Self-Aware Cyber-Physical Systems-on-Chip

Invited Paper

Nikil Dutt
University of California Irvine
CA, USA
Email: dutt@uci.edu

Axel Jantsch
TU Wien
Vienna, Austria
Email: axel.jantsch@tuwien.ac.at

Santanu Sarma
University of California Irvine
CA, USA
Email: santanu@uci.edu

***Abstract*—Self-awareness has a long history in biology, psychology, medicine, and more recently in engineering and computing, where self-aware features are used to enable adaptivity to improve a system's functional value, performance and robustness. With complex many-core Systems-on-Chip (SoCs) facing the conflicting requirements of performance, resiliency, energy, heat, cost, security, etc. – in the face of highly dynamic operational behaviors coupled with process, environment, and workload variabilities – there is an emerging need for self-awareness in these complex SoCs. Unlike traditional MultiProcessor Systems-on-Chip (MPSoCs), self-aware SoCs must deploy an intelligent co-design of the control, communication, and computing infrastructure that interacts with the physical environment in real-time in order to modify the system's behavior so as to adaptively achieve desired objectives and Quality-of-Service (QoS). Self-aware SoCs require a combination of ubiquitous sensing and actuation, health-monitoring, and statistical model-building to enable the SoC's adaptation over time and space. After defining the notion of self-awareness in computing, this paper presents the Cyber-Physical System-on-Chip (CPSoC) concept as an exemplar of a self-aware SoC that intrinsically couples on-chip and cross-layer sensing and actuation using a sensor-actuator rich fabric to enable self-awareness.**

I. Introduction

The concept of self-awareness has long inspired computer scientists and proponents of artificial intelligence because of our own, individual self-inspecting experience and observations in psychology, biology and brain sciences. Already at the dawn of theoretical psychology self-awareness and consciousness have been carefully examined by James introducing the physical, mental, and spiritual selves, and ego [8] and Freud differentiating between unconsciousness, preconsciousness and consciousness [6]. Since then psychology has developed a detailed understanding and rich terminology of phenomena of consciousness [17]. With the emergence of cognitive theories elaborate cognitive architectures have been proposed to explain the emergence of the observed phenomena [27]. The Global Workspace theory (GW) advanced by Baars in the 1980s [1] enjoys broad support by cognitive scientists because it correctly predicts many psychological and neuropsychological phenomena [2]. It contends that consciousness is a global resource allocator providing access to the global workspace and many specialized processing modules that operate unconsciously and in parallel. A salient feature of GW is that various *contexts* powerfully shape the processing of the conscious content and the specialized modules. An important and always present context is the keen awareness of the subjects own condition (self-awareness) and the current environment (situation-awareness).

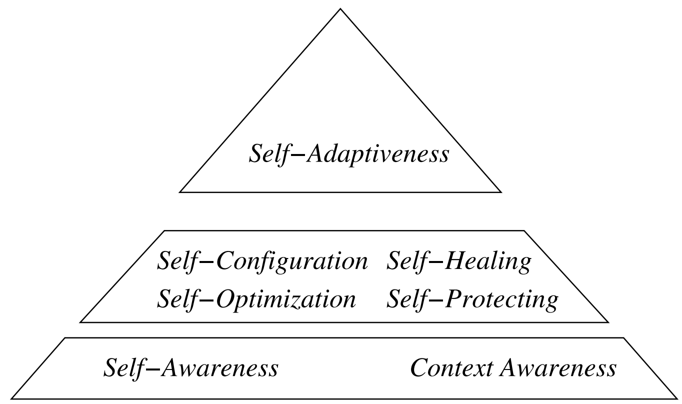

Fig. 1. Hierarchy of self-* properties [19].

The assumption of GW and most cognitive models that awareness is functional and benefits the individual motivates the many attempts of computer scientists and engineers to equip computers with awareness in the hope to improve their functional value, performance and robustness.

Responding to increasing software size and complexity during the 1990s researchers have called for more smartness. Laddaga contends that self adaptive software understands "*what it does; how it does it; how to evaluate its own performance; and thus how to respond to changing conditions*" [12]. Reacting to the challenges DARPA published a Broad Agency Announcement of Self-adaptive software [12] and IBM developed a vision of autonomic computing [11]. In the sequel many self-* properties have been proposed and studied. As figure 1 illustrates, self-adaptiveness depends upon the realization of several more specialized properties such as self-protecting and self-configuration which in turn depend on the awareness of the system of its own state (self-awareness) and its environment (context-awareness). These latter, lower level properties were often considered to be easily realized by collecting raw state information such as the detection of faults and processing delays [18]. However, this narrow definition of awareness is in contrast to our own intuition and to the richness of this notion in psychology and cognitive science. Assuming that under relentless evolutionary pressure all costly features are functional and beneficial other researchers have developed more elaborate concepts of awareness [3], [14].

Following a similar approach we have defined a set of properties that we deem necessary or relevant for self-awareness [4], [9]:

- **Semantic Interpretation** includes an appropriate abstraction of the primary input data and a disambiguation of possible interpretations.
- **Desirability Scale** provides a uniform goodness-scale for the assessment of all observed properties.
- **Semantic Attribution** maps properties into the desirability scale suggesting how good or bad an observation is for the system.
- **History of a Property:** Awareness of a property implies awareness of its change over time.
- **Goals** provide the context in which interpretation and semantic attribution is meaningful.
- **The Purpose** of a smart embedded systems is to achieve all its goals.
- **Expectation on Environment:** The system expects a specific environment and detects if the environment deviates significantly from expectations.
- **Expectation on Subject:** Similarly, the system's own state and condition are continuously assessed to detect deviations, degradation, performance and malfunctions.
- **Inspection Engine:** Continuously monitoring and assessing the situation requires a specific machinery that integrates all observations into a single, consistent world.

In the rest of the paper we present the Cyber-Physical System-on-Chip (CPSoC) as an exemplar self-aware SoC platform, and then relate and illustrate properties of self-awareness using this CPSoC exemplar.

II. CPSoC

CyberPhysical-System-on-Chips (CPSoC) [22], [24], is a smart embedded system paradigm that combines a sensor-actuator-rich self-aware computing-communication-control (C3) centric paradigm with an adaptive & reflective middleware (a flexible hardware-software stack and interface between the application and OS layer) to control the manifestations of computations (e.g., aging, overheating, parameter variability etc.) on the physical characteristics of the chip itself and the outside interacting environment. Inspired by the adaptive and learning abilities of autonomous computing [10] and C3 paradigm of CPSs [13], CPSoC provides a computing framework that assures the dependability of the cyber/information processing (i.e., the cyber aspects such as integrity, correctness, accuracy, timing, reliability and security) while simultaneously addressing the physical manifestations (in performance, power, thermal, aging, wear-out, material degradation, and reliability and dependability) of the information processing on the underlying computing platform. Note unlike the reference architecture proposed by Lewis et al. [15], CPSoC aims to coalesce these two traditionally disjoint aspects/abstractions of the cyber/information world and the underlying physical computing worlds into a unified abstraction of computing by using cross-layer virtual/physical sensing and actuation to enable a C3 centric self-aware computing platform.

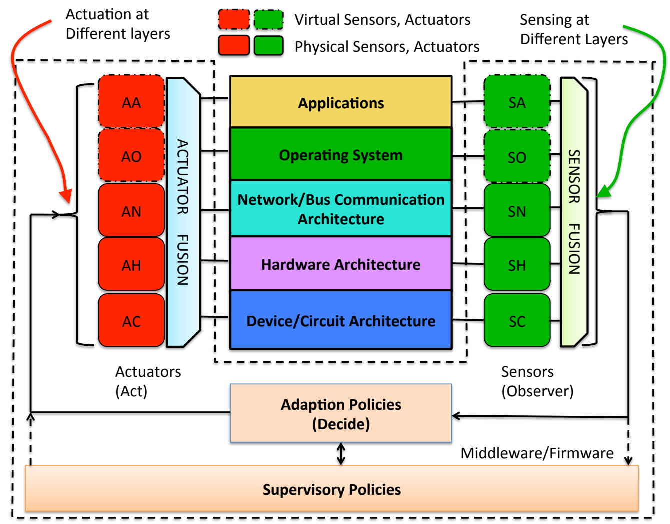

Fig. 2. Cross-layer virtual sensing and actuation at different layers of CPSoC [21], [22].

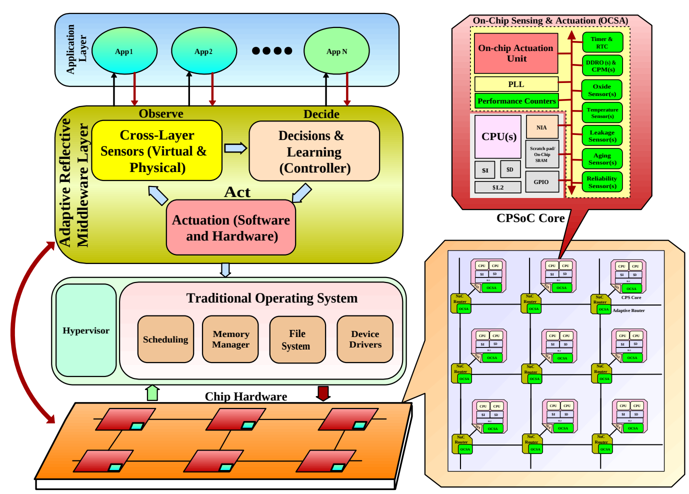

Fig. 3. CPSoC architecture with adaptive Core, NoC, and the Observe-Decide-Act Loop as Adaptive, Reflexive Middleware [21], [22].

The CPSoC architecture consists of a combination of sensor-actuator-rich computation platform supported by adaptive NoCs (cNoC – communication NoC; and sNoC – sensor NoC), Introspective Sentient Units (ISU), and an adaptive & reflective middleware to manage and control both the cyber/information and physical environment and characteristics of the chip [22], [24]. 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 MPSoC in addition to on-chip sensing and actuation to enable the ODA (Observe-Decide-Act) paradigm. Unlike traditional MPSoC, 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 Fig. 2. 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 Fig. 4.

CPSoC distinctly differs from a traditional MPSoC in several ways. 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 lack of suitable abstractions at multiple layers. Furthermore, they sacrifice usable performance and energy opportunities 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. CPSoC overcomes these limitations as detailed below.

A. CPSoC Features

The CPSoC framework supports four key ideas: 1) physical and virtual sensing and actuation 2) Simple and self-aware adaptations 3) multi or cross-layer interactions and interventions 4) predictive modeling and learning. We briefly describe these below. (A detailed description is in our Technical Report [24].)

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, performance counters, as well as voltage, current, and power sensors [22], [24]) on the lower three layers as shown by the on-chip-sensing-and-actuation block (OCSN) in Fig. 3. On the other hand, **virtual sensing** is a physical-sensor-less sensing of immeasurable parameters using computation [23]. It can be viewed as a software sensor that provides indirect measurement of abstract conditions, contexts, inferences or estimates by processing (e.g., combining, aggregating, or predicting) sensed data from either a set of homogeneous or heterogeneous sensors. It is also a computational technique that enhances and/or adds sensing capability, introduces sensing options, increases sensitivity, enables efficient sensor resource uses, and overcomes physical placement and cost restrictions. When combined with different kinds of sensors, virtual sensing enables consensus to resolve faults and errors while providing a test bed for on-chip sensor fusion [26].

Similarly, we define virtual actuations [22], [24](e.g., application duty cycling, algorithmic choice, checkpointing) that are software/hardware interventions that can predictively influence system design objectives such as performance, power, and reliability. Virtual actuation can be combined with physical actuation mechanisms commonly adopted in modern chips (e.g., DVFS and adaptive body biasing (ABB) to control the chip performance, power, and parametric variations); the notion of actuator fusion in CPSoC represents virtual and physical actuations that are combined across different layers of abstraction [22], [24].

2) Simple and Self-Aware Adaptations: Self-awareness is used to describe the ability of the CPSoC to observe its own internal behaviors as well as external systems it interacts with such that it is capable of making judicious decisions that optimize performance and other quality of service (QoS) metrics [10]. Self-aware systems will be capable of adapting their behavior and resources to automatically find the best way to accomplish a given goal despite changing environmental conditions and demands. A self-aware system must be able to monitor its behavior to update one or more of its components (hardware architecture, operating system and running applications), to achieve its goals.

Two key attributes of the self-aware CPSoC are adaptation of each layer and multiple cooperative ODA (Observe-Decide-Act) 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 [22], [24] offers extra dimensions of control and solutions in comparison to traditional MPSoC architecture. These cooperative and hierarchical control loops –e.g., the combination of traditional control loop (dotted lower box in Fig. 4) together with virtual sensing enabled optimized loop (upper loop in Fig. 4) – effectively translate user goals or QoS into one or more multiple design objectives [22], [24].

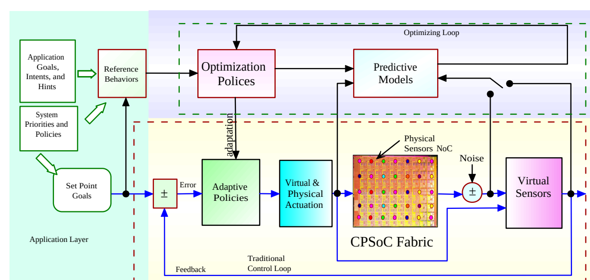

Fig. 4. Adaptation using predictive control model and policies in CPSoC [21], [22].

3) Predictive Models and On-line Learning: Predictive modeling and on-line learning abilities of the system behavior as well as internal and external (environmental) states provide 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 [16]. We specifically use statistical and neural network approaches [5], [7] such that the model accuracy can be traded-off for model computational complexity. We use regression based linear predictors and nonlinear neural predictors to build models of the system performance, power and energy consumption using the cross-layer events, hardware counters, and on-chip sensor data. In addition, use of coupling parameters (a metric that quantifies the interactions between layers) helps to develop application and cross-layer interaction models for nominal and abnormal operations. We use the predictive and learning abilities of CPSoC to improve autonomy in managing the system resources and assisting proactive resource utilization in the run-time system [22], [24].

III. Self-Awareness in CPSoC

In section I we described a necessary set of properties for a system to become self-aware. For a system to be self-aware in the ideal sense, it should exhibit all of these properties at the highest levels of fidelity and with maximal range of attributes. However, depending on the context of the engineered system to be designed, it may not need all of these attributes, nor does it need to have maximal coverage in terms of the fidelity and range of attributes. Indeed, in the SoC context, we are typically resource constrained, with limited area, power, and thermal budgets. Thus we need to consider carefully how to incorporate different self-awareness attributes, and at what levels they can be designed within the overall envelope of all the SoC design constraints. Towards that end, we describe below how the CPSoC exemplar has incorporated the self-aware properties described earlier. Recall that CPSoC supports four main concepts: (1) physical and virtual sensing and actuation, (2) self-monitoring and adaptation, (3) cross-layer interactions, (4) predictive modeling and learning.

Physical and virtual sensing collects the primary data but also does a great deal of abstraction with specific goals in mind. Disambiguation is performed partially and implicitly but not in a formal manner. Hence, CPSoC offers implicit **semantic interpretation**.

A **desirability scale** is built into in the system for each sensor or their combination in the virtual sensing approach. All the functions – from the collection of sensory data to the control algorithms and the actuation mechanisms – use this desirability scale for accurate sensemaking and deriving insights, (e.g., how the system should perform and what constitutes a malfunction). CPSoC may appear to use an implicit desirability scale but in fact uses an explicit one through mapping functions and calibration tables as discussed in the implementation of a temperature sensor in [20], [24]. An explicit mapping function (based on look-up tables (LUTs)) in a ring oscillator based thermal sensor transforms the ring oscillator frequency to that of actual temperature reading on the die (measured in ôC) is used as the desirability scale for thermal awareness. Moreover, as the mapping functions and LUTs are virtualized in software [24], the provision for re-purposing these sensors, for example, a ring oscillator based leakage power or aging sensor can be realized. An explicit desirability scale has the advantage of increased flexibility by decoupling observations from decisions. When new types of sensory data and new observations have to be interpreted, they are explicitly mapped onto a desirability scale allowing the control and decision algorithms to remain unchanged while still taking the new information into account. Hence, we consider that CPSoC has an explicit desirability scale while performing **semantic attribution** implicitly.

CPSoC keeps track of the **historic evolution of properties** by using explicit notions of epoch and sampling time [22], [25]; hence it embodies this dimension of awareness. As an example, the state space dynamic models in [21], [22] uses explicit notions of states in the previous epoch to predict the states value in the current epoch. Additionally, sensor data across multiple epochs are stored as history in order to make an assessment of the average behavior of certain states of interest.

CPSoC has clear **goals** such as the maximization of energy efficiency and the ability to detect and tolerate faults and failing components. Again, these goals are configurable and can be specified, selected, or changed in the control algorithms, which make them efficient and effective and provides flexibility to adapt to new goals during the system's lifetime. As an example, the online Simulated Annealing based optimization scheme in [22] is made configurable to accept or change the objective function at run-time by using a Linux system call for performance maximization while achieving energy efficiency.

The **purpose** is only defined partially, because CPSoC is a *platform* that can be used in a range of applications. By definition a platform will always only define some the system's goals, such as detecting and tolerating faults, but will leave the definition of other goals to the application.

CPSoC being a platform with system level goals (e.g, thermal efficiency) is aware of some aspects of the environment (e.g. the ambient temperature) and has consequently **expectations on the environment** in addition to the **expectations on itself** as encoded in the sensing and control algorithms. However, these expectations are implicit and reactions to environmental changes are limited to specific cases corresponding to the system-level goals.

The **inspection engine** is based on prediction models (or their variants) as in Fig. 4 , thus providing the capability to inspect state variables in time and space.

In summary, the design of the CPSoC framework has carefully considered how to balance the needs of self-awareness in an SoC context, and has realized different self-aware attributes with the dual goals of maximizing self-awareness, while minimizing overheads and simultaneously meeting the complex, interdependent set of constraints faced by the system. Hence, it can be considered as a self-aware system as defined above and in our earlier work [4], [9], to the extent reasonable in an SoC context. Indeed, it should be noted that more awareness is not necessarily better. There is a trade-off between the cost of awareness, the efficiency of implementation, and the flexibility and generality that would come with higher degrees of awareness. This is particularly important in the specific context of self-aware SoCs that must meet a multitude of cross-purpose constraints.

IV. Conclusions

While there has been a large body of work in attempting to achieve adaptivity through self-aware computing systems, the phrase "self-awareness" has been used rather loosely, typically with no well defined model of what properties a self-aware system must exhibit. Towards that end we first reviewed the concept of self-awareness as applied to the domain of computing systems. Since there is no agreed-upon definition of self-awareness within the computing realm, we then defined a set of necessary or relevant properties for a computing system to be considered self-aware. Emerging SoCs arguably face an even more complex set of conflicting constraints, in the face of highly dynamic workloads, as well as process and environmental variability. Furthermore, with increasing complexity of, and heterogeneity in the SoC platform architecture, there is a critical need for these SoCs to be self-aware, and perform in an adaptive manner. We presented the Cyber-Physical System-on-Chip (CPSoC) platform as an exemplar of a self-aware, heterogeneous many-core SoC platform. CPSoC achieves self-aware adaptation through a principled orchestration of ubiquitous (virtual) sensing and actuation, coupled with health-monitoring and statistical model building. We then briefly described how each of the self-awareness properties are manifested in the CPSoC platform. Since these facilities must be tightly woven into the SoC's hardware and software fabric, CPSoC's self-awareness properties have been engineered carefully to prevent the excessive overheads of intrusive sensing/actuation. We believe the CPSoC exemplar provides one view of a self-aware SoC. Other papers in this special session provide additional perspectives on building self-aware SoC platforms.

V. Acknowledgments

This work was partially supported by the NSF Variability Expedition award CCF-1029783.

References

[1] B. Baars. *A Cognitive Theory of Consciousness*. Cambridge University Press, 1989.

[2] B. Baars. The conscious access hypothesis: origins and recent evidence. *Trends in Cognitive Science*, 6(1):47–52, 2002.

[3] T. Chen, F. Faniyi, R. Bahsoon, P. R. Lewis, X. Yao, L. L. Minku, and L. Esterle. The handbook of engineering self-aware and self-expressive systems. *Computing Research Repository (CoRR)*, abs/1409.1793, 2014.

[4] N. Dutt, A. Jantsch, and S. Sarma. Towards smart embedded systems: A self-aware system-on-chip perspective. *ACM Transactions on Embedded Computing Systems, Special Issue on Innovative Design Methods for Smart Embedded Systems*, 2015. invited.

[5] L. V. Fausett. *Fundamentals of neural networks*. Prentice-Hall, 1994.

[6] S. Freud. *Fragments of an analysis of a case of hysteria*, volume 7. Hogarth, London, 1953. Originally published in 1905.

[7] S. S. Haykin, S. S. Haykin, S. S. Haykin, and S. S. Haykin. *Neural networks and learning machines*, volume 3. Pearson Education Upper Saddle River, 2009.

[8] W. James. *The Principles of Psychology*, volume I. Dover, New York, 1950. Originally published in 1890.

[9] A. Jantsch and K. Tammemäe. A framework of awareness for artificial subjects. In *Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis*, CODES '14, pages 20:1–20:3, New York, NY, USA, 2014. ACM.

[10] J. Kephart et al. The vision of autonomic computing. *Computer*, 36(1):41 – 50, jan 2003.

[11] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[12] R. Laddaga. Active software. In *Self-Adaptive Software*, volume 1936 of *Lecture Notes in Computer Science*, pages 11–26. Springer, July 2001.

[13] E. Lee. Cyber physical systems: Design challenges. In *ISORC, 2008*, pages 363 –369, may 2008.

[14] P. Lewis, A. Chandra, S. Parsons, E. Robinson, K. Glette, R. Bahsoon, J. Torresen, and X. Yao. A survey of self-awareness and its application in computing systems. In *Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2011 Fifth IEEE Conference on*, pages 102–107, October 2011.

[15] P. R. Lewis, A. Chandra, F. Faniyi, K. Glette, T. Chen, R. Bahsoon, J. Torresen, and X. Yao. Architectural aspects of self-aware and self-expressive computing systems. *IEEE Computer*, August 2015.

[16] L. Ljung. *System identification*. Springer, 1998.

[17] A. Morin. Levels of consciousness and self-awareness: A comparison and integration of various neurocognitive views. *Consciousness and Cognition*, 15(2):358 – 371, 2006.

[18] H. Psaier and S. Dustdar. A survey on self-healing systems: approaches and systems. *Computing*, 91(1):43–73, 2011.

[19] M. Salehie and L. Tahvildari. Self-adaptive software: Landscape and research challenges. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 4(2):14, 2009.

[20] S. Sarma and N. Dutt. FPGA emulation and prototyping of a cyberphysical-system-on-chip (CPSoC). In *Proceedings of the International Symposium on Rapid System Prototyping (RSP)*, New Delhi, India, October 2014.

[21] S. Sarma, N. Dutt, P. Gupta, A. Nicolau, and N. Venkatasubramanian. On-chip self-awareness using cyberphysical-systems-on-chip (CPSoC). In *Proceedings of the 12th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, New Delhi, India, October 2014.

[22] S. Sarma, N. Dutt, P. Gupta, A. Nicolau, and N. Venkatasubramanian. Cyberphysical-system-on-chip (cpsoc): A self-aware mpsoc paradigm with cross-layer virtual sensing and actuation. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2015*, March 2015.

[23] 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, pages 4:1–4:7, New York, NY, USA, 2012. ACM.

[24] S. Sarma, N. Dutt, N. Venkatasubramanian, A. Nicolau, and P. Gupta. Cyberphysical-system-on-chip (CPSoC): Sensor-actuator rich self-aware computational platform. Technical report, Center for Embedded Computer Systems University of California, Irvine, CA 92697-2620, USA, May 2013. CECS Technical Report No: CECS-TR-13-06.

[25] S. Sarma, T. Muck, L. A. Bathen, N. Dutt, and A. Nicolau. Smartbalance: A sensing-driven linux load balancer for energy efficiency of heterogeneous mpsocs. In *DAC 2015*, Jun 2015.

[26] S.Sarma, N.Dutt, and P.Gupta. Strength of diversity: Exploiting cheap heterogeneous noisy sensors for accurate full-chip thermal estimation. Technical Report CECS-TR-14-011, Univeristy of California Irvine, Jan 2014.

[27] D. Vernon, G. Metta, and G. Sandini. A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. *Evolutionary Computation, IEEE Transactions on*, 11(2):151–180, April 2007.