has focused mainly on the moving parts of the vehicle. With the integration of new smart technologies and the moves towards electric energy to power these vehicles, diagnosis and reliability studies needed to be adapted to take into account multiple engines setup, the batteries, and hybrid system control that came along. Moreover, the development of onboard driving assistance devices and autonomous vehicles has motivated the scientific community to develop monitoring algorithms for embedded electronic systems. These Systems-on-Chips (SoCs) are generally embedded in a complex environment, with cycles of heating and cooling related to the operation of the engines, as well as vibratory conditions with high variability, which might cause accelerated aging of these devices compared to the average lives announced by the manufacturers. The purpose of this paper is to develop and test a monitoring scheme for main components of the SoCs embedded in safety-critical systems i.e. central processing units (CPU) and graphics processing units (GPU).

A majority of the existing works in this field are based on causal models, like the directed graph by Zhang (2005) or the fault tree by Wang et al. (2011), for instance. In such models, for the system to function properly all of its components have to be fully and correctly operational Steininger (2000). However, an enhanced dependency model presented by Cui et al. avoids the shortcomings of the existing works mainly by allowing for the disregarding and the elimination of multiple faults, by including symbol switches representing mechanisms to disconnect one part from the main body of the model in their dependency graphical model (DGM).

The review published by Gizopoulos et al. (2011) is a thorough study of online error detection works done on multicore processors. These approaches are classified into four main categories: redundant execution Aggarwal and Ranganathan (2007); LaFrieda et al. (2007); Mukherjee et al. (2002), periodic Built-In Self-Test (BIST) approaches Shyam et al. (2006), dynamic verification approaches Austin (1999); Meixner et al. (2007), and anomaly detection approaches Wang and Patel (2006); Li et al. (2008). In one of the main conclusions of this review, it showcased the success of the dynamic verification approaches in detecting both transient and permanent faults, and also design bugs. A more general overview of all diagnosis and fault-tolerant techniques can be found in an extensive survey established by Gao et al. (2015a,b).

This paper is a follow up on the work we presented in Djedidi et al. (2017), in which we proposed and validated and incremental interconnected model that describes the dynamics of the frequencies, the voltages, the temperatures, and the power consumption of an ARM-based SoC. The processor used the most in autonomous vehicles. In this work, we use this model and its outputs as inputs for the monitoring algorithms for the early detection of faults and drifts in the system.

⋆ This work is a part of the MMCD project supported and funded by the Banque Publique d'Investissement (BPI), to whom we address our thanks.

By definition, the role of the monitoring subsystem is to flag errors and irregularities found in the surveilled variables and features. It also exploits the incremental structure of the model and the specialized nature of each subsystem (each one estimate only one variable) to detect and isolate faulty components.

The main advantage of the hereafter proposed monitoring algorithm, is its reliance on data already provided by the system itself. Thus, it can be deployed on all current and forthcoming SoCs, after model training. Once running, one can easily follow its fault indicators to monitor over the state of the device, intercept errors, investigate the effect of these errors, and even view wear-traits for predictive maintenance planning and remaining useful life calculations.

In the next section, the general proposed monitoring approach is presented. In section 3, we explain the fault detection and isolation (FDI) algorithm, detailing residual generation and evaluation, and then illustrating the decision-making process. Section 4 is dedicated to the presentation and discussion of the obtained experimental results, where we validate the FDI algorithm by analyzing residuals in normal and faulty scenarios. Finally, the last section is a conclusion highlighting the results of this paper.

2. GENERAL METHODOLOGY

In the introduction, we mentioned monitoring and FDI methods that rely upon—amongst others—built-in tests, redundancy, or verification. The method hereafter described is a complementary one; it monitors the system to provide an early detection of drifts in its functions caused by wear and over-solicitation. Moreover, analysis of these drift phenomena may allow, in addition to the early detection of faults, the study of the life-cycle of the system and factors accelerating its wear.

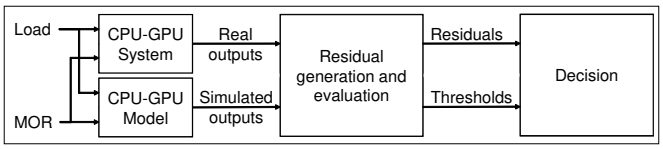

Fig. 1. A general block diagram of the monitoring approach.

Fig. 1 is a diagram of the monitoring algorithm we applied to a heterogeneous SoC. In this algorithm, the universal inputs for both the system and the estimation model are the CPU and GPU loads and the Memory (RAM) Occupation Rate (MOR). The load is defined as the relative busy time of the processor during a sampling period in percent. As for the MOR, we define it as the ratio of the occupied RAM relative to its full size. These inputs allowed us to construct an estimation model (see Djedidi et al. (2017)). This model is built in a modular structure, as a set of interconnected subsystems. In the first set of modules, frequencies per-core and–consequently–voltages per-core are calculated according to the present computational load. They are then applied to the second set of modules, in which they are used to estimate the power consumption and temperature of the SoC. The detailed modeling process of each of the subsystems along with their validation and the advantages of this model are presented by Djedidi et al. (2017).

The aforementioned variables characterize the operating state of the SoC and are used as inputs to the monitoring algorithm (c.f. Fig. 1). The algorithm is based on analytical redundancy. In this technique, outputs from the system are compared to those from the estimation model which in this case is called a reference model. During normal operations, outputs from the system are equal—within a margin of error—to those from the model. If a fault or an error occurs, these outputs will diverge from each other. The differences between the set of the two outputs are fault indicators commonly known as residuals. The latter is then processed by signal processing and probabilistic techniques in order to avoid erroneous decisions due to modeling uncertainties.

Although analytical redundancy has been widely used for fault diagnosis in systems without software components, to our knowledge, it has not been used previously to monitor system with both hardware and software components.

Finally, thanks to the modular structure of the proposed estimation model, each generated residual is associated with clearly identified modules, allowing an easy isolation of faulty subsystems.

3. RESIDUALS PROCESSING

The monitoring method, in this work, relies on the processing of residuals which consists their generation and evaluation.

3.1 Residuals generation

Raw residuals are generated from the difference between measured and reference values from the model (c.f. Fig. 1), and are computed as shown in equation 1.

$$\begin{cases} r_T = (T_{estm}(t) - T_{meas}(t))/T_{estm}(t) \\ r_P = (P_{estm}(t) - P_{meas}(t))/P_{estm}(t) \\ r_{i_V} = (V_{i_{estm}}(t) - V_{i_{meas}}(t))/V_{i_{estm}}(t) \\ r_{i_f} = (f_{i_{estm}}(t) - f_{i_{meas}}(t))/f_{i_{estm}}(t) \end{cases} \quad (1)$$

i specifies the GPU or the core number of the CPU, while the the subscripts $_{estm}$ and $_{meas}$ denote model estimations or measured values, respectively. These residuals are the dimensionless estimation errors and are the appropriate choice for the detection of deviations in the functioning of the system, especially progressive ones which are the main indicator of degradation.

3.2 Residuals evaluation

During normal operations, theoretically, estimations are equal to measurements, and residuals are equal to zero. In this case, non-zero residuals would only occur, if the outputs of the system deviate from those of the of the reference model when an unforeseen event or a problem occurs. But, on a real system, such non-zero residuals would also occur due to estimations errors. Hence, to avoid false flags, residuals also undergo an evaluation process.

Since the residuals originate from measurements and estimations, we use signal-based approaches to evaluate them,

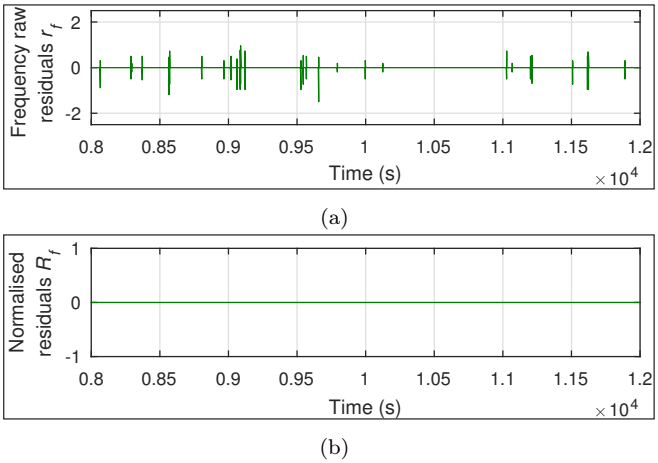

Fig. 2. Frequency residuals during normal operation. (a): raw residuals r_f, (b): normalized residuals R_f.

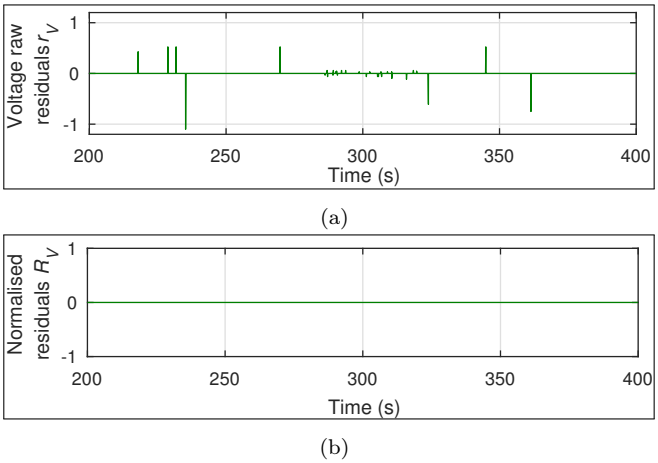

Fig. 3. Voltage residuals during normal operation. (a): raw residuals. (b): normalized residuals.

rather than models-based ones Adrot et al. (1999); Djeziri et al. (2009).

Signal-based methods ignore the origin of the signal—in this case, the residuals—but rather use its statistical and probabilistic properties of in normal operation as a reference to make a decision when an abnormal behavior occurs (generate an alarm, for instance) Djeziri et al. (2017); Basseville (1998); Ge et al. (2009); Benmoussa et al. (2010).

Evaluation of the frequency and voltage residuals Since these residuals are mainly equal to zero except for the spikes that are due to the estimation lag (Fig. 2a and Fig. 3a). To avoid the false alarms caused by this lag, we set a maximum tolerated delay value τ_{ref} from the data, and use it to calculate normalized residuals R_f and R_V (Equation 2).

$$R_x = \begin{cases} 1 & \text{if } r_x \neq 0 \text{ and } \tau_{x_d} > \tau_{f_{ref}}, \\ 0 & \text{elsewhere} \end{cases} \tag{2}$$

$x = \{f, v\}$ stands for either the frequency or the voltage, and τ_{x_d} is the value of the current measured delay.

Evaluation of the power and temperature residuals Fig. 4a and Fig. 5a clearly display a high-frequency noise in r_P and r_T. The average of this noise is close to zero, which proves that it is mainly due to estimation errors, considering that drifts of characteristics manifest mainly in the average value of the signal. Henceforth, rather than raw residuals, the moving average of the residuals is used to generate alarms. The moving average is the mean of the signal value in a window of n samples. Furthermore, the normal distribution law states that 99% of the signals population will be bounded in an envelope between two thresholds values that are the positive and negative values of the mean (μ) plus threefolds the standard deviation (σ). Hence, power residuals become (Temperature averaged residuals and thresholds are obtained using the same formula):

$$\begin{cases} r_{m_P} & = \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} r_P \quad , \\ th^+_{r_P} & = \quad \|\mu_{r_{m_P}} + 3 \times \sigma_{r_{m_P}}\|, \\ th^-_{r_P} & = -\|\mu_{r_{m_P}} + 3 \times \sigma_{r_{m_P}}\| \end{cases} \tag{3}$$

The averaged residuals are then normalized into R_P and R_T as follows:

$$\begin{cases} R_x = 1 & \text{if} \quad \|r_{x_m}\| > \|th_{r_x}\|, \\ R_x = 0 & \text{elsewhere} \end{cases} \tag{4}$$

$X = \{P, T\}$ stands for the power or the temperature.

3.3 Isolation of faults

In this work, fault isolation is a direct consequence of the nature of the model. Indeed, since each subsystem is only implicated with one estimation, faults will be first reported by the faulty component's subsystem in its alarms.

After the algorithm detects a fault, it can be either isolated or left to propagate and analyze its effect. For the purpose of this work, faults are isolated by the replacing outputs from the faulty subsystem's model by direct readings allowing for the rest of the subsystems in the model to continue generating the same outputs as measured.

4. EXPERIMENTAL RESULTS AND VALIDATION

4.1 Test of the monitoring algorithm

For experimental validation, we used two test boards; a commercial ARM-based system and a test and development board. The commercial ARM-based system is equipped with a SoC that has a quad-core ARM processor with variable frequencies ranging between 0.3–2.45 GHz, a GPU with frequencies ranging between 200 –578 MHz. The SoC is covered by the system's 2 GB low power DDR3 RAM. The test and development board has a one core ARM-based CPU and 1 GB of RAM.

Fig. 2 and Fig. 3 display the raw r and normalized R residuals of the frequency and voltage, respectively.The normalized residuals for both variables show how the

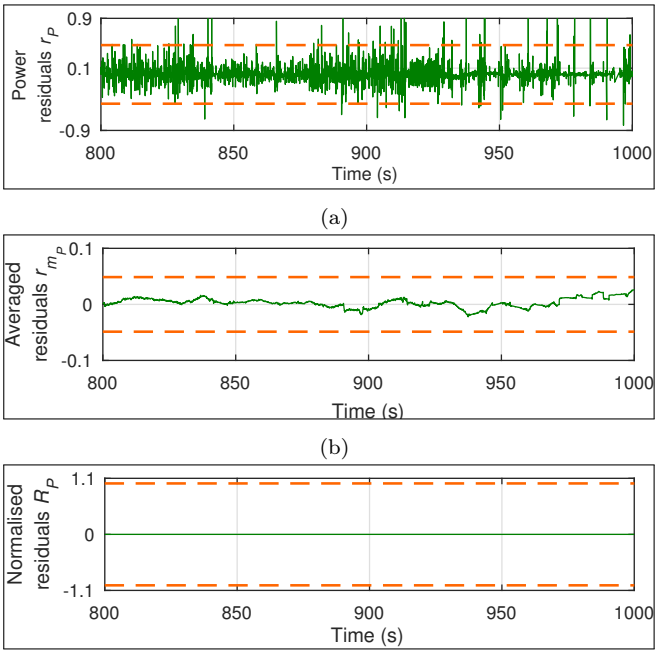

Fig. 4. Power residuals in normal operation. (a): Raw residuals. (b): Averaged Residuals. (c): Normalized residuals.

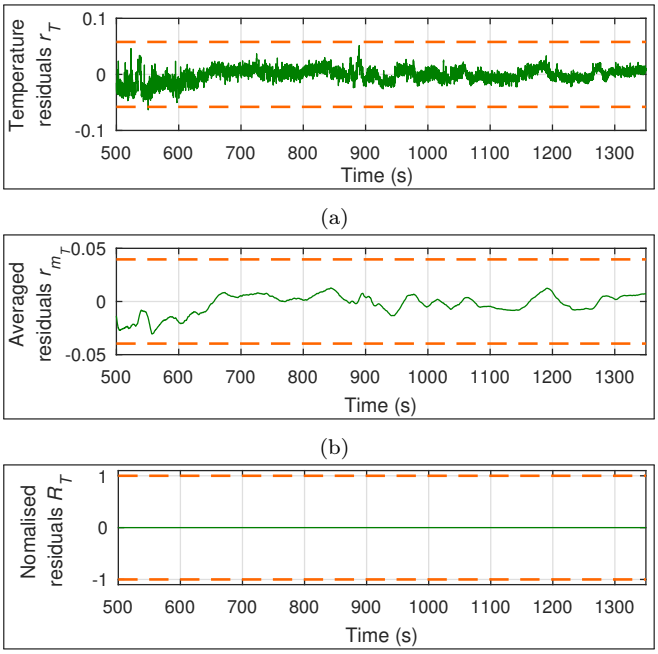

Fig. 5. Temperature residuals in normal operation. (a): Raw residuals. (b): Averaged Residuals. (c): normalized residuals.

addition of the delay thresholds eliminated all spikes, and no false alarms were recorded.

As it was the case with the frequency and voltage residuals, Fig. 4 and Fig. 5 display the raw residuals r_P, the added averaged residuals r_{m_P} and normalized residuals R_P in normal operation, and demonstrate the effectiveness of the processing method for both the power and temperature residuals, considering how it prevented the rise of false alarms by 0.9% of the residuals that were out of the threshold envelope.

4.2 Faulty scenarios

In order to validate the monitoring algorithm, we tested it against two faulty scenarios. The experimental results presented in this section were obtained with data recorded during those scenarios, which are chosen to simulate faults originating from the environment and or signs of wear of the system.

Faulty component or hardware fault Power consumption in electronic boards is regulated to avoid damaging components by high voltages or overheat. Thus, a higher power consumption than normal would be a sign of wear and fatigue, an increased resistivity due to the environment (accumulation of dust, for instance), or even faulty and damaged components. The presence of a new component or peripheral (an unauthorized one, for instance) would also cause a power usage increase, as would the absence of one of the system's peripherals manifest as a decrease. Since our model is only trained to estimate the power consumed by the SoC, to have a noticeable difference in power consumption, the Screen with high brightness, the LED flashlight, and 4G communications are activated.

Fig.6 shows the estimated and measured values of power consumption, and Fig 7 demonstrates how the monitoring algorithm detected the abnormal rise in power consumption (at the level of the SoC). It also shows, in part 7a, how raw residuals only picked momentary spikes, which could have been interpreted as estimation errors. But the averaged value of residuals clearly indicates that they are well over the threshold and hence the algorithm generates an alarm.

Environmental faults Faults caused by the environment generally manifest in the form of the overheated or overcooled surrounding. Being the most common one, overheating can be caused by a multitude of reasons ranging from a faulty cooling system to electrostatic charge, and even radiation.

In this scenario, the mobile was sealed in a waterproof bag and submerged into an 80 °C hot water bath.

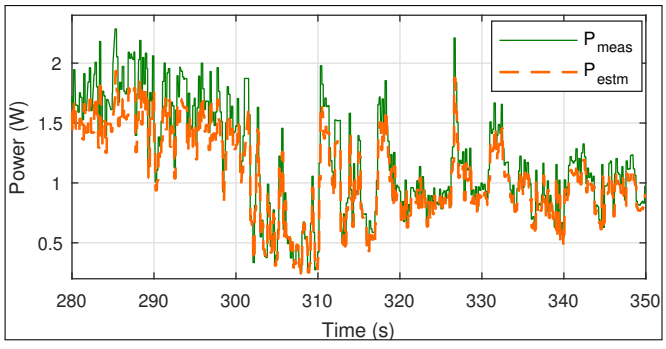

Fig. 6. Measured and estimated power values during the power faulty scenario experiment.

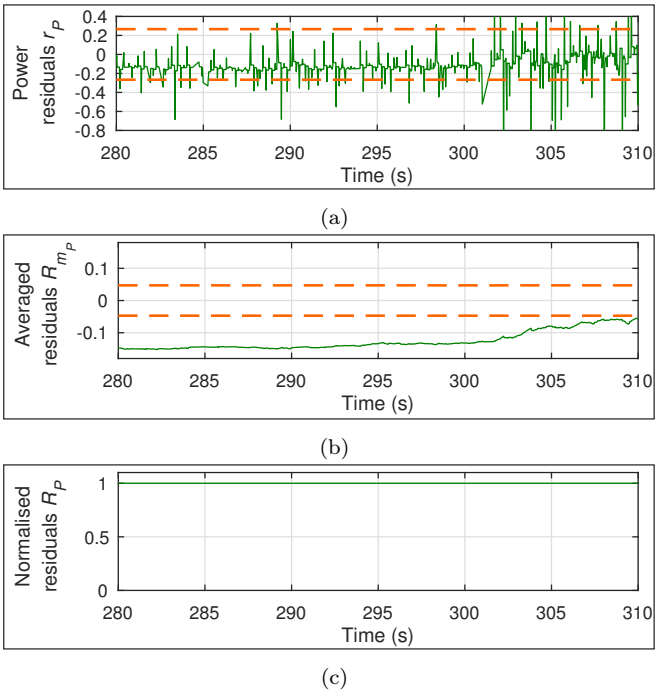

Fig. 7. Values of the residuals r_P, r_{m_P} and R_P during the faulty power scenario experiment.

In Fig. 8, measured values start diverging from estimated ones at around $t = 178\,\text{s}$, about 20 s after the submersion of the phone into the water. Fig. 9 displays the profiles of the residuals during the overheating. The latter is detected around $t = 183\,\text{s}$ where the residuals r_{m_T} goes beyond the normal operating envelope. Then, an alarm is generated (R_T rises from 0 to 1).

5. CONCLUSION

A data-driven approach is proposed in this paper for the detection and isolation of drifts in the characteristics of embedded electronic SoC. The method is based on the creation of redundancy through qualitative models for which each characteristic was built in an incremental interconnected structure, that ensures good isolation of faults.

Drift indicators are then generated by comparing the actual output of the system with the reference output generated by the model. Further residual treatment allows

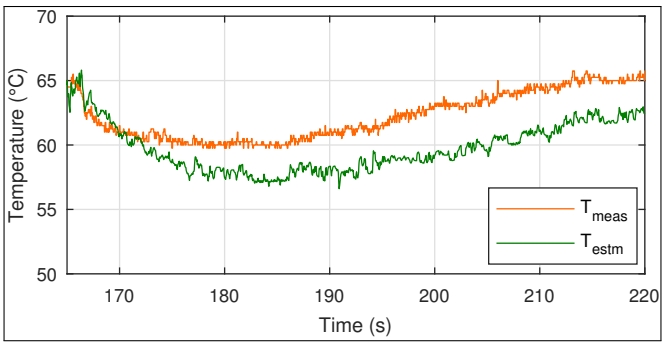

Fig. 8. Measured and estimated temperature values of the system while put in a heated environment. Divergence starts around $t = 175$s.

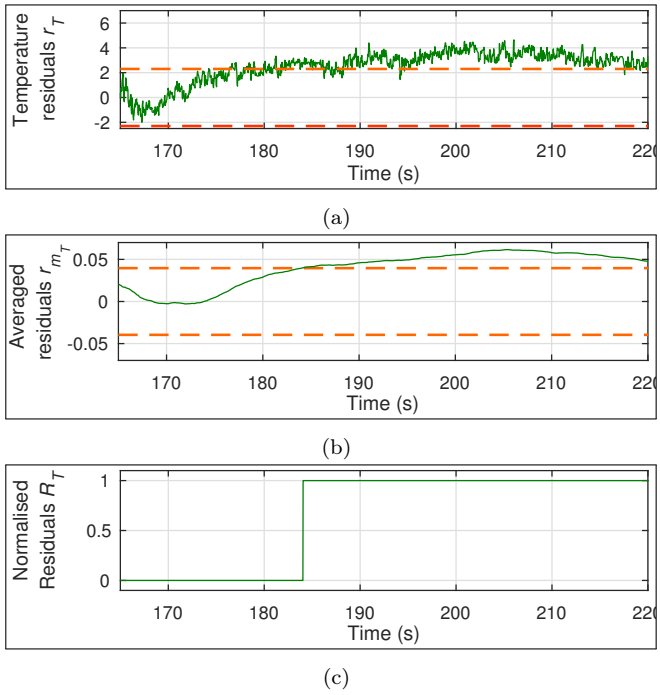

Fig. 9. Values of the residuals $_T$, r_{m_T} and R_T during in the presence of overheating. The alarm generated at $t = 185\,\text{s}$.

for the algorithm to generate normal operation thresholds. These thresholds, if surpassed, would indicate the presence of a drift in the behavior of the system.

The experimental results obtained on a CPU-GPU SoC show the ease of implementation and the effectiveness of the proposed approach.

ACKNOWLEDGEMENTS

The authors would like to adresses their thanks to the "FUI 19" project partners: INRIA, IRTS, and Nolam ES for the helpful discussions.

REFERENCES

Adrot, O., Maquin, D., and Ragot, J. (1999). Fault detection with model parameter structured uncertainties. In *5th European Control Conference, ECC'99*, CDROM.

Aggarwal, N. and Ranganathan, P. (2007). Configurable isolation: building high availability systems with commodity multi-core processors. In *Acm Sigarch ...*, volume 35, 470–481. ACM Press, New York, New York, USA. doi:10.1145/1250662.1250720.

Austin, T.M. (1999). DIVA: A reliable substrate for deep submicron microarchitecture design. In *32nd Annual Int. Symp. on Microarchitecture, (MICRO-32)*, November, 196–207. IEEE Comput. Soc. doi: 10.1109/MICRO.1999.809458.

Basseville, M. (1998). On-board Component Fault Detection and Isolation Using the Statistical Local Approach. *Automatica*, 34(11), 1391–1415. doi:10.1016/S0005-1098(98)00086-7.

Benmoussa, S., Djeziri, M.A., Ould Bouamama, B., and Merzouki, R. (2010). Empirical mode decomposition applied to fluid leak detection and isolation in process

engineering. In *18th Mediterranean Conference on Control and Automation, MED'10*, 1537–1542. IEEE. doi: 10.1109/MED.2010.5547829.

Cui, Y., Shi, J., and Wang, Z. (2015). An analytical model of electronic fault diagnosis on extension of the dependency theory. *Reliability Engineering & System Safety*, 133, 192–202. doi:10.1016/j.ress.2014.09.015.

Djedidi, O., Djeziri, M.A., M'Sirdi, N.K., and Naamane, A. (2017). Modular Modelling of an Embedded Mobile CPU-GPU Chip for Feature Estimation. In *Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics*, 338–345. SCITEPRESS - Science and Technology Publications, Mardrid, Spain. doi:10.5220/0006470803380345.

Djeziri, M.A., Ould Bouamama, B., and Merzouki, R. (2009). Modelling and robust FDI of steam generator using uncertain bond graph model. *Journal of Process Control*, 19(1), 149–162. doi: 10.1016/j.jprocont.2007.12.009.

Djeziri, M., Benmoussa, S., and Sanchez, R. (2017). Hybrid method for remaining useful life prediction in wind turbine systems. *Renewable Energy*, 116, 173–187. doi: 10.1016/j.renene.2017.05.020.

Gao, Z., Cecati, C., and Ding, S.X. (2015a). A Survey of Fault Diagnosis and Fault-Tolerant Techniques Part I: Fault Diagnosis. *IEEE Transactions On Industrial Electronics*, 62(6), 3768 – 3774. doi:10.1109/TIE.2015.2417501. URL `http://ieeexplore.ieee.org/document/7069265/`.

Gao, Z., Cecati, C., and Ding, S.X. (2015b). A survey of fault diagnosis and fault-tolerant techniques-part II: Fault diagnosis with knowledge-based and hybrid/active approaches. *IEEE Transactions on Industrial Electronics*, 62(6), 3768–3774. doi:10.1109/TIE.2015.2419013. URL `http://ieeexplore.ieee.org/document/7076586/`.

Ge, C., Yang, H., Ye, H., and Wang, G. (2009). A Fast Leak Locating Method Based on Wavelet Transform. *Tsinghua Science and Technology*, 14(5), 551–555. doi: 10.1016/S1007-0214(09)70116-6.

Gizopoulos, D., Psarakis, M., Adve, S.V., Ramachandran, P., Hari, S.K.S., Sorin, D., Meixner, A., Biswas, A., and Vera, X. (2011). Architectures for online error detection and recovery in multicore processors. *2011 Design, Automation & Test in Europe*, (c), 1–6. doi: 10.1109/DATE.2011.5763096.

LaFrieda, C., Ipek, E., Martínez, J.F., and Manohar, R. (2007). Utilizing dynamically coupled cores to form a resilient chip multiprocessor. In *Proceedings of the International Conference on Dependable Systems and Networks*, 317–326. IEEE. doi:10.1109/DSN.2007.100.

Li, M.L., Ramachandran, P., Sahoo, S.K., Adve, S.V., Adve, V.S., and Zhou, Y. (2008). Understanding the Propagation of Hard Errors to Software and Implications for Resilient System Design. *SIGOPS Oper. Syst. Rev.*, 42(2), 265–276. doi:10.1145/1353535.1346315.

Meixner, A., Bauer, M.E., and Sorin, D. (2007). Argus: Low-Cost, Comprehensive Error Detection in Simple Cores. In *40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007)*, 210–222. IEEE. doi:10.1109/MICRO.2007.18.

Mukherjee, S.S., Kontz, M., and Reinhardt, S.K. (2002). Detailed design and evaluation of redundant multithreading alternatives. In *29th Annual International Symposium on Computer Architecture, 2002. Proceedings*, 99–110. IEEE. doi:10.1109/ISCA.2002.1003566.

Shyam, S., Constantinides, K., Phadke, S., Bertacco, V., and Austin, T. (2006). Ultra low-cost defect protection for microprocessor pipelines. In *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems - ASPLOS-XII*, volume 41, 73. ACM Press, New York, New York, USA. doi:10.1145/1168857.1168868.

Steininger, A. (2000). Testing and built-in self-test – A survey. *Journal of Systems Architecture*, 46(9), 721–747. doi:10.1016/S1383-7621(99)00041-7.

Wang, F.W., Shi, J.Y., and Wang, L. (2011). Method of diagnostic tree design for system-level faults based on dependency matrix and fault tree. In *2011 IEEE 18th International Conference on Industrial Engineering and Engineering Management, IE and EM 2011*, PART 2, 1113–1117. IEEE. doi:10.1109/IEEM.2011.6035351.

Wang, N.J. and Patel, S.J. (2006). ReStore: Symptom-based soft error detection in microprocessors. *IEEE Transactions on Dependable and Secure Computing*, 3(3), 188–201. doi:10.1109/TDSC.2006.40.

Zhang, G. (2005). *Optimum Sensor Localization/Selection In A Diagnostic/Prognostic Architecture*. Phd dissertation, University System of Georgia.