Empowering Autonomy through Self-awareness in MPSoCs

Nikil Dutt[1], Amir M. Rahmani[1,2], Axel Jantsch[2]
[1]Department of Computer Science, University of California, Irvine, USA
[2]Institute of Computer Technology, TU Wien, Vienna, Austria
Email: {dutt, amirr1}@uci.edu, axel.jatsch@tuwien.ac.at

***Abstract*—In order to meet the significant challenges in future MPSoCs they need be become more adaptive and autonomous. We argue that a self-aware MPSoC can efficiently manage itself even in the presence of aging, varying application demands, functional and non-functional aberrations, and thus achieve a high level of adaption and autonomy. We review CPSoC as an exemplar of a self-aware MPSoC and demonstrate the benefits of these concepts. Finally, we discuss future directions and the main challenges of the research on self-aware, autonomous and adaptive MPSoCs.**

I. Introduction

Multiprocessor System-on-Chips (MPSoCs) have been widely adopted for embedded signal processing, multimedia computing, and application-specific designs [1]. Although they have a long history of use in embedded computing, we are seeing increasing adoption of MPSoCs as viable platforms for general (e.g., server/desktop) computing with key architectural driving factors such as scalability, multitasking, programmability, real-time performance, and low-power operation.

However, to maintain their continued viability as preferred computational platforms across a wide spectrum of applications and usage scenarios, MPSoCs must address a host of diverse challenges (Figure 1): a) *Varying Application and User Demands*, both within and between applications; b) *Functional Aberrations*, resulting from design errors, malicious attacks, etc.; and c) *Non-functional Aberrations*, caused by environmental effects (e.g., radiation, temperature), etc. Indeed, MPSoCs face exponentially increasing power density and heating, creating drastic and harsh environments (e.g., hotspots), that manifest as aging as well as wear-out phenomena (e.g., NBTI, HCI, TDDB, Electromigration etc. [2], [3]) which further increase susceptibility to errors in time over its lifetime. Furthermore, aggressive technology causes substantial manufacturing-induced variability effects, resulting in significant fluctuations of critical device/circuit parameters over time, resulting in diminishing yield and reliability [4].

In this paper we posit that **autonomy** (the ability to operate independently, without external control) and **adaptivity** (the ability to effect run-time changes and handle unexpected events) are key attributes that future MPSoCs must possess. Using autonomy and adaptivity, MPSoCs will be empowered to address the myriad challenges outlined in Figure 1. At first glance, limited adaptivity can be achieved through Runtime Monitoring and Management of MPSoCs. Indeed we are seeing current trends in research [2] that attempt to use a sensor-rich MPSoC platform, coupled with cross-layer orchestration between the application, runtime, compiler, architecture and circuit layers to manage variability [2]. However there are few efforts that address autonomy through self-awareness.

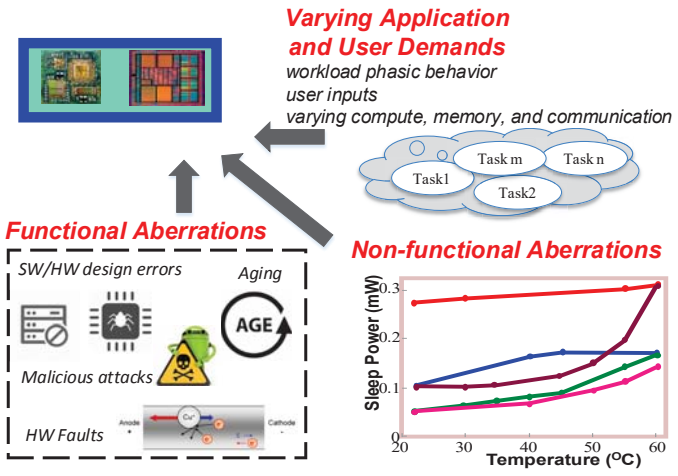

Fig. 1. Challenges faced by future MPSoCs

Existing systems are typically designed for end use-cases with particular goals in mind, such as: high-performance computing (Figure 2a), mobile computing constrained by energy/power (Figure 2b), space applications driven by resiliency (Figure 2c), and many others (e.g., Figure 2d-f). However, we see from Figure 1 that MPSoCs will need to deal with wide aberrations in functional as well as non-functional characteristics. Furthermore, existing systems are still brittle in the face of unexpected events or failures. Therefore MPSoCs will need to adapt to these dynamic changes autonomously while satisfying user goals and constraints. Taking inspiration from biology and nature, we believe that principles of self-awareness are key to empowering autonomy and adaptivity for achieving dynamic Quality-of-Service (QoS) demands (Figure 2f). In the following sections we describe self-awareness and autonomy from an MPSoC perspective, present an exemplar MPSoC (CPSoC) that achieves some of these goals, and conclude with open challenges.

II. Self-awareness and Autonomy

Self-adaptivity has been the focus of much research in software engineering for at least 20 years, with the goal of managing the complexity of large-scale software systems [5]. Manual troubleshooting, reconfiguration, and maintenance are demanding and error prone. Above a certain complexity of the system, it becomes infeasible. Self-adaptive behavior is triggered either by changes in the system's self (internal causes like faults or mode transitions), or by changes in the system's context (external events like changes in user request rates or user objectives) [5].

In MPSoCs, the main problem is the severity of resource

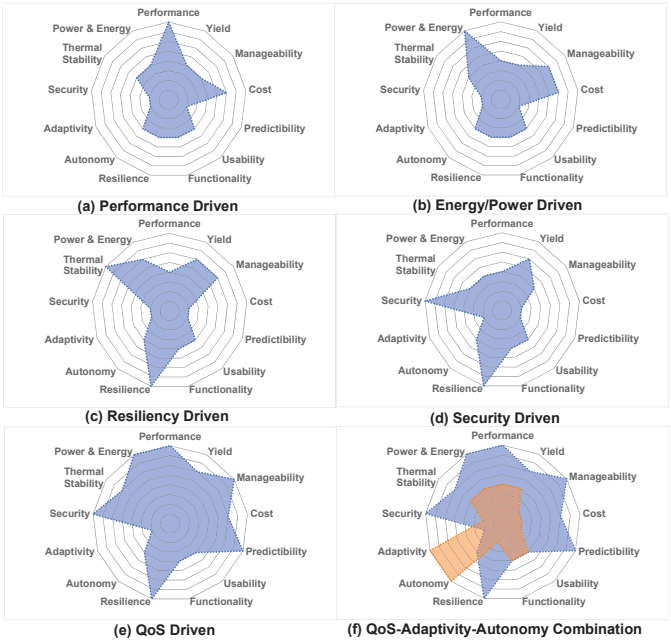

Fig. 2. MPSoCs need to adapt to dynamic changes autonomously

constraints, the large number of objectives, and their interaction, and the unpredictable environmental conditions of their deployment. Prevailing MPSoCs are moving towards unprecedented parallelism and heterogeneous many/multicore architectures with several hundreds or even thousands of cores. Traditional management, maintenance, diagnostics, and repair of most of such systems can easily lead to resource underutilization or safety issues for the chip. Thus, there is a growing need that MPSoCs have a better understanding of their own state, their behavior, their performance, and the surrounding conditions. We call this "better understanding" awareness, which improves the behavior of systems, making them more robust and reducing processing, communication and energy requirements. A variety of bio-inspired approaches have been proposed for the operation, modeling, design, optimization, and verification of embedded systems and SoCs, as a recent collection illustrates [6]. However, designing and implementing self-awareness in an ad-hoc manner for every new system is not feasible. Introducing awareness as a separate concept in the SoC infrastructure promises to simplify development and operation of such systems. We describe a first attempt at a self-aware MPSoC below.

III. CyberPhysical-System-On-Chip (CPSoC)

In [7], we presented the CyberPhysical Systems-on-Chip (CPSoC) as an exemplar self-aware MPSoC that enhances traditional MPSoCs with a sensor-actuator-rich platform deploying a closed loop paradigm emulating large-scale Cyber-Physical Systems, enhanced with smartness through adaptivity and limited self-awareness [8]. CPSoC was developed primarily in the context of managing and exploiting hardware variability using the under-designed and opportunistic (UNO) computing paradigm [2].

The high-level system architecture of CPSoC is shown in Figure 3. The middle of this figure shows various levels of abstraction for the CPSoC platform, from the lowest

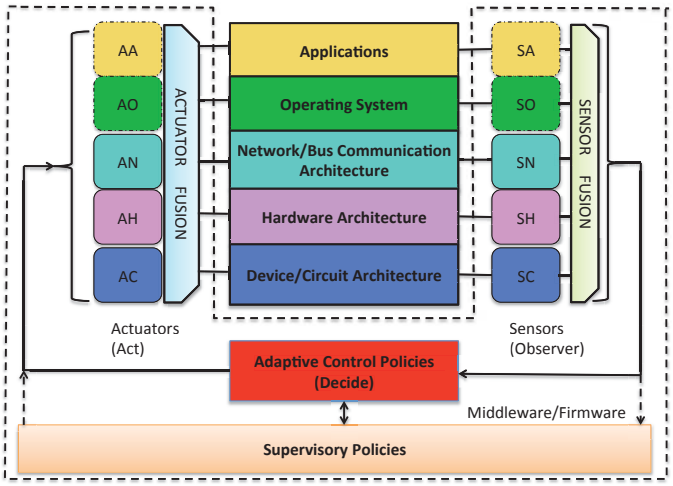

Fig. 3. Cross-layer virtual sensing and actuation at different layers of CPSoC.

(device/circuit level) layering up to the highest (application level). At each abstraction level, the CPSoC platform gathers information (through sensor fusion) using virtual and physical sensors, and in turn actuates (through actuator fusion) via virtual and physical actuators. The CPSoC architecture supports two classes of feedback loops: adaptive control (Red box in Figure 3) and self-aware supervisory control that generates supervisory policies (Tan box in Figure 3). These feedback loops are embedded within the adaptive, reflective middleware that orchestrates cross-layer sensing and actuation.

Figure 4 shows a more detailed view of the CPSoC architecture. On the top right of the figure is a template of an individual CPSoC computational Core, comprised of the computational units, memories, interfaces, and the on-chip sensing and actuations (OCSA) block that allows ubiquitous sensing and actuation at the CPSoC-Core level. These CPSoc-Cores are tiled into a (homogeneous or heterogeneous) CP-SoC computational fabric (lower right of Figure 4), using a Network-on-Chip (NoC) interconnect. Note that each router box in the NoC is also equipped with a sensing-and-actuation block (colored green) that enables monitoring and actuation at each NoC router. The left side of Figure 4 expands the abstraction layers of Figure 3, showing the CPSoC tiled hardware fabric at the lowest layer, and the applications executing on this platform at the highest layer. The adaptive, reflective middleware layer (yellow box on the left side of Figure 4) orchestrates the distributed sensing and actuation approach, where each component and core can make local decisions to manage the fabric.

While it may appear that the sensing and actuation mechanisms in the CPSoC fabric might share the traditional best effort core-to-core communication NoC, sensing and actuation needs vary in their bandwidth (low) and latency needs (periodic but real-time). Therefore CPSoC deploys a dedicated custom on-chip sensor network (sNoc) to efficiently enable distributed on-chip sensing and actuation, as shown in Figure 5. CPSoC's introspective sentient units (ISU) collect, monitor, and process the sensing data to enable assessment of the system's present states and context as well as future states. Each ISU is a processor-based system interfaced to the sNoC where the virtual sensing/actuation is performed. The information from the ISUs are distributed to the computational cores.

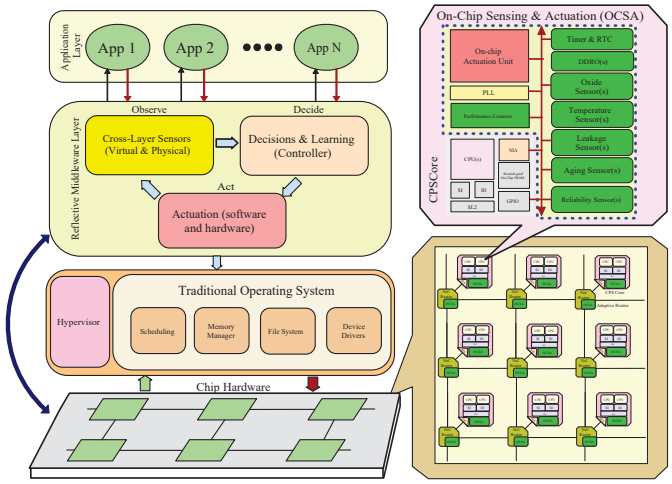

Fig. 4. CPSoC architecture with adaptive Core, NoC, and the ODA Loop as Middleware [9]

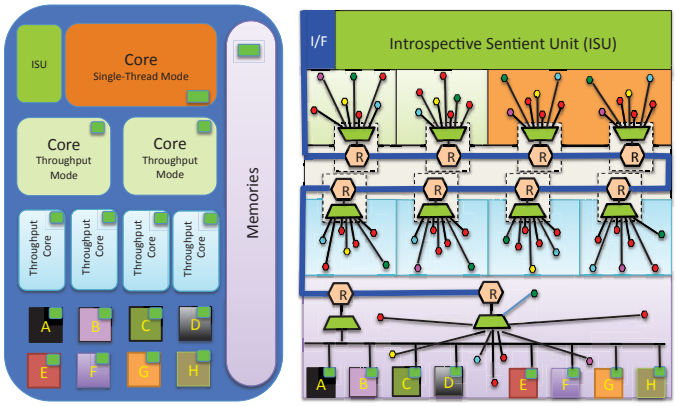

Fig. 5. CPSoC with the distributed sensor network (sNoC) and the introspective sentient unit (ISU). A ring network with a time division multiplexed (TDM) router connects all the distributed sensors of different types in cores, memories and accelerators. [9]

Thus the CPSoC architecture consists of a combination of a sensor-actuator-rich computational platform supported by adaptive NoCs, Introspective Sentient Units (ISU), and an adaptive & reflective middleware to manage and control both the information (cyber) and physical environment and characteristics of the chip. As shown in Figure 3 the CPSoC architecture is broadly divided into several layers of abstraction, for example, applications, operating system, network and bus communication, hardware, and the circuit / device layers. CPSoC inherits most features of classical MPSoCs, with the addition of on-chip sensing and actuation to enable the Observe-Decide-Act (ODA) CPS control paradigm. Unlike traditional MPSoCs [1], each layer of the CPSoC can be made self-aware and adaptive, by a combination of software and physical sensors and actuators as shown in Figure 3. These layer-specific feedback loops are integrated into a flexible stack which can be implemented either as firmware or middleware as shown by the dotted line in Figure 3.

CPSoC distinctly differs from a traditional MPSoC in several ways. For instance, traditional MPSoC paradigms lack the ability to sense the system states and behaviors across layers of the system stack due to lack of architectural support; they are incapable of exploiting and exposing process and workload variations due to a lack of suitable abstractions at multiple layers. Furthermore, they sacrifice usable performance and energy potentials by adopting worst case design (guard-bands), and lack support for multi-level actuation mechanisms and adaptations to aggressively meet competing and conflicting demands. Moreover, traditional MPSoCs lack self-learning mechanisms that can anticipate failures and predict vulnerabilities.

The CPSoC architecture overcomes these limitations by supporting three key ideas: **1) Cross-Layer Virtual and Physical Sensing & Actuation:** CPSoCs are sensor-actuator-rich MPSoCs that include several on-chip physical sensors (e.g., aging, oxide breakdown, leakage, reliability, temperature) on the lower three layers as shown by the on-chip-sensing-and-actuation block (OCSN) and the Introspective Sensing Units (ISUs) in Figures 4 and 5. Virtual sensing and actuation [10] is accomplished across the abstraction stack. For instance virtual actuations such as application duty cycling, and checkpointing are software/hardware interventions that can predictively influence system design objectives. Virtual actuation can be combined with physical actuation mechanisms commonly adopted in modern chips [11]. **2) Simple and Self-Aware Adaptations:** Two key attributes of the self-aware CPSoC are adaptation of each layer and multiple cooperative Observe-Decide-Act (ODA) loops. As an example, the unification of an adaptive computing platform (with combined DVFS, ABB, and other actuation means) along with a bandwidth adaptive NoC offers extra dimensions of control and solutions in comparison to traditional MPSoC architecture. **3) Predictive Models and On-line Learning:** Predictive modeling and on-line learning abilities enhance self-modeling abilities in the CPSoC paradigm. The system behavior and states can be built using on-line or off-line linear or non-linear models in time or frequency domains [12]. CPSoC's predictive and learning abilities improve autonomy for managing system resources and assisting proactive resource utilization [7].

The CPSoC concept has been applied in several contexts: energy efficiency, temperature control and adaptation, aging, etc. For instance, in SmartBalance [13], we deployed a Variability-Aware Dynamic Task Allocation and Scheduling approach where the native Linux kernel was enhanced with introspective and predictive models to improve system performance and power efficiency. Initial experimental results using the variability- and heterogeneity-enhanced Linux kernel demonstrated improvements of 20% (for workload variability alone) and 35-50% (considering workload and process variability jointly) in system efficiency (performance per Watt) over a standard Linux kernel for a quad-core machine. As another example, we showed how to improve the Resilience of CPSoC using temperature control and adaptation [14]. We deployed a neural network based system identification and model predictive control within CPSoC's adaptive and reflective middleware layer to precisely control and adapt to system dynamics and manage the CPSoC's mean time to failure (MTTF). We observed that the MTTF improves between 2-5 years across all the benchmarks tested using simple runtime adaptation policies as detailed in [14].

While CPSoC has proven to be a good initial exemplar for a self-aware SoC platform, it handles to a limited extent the

challenges of varying application/user demands and functional aberrations (Challenges a and partially b in Figure 1). The self-awareness models in CPSoC did not consider malicious attacks, functional design errors and non-functional aberrations, and in that sense it is still not fully autonomous and lays the ground for future work to address many of the open challenges we describe next.

IV. Open Problems and Conclusions

Although CPSoC constitutes an excellent platform for self-aware, autonomous and adaptive SoCs, several desirable features are still missing. Neither design errors in the hardware or software (bugs) nor malicious attacks can be detected. Promising approaches based on assertions and run-time monitoring [15] would, if integrated in CPSoC, extend the scope of self-awareness and could allow the system to adapt sensibly when design errors or a hostile attack is detected.

CPSoC and all similar approaches have a limited repertoire to detect and to react to aberrations of all kinds. When detected, aberrations are represented as binary symbols and a mostly hard-coded reaction is executed. When an aberration is happening slowly and gradually over longer time periods (e.g. slowing down due to aging), a more appropriate reaction would be register this change, and also its trend, long before it becomes a failure. Higher levels in the system can be notified about this ongoing trend and new applications could be rejected if their performance demands exceed the expected system's capacity in the near future. Alternatively, the priorities of the system could be rearranged to avoid high temperatures and to counter the ongoing trend. Such a scenario requires a richer representation of the system's capabilities and, in particular the changes of these capabilities over time. Also, an explicit hierarchy of goals and their dynamic reordering would allow for appropriate responses to aberrations and trends.

Moreover, we cannot ignore that the data collected by real and virtual sensors vary in their accuracy and reliability. Hence, in addition to collecting sensor data we also have to collect meta-data about this data in order to make correct decisions [16].

Collecting and processing data by real and virtual sensors is expensive and should be avoided if the data is not useful. Attention mechanisms [17] can direct and limit these activities to save energy and processing resources without compromising the quality of monitoring.

This discussion illustrates how to expand the scope of self-awareness for the benefit of increased adaptivity and autonomy of the system. Furthermore, dynamic learning should play a much bigger role because even the adaption mechanisms themselves have to adapt to the specific situation, environment, and applications. However, much more research is needed to explore this space and find efficient solutions and to understand the involved trade-offs in sufficient depth and detail.

Finally it should be noted, that the expansion of autonomy and adaptation brings new challenges. Validation of highly adaptive systems is radically different from validation of systems with a well specified and static functionality. Thus, the deployment of more self-aware, adaptive and autonomous MPSoCs will most likely be constrained by our capacity to validate them and to guarantee safety and real-time properties.

Acknowledgements

We thank Dr. Santanu Sarma for the conception and development of the CPSoC platform and prototype. We also acknowledge financial support by the Marie Curie Actions of the European Union's H2020 Programme.

References

[1] W. Wolf, A. A. Jerraya, and G. Martin, "Multiprocessor system-on-chip (mpsoc) technology," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 10, pp. 1701–1713, 2008.

[2] P. Gupta *et al.*, "Underdesigned and opportunistic computing in presence of hardware variability," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 32, no. 1, pp. 8–23, 2013.

[3] A. Rahmani, P. Liljeberg, A. Hemani, A. Jantsch, and H. Tenhunen, *The Dark Side of Silicon*. Springer, Switzerland, 1st edition ed., 2016.

[4] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, "Parameter variations and impact on circuits and microarchitecture," in *Proc. of Design Automation Conference*, pp. 338 – 342, 2003.

[5] B. H. Cheng, R. de Lemos, H. Giese, P. Inverardi, and J. Magee, "Software engineering for self-adaptive systems: A research roadmap," pp. 1–26, 2009.

[6] P. Cong-Vinh, *Autonomic Networking-on-Chip: Bio-Inspired Specification, Development, and Verification*. CRC Press, 2011.

[7] S. Sarma, N. Dutt, P. Gupta, N. Venkatasubramanian, and A. Nicolau, "Cyberphysical-system-on-chip (CPSoC): A Self-aware MPSoC Paradigm with Cross-layer Virtual Sensing and Actuation," in *Proc. of the Design, Automation & Test in Europe Conference*, pp. 625–628, 2015.

[8] N. Dutt, A. Jantsch, and S. Sarma, "Toward Smart Embedded Systems: A Self-aware System-on-Chip (SoC) Perspective," *ACM Trans. Embed. Comput. Syst.*, vol. 15, no. 2, pp. 22:1–22:27, 2016.

[9] S. Sarma, N. Dutt, N. Venkatasubramanian, A. Nicolau, and P. Gupta, "Cyber Physical-System-On-Chip (CPSoC): Sensor-Actuator Rich Self-Aware Computational Platform," tech. rep., Univeristy of California Irvine, 2013.

[10] S. Sarma, N. Dutt, and N. Venkatasubramanian, "Cross-layer virtual observers for embedded multiprocessor system-on-chip (mpsoc)," in *Proceedings of the 11th International Workshop on Adaptive and Reflective Middleware*, ARM '12, (New York, NY, USA), pp. 4:1–4:7, ACM, 2012.

[11] S. Sarma, N. Dutt, and P. Gupta, "Strength of diversity: heterogeneous noisy sensor allocation and placement for optimal reconstruction and accurate fullchip thermal estimation," tech. rep., Univeristy of California Irvine, 2014.

[12] L. Ljung, ed., *System Identification (2Nd Ed.): Theory for the User*. Prentice Hall PTR, 1999.

[13] S. Sarma, T. Muck, L. A. D. Bathen, N. Dutt, and A. Nicolau, "SmartBalance: A Sensing-driven Linux Load Balancer for Energy Efficiency of Heterogeneous MPSoCs," in *Proc. of the Annual Design Automation Conference*, pp. 109:1–109:6, 2015.

[14] S. Sarma and N. Dutt, "Self-healing policies for aging resilience in cyberphysical system-on-chip (CPSoC)," *University of California, Irvine, Tech. Rep. CECS TR-14-03*, 2014.

[15] R. Medhat, B. Bonakdarpour, D. Kumar, and S. Fischmeister, "Runtime monitoring of cyber-physical systems under timing and memory constraints," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 14, no. 4, 2015.

[16] N. TaheriNejad, A. Jantsch, and D. Pollreisz, "Comprehensive observation and its role in self-awareness - an emotion recognition system example," in *Proceedings of the Federated Conference on Computer Science and Information Systems*, (Gdansk, Poland), September 2016.

[17] J.-S. Preden, K. Tammemäe, A. Jantsch, M. Leier, A. Riid, and E. Calis, "The benefits of self-awareness and attention in fog and mist computing," *IEEE Computer, Special Issue on Self-Aware/Expressive Computing Systems*, pp. 37–45, July 2015.