HAL
open science

Adaptive Estimation of the Thermal Behavior of CPU-GPU SoCs for Prediction and Diagnosis

Oussama Djedidi, Nacer K M'Sirdi, Aziz Naamane

To cite this version:

Oussama Djedidi, Nacer K M'Sirdi, Aziz Naamane. Adaptive Estimation of the Thermal Behavior of CPU-GPU SoCs for Prediction and Diagnosis. IMAACA 2019 – Proceedings of the International Conference on Integrated Modeling and Analysis in Applied Control and Automation, 2019, Sep 2019, Lisbon, Portugal. pp.93-98. hal-02311475v1

HAL Id: hal-02311475

https://amu.hal.science/hal-02311475v1

Submitted on 11 Oct 2019 (v1), last revised 11 Oct 2019 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ADAPTIVE ESTIMATION OF THE THERMAL BEHAVIOR OF CPU-GPU SOCS FOR PREDICTION AND DIAGNOSIS

Oussama Djedidi[(a)], Nacer K. M'Sirdi[(b)], Aziz Naamane[(c)]

[(a),(b),(c)] Aix Marseille University, Université de Toulon, CNRS, LIS UMR 7020, SASV, Marseille, France

[(a)]oussama.djedidi@lis-lab.fr, [(b)]nacer.msirdi@lis-lab.fr, [(c)]aziz.naamane@lis-lab.fr

ABSTRACT

This paper proposes a dynamic behavioral model for temperature variations of systems on chips (SoC) in embedded systems. We use identification techniques (ARMAX modeling) to construct a data-driven online temperature model that estimates the temperature according to the CPU and GPU frequencies, the used RAM and the power consumed by the chip. Furthermore, we used two the Recursive Least Squares (RLS) to estimate the parameters of the ARMAX model. This method allows us to update the parameters of the model online in case of a change in the system or its characteristics. Finally, we validate the temperature model and compare between booth estimation methods.

Keywords: Identification, Embedded systems, Control

1. INTRODUCTION

Since their introduction in 1971, microprocessors have evolved from simple calculators to the center of all technological innovations (Faggin, Hoff, Mazor, and Shima, 1996). The ubiquity of microprocessor-based systems has pushed for the study of their behavior and reliability, notably when used in safety-critical and sensitive systems. Thus, the modeling and diagnosis of the microprocessor-based systems is, now, an ongoing scientific and engineering endeavor.

These systems are evermore evolving and increasing in complexity both on the microarchitectural and process levels, giving rise to new challenges with every new generation. This paper is a part of a project that explores yet another evolution enabled by these systems; the development of avionic cockpits operated by touchscreens (Figure 1). The embedded SoC used in such a critical system is required to be failproof, which require them to be studied from all physical and software aspects. In this particular work, we focus on the thermal behavior of the SoC behind the touchscreen.

Of the many aspects of modeling systems-on-chips (SoC), the temperature is one of the few that links the software, mechatronic and physical characteristics. Hence, many of the recent work studying it were studied on the thermal effects on system radiality (Löfwenmark and Nadjm-Tehrani, 2018), its management for a better reliability (Niu and Zhu, 2017, Zhou et al., 2018) or better scheduling and power management (Li, Yu, and Song, 2018). Our work, however, is oriented towards the real-time surveillance of the chip (Djedidi, Djeziri, and M'Sirdi, 2018). It concentrates on the monitoring of the chip to detect the presence of any anomalies of abnormal behavior.

Figure 1: A prototype of the cockpit of the future by Thales Avionics (Thales, 2017).

In the next section, we further detail the goal of our and put into the context of our previous works. In section 3, we discuss the thermal modeling of CPU-GPU, then we present and apply identification-based modeling to model the thermal devious of the SoC in section 4. Finally, the results and concluding remarks are presented in sections 5 and 6.

2. THE STUDIED SYSTEM

The objective of this work is the mechatronic modeling of the CPU-GPU SoC in embedded systems to predict their behavior. This behavior prediction can then be used for monitoring and diagnosis. This work is also a continuation of the works by Djedidi et al. (2017) and Djedidi, Djeziri, and M'Sirdi (2018), where the authors worked on the modeling and monitoring of systems designed for safety-critical environments. In the first study, Djedidi et al. (2017) developed an incremental interconnected modeling approach, to estimate key variables that determine the operating state of the system (Frequencies and voltages of the CPU, and GPU, Memory Occupation Rate (MOR), Chip Temperature and power consumption). Figure 2 shows a generalized diagram of the established model for mobile CPU-GPU SoC with n CPU cores.

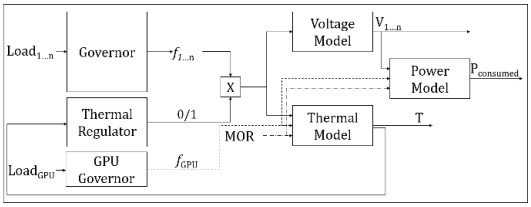

Figure 2: Synoptic diagram of the incremental interconnected model for a CPU-GPU SoC (Djedidi et al., 2017).

The developed model was then used to monitor the state the SoC and for the online detection of several types of faults such as software bugs and environmental faults (Djedidi et al., 2018).

This work focuses on temperature modeling and estimation. It aims to build a model that can be used to predict the temperature values of the SoC according to the current workload. The model is also to be integrated in the interconnected modelling framework as the thermal model (Figure 2). Finally, it is also intended to be used for the diagnosis of the chip in the future.

The case study we used to validate, and test model is a safety-critical certified development board (Figure 3). The board runs on Linux and is Android capable. It has a one core ARM Cortex-A9 processor and is equipped with 1 Gb of RAM (Freescale Semiconductor Inc, 2012b).

Figure 3: View of the test installation with the development board in the middle connected to the monitoring PC.

3. THERMAL MODELING

To model the thermal behavior of an embedded system, the first step is to follow the heat flow.

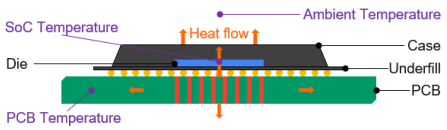

Figure 4: Simplified cross section of a typical SoC with a die containing the CPU, GPU and RAM, installed on a PCB.

Figure 4 shows, how in the SoC, heat is generated by the circuitry containing the processor cores and the RAM. It is then transferred through conduction in two directions to the silicon case (top), the underfill and the C4 bumps (bottom). The latter two would then conduct the heat to the substrate which itself would conduct it to the printed circuit board (PCB). Finally, the heat is dissipated by the case and the PCB to the air through convection and radiation. In these circuits, heat transfer occurs mostly from one layer to another (vertically, in the diagram).

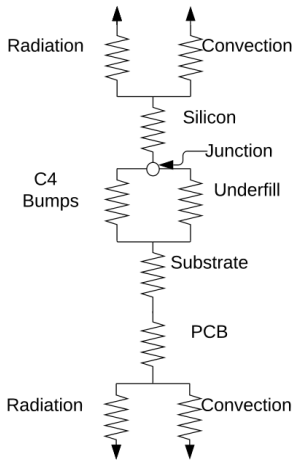

Figure 5: Equivalent thermal resistance circuit of a typical integrated circuit of an SoC.

Figure 5 shows the equivalent thermal resistance circuit (Freescale Semiconductor Inc, 2012a, Wang, Sun, and Pan, 2017). The thermal resistance circuit can be used to build a model that describes the evolution of the temperature from one layer to another. The heat (Q) can be written as:

$$Q = Q_{J/Si} + Q_{J/U} + Q_{J/C4} + Q_{Si/Air} + Q_{C4/Su} + Q_{U/Su} + Q_{Su/PCB} + Q_{PCB/Air} \quad (1)$$

Where Q is equal, in each layer, to the temperature difference divided by the thermal resistance of the said layer (Wang et al., 2017). For instance, the heat transfer between the junction and the silicon encasing is equal to:

$$\mathrm{Q_{J/Si}} = \frac{\mathrm{T_j} - \mathrm{T_{Si}}}{R_{Si}} \quad (2)$$

Where R_{Si} is the thermal resistance of the silicon. Its value can be determined either by studying the material property (area and thermal conductivity), or empirically through identification.

Models built using this method are crucial for the thermal management of the SoC. They enable engineers and system builders to correctly design the optimal cooling method (heatsink, fan, vapor chambers…). However, they also require a deep knowledge of the system and an estimation of the energy drawn by the SoC and transformed into heat which increases the complexity of the modeling process. It also does not allow for the prediction of the temperature of the SoC according to operating conditions (workload and frequency), nor the change of its value with time and degradation.

Identified models, on the other hand rely mostly on a combination of human expertise and observations to choose which inputs correlate best with the output. In this work, we use an auto-regressive model with exogenous input to predict the temperature of the SoC.

4. IDENTIFICATION-BASED MODELING

ARMAX models are polynomial models used to estimate or predict the output depending on its previous values alongside those of the input vector (Landau and Gianluca, 2006). Our choice settled on a polynomial model—precisely an ARMAX one—because they are dynamic and also fast enough to be used to generate online estimation at frequencies up to 50 Hz. Furthermore, since these models are dynamic, they can also be used to accurately simulate the behavior of the system offline and predict the operating temperature of the SoC.

A discrete ARMAX process can be described by the difference equation (Landau and Gianluca, 2006):

$$\begin{aligned} y(k) = a_1 y(k-1) + \cdots + a_{n_a} y(k_n) \\ + b_{1,1} u_1(k-\tau) + \cdots \\ + b_{m,n_b} u_m(k-\tau-n_b) \\ + e(t) + c_1 e(t-1) + \cdots \\ + c_{n_c} e(t-n_c) \end{aligned} \tag{3}$$

where $y(k)$ represents the output, $u(k)$ the input vector with m width, and $e(t)$ the error value. The parameters $[a_1, \dots, a_{n_a}]$ are the regression parameters, $[b_{1,1}, \dots, b_{m,n_b}]$ are the input parameters, and $[c_1, \dots, c_{n_c}]$ are the moving average parameters. Finally, n_a, n_b and n_c are the orders of the model, and τ is the input delay.

In our case study, the output to be estimated is the temperature T of the SoC, and the inputs are the frequencies of the cores and the memory occupation rate (MOR). These inputs are the variables that correlate the most with the temperature (Mercati, Paterna, Bartolini, Benini, and Rosing, 2017, Niu and Zhu, 2017, Zhou et al., 2018).

+

$$\varepsilon_k = y_k - \hat{y}_k \tag{4}$$

$\hat{y}(k)$ is the predicted output. We rewrite equation (3) as a discrete linear model:

$$A(q^{-1})y(k) = B(q^{-1})u(k-\tau) + C(q^{-1})e(k) \tag{5}$$

with q^{-1} being the delay operator, and $A(q^{-1}) = 1 + \sum_{i=1}^{n_a} a_i\, q^{-i}$, $B(q^{-1}) = 1 + \sum_{i=1}^{n_b} b_{j,i}\, q^{-i}$, and $C = 1 + \sum_{i=1}^{n_c} c_{j,i}\, q^{-i}$. Thus, predicted output at the sample k becomes:

$$\hat{y}(k) = \frac{B(q^{-1})}{A(q^{-1})}\, u(k-\tau) + \frac{C(q^{-1})}{A(q^{-1})} e(k) \tag{6}$$

Based upon this, we construct an adaptive predictor. This latter follows the model described in equation (6):

$$\hat{y}_k = \hat{\theta}\varphi_{k-1} \tag{7}$$

with $\hat{y}_k$ being the vector of the temperature value, $\varphi_{k-1} = [y(k-1), \dots, y(k-n_a), u_1(k-\tau), \dots, u_m(k-\tau-n_b), \varepsilon(k-1), \dots, \varepsilon(k-n_c)]$ being a vector composed of the output feedback, the inputs, and the prediction error, and $\hat{\theta} = [-a_1, \dots, c_{n_c}]$ being the vector the parameters vector.

The listing presented in Algorithm 1 is a pseudo-code describing how the parameters of the ARMAX model will be calculated with each iteration for a whole test vector $y(k)$ (Landau, M'Sirdi, and M'Saad, 1986).

Algorithm 1: Pseudo-code for the recursive Least square algorithm.

```
Begin
Define orders: n_a, n_b, n_c
Define the input delay: τ
                                        // Initialization of φ_{k-1}
Initialize φ_{y_{k-1}}
Initialize φ_{u_{k-τ}}
Initialize φ_{ε_{k-1}}
φ_{k-1} ← [φ_{y_{k-1}}, φ_{u_{k-τ}}, φ_{ε_{k-1}}]
θ̂_{k-1} ← zeros(φ_{y_{k-1}}.length, 1)
                                        // Initialization of an empty vector
F ← 100 × eye(φ_{y_{k-1}}.length)
y(k) ←Read(Output)
While y(k) ≠ null do
    ŷ(k) ← θ̂_{k-1} φ_{k-1}
    ε(k) = y(k) − ŷ(k)
                                        // Recalculation of the parameters vector
    G ← F · φ_{k-1}
    norm ← 1 + φ_{k-1} · G
    F ← F − (G·G^T)/norm
    θ̂_{k-1} ← θ̂_{k-1} + G · ε(k)
                                        // Updating φ_{k-1}
    φ_{y_{k-1}}.addFirst(−y(k))
    φ_{y_{k-1}}.poll(n_a + 1)
    φ_{u_{k-τ}}.addFirst(Read(Input(1:m)))
    φ_{u_{k-τ}}.poll(n_b + 1: n_b + 1 + m)
    φ_{ε_{k-1}}.addFirst(ε(k))
    φ_{ε_{k-1}}.poll(n_c + 1)
    φ_{k-1} ← [φ_{y_{k-1}}, φ_{u_{k-τ}}, φ_{ε_{k-1}}]
                                        \\ Reading the next output
    y(k) ←Read(Output)
End
```

5. RESULTS AND DISCUSSION

The results presented in this section are established during controlled experiments. The experiment starts when the data acquisition starts. It begins with two standards benchmarks: AnTuTu (AnTuTu, 2019) and 3DMark (Futuremark Oy, 2019), then goes on to playing and interacting with a Sudoku game, followed by 4 minutes of web browsing, HD video playback, and 4 minutes of standby time.

During this scenario, data is gathered and sent to the monitoring PC where the $\text{ARMAX}_{\text{RLS}}$ model is trained at the same time with a predefined set of orders. Once the best set of orders is found, multiple trials are again launched with a different number of training samples each time. Once the training is finished, the accuracy of the model is then validated online.

Finally, to compare the methodologies, a similar ARMAX model is trained offline with the traditional

least squares method ($ARMAX_{LS}$) using the same data as the equivalent $ARMAX_{RLS}$. The model is then validated, again, with same data used to validate the equivalent $ARMAX_{RLS}$ model.

The results presented for the accuracy of the model are the results obtained from validation trials and sets containing about 11×10^4 samples (about 3000 s).

5.1. Order selection

The first set of trials was launched with different sets of orders. The best set is chosen according to two criteria; the Mean Absolute Percentage Error (MAPE) and the average time required to generate estimations by the model.

While higher estimation accuracy is a virtue, models with higher orders may require longer times to generate estimations which can lead to missing the changes of variables values. The time required to generate estimation is also heavily affected by OS scheduling and interruptions on both the device and the monitoring PC. Thus, the optimal model needs to satisfy both accuracy and speed of estimation constraints.

Table 1: Evolution of the accuracy and the time needed to generated estimation of the $ARMAX_{RLS}$ model according to its orders.

Orders of the model			MAPE (%)	Average Sampling Time (s)
n_a	n_b	n_c		
2	2	2	11.1853	18×10^{-3}
3	3	3	7.9672	18×10^{-3}
4	4	4	1.3757	$\sim 30 \times 10^{-3}$
4	**4**	**2**	**0.8377**	$\mathbf{\sim 25 \times 10^{-3}}$
5	5	2	0.7215	$\sim 65 \times 10^{-3}$
6	6	2	1.3667	$\sim 135 \times 10^{-3}$

Table 1 displays the MAPE and the average sampling time for several sets of model orders $[n_a, n_b, n_c]$. The data in the tables show that the accuracy of the model increases with the increase of the orders up until [4,4,4], where a lower order for the moving average actually results in an increase in accuracy. Furthermore, Table 1 also show how the average sampling time increases with the order of the model, until it even starts affecting the accuracy of the model due to the longer wait time for estimations. Hence, from these experimental data, the best model orders according to both the accuracy and sampling time are [4,4,2].

5.2. Number of samples

In theory, the ideal training set would contain data representing all possible information about the system. However, in practice, the information in the training set is limited by the sample number and information contained in that sample. Table 2 shows how the accuracy of the model increases with the number of samples. However, it also shows that this increase in the accuracy is not absolute, and the accuracy might decrease even with increase number of samples. This is also shown in the comparisons shown in Figure 6 and Figure 7. Hence, the solution to obtain the best model (In this case, $n = 3000$) is to by comparison of the Mean Squared Error (MSE) as shown in Algorithm 2.

Algorithm 2: Pseudo-code the selection of the best model.

```
Begin
MSE_current = Σ_{i=1}^{n} (y_i − ŷ_i)^2 / n
                                    // n is the number of samples
If (MSE_Current < MSE_best) then
        MSE_best ← MSE_Current
        Model_best ← Model_current
End
```

Table 2: The influence of the number of samples on the accuracy of the model.

Number of samples	MAE (°C)	MAPE (%)
500	64.34	169.1200
1000	57.77	151.7027
1500	52.46	137.6220
2000	0.92	2.3688
2500	0.64	1.5077
3000	**0.34**	**0.8377**
3500	0.65	1.5100
4000	0.57	1.5005

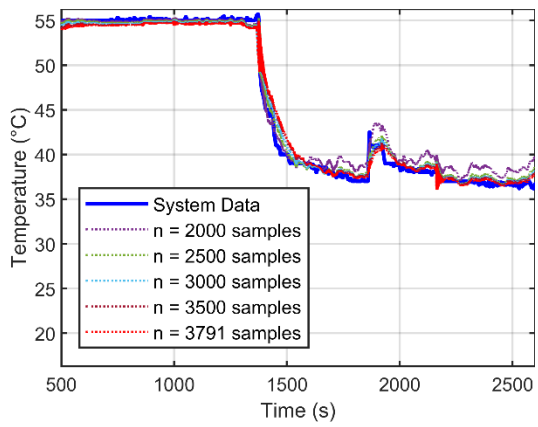

Figure 6: Estimations generated by multiple $ARMAX_{RLS}$ models with different sample numbers in their training sets against system measurements.

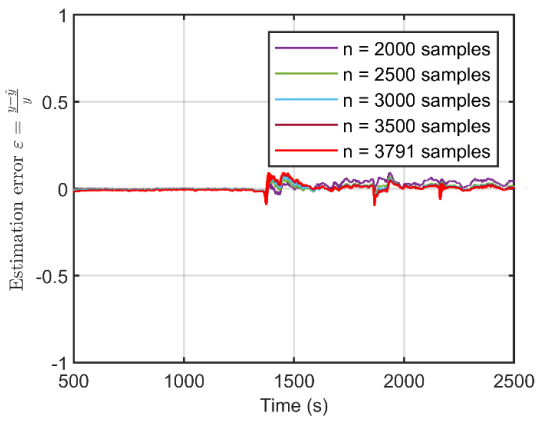

Figure 7: Estimation error (after model convergence) of the $ARMAX_{RLS}$ according to the number of samples (n) used in training.

5.3. Comparison with $ARMAX_{LS}$

Most of the ARMAX models are trained using least-squares method. Using the same data with which the $ARMAX_{RLS}$ model was trained, a new $ARMAX_{LS}$ was also trained with [4,4,2] as a set of orders. The model was then tested and validated using the same test data used to validate $ARMAX_{RLS}$ one. Table 3 show a comparison of the MAE of the $ARMAX_{RLS}$ and $ARMAX_{LS}$ models. While the $ARMAX_{LS}$ shows a slight advantage in its offline validation results, online estimation—our case use— demonstrates an advantage for the $ARMAX_{RLS}$. A further comparison of the estimations and estimation errors are shown in Figure 8 and Figure 9.

Table 3: The MAE and MAPE validation results for the $ARMAX_{RLS}$ and $ARMAX_{LS}$ models.

Model	Number of samples	MAE (°C)	MAPE (%)
$ARMAX_{RLS}$	3000	0.34	0.8377
$ARMAX_{LS}$ (Offline)	4000	0.27	0.6324
$ARMAX_{LS}$ (Online)	4000	0.56	1.3757

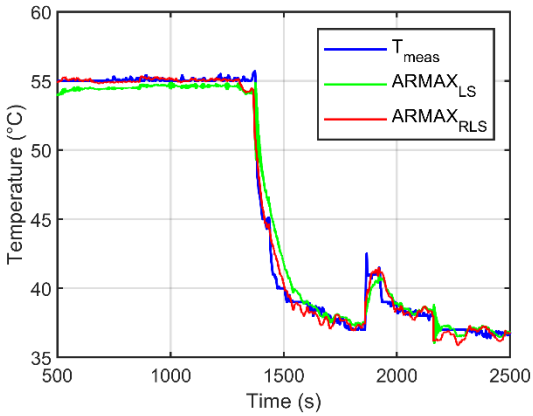

Figure 8: Estimations generated by the $ARMAX_{RLS}$ and the $ARMAX_{LS}$ models against system measurements.

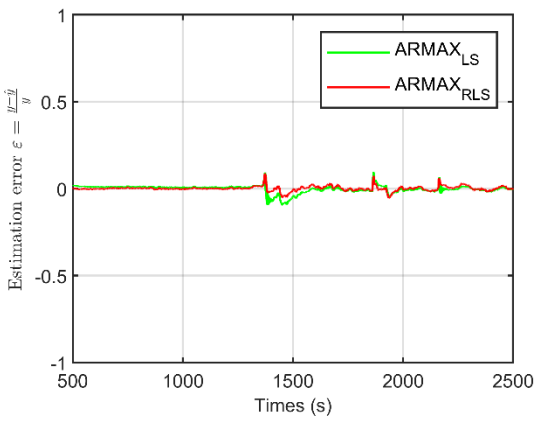

Figure 9: Estimation error (after model convergence) of the $ARMAX_{RLS}$ and the $ARMAX_{LS}$ models.

All the results mentioned above clearly validate the $ARMAX_{RLS}$. However, the high accuracy of the model (being 99.1623%), along with its speed of estimations are not the only advantages of this model. One last advantage is the capacity to retrain the model at will without stopping the monitoring process this can prove useful when a change in operating condition or a drop in the accuracy occur.

6. CONCLUSION

In this paper, we have built and validated ARMAX model to predict the temperature of embedded SoCs according to the workload and operating conditions. The model is trained using RLS method which offer two clear advantages over the traditional LS method. These advantages are a better online accuracy, and the capacity of training and retraining the model online without having to stop the monitoring process.

The model offers high accuracy with a mean absolute error of only 0.34°C, and also satisfy the required sampling time.

Having validated the model with satisfactory results, it will now integrated in the interconnected incremental framework we previously developed (Djedidi et al., 2018, 2017). In future works, we will be studying the effects of the temperature on the reliability of the system, and plan on using the $ARMAX_{RLS}$ model in the diagnosis of the state of health of the SoC.

REFERENCES

AnTuTu. , 2019. AnTuTu Benchmark - Android Apps on Google Play Retrieved from https://play.google.com/store/apps/details?id=com.antutu.ABenchMark.

Djedidi, O., Djeziri, M. A., and M'Sirdi, N. K. , 2018. Data-Driven Approach for Feature Drift Detection in Embedded Electronic Devices. IFAC-PapersOnLine, 51(24), 1024–1029.

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 (Vol. 1, pp. 338–345). Mardrid, Spain: SciTePress.

Faggin, F., Hoff, M. E., Mazor, S., and Shima, M. , 1996. History of the 4004. IEEE Micro, 16(6), 10–20.

Freescale Semiconductor Inc. , 2012a. i.MX 6 Series Thermal Management Guidelines.

Freescale Semiconductor Inc. , 2012b. i.MX 6SoloX Automotive and Infotainment Applications Processors - Data Sheet. Freescale Semiconductor.

Benchmark - Android Apps on Google Play Retrieved from https://play.google.com/store/apps/details?id=com.futuremark.dmandroid.application.

Landau, I. D., and Gianluca, Z. (Eds.). , 2006. System Identification: The Bases BT - Digital Control Systems: Design, Identification and Implementation (pp. 201–245). London: Springer London.

Landau, I. D., M'Sirdi, N., and M'Saad, M. , 1986. Techniques de modélisation récursive pour

l'analyse spectrale paramétrique adaptative. Revue de Traitement Du Signal, 3, 183–204.

Li, T., Yu, G., and Song, J. , 2018. Minimizing energy by thermal-aware task assignment and speed scaling in heterogeneous MPSoC systems. Journal of Systems Architecture, 89, 118–130.

Löfwenmark, A., and Nadjm-Tehrani, S. , 2018. Fault and timing analysis in critical multi-core systems: A survey with an avionics perspective. Journal of Systems Architecture, 87, 1–11.

Mercati, P., Paterna, F., Bartolini, A., Benini, L., and Rosing, T. Š. , 2017. WARM: Workload-Aware Reliability Management in Linux/Android. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 36(9), 1557–1570.

Niu, L., and Zhu, D. , 2017. Reliability-aware scheduling for reducing system-wide energy consumption for weakly hard real-time systems. Journal of Systems Architecture, 78, 30–54.

Thales. , 2017. What is new on Avionics 2020? | Thales Aerospace BlogThales Aerospace Blog Retrieved from http://onboard.thalesgroup.com/new-avionics-2020/.

Wang, K. J., Sun, H. C., and Pan, Z. L. , 2017. An analytical thermal model for Three-Dimensional integrated Circuits with integrated micro-channel cooling. Thermal Science, 21(4), 1601–1606.

Zhou, J., Yan, J., Cao, K., Tan, Y., Wei, T., Chen, M., … Hu, S. , 2018. Thermal-aware correlated two-level scheduling of real-time tasks with reduced processor energy on heterogeneous MPSoCs. Journal of Systems Architecture, 82, 1–11.