ADAPTIVE AND EVOLVABLE HARDWARE AND SYSTEMS: THE STATE OF THE ART AND THE PROSPECTUS FOR FUTURE DEVELOPMENT

Mircea Gh. Negoita, Lukas Sekanina, Adrian Stoica

Abstract:

This paper is an overview on the Evolvable Hardware (EHW) - the exciting and rapidly expanding industrial application area of the Evolutionary Computing (EC), of the Genetic Algorithms especially. The content of the work has the following structure: the first part includes generalities on industrial applications of EC, and the importance of EHW in this frame; the second part presents the outstanding technological support making possible the implementation of system adaptation in hardware. Different kind of programmable circuits arrays are introduced. The third part tackles the most known EC based methods for EHW implementation; the fourth part deals with some concrete elements of the EHW design, including the current limits in evolutionary design of digital circuits. The last part is focused on some concluding remarks with regard to future perspectives of the area. A list of references used in this work was inserted at the end.

Keywords: *Evolvable Hardware (EHW), Evolutionary Design, Reconfigurable Hardware, Field Programmable Analogue Arrays (FPAA).*

1. Introduction

Nevertheless, it is hardware implementation of the most benefit for the society and indeed most revolutionizing application of **EC** by leading to the so-called **EHW**. These new **EC** based methodologies make possible the hardware implementation of both genetic encoding and artificial evolution, having a new brand of machines as a result. This type of machines is evolved to attain a desired behavior that means they have a *behavioral computational intelligence*. There is no more difference between adaptation and design concerning these machines, these two concepts representing no longer opposite concepts. A dream of technology far years ago currently became reality: adaptation transfer from software to hardware is possible by the end. Much more, the electronics engineering as a profession was radically changed: the most based on soldering assembling manufacturing technologies are largely replaced now by programming circuitry-based technologies, including EHW technologies.

1.1. Definitions, General Consideration and Classification of EHW

A *definition of* the **EHW** may be as follows: a subdomain of artificial evolution rep- resented by a *design methodology* (consortium of methods) involving the application of **EC** to the synthesis of digital and analogue electronic circuits and systems. A more agreed definition among the practitioners might be: **EHW** means *programmable hardware* that can be evolved [2]. But some members of the scientific community acting in the area consider the term *evolutionary circuit design* more descriptive for EHW features. Much more, another term is used nowadays for the same work - *evolvare* concerning to this evolvable ware with hardware implementation. This leads to a future perspective of using the term *bioware* concerning to a possible evolving ware with biologic environments implementation. Even some other environments are seen as possible evolvable media: *wetware* - real chemical compounds are to be used as building blocks or *nanotechnology* - relied on molecular scale engineering.

This new design methodology for the electronic circuits and systems is not a fash ion. It is suitable to the special uncertain, imprecise or incomplete defined real-world problems, claiming a continuous adaptation and evolution too. An increased efficiency of the methodology may be obtained by its application in the soft-computing framework that means in aggregation with other intelligent technologies [3; 4] such as fuzzy logic (**FS**) and neural networks (**NN**), Artificial Immune Systems (**AIS**), evolutionary algorithms (**EA**). The reason of **EHW** using the above mentioned type of application is relied on the main advantage over the traditional engineering techniques for the electronic circuit design, namely the fact that the designer's job is very much simplified following an algorithm [5] with a step sequence as below:

STEP 1 - *problem specification* - requirements specification of the circuit to be designed specification of basic (structural) elements of the circuit;

STEP 2 - *genome codification* - an adequate (genotypic) encoding of basic elements to properly achieve the circuit description;

STEP 3 - *fitness calculation* - specification of testing scheme used to calculate the genome fitness;

STEP 4 - *evolution (automatically generation of the required circuit)* - generation of the desired circuit.

Despite of the above-mentioned advantage of **EHW** methodology, its disadvantage is not to be neglected: (sometimes) suboptimal results are got over those ones of the classical methods that remain preferable in case of well-defined problems. The designer himself is involved by acting directly during the first three steps, while the fourth step is an *automatically generation* of the circuit. The flow modality of both step *3* and step *4* leads to same categorizing classes criteria for **EHW** (see [5]).

The practical **EHW** classification [6], can be adopted by: hardware type; controller type; Objective Function.

Based on the type of hardware and type of changes, **EHW** is classified as follows:

- ***R****econfigurable* ***d****igital* ***h****ardware* (**RDH**) - means digital hardware that changes quasi-instantly (short transition time compared with to processing time) and directly (single transition, no meaningful inter-mediary transition states) from the initial to the final configuration;
- ***M****orphable* ***d****igital* ***h****ardware* (**MDH**) - means digital hardware that changes from one complete configuration to another through a set of intermediate states, each having a functional role;
- ***R****econfigurable* ***a****nalog* ***h****ardware* (**RAH**) - is analog hardware with a dynamics of changes that is similar to RDH from the initial to final configuration;
- ***M****orphable* ***a****nalog* ***h****ardware* (**MAH**) - is analog hardware that changes gradually without switches;
- ***A****djustable/***T***unable/***P***arametric* ***HW*** (**ATPHW**) - is hardware in which the changes influence the parameters of a function.

Classification by controller type leads to:

- ***E****volutionary* ***h****ard***w***are* **- EHW** in which the controller employs evolution - any algorithms;
- ***E****mbryonic* ***h****ard***w***are* **- EHW** in which the controller employs a embryonic - initiated growth mechanism.

The classification by Objective Function is a more general one, covering all kind of Adaptive Systems, not the EHW only:

- ***External adaptation*** - featured by adaptive behavior in the presence of stimuli originating in the surrounding environment;
- ***Internal adaptation*** - featured by adaptive behavior in the presence of a disturbance located in the system itself;
- ***Darwinian adaptation*** - featured by adaptive behavior when the response is directed toward modifying the object;
- ***Singerian adaptation*** - adaptive behavior when the response is directed towards modifying the environment;
- ***Smart System*** (**SS**) - a system aware of its state, operation and changing environment. It can predict what will happen at a certain future. This knowledge can lead to adaptation;
- ***Smart adaptive system*** (**SAS**) - can adapt to a changing environment, a similar setting without being "ported" to it, adapt to a new/unknown application.

Dramatic changes happen in the relation between hardware and the application environment, and this in case of *malicious faults* or need for *emergent new functions* that claim for in-situ synthesis of a totally new hardware configuration. **EHW** is suitable for flexibility and survivability of autonomous systems as that ones developed by NASA JPL. **EHW** *survivability* means to maintain functionality coping with changes in hardware characteristics under the circumstances of adverse environmental conditions as for example: temperature variations, radiation impacts, aging and malfunctions. **EHW** *flexibility* means the availability to create new functionality required by changes in requirements or environment.

The application developer may meet different design tasks to be evolved. As the case, the design to be evolved could be: a program, a model of hardware or the hardware itself. Algorithms that run outside the reconfigurable hardware, mainly feature the actual **EHW** state of the art, but also some chip level attempts were done. The path from chromosome to behavior data-file is different in case of intrinsic and *extrinsic* (see more in [1]). **EHW** Evolution in Simulation has some typical features and parameters value as follows [5]: computationally intensive (640,000 individuals for about 1000 generations); runs over tens of hours, expected about 3 min in 2010 on desktop PC for experiments with netlists of about 50 nodes; SPICE scales badly, namely time increases nonlinearly with as a function of nodes in netlist, in about a subquadratic to quadratic way; no existing hardware resources allow porting the technique to evolution directly in hardware (and not sure will work in hardware). See in [1] more details on the huge advantage of the on- chip - versus CAD/synthesis tools.

1.2. The Technological Support of EHW

The appearance of programmable integrated circuits, especially their new generation - ***f****ield* ***p****rogrammable* ***g****ate* ***a****rrays* (**FPGA**s) and most recently reconfigurable ***a****nalogue* ***a****rrays* (**FPAA**s) and ***f****ield-***p***rogrammable* ***i****nterconnection* ***c****ircuits* (**FPIC**s) or configurable digital chips at the functional block level, (*open-architecture* **FPGA**s) make possible for most companies to evolve circuits as the case would be. The appearance of reconfigurable ***a****nalogue* ***a****rrays* (**FPAA**s) was crucial for **EHW** technological support. The analog reconfigurable hardware allows prevention or removal of essential fabrication mismatches and other refined technological problems by evolving circuits. The programmability of **FPAA**s is limited to just only allow configuration around op-amp level. But the applications require also for many interesting circuit topologies to be evolved below the op-amp level. This application requirement led to another kind of programmable (reconfigurable) hardware relied on *evolution-oriented devices* that have some advantages over **FPAA**, namely: can reprogram many times, can understand what's inside and are featured by a flexible programmability. This is the so called *custom made* **EHW**-*oriented reconfigurable hardware*. See in [1] more details regarding the types of *custom made* **EHW**-*oriented reconfigurable hardware*.

2. The Structure of an EHW System EC Based of EHW Implementation

The language for programming reconfigurable hardware must define: an *alphabet*, expressing choices of cells and, a *vocabulary/grammar*, expressing the rules of in terconnect. An **EHW** system, any is its destination, either for demonstrations, prototype experiments or real time implementation, must be structured in from of two main components: the *reconfigurable hardware* (**RH**) and the *reconfigurable mechanism* (**RM**) [7]. Regarding the practical ways of implementing an evolvable system, real-world applications requirements are toward a reliable solution featured by compactness, low-power consumption and autonomy. The evolution on **JPL SABLES** (***S**tand-**A**lone **B**oard-**L**evel **E**volvable **S**ystem*) [8] is a solution that proved to be effective in various applications, see Fig. 1.

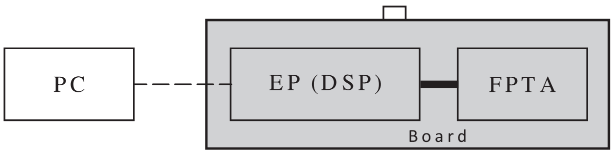

Fig. 1. The simplified bloc diagram of JPL SABLES [14].

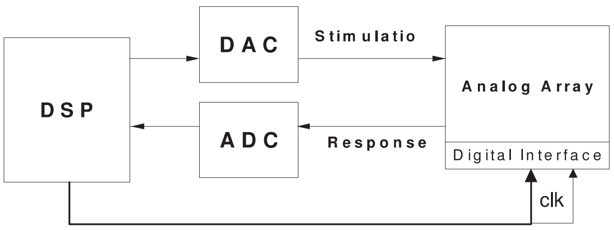

Fig. 2. The information flow between DSP and FPTA in JPL SABLE.

The main integrated components by the JPL **SABLES** are: the transistor-level **RH** - a JPL FPTA, and the **RM** - the evolutionary algorithm, implemented by a TI DSP acting as a controller for reconfiguration. The information flow and implementation of **JPL SABLES** is featured - see Fig. 2 - both by autonomy and speed in providing on-chip circuit reconfiguration: about 1000 circuit evaluations are performed per second. Another parameters of **JPL SABLES** performance are the followings: 1-2 orders of magnitude reduction in memory and about 4 orders of magnitude improvement in speed compared to systems evolving in simulations.

The final aim of **EC** techniques development and of their silicon implementation is to create architectures towards an artificial brain, a computer having its own ability of reasoning or decision making, but also being able of emergent functionality creation, or having the possibility of self-creation and evaluation of its own structure. The main types of **EHW** architectures with intrinsic **EC** logic elements are: *embryological* architectures; *emergent functionality* architectures; evolvable *fault tolerant* systems; *parallel evolvable architectures* (of Higuchi type) [1].

Other practical considerations regarding the concept of **EHW** architectures and hard- ware **GA** implementation, including the engineering compromise between the performances and the GA architectural complexity is to be found in [9]. *Gate Level Evolution* in **FPGA**s at the intrinsic level was applied in [10]. In the *Functional Level Evolution* the circuits are composed of high-level functional blocks such as adders and multiplier [11]. *Incremental Evolution* was applied in order to evolve circuits of high complexity. A *divide-and conquers* approach for the evolution of systems (also known under the name of "increased complexity evolution") is introduced in [12]. **EHW** A*rchitectures at Functional Level*. This aim is feasible on a specialized dedicated FPGA architecture proposed by Higuchi [13] and called **F^{2}PGA**. This architecture is relied on a network of *switch settings* that allow the basic cells inside the device to be connected as the case. A basic cell in this kind of **FPGA** is called ***Program**mable **F**loating **U**nit* (**PFU**) because of its availability of performing a large function variety: adding, subtracting, multiplication, cosine and sine using floating point numbers. A **F^{2}PGA** programming word is a variable length chromosome containing both the **PFU**'s function programming and the crossbar witch settings.

3. Evolutionary Design of Digital Circuits: Where Are Current Limits?

It is important to understand that *evolutionary circuit* design and *evolvable hardware* (**EHW**) are two different and distinct approaches [14]. *Evolutionary circuit design* performs the evolution (the design) of a single circuit. The aim is typically to design novel implementations that are better (in terms of area, speed, power consumption) than conventional deigns and/or to design circuits with additional features such as fault-tolerance, testability, polymorphic behavior, that are difficult to design by conventional methods. ***E**volvable **h**ardware* (**EHW**) involves an **EC** responsible for continual adaptation. **EHW** is applied to high-performance and adaptive systems in which the problem specification is unknown beforehand and can vary in time [15], [16].

The main issues in the evolutionary circuit design are: the bias in the design method; the chromosome size versus complexity of circuits; the fitness calculation as a bottleneck; the level of innovation [14].

It is important to explore the relation between the size of chromosome and the complexity of evolved circuits. Long chromosomes imply large search spaces that are difficult to search. Computing resources that are currently available determine the size of the search space that can be explored. This is known as the problem of *scalability of representation*. In case of evolving the combinational circuits, the evaluation time of a candidate circuit grows exponentially with the increasing number of inputs. Hence, the evaluation time becomes a main bottleneck of the evolutionary approach even if the circuit consists of a few components. This is the problem of *scalability of evaluation*. In order to reduce this time of evaluation, only a subset of all possible input vectors can be utilized. It is impossible to measure the level of innovation in an evolved circuit; the level of innovation does not depend on the approach utilized to evolve the circuit and on the complexity of the evolved circuits.

The *Gate-Level Evolution* can produce circuits of complexity up to about 100 gates. The *Application-specific Encoding* (computer & swapping medians or sorting networks in **FPGA**) can be used to obtain more complex circuits. The *Functional Level Approaches* produce circuits of an unlimited complexity and depends on the complexity of components used as building blocks. *Development Approaches* produce arbitrarily complex solutions using chromosomes of a short length. The advantage is that the area on a chip is minimized. The limitation is that the obtained solutions are featured by regularity; this causes redundancy. *Transistor Level* evolution produces only small digital circuits. These elementary circuits are implemented using a large range of unconventional approaches. The complexity of evolved circuit only partially depends on the size of the chromosomes. It mainly depends on the size of objects encoded in the chromosome or handled by instructions encoded in the chromosome (see the developmental approaches), Fig.3, after [14].

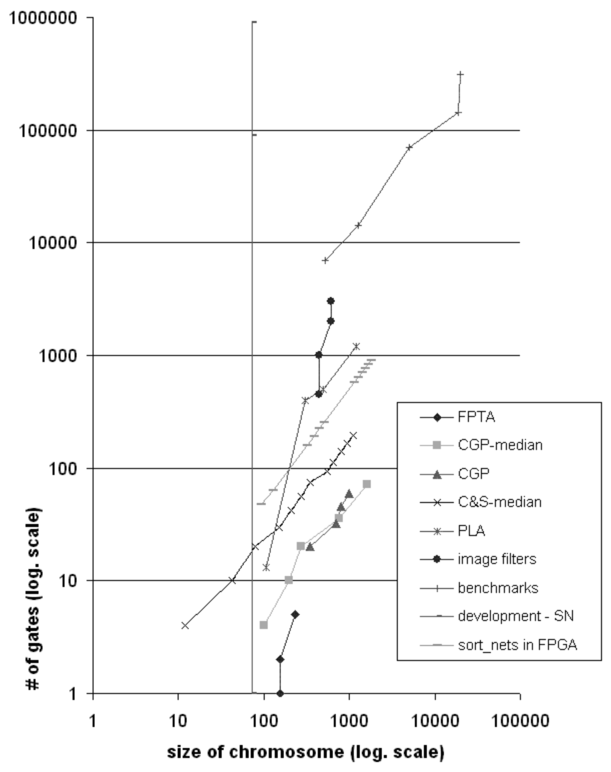

Fig. 3. The complexity of the evolved circuits versus the size of chromosomes.

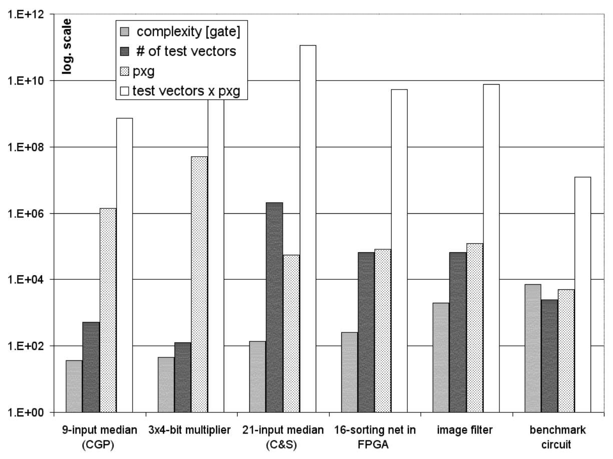

Fig. 4. Evolved circuits complexity versus the configuration bitstream.

Another relevant view is made by the comparison of computational effort considering circuits that have similar size of the configuration bitstream [14], see Fig. 4, where the chromosomes are of a size in the range of 512 -868. The legend of this table is as follows: *in* - the number of inputs; *out* - the number of outputs; *chromsize* - the size of chromosome; *gates* - the number of gates the circuit is composed; *testvect* - the number of test vectors used in the fitness function to evaluate the circuit; *popsize* - the population size; *gnrs* - the average number of generations to evolve the circuit. Some interpretations of different evolutionary design methods are as follows [14]:

- the evaluation of a 21-input median circuit is the most time consuming;
- *pxg = popsize x gnrs*, is a maximal parameter for CGP although the complexity of these gate-level circuits is not so high; surprisingly, pxg decreases with the complexity of evolved circuits;
- the *"time of evolution" = pxg x test-vect*; this parameter is very similar for almost all circuits. This is important, because the time of evolution indicates how much time/resources the designers are able to invest to the evolutionary design process.

4. Concluding Remarks

The most spectacular **EC** application in the **CI** framework is the (**EHW**). It opened a revolutionary eve in technology and in social life development by its radical implications on engineering design and automation. A dream of humanity became reality - the *systems adaptivity* was implemented (transferred) by **EHW** *from software to hardware*. A drastic time saving way from design to real world application of intelligent hardware is used, no *more difference* exists *between design* and *adaptation* concerning **EHW** based machines having a *behavioral computational intelligence*. *Electronics engineering was fundamentally changed* as a profession by using **EHW** custom design technologies instead of soldering based manufacturing.

A survey of the state of the art of evolutionary design of digital circuits was made. The investigation was mainly focused on level of complexity of the evolved circuits with respect to the size of the search space. The practical conclusion is that innovative results can be obtained by all mentioned approaches. There is no reason to prefer some approaches.

5. Annex

Field Programmable Analogue Circuits for Evolvable and Adaptive Hardware Field Programmable Analogue Arrays(FPAA).

The appearance of ***r****econfigurable* ***a****nalogue* ***a****rrays* (**FPAA**s) was crucial for the technological support required by companies involved in electronics research and development as well as in manufacturing. The analog reconfigurable hardware allows prevention or removal of essential fabrication mismatches and other refined technological problems by evolving circuits as the case. Analog reconfigurable hardware has actually a huge weight in **EHW** environment, see Fig. A.1, where the actual hardware platforms for **EHW** development and their authors are enumerated (Negoita and Stoica, 2004).

FPAA reconfigurability relies on switched capacitors, typical to the main providers as Pilkington, Motorola see - Motorola MPAA020 - or Anadigm (see Fig. A.2).

Zetex **TRAC** provides a ***T****otally* ***R****econfigurable* ***A****nalog* ***C****ircuit* structured in form of 20 cells, each being an op-amp with a small reconfigurable network. Such a cell performs any of the following analog operations: add, negate, subtract, multiply, pass, log, antilog, rectify, or basic inverting op-amp for use with external components.

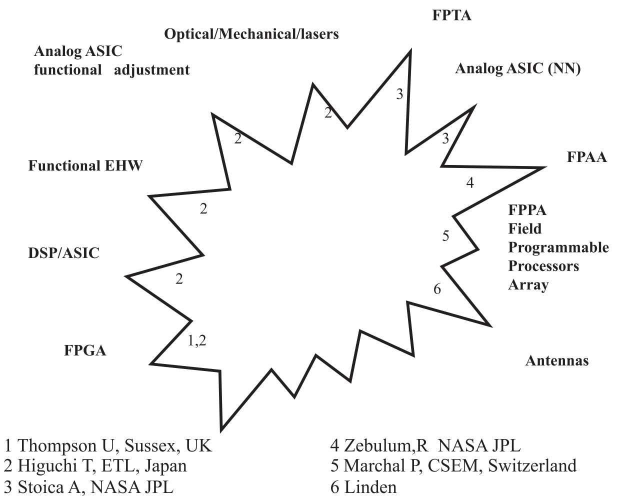

Fig A.1. The main hardware platforms for EHW development and their authors/providers.

Lattice ispPAC10 - see Fig. A.3 - contains four programmable analog modules and a programmable system. This circuit is an **EHW** analog reconfigurable hardware that can be configured to implement 2nd and 4th order active LP and BP filters in the 10 kHz-100 kHz range.

High complex architectures are featuring some analog reconfigurable hardware of the Anadigm **FPAA** family, as for example AN220E04.

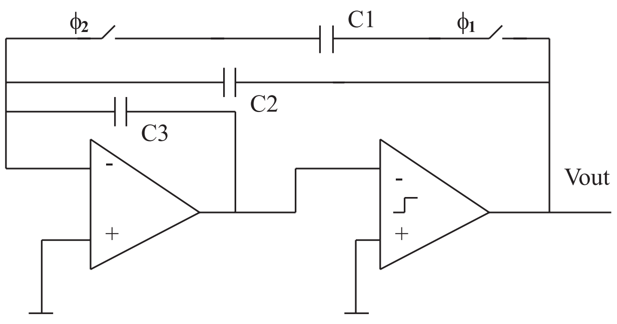

Fig. A.2 A typical FPAA switched capacitor configuration.

Field Programmable Transistor Arrays (FPTA) produced by NASA JPL.

The programmability of **FPAA**s is limited to only allow configuration around op-amp level. But the applications require also for many interesting circuit topologies to be evolved below the op-amp level. This application requirement led to another kind of programmable (reconfigurable) hardware relied on *evolution-oriented devices* that have some advantages over **FPAA**, namely: can reprogram many times, can understand what's inside and are featured by a flexible programmability. This is the so-called *custom made* **EHW***-oriented reconfigurable hardware*. Some briefly comments are to be made regarding the types of *custom made* **EHW***-oriented reconfigurable hardware* (Negoita and Stoica, 2004):

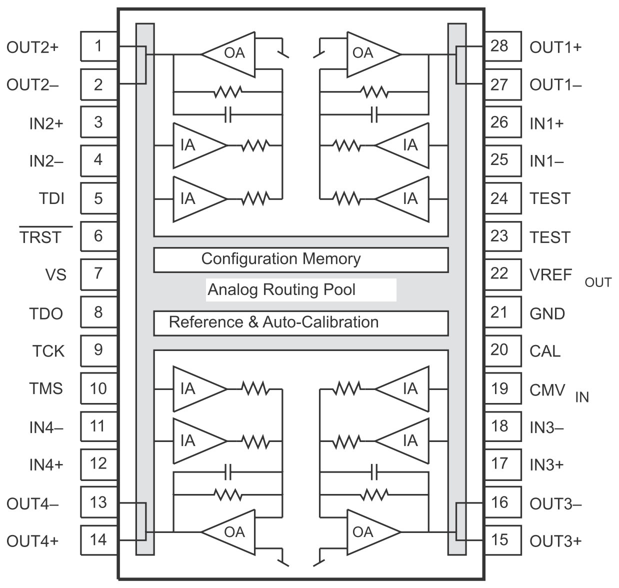

Fig.A.3 The block diagram of Lattice ispPAC10.

- **JPL PTA**s are chips of reconfigurable hardware at transistor level, both analogue and digital;

- **FPTA chip of Heidelberg University** is an array of 16x16 transistors, performing a programmability in connectivity and channel length;

- **JPL'98 FPTA-0** chip is a programmable transistor array cell with 24 programmable switches, a sufficient number for meaningful circuit topologies. All three terminals of a transistor cell are connected via switches to expansion terminals. The chromosomes give the value HIGH-LOW of the switches (not only ON-OFF).

- **JPL'2001 FPTA-2** chip is a second generation reconfigurable array chip, a programmable array of transistor array cells implementing an evolution-oriented reconfigurable architecture, featured by a NESW interconnection amongst 64 integrated cells (an 8x8 matrix of reconfigurable cells), each of the cells having 44 transistors. It is the first chip integrating reconfigurable processing circuitry with sensing: an array of 16x8 photodetectors distributed within the cells is also integrated on this chip. **FPTA-2** chip is able of receiving 96 analog/digital inputs and provides 64 analog/digital outputs; it is the first **FPMA** (***F****ield* ***P****rogrammable* ***M****ixed-signal* ***A****rray*).

- **PAMA** (Programmable Analog Multiplexer Array) chip is an analog platform based on analog multi-plexers/demultiplexers (Santini, Zebulum *et al.* 2001). The multiplexers/demultiplexers are fixed elements that perform the interconnections of the different discrete electronic devices that can be plugged into the board. The platform performs intrinsic evolution of analog circuits through a **RM** - represented by a **GA**. Each gene configures the select input signals of a particular analog multiplexer. A multifunction I/O board is connected to the PC bus to perform the A/D conversion and the chromosome download. The control bit strings (**GA** chromosomes) are downloaded to the **RH**. The circuit evaluation runs as follows: circuit responses are compared against specifications of a desired response, this comparison being followed by a circuits ranking based on how close they come to satisfying the target.

PAMA provides a practical environment to evolve generic analogue circuits based on discrete components, without the need of simulators. This is in fact a very useful prototype platform allowing a larg number of component terminals and evolution of a great number of circuits. The circuit evaluation speed is featured by 6 minutes to evolve a certain circuit.

Other strong features is that **PAMA** platform provides protection against illegal configuration that may damage electronic components and confers the possibility to analyse circuits which have been evolved, due to access to individual circuit elements with test equipment.

An innovative powerful ***P**rogrammable **S**ystem-**o**n-**C**hip* (**PSoC**) is the family **PSoC**™ CY8C25122/CY8C26233/CY8C26443/CY8C26643 of configurable mixed-signal arrays provided by Cypress Connects.

PSoC is a chip, reconfigurable device that replaces the components of a multiple **MCU**-based system components. A chip of this kind includes configurable analog and digital peripheral blocks, a fast **CPU**, Flash program memory and **SRAM** data memory in a range of convenient pin-outs and memory sizes.

ACKNOWLEDGMENTS
The support of the **EHW Group at NASA JPL** was crucial for tackling research objectives surveyed in this work that was suggested by the **KES International Advisory Board**.

AUTHORS
Mircea Gh. Negoita* - KES International, 2nd Floor, 145-157 St John Street, London, EC1V4PY, United Kingdom. E-mail: m.negoita@hotmail.com.
Lukas Sekanina - Faculty of Information Technology, Brno University of Technology, Bozetechova 2, 612 66 Brno, Czech Republic . E-mail: sekanina@fit.vutbr.cz.
Adrian Stoica - NASA Jet Propulsion Laboratory (JPL), USA, 4800 Oak Grove Drive MS 303-300, Pasadena, CA 91109. E-mail: adrian.stoica@jpl.nasa.gov.
* Corresponding author

References

[1] Negoita Gh. M. (ed.), *What is Evolvable Hardware, a Lecture*, KES 2008 Post-Doc School on EHW and AHS, Zagreb, Croatia, September 2008.

[2] Torresen J., "Evolvable Hardware The Coming Hardware Design Method?", In: Kasabov, (ed.), *Neuro-fuzzy Tools and Techniques*, Springer: Heidelberg, 1997.

[3] Negoita M.Gh., Neagu C.D., Palade V., Computational Intelligence. In: *Engineering of Hybrid Systems, Springer;* Heidelberg, 2004.

[4] Negoita M.Gh., "A Modern Solution for the Technology of Preventing and Alarm Systems: EC in EHW Implementation". In: *Proceedings of The Second Dunav Preving International Conference on Preventing and Alarm Systems*, Belgrade, 1997, pp. 201-209.

[5] Negoita M.Gh., Stoica A., "Evolvable Hardware (EHW) in the Framework of Computational Intelligence-Implications on Engineering Design and Intelligence of Autonomous Systems". In: *Tutorial Course*, *KBCS 2004*, Hyderabad, India, December 2004, pp. 4-52.

[6] Stoica A., Radu A., "Adaptive and Evolvable Hardware A Multifaceted Analysis". In: Arslan, T., Stoica, A., *et al.* (eds.) *Proceedings of 2007 NASA/ESA Conference on Adaptive Hardware and Systems*, Edinburgh, UK, 5th-8th August, 2007, pp. 486-496.

[7] Stoica A., Zebulum R.S., *et al.*, "Evolving Circuits in Seconds: Experiments with a Stand-Alone Board-Level Evolvable System". In: *Proceedings of The 2002 NASA/DoD Conference on EHW*, Alexandria, Virginia, 15th-18th July, 2002, pp. 67-74.

[8] Zebulum R.S., Keymeulen D., Duong V., Guo X., Ferguson M.I., Stoica A., "Experimental Results in Evolutionary Fault-Recovery for Field Programmable Analog Devices". In: *Proceedings of The 2003 NASA/DoD Conference on Evolvable Hardware*, Chicago, Illinois, 9th-11th July, 2003, pp. 182-188.

[9] Mihaila D., Fagarasan F., Negoita M.Gh., Architectural Implications of Genetic Algorithms Complexity in Evolvable Hardware Implementation. In: *Proceedings of the European Congress EUFIT 1996*, September, Aachen, Germany, vol. 1, 1996, pp. 400-404.

[10] Thompson A., Layzell P., Zebulum S., "Exploration in Design Space: Unconventional Electronics Design Through Artificial Evolution", *IEEE Transactions on Evolutionary Computation*, vol. 3, vol. 3, 1999, pp. 167-196.

[11] Murakawa M., *et al.*, "Evolvable Hardware at Functional Level". In: *Ebeling*, W., Rechenberg, I., Voigt, H.-M., Schwefel, H.-P. (eds.) PPSN 1996. LNCS, vol. 1141, 1996, Springer: Heidelberg (1996) pp. 62-71.

[12] Torresen J., "A Scalable Approach to Evolvable Hardware", *Genetic Programming and Evolvasble Machines*, vol. 3, vol. 3, 2002, pp. 259-282.

[13] Higuchi T., "Evolvable Hardware with Genetic Learning". In: *Proceedings of IEEE International Symposium on Circuits and Systems*, ISCAS 1996, Atlanta, USA, 13th May 1996.

[14] Sekanina L., "Evolutionary Design of Digital Circuits: Where are Current Limits?" In: Stoica A., Arslan T. (eds.) *Proceedings of First NASA/ESA Conference on Adaptive Hardware and Systems*, Istanbul, Turkey, 15th-18th June, 2006, , pp. 171-177.

[15] Sekanina L., *Evolvable Components: From Theory to Hardware Implementation*, Springer: Heidelberg, 2004.

[16] Torresen J., Bake W.J., Sekanina L., "Recognizing Speed Limit Sign Numbers by EHW". In: Raidl G.R., Cagnoni S., Branke J., Corne D.W., Drechsler R., Jin Y., Johnson C.G., Machado P., Marchiori E., Rothlauf F., Smith G.D., Squillero G. (eds.) *EvoWorkshops 2004*, LNCS, vol. 3005, 2004, Springer, Heidelberg, pp. 682-691.