Chapter 9

Self-Organizing Architectures for Digital Signal Processing

Daniele Peri and Salvatore Gaglio

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/53334

1. Introduction

Technological bounds in digital circuits integration in the last decades have been fostering the development of massively parallel architectures for tasks that had not been touched before by traditional parallel paradigms. Even in personal computers, as well as in consumer and mobile devices, it is common to find powerful processing units composed of processing elements in the range of the hundreds to the thousands.

The request for mobile devices, that are self-powered, almost permanently switched on and connected through wireless networks, as well as environmental friendliness constraints, obviously urges to reduce energy consumption of processing units.

On the other hand, applications continuously keep pushing forward computing power needs. A number of such applications are actually performed on application specific or on almost general-purpose parallel multi-core unit, as in the case of 3D graphics, sound processing, and the like, in the multimedia arena.

The current industrial trend aims to increase computing power and energetic efficiency by adding cores to both main processors and specialized units. A number of experimental architectures have been proposed that try to achieve the same goal by exploiting different designs. Coarse and fine grained architectures, and more in general, reconfigurable architectures have been proposed to make the hardware adapt to the required tasks instead of using specialized software running on general purpose processing elements. This has especially been the case in computer vision, and intelligent systems in general.

More interestingly, in these fields, the quest for massively parallel and energy efficient hardware implementations, coupled with biological models of reference, may pour interest in reviewing well and lesser studied approaches that are centered on self-organizing processing

structures. Indeed, current research on pattern recognition shows significant interest in highly structured models built on large numbers of processing nodes trained by demanding algorithms that can, at least partially, be implemented in a parallel fashion.

In this chapter we provide a review of self-organization as it may appears at the various abstract levels of computational architectures, and including applications to real-world problems.

We start outlining the properties related to complexity and self-organization in natural and artificial systems. Then we de-scribe the computational models that are better suited to study self-organizing systems. We then discuss self-organization at the hardware level. Finally, we look at networked systems paying particular attention to distributed sensing networks.

2. Self-organization and self-organizing systems

Human speculation on the visible order of things, either living or not, is so ancient to be considered one of the fundamental questions of mankind. Science has always been exploring the complex structure of Nature, adding pieces to pieces to its infinite puzzle.

Meanwhile, technologies evolve benefiting from new findings, sometimes trying either successfully or ingenuously to duplicate Nature's work. Improvements in technologies then reflect on further science advancements, closing the loop.

Order, self-organization, adaptation, evolution, emergence and several other terms reminds us that as artificial systems advance, gaining complexity, it is expected for them to be compared to natural ones in both structure and function.

At the time of the vacuum tube digital computer introduction in the 1940s, McCulloch and Pitts had already proposed their neuron model, while cyberneticists were starting to recognize their interdisciplinary studies on natural and artificial systems as a brand new field.

With their simple and primitive circuits, made of few thousands discrete components, digital computers were certainly "complex" with respect to the available technology, but orders of magnitude simpler and unstructured than their biological computational counterparts made of billions of neurons arranged by some "Self-Organization" process.

Anyway, it did not took much to Von Neumann to start exploring the theory of Cellular Automata and self-reproducing machines, attempting to bridge natural and artificial computational models.

Rosenblatt's perceptron was another attempt to propose a biologically inspired computational framework. As a confirmation of the difficulties in reverse-engineering Nature, it took a few decades for Artificial Neural Networks built with perceptrons to become viable means to tackle useful computational tasks.

http://dx.doi.org/10.5772/53334

Of all the connotations given to perceptron networks, "adaptive" has been certainly one of the most adopted, however many of the terms cited before have found some kind of use with ANNs [1].

With respect to the "self-organization" term, its use has been used so widespread in computer and information processing literature that the effort to determine its introduction is quite pointless.

One of the most well known uses of the term can be traced back to Kohonen's "Self-Organizing Maps" (SOMs) [2]. Differently from perceptron based Artificial Neural Networks, trained with supervised algorithms, Kohonen proposed an unsupervised method whose geometrical representation is that of a continuous rearrangement of points of the feature space around auto-determined cluster centers represented as cells in a two-dimensional array. In the topological representation of the evolving map during learning, centers can be visualized as they move forcing the two-dimensional map to stretch in the effort to cover the feature space. Even if SOMs are not a derivation of any biological model, some parallelism with the visual neocortex both in terms of function and structure has been drawn [3].

Given the impact of SOMs in machine learning and the excitement produced by a simple and effective unsupervised learning algorithm, it is not surprising that a large number of papers followed in Kohonen's, and that research on SOMs is still carried on actively. Fritzke proposed structures that grow from a small number of cells [4] modulating the network structure accordingly to the unknown probability distribution of the input.

Other research on SOMs, similarly to the evolution of multi-layered supervised neural networks, introduced some hierarchical organization, as Choi and Park did with their "Self-Creating and Organizing Neural Network" [5], or Rauber et al. with their "Growing Hierarchical SOM" [6], or followed the path of hardware implementation of SOMs either in analog [7], or digital form [8]. A surveillance application was proposed by Chacon-Murguia and Gonzalez-Duarte that mixes SOMs with neuro-fuzzy networks to detect objects in dynamic background for surveillance purposes [9].

At some point, Dingle and Jones proposed the Chaotic Self Organizing Feature Map [10] based on recurrent functions leading to chaotic behavior. Continuing this research, more recently Silva et al. proposed a self-organizing recursive architecture for continuous learning [11]. The importance of chaotic dynamics in self-organization will re-emerge in the discussion about computational frameworks.

The relevance of the previously cited work notwithstanding, "self-organization" –in the biological sense– capabilities, should rather be attributed to systems capable of self-assembling from simple elementary units, finding their coordination by direct interactions governed by simple mathematical rules – intrinsic of Nature, it could be stated. Such systems should rather find their biological model in the "prebiotic soup", in which chemical interactions between atoms and then compounds, lead to the organization of cells, tissues and complex organisms.

Random Boolean Networks (RBNs) were originally introduced by Kauffmann to model gene regulation mechanisms [12]. In Kauffman's Biology-centered view, evolution is result-

ing from the Darwinian selection mechanisms coupled with self-organization, adding the concept of "anti-chaos" to the already large and unsettled realm of complex systems.

Cells in biological systems share the same genetic information, nevertheless they differentiate on the basis of specific activation patterns of that very same information. From a computer engineering point of view, this is an upside-down perspective, as changing the software has been the way to make machines adapt to problems.

Indeed, the dichotomy of hardware and software has been the key of early digital computers evolution, permitting to get rid of the hand-wired logic of primordial calculators. Incidentally, if we put apart for a moment most of the involved technological considerations, Von Neumann's pioneering work on self-reproducing automata was a fifty years forward leap to meet the biological research at some common point.

More or less in the same years as Kauffman, Wolfram meticulously described the complex behavior of one-dimensional cellular automata (Figure 1) pointing out the emergence of self-organization [13, 14].

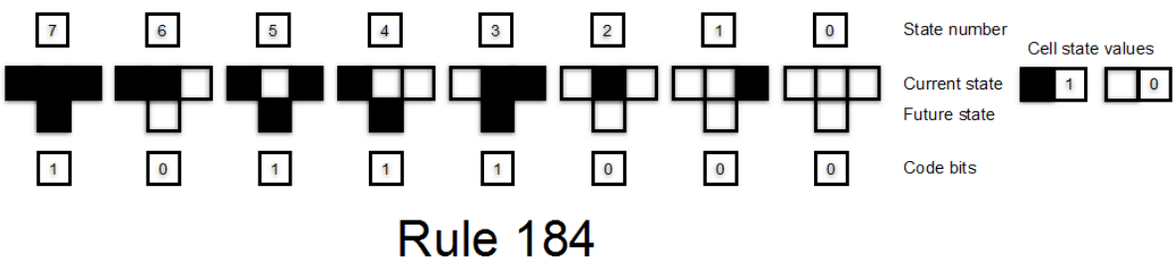

Figure 1. Simple one-dimensional binary cellular automata with two-cell neighborhood are named after the code introduced by Wolfram. The eight possible states for each cell and its neighborhood are arranged from right to left accordingly to the decimal interpretation of the three bit current values. The eight bits describing the future states are then interpreted as a decimal number. The code may be generalized to elementary cellular automata with any number of states, dimensionality, and neighborhood size.

Coincidentally, at that time the influence of Mandelbrot's work on fractals was at its peak, as well as the interest for simple formulae able to produce results so similar to those of natural processes [15]. That was certainly a rather inspiring time for those who are subject to the fascination of complexity arising from simplicity but, indeed, this feeling has been pervading the research in information systems for decades.

It was also the time of the advent of networking and – a few more years would have taken the Web to be brought to life – Internet. The latter has the mark of a "self-organizing" system well in its roots, and even in its name, in some way. Then, in a short time lapse, wireless networks broadened the communication horizon once again providing us with mobile systems.

The realm of computers thus has reached a point where interconnected systems at macro scale coexist with the micro scale of the circuits they are built upon, while the nano-dimensionality is being intensively explored. Compared to the many scales adopted to observe biological systems at their molecular, cellular, tissutal and macroscopic levels, this is still a

http://dx.doi.org/10.5772/53334

very coarse and rigid stratification, nevertheless provides an interesting parallelism and some points to look at in the distance.

Either in the biological or in the computer realm, more levels bring more complexity. Systems are of different kinds and it needs some "handshaking", as in the networking jargon, to allow communication. Systems need to share resources, and then some sort of arbitration is needed. A large part of the engineering of information systems has thus become the design of communication and arbitration protocols to make system "self-organize".

Heylighen and Gershenson invoked self-organization in computers as a necessity to cope with an increasing complexity in information systems that creates a "bottleneck" limiting further progress [16]. They discussed inter-networking, and the rapid changes in hardware, software and protocols to deal with it, as only exacerbating the difficulties for human developers to keep everything under their own control. They then described a few qualitative principles to introduce self-organization in highly engineered and inter-networked information systems, with some references to current applications such as the hyperlinks-based Web, and, with some projections to the future, even software development paradigms.

Kohonen's networks, the medium access control and routing protocols of the many computer network types, and Kauffman's RBNs, all of them express self-organization of some degree. The heterogeneity of the three examples is evident, though. Some effort has been taken to formalize this hardly sought property of systems. Gershenson and Heylighen, moving from classical considerations based on thermodynamics, and then considering statistical entropy, provided an insight on what conditions should describe the emergence of self-organization in observed systems.

They concluded that the term "self-organization" may rather describe a way to look at systems than a class of systems [17].

3. Computational models

As anticipated, Random Boolean Networks (RBNs) trace in their biological model of inspiration their self-organizing abilities. RBNs consist in a network of N nodes with Boolean state, each having K of Boolean input connections. Both parameters N and K are fixed. Because of these characterizing parameters RBNs have also been called NK networks. Each node state is updated at discrete time steps accordingly to a Boolean function of the inputs. A variable number of Boolean outputs, propagating the node state, may departs from each node towards other nodes' inputs, arbitrarily. Indeed, both connections and the Boolean state update function are chosen randomly during initialization and never changed (Figure 2).

Kauffman discussed RBNs as finite discrete dynamical systems in terms of the sequences of states the networks run through. Given that 2^N states can be assumed by RBN, and that for each state there is only one possible successor, the network will run through finite-length cyclic sequences called *state cycles*, that are the *dynamical attractors* of the system.

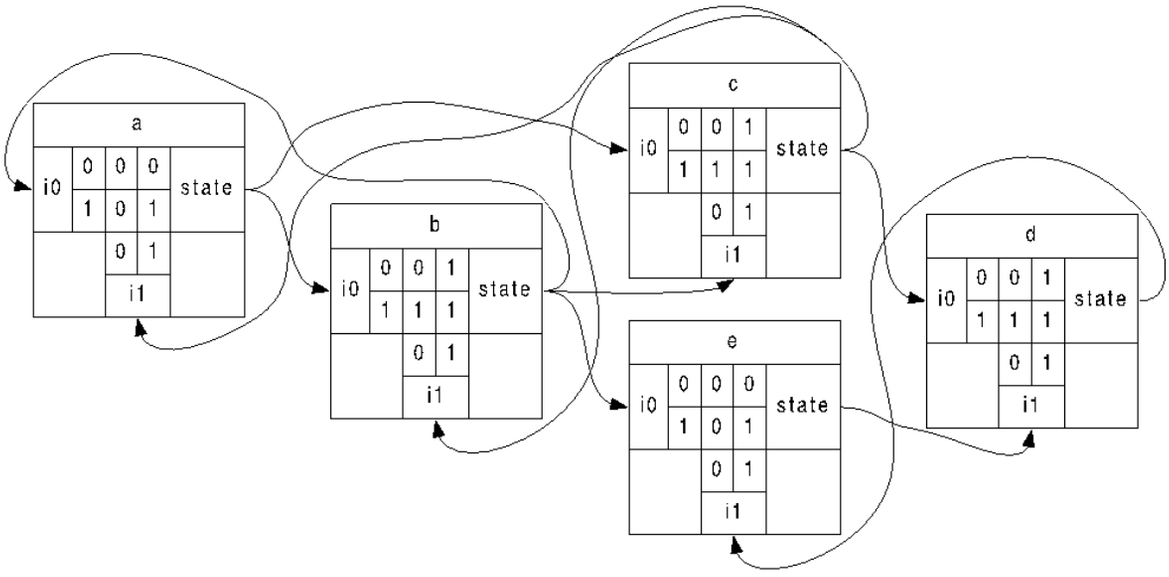

Figure 2. A Random Boolean Network with two input nodes. After each discrete time step, each node state is updated accordingly to the Boolean function of the inputs i0 and i1. Node state is fed to the output at the following step.

The behavior of a RBN can be represented by the state transition diagram, a directed graph having a connected component for each state cycle. Not all the states in each of these subgraphs are part of the respective cycle, as states having no antecedents – the so-called *garden-of-Eden* states – may be present; they instead compose the state cycle's *basin of attraction* (Figure 3).

Properties of the *state cycles*, such as cycle length, asymptotic patterns, and *basins of attraction* were used to classify the interesting complex behaviors of these simple models for different values of K [12]. Some basic findings, still providing some insights into the self-organization abilities of RBNs are reported in Table 1. When the network is completely interconnected ($K = N$), and the sensitivity to initial conditions is at its maximum, *state cycle* lengths become large as N increase, yet their number keeps being comparatively small.

When K is equal or greater than 5, RBNs keeps showing chaotic behavior. A few concepts need to be introduced to analyze these results. The *internal homogeneity* P of a Boolean function of K inputs is defined as the ratio $M/2^K$, with M beeing the maximum between the number of 1's and 0's in the output column of the function's truth table. The *bias* B is then defined as $1/\sqrt{P}$.

In contrast with the first two *chaotic* cases, when $K = 2$, RBNs show the emergence of "spontanous order" as both the cycle length and number of attractors scales with the square root of N. Moreover, these networks show other important properties that result in higher stability over perturbations of the activity of the nodes. Indeed, more recently, a linear dependance was found sampling larger networks [18]. For $K = 1$ the RBNs show a similar growth of the cycle length and an exponential rise of the number of attractors.

http://dx.doi.org/10.5772/53334

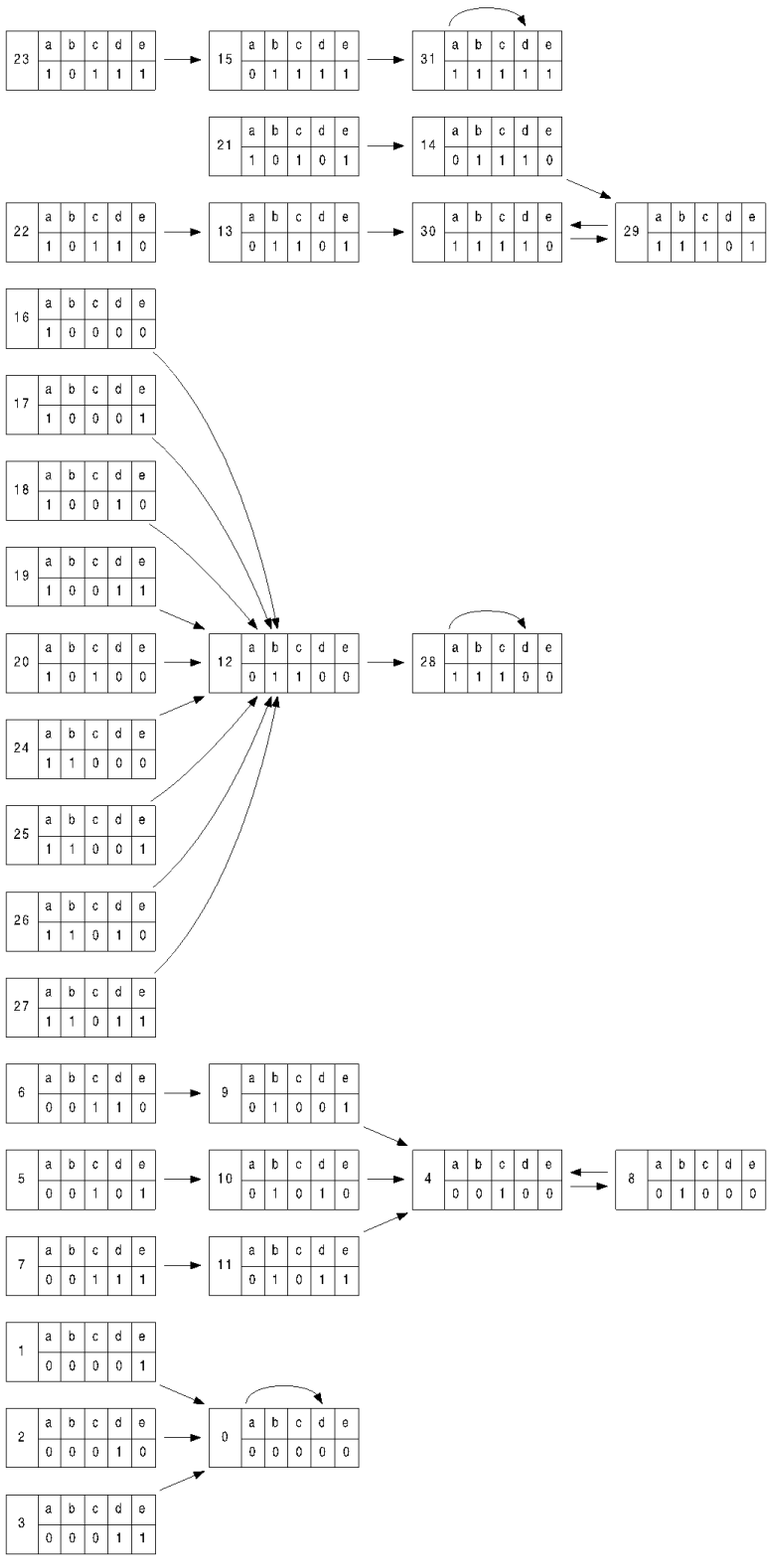

Figure 3. The state transition diagram of the RBN showed in Figure 2. States are numbered from 0 to 31 according to the binary representation of the five network nodes' values. The graph is partitioned into four unconnected components, one for each *state cycle*. The network will finally be attracted into one of the four *state cycles*: (31), (30, 29), (28), (4, 5), (0). On the left side, the 17 states that are unreachable from any other state are showed.

	State cycle length	Number of state cycle attractors
$K = N$	$\frac{2^{N/2}}{2}$	$\frac{N}{e}$
$K \geq 5$	$\frac{2^{\frac{BN}{2}}}{2}$ $(B > 1)$	$\sim N\left[\frac{\log\left(\frac{1}{1/2 \pm (P(K) - 1/2)}\right)}{2}\right]$
$K = 2$	$\sqrt{N}$	$\sqrt{N}$
$K = 1$	$\sqrt{\frac{\pi}{2}}\sqrt{N}$	Exponential in N

Table 1. Properties of RBNs for different values of K. The state cycle length is the median length of the state cycles. B is the network *bias* while $P(K)$ is the mean *internal homogeneity* of all the Boolean functions of K inputs.

The boundaries among the ordered, critical and chaotic dynamical phases, yet not quite analytically assessed, still inspire new studies. An updated introduction to RBNs, including references to Asynchronous RBNs (ARBNs), Deterministic Asynchronous RBNs (DARBNs) and other variants of RBNs, can be found in form of a tutorial in [18]. Gershenson described several methods to guide the self-organization of RBNs [19]. The need of a "guiding" process seems somewhat contradictory with the premise in the title. Indeed, he investigated the mechanisms through which natural selection may intervene in the self-organization of biological structures, suggesting engineers may use the same parameters characterized in computational frameworks, such as RBNs and Cellular Automata.

Getting back to the gene regulation mechanisms the RBNs were designed to model, "self-organization" succeeded in "reducing the complexity" provided by the tens of thousands genes in the human genome to the mere hundreds types of human cells. RBNs are finite state space, deterministic, dynamical, discrete time systems whose self-organizing property derives from having attractors, i.e. states that can be revisited by the network. RBN can be in either a ordered or a chaotic dynamical phase, transitions are possible and the transition from one to the other is "characterized by its criticality".

A static, stable phase preserves information but is prevented from computing or adapting. A chaotic phase provides the requested variability for computing and adapting, but is incapable of preserving information. As the critical "interface" between the two phases provides the advantages of both phases, guiding the RBN towards self-organizations means finding the necessary conditions to make RBNs evolve towards the critical regime. Gershenson then considered several factors that can induce such evolution and gives a few hints on how criticality could help improve adaptability, ability to evolve, and robustness of RBNs.

http://dx.doi.org/10.5772/53334

4. Cellular automata

From definition it is evident that cellular automata are a special case of RBNs in which each node receives inputs only from neighbors. Their simpler topology and consequent implementation has given an appeal to these minimal models manifesting self-organization that goes beyond "recreational" applications such as Conway's "Game of Life".

Some theoretical extensions include Probabilistic Cellular Automata (PCA), and Fault-tolerant Cellular Automata (FCA), studied by Gács [20] in the context of the problem of reliable computation with unreliable components. In particular, the error probability of each component is not required to decrease as the number of components increases, and the faults affect the local state but not the transition function.

Even if they are purely theoretical, such models may be useful in designing massively parallel "self-organizing" architectures, as due to the distributed nature of information in cellular automata, "self-stabilization" is required beside traditional error-correction techniques.

Cellular automata have had many applications to real-world problems. No surprisingly, several biological models have been simulated with cellular automata. Shumate et al. described a simulation of Ductal Carcinoma in Situ [21]. Chaudary et al. proposed a simulation model for Tumorigenesis [22]. Shimokawa and Muraki investigated a simple model of nerve excitement propagation [23]. Sakamoto et al. proposed a method for surgery simulation based on voxel automata [24].

Cellular automata have also been used to model complex dynamics such as those of urban traffic [25–27]. Recent applications of cellular automata to image processing include super pixel segmentation [28], image compression [29], and computer graphics [30].

Cellular automata also continue to be used in more theoretical studies on algorithms [31, 32].

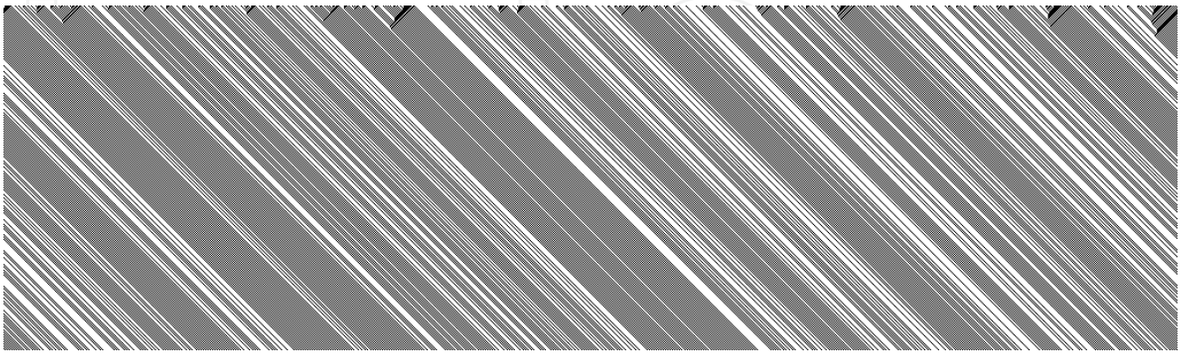

Figure 4. Rule 184 (see Figure 1) is one of most used cellular automaton in traffic simulation. The distribution of vehicles and spaces in a road lane is modeled as black and white cells, respectively, in each image row. The topmost cell row depicts the initial distribution (with a black/white ratio of 3/8) let evolve over 300 iterations. After few iterations still presenting random behavior, visible in form of triangular structures, the regularization ability of the rule is manifest as vehicles move to the right at constant speed.

5. Hardware

The balance between the hardware and software components in signal processing applications has always been a trade-off between the flexibility of the microprocessor-based solutions and the performance of ASIC implementations.

Taking the aforementioned SOMs and ANNs into account, literature abounds in hardware implementations that are motivated by scarce performance of the analogous sequential methods. In the 1980s, aiming at parallel real-time signal processing with the then available analog very-large-scale integration (VLSI) technology, Chua introduced his Cellular Neural Networks (CNN) [33], describing some application to image processing tasks [34]. Subsequently, Yang et al. showed a VLSI implementation of CNNs [35].

A couple of decades later, Ruckert et al. discussed massively parallel implementations of artificial neural networks at ultra-large-scale integration (ULSI) [36], later showing a massively parallel architecture for SOMs [37].

Hopfield had started its seminal work on recurrent neural networks posing himself the question whether "the ability of large collection of neurons to perform computational tasks may in part be a spontaneous collective consequence of having a large number of interacting simple neurons". He concluded that this is actually the case, and that implementation of such models could lead to integrated circuits that are more fault-tolerant than normal circuits [38].

Weightless Neural Networks (WNNs), being based on random access memories, provide another ANN paradigm inherently tied to circuit-level implementation whose origins trace back to Alexander's "Self-adaptive universal logic circuits" [39].

Though all of these are examples of systems having self-adapting qualities, self-organization at the hardware level – the "microscopic" layer in the our biological analogy– had simply not been possible until the advent of reconfigurable circuits, such as field programmable gate arrays (FPGAs), and coarse-grain reconfigurable arrays, added a new degree of configurability, and related complexity, to computer systems. A survey on reconfigurable hardware with emphasis on real-time configuration is provided by Shoa and Shirani [40].

5.1. Coarse-grained and fine-grained architectures

Hartenstein, reviewing most of the "coarse-grained reconfigurable architectures" of a decade (circa 2000) [41], suggested that with the explosion of design costs and reduction of production life cycles, performance becomes relatively less important in the design of computing devices. Instead, extension of product longevity, "reduction of support turnaround, in-system debugging, profiling, verification, tuning, field-maintenance, and field-upgrade" time by employing reconfigurable arrays is much more important. Hartenstein, dismissing "von Neumann" architectures as obsolete, in the light of the dominance of host/accelerators designs, proposed a new coarse-grained soft machine paradigm, in which a so called "co-compilation" provides instructions for the host and data-path configuration information at the same time.

http://dx.doi.org/10.5772/53334

Other approaches to the mapping of high-level coarse-grained mapping from high-level synthesis are present in literature [42]. Recently, implementations as System On a Chip design of highly reconfigurable arrays have been described with applications to face detection [43], Internet protocols processing [44], FIR filters and ICA [45]. Several SOMs, CNNs and derivative neural network models have been designed for reconfigurable hardware [46, 47].

Fine-grained systems bring configurability close to the gate or transistor level, permitting analog, digital, and hybrid implementations. In contrast to coarse-grained systems, data path width is reduced to the bare minimum, with the advantages of increased flexibility and lower costs, but the general purpose routing is generally less energy efficient.

Nanotechnologies aim at even finer degrees of integration, and it is reasonable to assume that at the hardware level new computational paradigms may emerge because of that. However, as in the pioneering stage of any technology, the span from theory and implementation may not be short. Lin et al. proposed a hybrid FPGA architecture based on nano-RAM [48] with run-time configuration abilities and high logic density, to revert later to a CMOS SRAM implementation for immaturity of the nano-RAM fabrication processes [49].

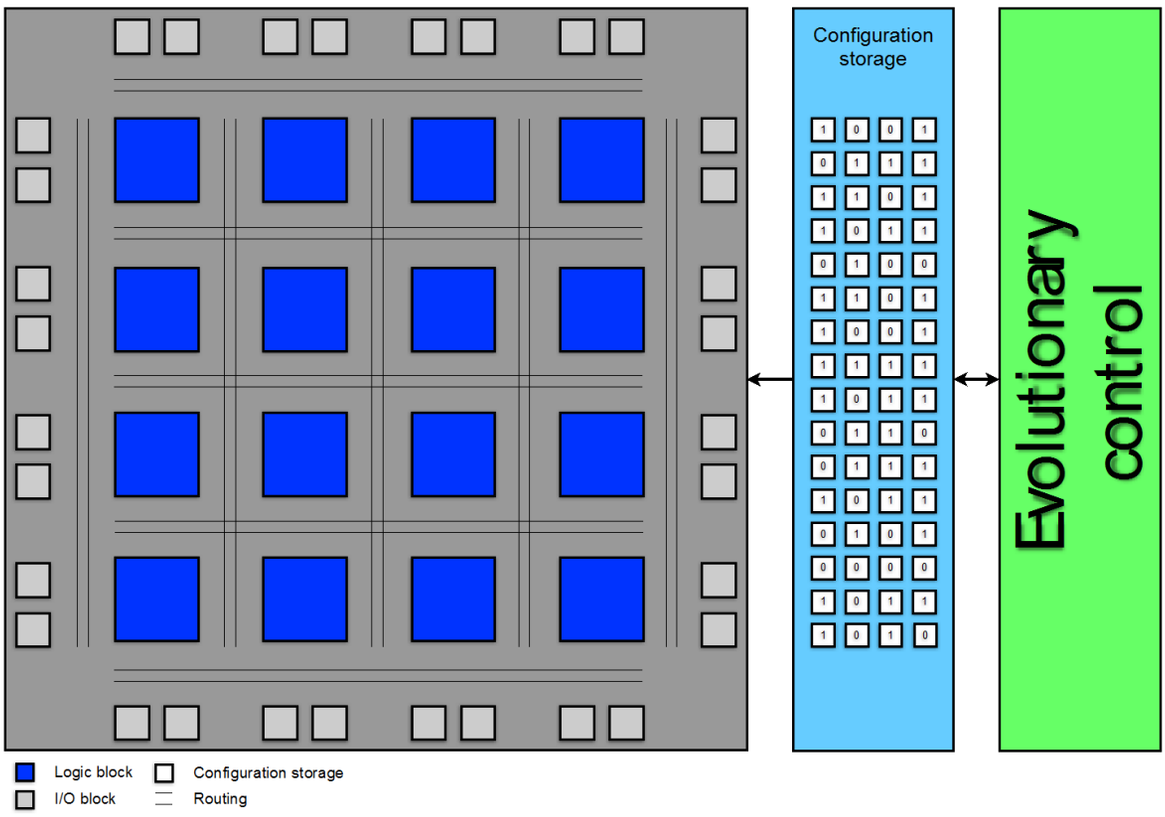

Figure 5. A schematic depiction of Evolvable Hardware. A reconfigurable device (i.e. FPGA), in gray, is coupled with a configuration storage (light blue). Configuration is updated by the control block (green) in real time according to some fitness function, as in genetic programming.

5.2. Evolvable hardware

Reconfigurable hardware is turned into a specific implementation by loading bitstreams compiled from "soft cores" coded in some hardware description language. Being a totally software matter, a host processor can perform real-time reconfiguration when needed. New paradigms blending traditional machine language and reconfigurable hardware bitstreams become thus possible.

The idea that hardware could change autonomously its own configuration seeking the best one according to some fitness function was called evolvable hardware, recurring –once again– to a biological metaphor [50, 51]. Continuing with the metaphor, the bitstream takes the role of the digital DNA (Figure 5).

Approaches from genetic and evolutionary programming are attempted on the hardware configuration bitstream. Interesting applications of EHWs to pattern recognition are those presented by Glette et al. [52, 53].

Even though most work on EHW concerns digital implementations, some evolution-oriented analog implementation are reported in literature, such as the evolvable hardware architecture based on field programmable transistor arrays [54], and quantum-inspired paradigm to be implemented in evolutionary analog hardware [55].

The enthusiasm of the early 2000s notwithstanding, EHW has not yet delivered what promised. Cancare et al. [56] investigating the reasons of this apparently missed success, and citing scalability issues as the most prominent, propose to abandon generic genetic algorithms and look at hierarchical evolution and linkage learning, encouraging support from the Evolutionary Computation community.

6. Networks

Computer networks provide many examples of global behaviors emerging from interactions of elements without centralized control. At different levels of abstraction and implementation, from medium access control and routing, to the application level protocols, algorithms drive each independent network node so that some global goal, be it communication, co- ordination, or distributed processing, is achieved. Thus, non-surprisingly, "self-" prefixed and akin terms abound in related literature.

While computer networks in general are a rather natural field to study self-organization, and many analogies with biological systems may be detected, without broadening too much our discussion, we restrict our discussion considering only one example of network of very simple nodes in which distributed processing of locally collected data is the main goal: Wireless Sensor Networks (WSNs).

These systems are composed by a number of nodes, consisting in miniaturized, battery-operated, computational elements fitted with sensors to monitor the surrounding environment, that are connected through short distance radio links. Depending on the applications,

the number of nodes may vary sensibly from a few to thousands and more units. In many scenarios sensor nodes are dispersed in the environment thus their expendability becomes another important requisite.

As a consequence of these constraints on physical size, energy supply and cost per unit, processing resources are limited, and in most designs they only consist in simple microcontrollers. Comprehensive surveys on WSNs including sensor technologies, network protocols, and hardware and software architectures, are provided by Akyldiz et al. [57], and Yick et al. [58].

Even though WSNs were conceived as distributed sensing architectures, several examples are provided in literature about nodes also performing in-network pre-processing of raw sensed data [59]. The need for a trade-off between the limited available energy source, and the manifold application scenarios [60], typically calls for the application of self-organization techniques, breaking the boundaries between the traditional architectural layers in order to optimize the behavior of such nodes. Sohrabi et al. presented a number of algorithms and protocols for self-organization of wireless sensor networks [61]. Self-organization techniques to reduce energy consumption in ad-hoc networks of wireless devices were described by Olascuaga-Cabrera et al. [62]. With even more technological constraints than WSNs, Wireless Sensor Body Networks (WSBNs) consist of wearable and implantable devices. Health-monitoring usage of WSBNs is discussed by Hao and Foster [63].

Indeed, due to their ultra-low energy consumption requirements, WSBNs represent a very challenging scenario for sensor devices based on current, and even near future, general purpose processing elements, and implementing signal processing algorithms on nodes may prove unfeasible.

Alternative approaches based on application specific integrated circuits have been investigated [64]. Departing from the network oriented vision, and calling for the establishment of self-managing systems engineering, Beal et al. proposed the “amorphous medium” abstraction in which the deployed sensor network represents the physical space of the application to be engineered [65].

From an engineering perspective, the application goal is reached by programming the medium instead of the network. The former abstracts the computational model, turning sensor nodes into points of the physical space. A global behavior is described in a specifically crafted language, as it were to be executed by the abstract medium. Actually, the abstract description is compiled into code to be executed identically on each node. Besides executing the same code, nodes interact only with neighboring devices.

Beal et al. called this programming paradigm amorphous computing, revealing their inspiration to come from some properties of biological systems, such as morphogenesis and regeneration. More interestingly, even though with substantial topological differences, many similarities can be detected between the “amorphous-medium” and cellular automata.

7. Conclusions

The paradoxical fascination of simplicity producing complexity has traversed decades of research in information systems. Even more now that extremely high integration is pushing millions of highly modular circuits in few square millimeters, and inter-networking is the next –or rather, current – large scale integration.

Some research directions seem to suggest that breaking some of the fixed ties in engineered system, letting systems auto-organize in response to the environmental changes, as biological systems have been doing for millions of years, is the way to go to "put some order in the chaos". In support of these indications, self-organizing systems have provided interesting results in modeling complex processes, blurring a little the line between artificial and natural systems.

Other researches seek to extend self-organization to the extreme of self-healing systems able to recognize their own faults and self-repair, while biological applications confirm that taking into account self- organization when studying natural processes, while not an easy task, can provide more comprehensive and effective models.

If all these efforts move on the path towards truly intelligent systems, or even Artificial Life –as some have been suggesting for years– is yet to be discovered, nevertheless it is a very interesting path.

Author details

Daniele Peri and Salvatore Gaglio

DICGIM - University of Palermo, Italy, ICAR - CNR, Palermo, Italy

References

[1] Widrow B, Lehr M. 30 years of adaptive neural networks: perceptron, Madaline, and backpropagation. Proceedings of the IEEE 1990;78(9) 1415 –1442. doi:10.1109/5.58323.

[2] Kohonen T. The self-organizing map. Proceedings of the IEEE 1990;78(9) 1464 –1480. doi:10.1109/5.58325.

[3] Bednar JA, Kelkar A, Miikkulainen R. Scaling Self-Organizing Maps To Model Large Cortical Networks. Neuroinformatics 2001; 275–302.

[4] Fritzke B. Let it grow – self-organizing feature maps with problem dependent cell structure. In: Kohonen T, Ma kisara K, Simula O, Kangas J, editors, Artificial Neural Networks. North-Holland, Amsterdam, 1991;403–408.

[5] Choi DI, Park SH. Self-creating and organizing neural networks. Neural Networks, IEEE Transactions on 1994;5(4) 561 –575. doi:10.1109/72.298226.

[6] Rauber A, Merkl D, Dittenbach M. The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. Neural Networks, IEEE Transactions on 2002;13(6) 1331 – 1341. doi:10.1109/TNN.2002.804221.

[7] Macq D, Verleysen M, Jespers P, Legat JD. Analog implementation of a Kohonen map with on-chip learning. Neural Networks, IEEE Transactions on 1993;4(3) 456 – 461. doi:10.1109/72.217188.

[8] Ienne P, Thiran P, Vassilas N. Modified self-organizing feature map algorithms for efficient digital hardware implementation. Neural Networks, IEEE Transactions on 1997;8(2) 315 –330. doi:10.1109/72.557669.

[9] Chacon-Murguia M, Gonzalez-Duarte S. An Adaptive Neural-Fuzzy Approach for Object Detection in Dynamic Backgrounds for Surveillance Systems. Industrial Electronics, IEEE Transactions on 2012;59(8) 3286 –3298. doi:10.1109/TIE.2011.2106093.

[10] Dingle A, Andreae J, Jones R. The chaotic self-organizing map. In: Artificial Neural Networks and Expert Systems, 1993. Proceedings., First New Zealand International Two-Stream Conference on. 15 –18. doi:10.1109/ANNES.1993.323092.

[11] da Silva L, Sandmann H, Del-Moral-Hernandez E. A self-organizing architecture of recursive elements for continuous learning. In: Neural Networks, 2008. IJCNN 2008.

[12] Kauffman SA. The Origins of Order: Self-Organization and Selection in Evolution. 1 edition. Oxford University Press, USA, 1993.

[13] Wolfram S. Statistical mechanics of cellular automata. Rev. Mod. Phys. 1983;55 601–644. doi:10.1103/RevModPhys.55.601.

[14] Wolfram S. Universality and complexity in cellular automata. Physica D: Nonlinear Phenomena 1984;10(1-2) 1 – 35. doi:10.1016/0167-2789(84)90245-8.

[15] Mandelbrot BB. The Fractal Geometry of Nature. WH Freeman and Co., New York, 1982.

[16] Heylighen F, Gershenson C, Staab S, Flake G, Pennock D, Fain D, De Roure D, Aberer K, Shen WM, Dousse O, Thiran P. Neurons, viscose fluids, freshwater polyp hydra-and self-organizing information systems. Intelligent Systems, IEEE 2003;18(4) 72 – 86. doi:10.1109/MIS.2003.1217631.

[17] Gershenson C, Heylighen F. When Can We Call a System Self-Organizing? In: Banzhaf W, Ziegler J, Christaller T, Dittrich P, Kim J, editors, Advances in Artificial Life, volume 2801 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2003;606–614. doi:10.1007/978-3-540-39432-7 65.

[18] Gershenson C. Introduction to Random Boolean Networks. In: Bedau M, Husbands P, Hutton T, Kumar S, Suzuki H, editors, Workshop and Tutorial Proceedings, Ninth

International Conference on the Simulation and Synthesis of Living Systems (ALife IX). Boston, MA, 160–173.

[19] Gershenson C. Guiding the Self-organization of Random Boolean Networks. ArXiv e-prints 2010;1005.5733.

[20] Gacs P. Reliable cellular automata with self-organization. In: Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on. 90 –99. doi: 10.1109/SFCS.1997.646097.

[21] Shumate S, El-Shenawee M. Computational Model of Ductal Carcinoma In Situ: The Effects of Contact Inhibition on Pattern Formation. Biomedical Engineering, IEEE Transactions on 2009;56(5) 1341 –1347. doi:10.1109/TBME.2008.2005638.

[22] Chaudhary S, Shin SY, Won JK, Cho KH. Multiscale Modeling of Tumorigenesis Induced by Mitochondrial Incapacitation in Cell Death. Biomedical Engineering, IEEE Transactions on 2011;58(10) 3028 –3032. doi:10.1109/TBME.2011.2159713.

[23] Shimokawa K, Muraki S. A study on spatial and temporal visual simulation of nerve excitement propagation. In: Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on, volume 1. 217 –221 vol.1. doi: 10.1109/IJCNN.2000.857839.

[24] Sakamoto Y, Tuchiya K, Kato M. Deformation method for surgery simulation using voxel space automata. In: Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on, volume 4. 1026 – 1031 vol.4. doi:10.1109/ICSMC.1999.812551.

[25] Wei J, Wang A, Du N. Study of self-organizing control of traffic signals in an urban network based on cellular automata. Vehicular Technology, IEEE Transactions on 2005;54(2) 744 – 748. doi:10.1109/TVT.2004.841536.

[26] Rosenblueth DA, Gershenson C. A model of city traffic based on elementary cellular automata. Complex Systems 2011;19(4) 305–322.

[27] Gershenson C, Rosenblueth DA. Self-organizing traffic lights at multiple-street intersections. Complexity 2012;17(4) 23–39. doi:10.1002/cplx.20392.

[28] Wang D, Kwok N, Jia X, Fang G. A Cellular Automata approach for superpixel segmentation. In: Image and Signal Processing (CISP), 2011 4th International Congress on, volume 2. 1108 –1112. doi:10.1109/CISP.2011.6100339.

[29] Cappellari L, Milani S, Cruz-Reyes C, Calvagno G. Resolution Scalable Image Coding With Reversible Cellular Automata. Image Processing, IEEE Transactions on 2011; 20(5) 1461 –1468. doi:10.1109/TIP.2010.2090531.

[30] Debled-Rennesson I, Margenstern M. Cellular automata and discrete geometry. In: High Performance Computing and Simulation (HPCS), 2011 International Conference on. 780 –786. doi:10.1109/HPCSim.2011.5999908.

[31] OrHai M, Teuscher C. Spatial Sorting Algorithms for Parallel Computing in Networks. In: Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2011 Fifth IEEE Conference on. 73 –78. doi:10.1109/SASOW.2011.10.

[32] Maignan L, Gruau F. Convex Hulls on Cellular Spaces: Spatial Computing on Cellular Automata. In: Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2011 Fifth IEEE Conference on. 67 –72. doi:10.1109/SASOW.2011.14.

[33] Chua L, Yang L. Cellular neural networks: theory. Circuits and Systems, IEEE Transactions on 1988;35(10) 1257 –1272. doi:10.1109/31.7600.

[34] Chua L, Yang L. Cellular neural networks: applications. Circuits and Systems, IEEE Transactions on 1988;35(10) 1273 –1290. doi:10.1109/31.7601.

[35] Yang L, Chua L, Krieg K. VLSI implementation of cellular neural networks. In: Circuits and Systems, 1990., IEEE International Symposium on. 2425 –2427 vol.3. doi: 10.1109/ISCAS.1990.112500.

[36] Ruckert U. ULSI architectures for artificial neural networks. Micro, IEEE 2002;22(3) 10 –19. doi:10.1109/MM.2002.1013300.

[37] Porrmann M, Witkowski U, Ruckert U. A massively parallel architecture for self- organizing feature maps. Neural Networks, IEEE Transactions on 2003;14(5) 1110 –1121. doi:10.1109/TNN.2003.816368.

[38] Hopfield JJ. Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences 1982;79(8) 2554–2558.

[39] Aleksander I. Self-adaptive universal logic circuits. Electronics Letters 1966;2(8) 321 –322. doi:10.1049/el:19660270.

[40] Shoa A, Shirani S. Run-Time Reconfigurable Systems for Digital Signal Processing Applications: A Survey. The Journal of VLSI Signal Processing 2005;39 213–235. 10.1007/s11265-005-4841-x.

[41] Hartenstein R. A decade of reconfigurable computing: a visionary retrospective. In: Design, Automation and Test in Europe, 2001. Conference and Exhibition 2001. Proceedings. 642 –649. doi:10.1109/DATE.2001.915091.

[42] Lee G, Lee S, Choi K. Automatic mapping of application to coarse-grained re- configurable architecture based on high-level synthesis techniques. In: SoC De- sign Conference, 2008. ISOCC '08. International, volume 01. I–395 –I–398. doi: 10.1109/SOCDC.2008.4815655.

[43] He C, Papakonstantinou A, Chen D. A novel SoC architecture on FPGA for ultra fast face detection. In: Computer Design, 2009. ICCD 2009. IEEE International Conference on. 412 –418. doi:10.1109/ICCD.2009.5413122.

[44] Badawi M, Hemani A. A coarse-grained reconfigurable protocol processor. In: System on Chip (SoC), 2011 International Symposium on. 102 –107. doi:10.1109/ISSOC.2011. 6089688.

[45] Jain V, Bhanja S, Chapman G, Doddannagari L. A highly reconfigurable computing array: DSP plane of a 3D heterogeneous SoC. In: SOC Conference, 2005. Proceedings. IEEE International. 243 – 246. doi:10.1109/SOCC.2005.1554503.

[46] Hendry D, Duncan A, Lightowler N. IP core implementation of a self-organizing neural network. Neural Networks, IEEE Transactions on 2003;14(5) 1085 – 1096. doi: 10.1109/TNN.2003.816353.

[47] Starzyk J, Zhu Z, Liu TH. Self-organizing learning array. Neural Networks, IEEE Transactions on 2005;16(2) 355 –363. doi:10.1109/TNN.2004.842362.

[48] Zhang W, Jha NK, Shang L. A hybrid nano/CMOS dynamically reconfigurable system–Part I: Architecture. J. Emerg. Technol. Comput. Syst. 2009;5(4) 16:1–16:30. doi: 10.1145/1629091.1629092.

[49] Lin TJ, Zhang W, Jha NK. SRAM-Based NATURE: A Dynamically Reconfigurable FPGA Based on 10T Low-Power SRAMs. Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 2011;PP(99) 1 –5. doi:10.1109/TVLSI.2011.2169996.

[50] Yao X, Higuchi T. Promises and challenges of evolvable hardware. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 1999;29(1) 87 –97. doi:10.1109/5326.740672.

[51] Forbes N. Evolution on a chip: evolvable hardware aims to optimize circuit design. Computing in Science Engineering 2001;3(3) 6 –10. doi:10.1109/5992.919259.

[52] Glette K, Torresen J, Yasunaga M, Yamaguchi Y. On-Chip Evolution Using a Soft Processor Core Applied to Image Recognition. In: Adaptive Hardware and Systems, 2006. AHS 2006. First NASA/ESA Conference on. 373 –380. doi:10.1109/AHS.2006.55.

[53] Glette K, Torresen J, Hovin M. Intermediate Level FPGA Reconfiguration for an Online EHW Pattern Recognition System. In: Adaptive Hardware and Systems, 2009. AHS 2009. NASA/ESA Conference on. 19 –26. doi:10.1109/AHS.2009.46.

[54] Stoica A, Zebulum R, Keymeulen D, Tawel R, Daud T, Thakoor A. Reconfigurable VLSI architectures for evolvable hardware: from experimental field programmable transistor arrays to evolution-oriented chips. Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 2001;9(1) 227 –232. doi:10.1109/92.920839.

[55] Wang Y, Shi Y. The application of quantum-inspired evolutionary algorithm in analog evolvable hardware. In: Environmental Science and Information Application Technology (ESIAT), 2010 International Conference on, volume 2. 330 –334. doi: 10.1109/ESIAT.2010.5567359.

[56] Cancare F, Bhandari S, Bartolini D, Carminati M, Santambrogio M. A bird's eye view of FPGA-based Evolvable Hardware. In: Adaptive Hardware and Systems (AHS), 2011 NASA/ESA Conference on. 169 –175. doi:10.1109/AHS.2011.5963932.

[57] Akyildiz I, Su W, Sankarasubramaniam Y, Cayirci E. A survey on sensor networks. Communications Magazine, IEEE 2002;40(8) 102 – 114. doi:10.1109/MCOM.2002.1024422.

[58] Yick J, Mukherjee B, Ghosal D. Wireless sensor network survey. Computer Networks 2008;52(12) 2292 – 2330. doi:10.1016/j.comnet.2008.04.002.

[59] Gatani L, Lo Re G, Ortolani M. Robust and Efficient Data Gathering for Wireless Sensor Networks. In: Proceedings of the 39th Annual Hawaii International Conference on System Sciences, HICSS'06. IEEE Computer Society, 235–242.

[60] Anastasi G, Lo Re G, Ortolani M. WSNs for Structural Health Monitoring of His- torical Buildings. In: Proceedings of HSI'09. The 2nd Conference on Human System Interactions. IEEE, 574–579.

IntechOpen