HARPA: Tackling Physically Induced Performance Variability

Nikolaos Zompakis*, Michail Noltsis*, Lorena Ndreu†, Zacharias Hadjilambrou†, Panagiotis Englezakis †
Panagiota Nikolaou†, Antoni Portero‡, Simone Libutti§, Giuseppe Massari§, Federico Sassi‖
Alessandro Bacchini‖, Chrysostomos Nicopoulos†, Yiannakis Sazeides†, Radim Vavrik‡, Martin Golasowski‡
Jiri Sevcik‡, Vit Vondrak‡, Francky Catthoor¶, William Fornaciari§ and Dimitrios Soudris*
*MicroLab-ECE-NTUA, Greece
†University of Cyprus, Cyprus
‡IT4Innovations, National Supercomputing Center, the Czech Republic
§ DEIB - Politecnico di Milano, Italy
¶IMEC, Belgium
‖Camlin Italy s.r.l., Italy

***Abstract*—Continuously increasing application demands on both High Performance Computing (HPC) and Embedded Systems (ES) are driving the IC manufacturing industry on an everlasting scaling of devices in silicon. Nevertheless, integration and miniaturization of transistors comes with an important and non-negligible trade-off: time-zero and time-dependent performance variability. Increasing guard-bands to battle variability is not scalable, since worst-case design margins are prohibitive for downscaled technology nodes. This paper discusses the FP7-612069-HARPA project of the European Commission which aims to enable next-generation embedded and high-performance heterogeneous many-cores to cost-effectively confront variations by providing Dependable-Performance: correct functionality and timing guarantees throughout the expected lifetime of a platform under thermal, power, and energy constraints. The HARPA novelty is in seeking synergies in techniques that have been considered virtually exclusively in the ES or HPC domains (worst-case guaranteed partly proactive techniques in embedded, and dynamic best-effort reactive techniques in high-performance).**

I. Introduction

The evolution towards processors with many heterogeneous cores is impeded by variability related challenges. This is primarily due to the increasing susceptibility to static and dynamic variations in the performance of the silicon devices and wires. A variety of well-known approaches, such as tuning manufacturing process, column/row sparing [1], frequency binning and body-biasing [2], dynamic sparing [3], variation tolerant circuits [4], and lifetime guard-bands [5], have been adopted to provide chips with a number of mitigation techniques for such performance fluctuations. The aforementioned solutions provide a basic but key expectation for a reliable processor that throughout its lifetime performs correctly under any nominal operating condition.

Designers introduce circuits to detect and when possible recover from errors, such as error-correction codes for memory and residue codes for functional units, and also to diagnose when the hardware is deviating from performance perceptive, such as built-in-self-tests, and report misbehaving hardware through interrupts higher in the compute stack. Performance dependability is more difficult to provide, in general purpose systems, as we move to higher layers of the compute stack due to software induced variations. In contrast, for embedded systems that are mission critical, it is paramount to guarantee it, at least at the system level. Thus, any of the fabricated chips should at any moment of their lifetime meet the imposed hard real-time constraints at their observable system nodes. This does not necessitate a cycle-by-cycle invariability in most practical cases, because many internal errors will be masked. Regardless, ensuring this system level reliability is harder and harder to achieve for deeply scaled technology nodes.

Several research groups have proposed design-time techniques to ensure dependability and either prevent or mitigate reliability threats through resilient and tolerant designs, based mostly on hardware overhead. Guardbanding techniques were also developed on the hardware and middleware level. These techniques use monitors and some fixed margins acting as sentinels to control behavior and act proactively to guarantee performance within constraints [6]. Later, reactive techniques focusing on best-effort procedures were developed that react on specific situations to keep performance within desired margins [7]. Best-effort techniques, unlike worst-case techniques, fail to guarantee dependability in a strict manner but they exploit system's performance more efficiently.

As performance variability has been occupying reliability community for many years, several European Projects focusing on variability and dependability have been carried out or are currently in progress. In a previous study [8], we summarized many consortiums that aim to relevant domains. Important work is also carried out by the MoRV project [9], which focuses on developing aging models that accurately describe the reliability phenomena. The MEDIAN cost action [10] has created a European network that studies manufacturability and dependability issues in multicore architectures from technology to architecture level. Finally, one can also highlight the contribution of the PRiMe research program [11], addressing a cross-layer management of energy and reliability for many-core embedded systems. Outlining the main goals, the majority of these projects focus on the modeling and analysis of the

978-3-9815370-8-6/17/$31.00

hardware vulnerability and on deploying fault-tolerant designs and mitigation techniques that guarantee correct operation and performance dependability. In line with the aforementioned objectives, the FP7-612069 European project is involved in HARnessing Performance vAriability (HARPA), proposing mitigation techniques for performance variability solutions on hardware and middleware level in addition to the application level. We address several challenges in developing performance-dependable heterogeneous multi-core architectures, applicable to both embedded systems and HPC setups.

II. HARPA OBJECTIVE OVERVIEW

The HARPA team [8] is composed of industry and academic partners across Europe specialized in fields covering all the abstraction layers, from hardware to application level. HARPA project has developed a set of monitors/knobs in hardware and software designs that observes the performance unpredictability, triggering system reactions. A run-time engine, which is a middleware split between the Operating System (HARPA-OS) and the hardware actuators (HARPA-RT) provides run-time dependable performance guarantees. HARPA-OS applies resource allocation policies, arbitrating the OS calls in a second time granularity. HARPA-RT sits at a low level in the system stack, achieving a millisecond control on hardware resources. HARPA-OS and HARPA-RT cooperate to ensure the performance dependability goals keeping a prompt low-level control on hardware resources. Run-time reactive and proactive techniques have been deployed, ensuring that the combined monitor/scheduling/knob reaction latency never violates the application deadlines. These techniques are tested on industrial applications running on embedded platforms and a full-system evaluation framework simulating HPC setups.

III. PERFORMANCE VARIABILITY MODEL

One of the primary objectives of HARPA project is accomplished by means of an accurate yet efficient modeling of performance variability in scaled technologies. Capturing and simulating variability threats is an important step towards providing solutions in mitigating them.

A. Variability Model

Among different variability phenomena, attention is given to the BTI effects, since they are expected to be the dominant ones [14]. For this purpose, imec has designed and developed a proper estimation flow that models the dependence of variability at circuit and block-level on the operating conditions (temperature and voltage), workload and technology parameters [12]. Experimental results have shown this impact for 32nm technology node on OpenSPARC processor running different application benchmarks of interest. The flow is based on commercial ASIC tools and in-house scripts: from the transistor-level representation of the circuit, and the transistor-level (real) workload, BTI variation based on defect-centric models [15] is quantified, and its impact on the critical path delay of the reference circuit is then analyzed through standard STA run. Work is in progress in this direction to perform gate-level analysis instead of transistor-level, to prune the simulation time and to be even more compatible with standard ASIC digital design flows.

Although the impact on BTI is driven by the trends in current technology [13], the system-level perspective is a joint result of different failure mechanisms. However, coping with all these mechanisms, their mutual influence and dealing with system complexity, makes it not feasible to simulate. For this purpose, to close the gap with system-level analysis, imec has designed and deployed an experimental setup to *measure* the impacts of stress-induced variability at the system-level, from a functional perspective [14]. The proposed setup allows to directly measure from the hardware the time between successive faults while the reference platform is running the desired application. From TBF samples, it is then possible to predict the platform/application MTBF that represents the sensitivity of the platform to functional faults. By forcing harsh operating conditions (high temperature and operating voltage), the time for a fault to occur is reduced to manageable values.

B. Stress Model

After modeling aging effects on the device level and estimating their impact on the system level through the metric of MTBF, our goal is to ascend even further on the abstraction layer. Therefore, to achieve a realistic emulation of performance variability in the application level, UCY has deployed a test program (Bubbles) to stress different shared resources. Bubble stresses different levels of cache hierarchy by allocating and performing random memory accesses to an array, in such a way that data prefetching is impossible. Random memory accesses are produced by using a fast random number generator using linear shift registers (LFSR), in such a way that it prevents accessing the same address twice before accessing all the allocated array entries once, as it is referred in [15].

The main goal of Bubble is to emulate the performance degradation that an application can suffer due to time dependent variability on real HW. This is done by collocating the application of interest with Bubble that is contending on shared resources. Contention for shared resources and which level of cache hierarchy Bubble stresses, while co-running the host application, depends on the size of the array that it is accessing. Bubble is emulating the consequences of time dependent variability by stealing the available cache capacity in different granularities such as cache ways etc. It can also emulate the performance degradation due to periodic appearance of faults by periodically stressing the memory subsystem, while co-running with the application of interest. The periodic appearance of faults can be defined by a given Mean Time To Failure (MTTF) and Mean Time To Repair (MTTR) values. MTTF values can be provided from different studies in the field.

IV. HARPA ENGINE

While modeling and deeply understanding variability impact is an important part of the work, a fundamental objective of the project is to provide solutions to mitigate reliability threats and ensure dependable system performance. Towards this direction, the HARPA engine has been developed, implementing

various control frameworks accross the system stack. The goal is to exploit different manifestations of platform slack (i.e., slack in performance, power, energy, temperature, lifetime, and structures/components), in order to ascertain timing guarantees throughout the lifetime of the device.

A. HARPA OS

A component of the HARPA engine is the HARPA-OS, the system-wide resource manager developed by POLIMI [16]. This component must include control policies capable of providing a response in a time frame spanning from hundreds of milliseconds to a second.

The HARPA-OS is composed of two components: a system-wide manager that allocates resources to applications while taking into account power, temperature and performance degradation induced by faults and aging; and a Runtime Library (RTLib), which is linked with each application and manages their execution flow in a resource and performance-aware fashion. The flexible structure of the HARPA-OS plays a key role in pursuing these objectives. As shown in Figure 1, the HARPA-OS is fully customizable according to available monitors, e.g. power, temperature or faults; available knobs, e.g. cpufreq and Control Groups in the case of Linux OS, or specific run-times in the case of custom accelerators; and the system-wide optimization goals, which can be pursued using pluggable scheduling policies.

Due to the different optimization goals of ES and HPC scenarios, POLIMI has developed individual scheduling policies for each case. The ES goal is to maintain the processors' temperature and the battery level under a critical safety threshold, thus preventing a system shutdown. According to the system status, i.e. battery level and temperature, and the optimization goals, i.e. energy budget and performance requirements, the scheduling policy, identifies a suitable allocation configuration for the applications. This configuration is called Application Working Mode (AWM). Conversely, the goal in the HPC scenario is reducing power consumption, avoiding thermal hot-spots and counteracting performance degradation due to aging or faults; thanks to the interaction between the system-wide manager and the RTLib, this scheduling policy can dynamically change the resource allocation according to the runtime-variable performance requirements of the applications, while complying with the QoS goals.

B. Monitors/Knobs

Monitors supervise the system to verify if the running constraints are satisfied, while knobs are actuators that help the system recover from any violations in pertinent metrics. Monitors and knobs enable the Harpa-OS and HARPA-RT to detect system variability and to rapidly mitigate its consequences. However, some sub-systems, such as the Network-on-Chip (NoC), must rely on extremely swift mitigation actions, which are much faster than the reaction time of the Harpa-OS and HARPA-RT. Thus, in the case of the NoC, there is an imperative need for near-instantaneous action (at the granularity of only a few cycles), in order to minimize the impact on performance. Since the performance of the NoC is

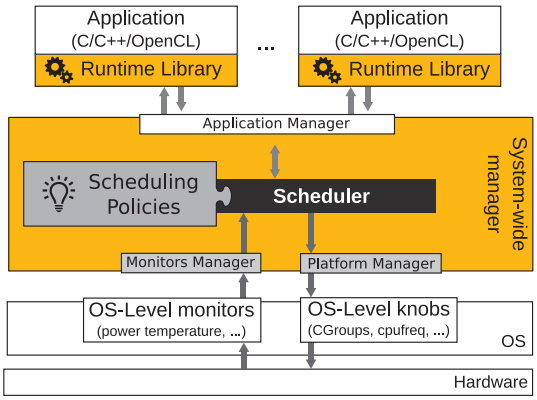

Fig. 1: HARPA-OS as a flexible resource manager: gray blocks allow it to interact with the monitors/knobs exposed by the target hardware and to pursue custom optimization goals.

critical to the overall performance of the microprocessor, NoC routers operate with extremely low latencies. Consequently, NoC monitors and knobs must typically rely on *autonomous* (stand-alone) monitor-and-knob mechanisms that can operate at such ultra-low latencies. Of course, this autonomous nature implies that these mechanisms do not need to communicate with the HARPA engine itself.

The proposed NoCAlert [17] architecture is a comprehensive online and real-time fault detection and localization mechanism that demonstrates 0% false negatives within the interconnect for the fault models and stimulus set used. NoCAlert focuses on the detection and localization of faults in any components of the NoC's control logic. Based on the concept of invariance checking, NoCAlert employs a group of lightweight microchecker modules that collectively implement real-time hardware assertions. An illegal output (invariance) is defined here as an operational decision that violates the functional correctness rule(s) of a particular component. The checkers operate concurrently with normal NoC operation, thus eliminating the need for periodic, or triggered-based, self-testing. NoCAlert can detect (a) single stuck-at (permanent) faults, and (b) single-event transient faults. Moreover, the proposed mechanism is also able to pinpoint (localize) the fault with extremely high accuracy. Most importantly, 97% of the transient and 90% of the permanent faults are detected instantaneously, within a single clock cycle upon fault manifestation. The fault localization accuracy ranges from 90% to 100%, depending on the desired localization granularity. Note that a localization accuracy of less than 100% simply means that NoCAlert provides more than one possible fault location, with the actual fault location always included in the reported candidate locations. Extensive cycle-accurate simulations in a 64-node CMP and analysis at the RTL netlist-level demonstrate the efficacy and efficiency of the proposed technique. Our hardware synthesis results indicate minimal area/power/timing overhead.

C. HARPA-RT

The HARPA-RT sits at a low level in the system stack and is in direct contact with the various monitors and knobs.

It has responsive control on hardware resources, enabling extremely fast adaptation to system behavior in the scale of some milliseconds, which is ideal for providing guarantees for hard-deadline applications and complements the comparatively slower responsiveness of the HARPA-OS.

ICCS deploys the HARPA-RT, exploiting a PID controller [18] that mitigates the performance degradation by applying respective DVFS schemes. The PID controller operates based on cycle budgets that a scenario detector provides. The key issue in this implementation is the scenario concept [19].

The scenario detector represents the interface between the HARPA-RT and the active application. Each running application situation (RTS) is fitted to a corresponding scenario cycle budget. The scenarios are defined at design time, grouping the application RTSs into clusters. Each cluster is represented by the fastest RTS that is exploited as a reference cycle budget in the PID controller to estimate the time deviation with an error-free operation. The RTS variation in the same scenario creates an impression of extra delay that leads the PID controller to overreact consuming extra resources. The scenario accuracy is defined by the number of scenarios (the more the better). However, the preciseness leads to more complex and costly scheduling. An inherited trade-off exists between an efficient resource exploitation and scheduling overhead [20].

The system scenarios implement the policy that the HARPA-OS imposes, achieving a trade-off between performance and resource utilization. For each policy a different set of scenario is activated. Each scenario-driven DVFS reaction equalizes the lost time due to the extra cycles of the error-correction interventions. Two factors define the efficiency of these actuations 1) the first is the aggressiveness regarding the operation frequency boosting that can be defined by modifying the scenario budgets and 2) the second is the reactive or proactive response that is correlated with the timing of the reaction. ICCS exploits these two features to enhances the system scenario approach with adaptation capabilities.

A scenario set can be modified at run-time optimizing the applied policy without interrupting the application execution by regrouping the RTSs or by modifying the scenario cycle budgets. The realization of a scenario adjustment considers the frequency norm of the error-correction interventions during the operation. This provides partially proactive reactions that ensure performance dependability. The approach achieves remarkable results absorbing delays that the stable scenarios is unable to equalize, having an extra energy impact. Indicative results will be presented in the following Application section. ICCS is working on a fully proactive approach that apart from performance variability, will eliminate the transition effects due to DVFS switching. This setup will realize the Dynamic Scenario concept. The goal is to keep the majority of the rescheduling decisions in a HARPA-RT level avoiding the time expensive calls of the OS.

V. Application Results- Case Studies

As previously stated, the HARPA engine provides solutions for dependable performance according to different levels of responsiveness. These mechanisms together, can handle the entire timing range required for practical applications. This section examines the effectiveness of the HARPA engine, tested under realistic applications drawn from the industry partnerns of the consortium.

A. Disaster and Flood Management Simulation

The first application under study is part of the FLOREON+ framework, developed by IT4I [21]. The application is used to simplify the process of disaster management and increase its operability and effectiveness. POLIMI-IT4I have developed a case study [22] to evaluate the resource allocation capabilities of the HARPA-OS on an HPC machine equipped with an x86-64 multi-core machine. The case study simulates the uncertainty (*Uncer*) of the rainfall-runoff (*RR*) model. The Uncertainty model computes 12K Monte Carlo samples with a ten minutes as time deadline (our defined QoS) that run concurrently with/and without a What-if-Analysis (*WIA*) model. The *WIA* starts in a δ_0 time and runs for a δ time. When both models run in the system, they have to compete for resources. Figure 2 presents the goal gap defined as the difference between the MC samples that are running in the system vs. the ideal MC samples to finish in the exact requested time (i.e. the perfect goal gap is zero).

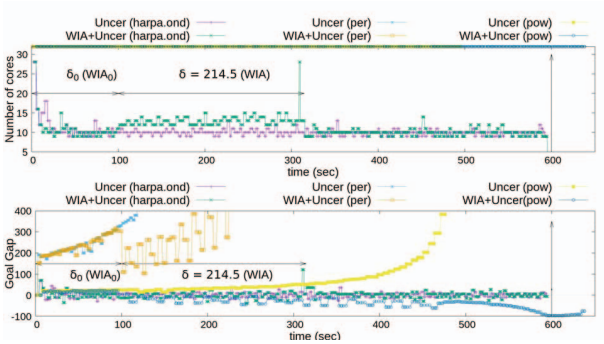

Fig. 2: Up: Number of resources allocated for performance, powersave and harpa governors with and without WIA. Down: Goal gap during execution time with and without WIA for the three governors (per, pow, harpa) [22].

A comparison between the HARPA-OS and two GNU/Linux resource allocators governors (*performance* and *powersave*) that provide a best effort solution for performance and power follows. Figure 2 presents the number of cores allocated by the governors. In the case of GNU/Linux governors, this value is always 32, the maximum number of cores that the system has. In the case of HARPA-OS governor, this value changes dynamically trying the goal gap to be zero. When the models run with *performance* governor, the model finalizes faster than the deadline requested. Accordingly, the *powersave* governor covers the goal gap in the first case, when the *WIA* model does not run, but it becomes negative (violating the performance requirement) when *WIA* model is producing HW constraints to the system. In the case of the HARPA-OS, the runtime always provides

the minimized resources to execute the models in the given time frame (before the deadline). The study [22] shows that the HARPA-OS mechanism has advantages regarding power consumption and a decrease in the temperature of the cores. Since total runtime is limited due to the QoS constraint, and the goal is to minimize power and heat.

For the same application in another case, Bubble has been used to mimic time dependent performance variability and evaluate a compensation technique that is based on monitoring the applications IPC (instructions per cycle) in an Intel Xeon node. A previous profiling of the application has been performed to keep track on the targeted IPC, in a fault-free scenario. In this use case, DVFS has been used as the performance variability mitigation technique to compensate any unexpected loss in Floreon+ IPC from the targeted one which is assumed to be 1.5. Figure 3 shows a case study where Bubble causes intermittent performance degradation to a Floreon+ thread. When Bubble contents on L1 data cache with Floreon+, IPC drops are observed and DVFS is used to increase the frequency. DVFS is able to compensate the application drop in IPC. When the degradation stops (Bubble stops stressing the shared resources), the IPC goes back to normal and the frequency drops down to the nominal value which is assumed to be 2.00GHz. This is an example of how Bubble can be used to emulate performance degradation and to evaluate compensation techniques such as DVFS.

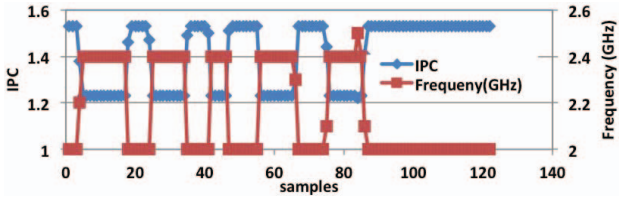

Fig. 3: Frequency and IPC fluctuation over performance-dependable DVFS schemes.

B. FREQUENCY SPECTRUM SENSING APPLICATION

A spectrum sensing application developed by THALES is our second case-study. Its purpose is to explore the frequency spectrum and gather information on free or unused frequency bands, in order for instance to perform the radio-frequency allocation. Specifically, the application gathers samples over a wide frequency range, performs filtering and splits the signal across different frequency bands for further analysis. Then, for each band, provision of statistical information is given, along with a view of signal carriers and spectral footprint. In general, the sensing function faces a number of challenges in terms of miniaturization, power consumption, and timing response and such demands are even more severe for mobile terminals.

ICCS deploys a simulator [18] that evaluates the HARPA-RT efficiency. The intervention of the error-correction mechanisms is modeled as injected cycle noise that the HARPA-RT is called to mitigate and ensure performance dependability. For our case study, cycle noise is modeled as a bivariate Gaussian distribution with mean Λ and M values (λ and μ) that represent respectively the rate and amplitude of the error-correction events in terms of clock cycles. The timing and power models of the simulator create traces from which delay and total energy can be deduced. The simulator instantiates two HARPA-RT implementations, one that exploits a static system scenario set and a second one that realizes the aforementioned adaptive approach (referred previously in HARPA-RT subsection).

Figure 4 presents the pareto optimal trade-offs between static and adaptive scenarios for a noise instantiation in respect with the overdue and the energy. Each point in the same curve, represents a scenario set with an individual number of scenarios that is evaluated with and without adaptivity. The two curves (adaptive vs static system scenarios) are close, presenting similar cost trade-offs for a specific overdue around 4%. The adaptive scenarios appear just a little worse trend (longer distance from the start of the metric axis) due to the extra implementation cost that they require. The adaptive scenario approach is of high value when strict dependability constraints are demanded. However, when timing constraints relax the static scenarios are more efficient.

Comparing a guardband approach that deals with the worst noise case (high μ and low λ) to a scenario-based DVFS approach that ensures performance dependability, Figure 5 highlights each percentage deviation in time for the aforementioned sensing application. For less aggressive noise (low μ and high λ), the scenarios present a delay up to 35% in respect with the guardband. This represents a quantification of the margins that the scenarios exploit to improve the resource utilization, without violating the deadlines. These specific margins provide energy savings up to 15%. The time deviation and the respecting gains decrease proportionally to the injected noise and become zero in the worst noise case when the most elevated frequency is activated.

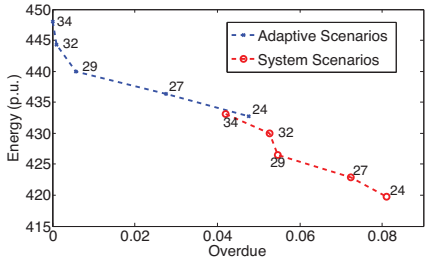

Fig. 4: Static - Adaptive scenarios Overdue Vs Energy.

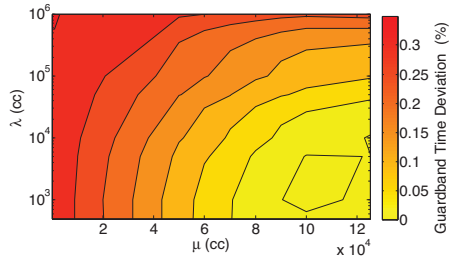

Fig. 5: Scenario Vs Guardband time deviation.

C. LANDSLIDE MONITORING

Camlin provides in the HARPA project a landslide and rock-wall monitoring embedded application. The application creates a smart-bridge for the Beesper network able to process data remotely, using machine learning and artificial intelligence techniques, which gathers using a wireless sensor network and on-site cameras. Beesper is a flexible system, based on Camlin Wireless Sensor Network (WSN) that needs to operate functionally correct over long period times. This application analyses a stream of images acquired by a camera. The main knob to tune the application is the rate at which it acquires new images.

HARPA-OS reconfigures the application aiming at maximizing the uptime of the system, the different AWM will be selected depending on the available power. Figure 6 highlights the AWMs transitions in critical conditions due to cloudy or rainy days. The blue line represents the power generated from the solar panel for charging the battery, while the red line represents the battery level. The green area defines the AWM evaluation. Starting from AWM_1 a sunny day, the battery level raised and the system moved to AWM_2. In the next two more sunny days, the system switched to AWM3 for a limited amount of time. The following days are cloudy and brought to a drop in the generated power and a fall of the battery level: the selected AWM first switched to AWM_1 and then to AWM_0 to prevent the shutdown of the system. After these critical days, the sun restores the battery to a safe level and allowed the transition to the AWM_2. Thanks to the HARPA-OS management, the Video Rockfall detection and the Extensometer Analysis were able to operate, keeping the minimum level of Quality of Service required from the application during cloudy days while performing at greater quality when power was available.

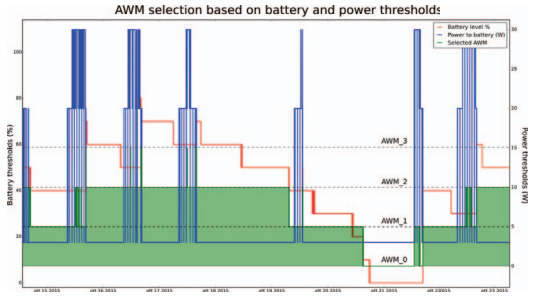

Fig. 6: Selection of the AWMs (green) based on the battery level (red) and the power generated from the solar panel (blue).

VI. CONCLUSION

This paper provides an updated summary of the FP7-612069 HARPA European project, its preliminary objectives and results. The project focuses on the HPC as well as the ES domain to present mitigation solutions to tackle performance variability and dependability threats. Industry partners have provided three applications of increased scientific importance with an incorporated need for dependable and reliable operation. Significant effort is spent on efficiently capturing variability-induced mechanisms and estimating the degradation of performance. The HARPA engine, the heart of this project, exploits a set of mechanisms from the hardware, architecture and software level to guarantee dependable and reliable performance of multicore and manycore systems.

VII. ACKNOWLEDGMENTS

Current work is supported by the FP7-612069- HARPA EU project.

REFERENCES

[1] H. Q. Le *et al.*, "Ibm power6 microarchitecture," *IBM Journal of Research and Development*, vol. 51, no. 6, pp. 639–662, 2007.
[2] S. Borkar *et al.*, "Parameter variations and impact on circuits and microarchitecture," in *Proceedings of the 40th annual Design Automation Conference*. ACM, 2003, pp. 338–342.
[3] C. McNairy and R. Bhatia, "Montecito: A dual-core, dual-thread itanium processor," *IEEE micro*, vol. 25, no. 2, pp. 10–20, 2005.
[4] J. P. Kulkarni *et al.*, "A 160 mv, fully differential, robust schmitt trigger based sub-threshold sram," in *Pr. of the 2007 international symposium on Low power electronics and design*. ACM, 2007, pp. 171–176.
[5] T. Austin, "Razor: a low-power pipeline based on circuit-level timing speculation," in *Proceedings of the 19th annual symposium on Integrated circuits and systems design*. ACM, 2006, pp. 13–13.
[6] K. Jeong *et al.*, "Impact of guardband reduction on design outcomes: A quantitative approach," *IEEE Transactions on Semiconductor Manufacturing*, vol. 22, no. 4, pp. 552–565, 2009.
[7] K. K. Pusukuri, R. Gupta, and L. N. Bhuyan, "Thread tranquilizer: Dynamically reducing performance variation," *ACM Trans. on Architecture and Code Optimization (TACO)*, vol. 8, no. 4, p. 46, 2012.
[8] D. Rodopoulos *et al.*, "Harpa: Solutions for dependable performance under physically induced performance variability," in *Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS), 2015 International Conference on*. IEEE, 2015, pp. 270–277.
[9] "Modeling of reliability under variability," https://morv-project.eu/wordpress.
[10] "Manufacturable and dependable multicore architectures at nanoscale," http://www.median-project.eu/.
[11] "Power-efficient, reliable, many-core embedded systems," http://www.prime-project.org/.
[12] D. Stamoulis *et al.*, "Capturing true workload dependency of bti-induced degradation in cpu components," in *Proceedings of the 26th Edition on Great Lakes Symposium on VLSI*, ser. GLSVLSI '16. New York, NY, USA: ACM, 2016, pp. 373–376.
[13] G. T *et al.*, "Recent advances in understanding the bias temperature instability," in *Electron Devices Meeting (IEDM), 2010 IEEE International*, Dec 2010, pp. 4.4.1–4.4.4.
[14] S. Corbetta *et al.*, "System-wide reliability analysis on real processor and application under vdd and t stress," in *Silcon Errors and Logic System Effects - SELSE 2016*, 2016.
[15] J. Mars *et al.*, "Bubble-up: Increasing utilization in modern warehouse scale computers via sensible co-locations," in *Proceedings of the 44th annual IEEE/ACM International Symposium on Microarchitecture*. ACM, 2011, pp. 248–259.
[16] P. Bellasi *et al.*, "A rtrm proposal for multi/many-core platforms and reconfigurable applications," in *Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2012 7th Int. Work. on*. IEEE, 2012.
[17] K. Chrysanthou *et al.*, "An online and real-time fault detection and localization mechanism for network-on-chip architectures," *ACM Trans. on Architecture and Code Opt.(TACO)*, vol. 13, no. 2, p. 22, 2016.
[18] D. Rodopoulos, F. Catthoor, and D. Soudris, "Tackling performance variability due to ras mechanisms with pid-controlled dvfs," *IEEE Computer Architecture Letters*, vol. 14, no. 2, pp. 156–159, 2015.
[19] S. V. Gheorghita *et al.*, "System-scenario-based design of dynamic embedded systems," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 14, no. 1, p. 3, 2009.
[20] N. Zompakis *et al.*, "Enabling efficient system configurations for dynamic wireless applications using system scenarios," *Int. journal of wireless information networks*, vol. 20, no. 2, pp. 140–156, 2013.
[21] S. Kuchar *et al.*, "Using high performance computing for online flood monitoring and prediction," *International Journal of Environmental, Ecological, Geological and Geophysical Engineering*, vol. 9, no. 5, pp. 267–272, 2015.
[22] A. Portero *et al.*, "Using an adaptive runtime system for time predictability power-aware in hpc-oriented applications," *The Third Workshop on Low-Power Dependable Computing (LPDC)*, 2016.