Evolution of analog circuits on Field Programmable Transistor Arrays

A. Stoica, D. Keymeulen, R. Zebulum,
A. Thakoor, T. Daud, G. Klimeck, Y. Jin, R. Tawel and V. Duong
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
adrian.stoica@jpl.nasa.gov

Abstract

Evolvable Hardware (EHW) refers to HW design and self-reconfiguration using evolutionary/genetic mechanisms. The paper presents an overview of some key concepts of EHW, describing also a set of selected applications. A fine-grained Field Programmable Transistor Array (FPTA) architecture for reconfigurable hardware is presented as an example of an initial effort toward evolution-oriented devices. Evolutionary experiments in simulations and with a FPTA chip in-the-loop demonstrate automatic synthesis of electronic circuits. Unconventional circuits, for which there are no textbook design guidelines, are particularly appealing to evolvable hardware. To illustrate this situation, one demonstrates here the evolution of circuits implementing parametrical connectives for fuzzy logics. In addition to synthesizing circuits for new functions, evolvable hardware can be used to preserve existing functions and achieve fault-tolerance, determining circuit configurations that circumvent the faults. In addition, we illustrate with an example how evolution can recover functionality lost due to an increase in temperature. In the particular case of space applications, these characteristics are extremely important for enabling spacecraft to survive harsh environments and to have long life.

1. Introduction

The application of evolution-inspired formalisms to hardware design and self-configuration leads to the concept of Evolvable Hardware (EHW). In the narrow sense, EHW refers to self-reconfiguration of electronic hardware by evolutionary/genetic reconfiguration mechanisms. In a broader sense, EHW refers to various forms of hardware from sensors and antennas to complete evolvable space systems that could adapt to changing environments and, moreover, increase their performance during their operational lifetime.

This paper presents the concept of evolution oriented devices and describes an effort toward building these devices and an evolvable system on a chip. A Field Programmable Transistor Array architecture is used as the experimental platform for evolutionary experiments. The platform is flexible and supports implementation of both analog and digital circuits. Previous works using the FPTA [1], [2] have illustrated the implementation of several conventional building blocks for electronic circuits such as logical gates, transconductance amplifiers, filters, Gaussian neuron. This paper illustrates the automatic design of the rather more unconventional circuits for combinatorial fuzzy logics and the fault-tolerant properties of the FPTA device.

The paper is organized as follows: Section 2 presents the components of an evolvable hardware system, providing a perspective on the evolution of the field. Section 3 surveys some important evolutionary experiments and applications of evolvable hardware. Section 4 presents an evolution-oriented architecture based on the concept of the FPTA. Section 5 illustrates how the FPTA can be used to evolve reconfigurable circuits for combinatorial fuzzy logic. Section 6 presents considerations related to the application of EHW in space systems and section 7 presents the architecture of an enhanced FPTA. Section 8 concludes this work.

2. Evolvable Hardware: From Roots to Buds

The concept of evolvable hardware was born partially inspired by the search/optimization/adaptation mechanisms and partially by the availability of reconfigurable devices such as Programmable Gate Array (PLA) [3] and later Field Programmable Gate Arrays (FPGA) [4]. Circuits can be evolved by reconfiguring programmable devices (which is called *intrinsic* EHW) or evolving software models – descriptions of the electronic HW (referred to as *extrinsic* EHW). Currently, evolutionary platforms are board level. These include programmable hardware that is reconfigured under the control of configuration bits determined by evolutionary algorithms running in software. It is likely that in the next 1-3 years, a number of platforms will integrate the reconfigurable hardware and the reconfiguration mechanism in an evolvable system-on-a-chip (SOC) solution. Finally, the path leads to the Intellectual Property

(IP) level and EHW solutions will become an integrated component in a variety of systems that will thus have an evolvable feature. This is illustrated in Figure 1.

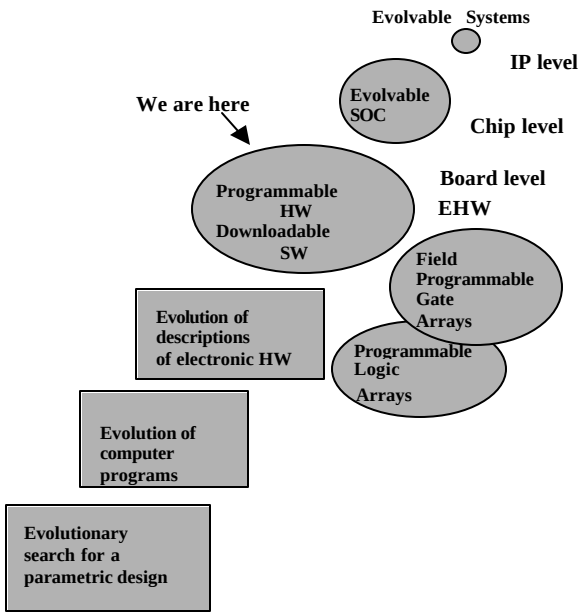


Figure 1 Evolutionary path for the evolvable hardware field: from design optimization to hardware IP cores for evolvable systems.

The main steps for evolutionary design of electronic circuits are illustrated in Figure 2. Each candidate circuit design is associated with a "genetic code" or chromosome. The simplest representation of a chromosome is a binary string, a succession of 0s and 1s that encode a circuit. The first step of evolutionary synthesis is to generate a random population of chromosomes. In extrinsic evolution the chromosomes are then converted into a model that gets simulated (e.g. by a circuit simulator such as SPICE) and produces responses that are compared against specifications.

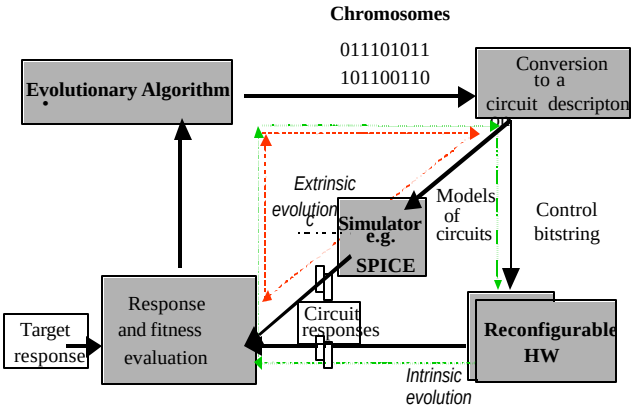


Figure 2 Evolutionary synthesis of electronic circuits

A more detailed description of the circuit representation will be given in section 4 of this article.

A solution determined by extrinsic evolution may eventually be downloaded or become blueprint for hardware. In intrinsic evolution the chromosomes are converted into control bitstrings, which are downloaded to program the reconfigurable device. The configuration bitstring determines the functionality of the cells of the programmable device and the interconnection pattern between cells. Circuit responses are compared against specifications of a target response and individuals are ranked based on how close they come to satisfying it. Preparation for a new iteration loop involves generation of a new population of individuals from the pool of the best individuals in the previous generation. Here, some individuals are taken as they were and some are modified by genetic operators, such as crossover and mutation. The process is repeated for a number of generations, resulting in increasingly better individuals. The process is usually ended after a given number of generations, or when the closeness to the target response has been reached. In practice, one or several solutions may be found among the individuals of the last generation.

3. Evolutionary Experiments

A variety of circuits have been synthesized through evolution. For example, Koza used Genetic Programming (GP) to grow an “embryonic” circuit to one that satisfies desired requirements [5]. This technique was used to evolve a variety of circuits, ranging from filters to controllers. Some of Koza’s evolved designs rediscover solutions that at some point in time were patented, illustrating thus the power of the GP to obtain solutions that normally require an intelligent/innovative human. Some researchers succeeded in hardware evolution. For example, evolution in hardware was demonstrated by Thompson [4], who used an FPGA as the programmable device, and a Genetic Algorithm (GA) as the evolutionary mechanism. Higuchi and colleagues in Japan have used evolvable hardware for a variety of applications, the most recent including the use of EHW to increase the yield of specific chips [6]. In particular, this technique is applicable to chips with very tight requirements and which are fabricated in a relatively new technology which still has poor yield. For example, this method was used to automatically tune (and thus bring to specifications and pass the tests increasing the yield) manufacturing process parameters for cellular phones filter chips [6] and to compensate for clock skew of fast processors [7]. More details on current work in evolvable hardware can be found in [8] and [9].

4. Toward Evolution-Oriented Reconfigurable Architectures

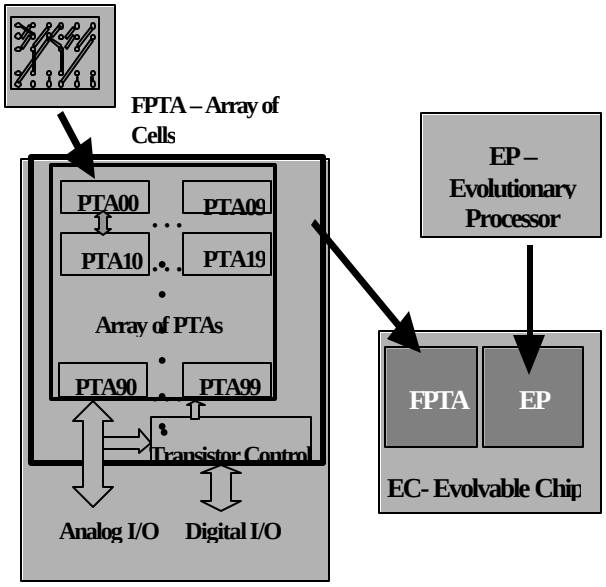
Current efforts toward hardware evolution have been limited to simple circuits. In particular for analog circuits, this limitation comes from a lack of appropriate reconfigurable analog devices to support the search. This precludes searches directly in hardware and requires evolving on hardware models. Such models require evaluation with circuit simulators such as SPICE; the simulators need to solve differential equations and, for anything beyond simple circuits, they require too much time for practical searches of millions of circuit solutions. A hardware implementation offers a big advantage in evaluation time for a circuit; the time for evaluation is determined by the goal function. For example, considering an A/D converter operating at a 100 kHz sampling rate, its electronic response is available within 1mili-second (100 samples data), compared to (an over-optimistic) 1 second on a fast computer running SPICE; this advantage increases with the complexity of the circuits. In this case the 10^3 speedup would allow evaluations of populations of millions of individuals in seconds instead of days.

Increasingly more complex Field Programmable Devices offer powerful solutions to applications in digital signal processing, programmable interfaces, filtering, etc. However, for efficiency in EHW applications, future devices would benefit from implementing evolution-oriented reconfigurable architectures (EORA). One of the most important features for EORA relates to the *granularity* of the programmable chip. Field Programmable Analog Arrays (FPAAs) offer only coarse granularity which is a clear limitation; FPGAs are offered both in versions with coarse-grained and fine-grained architectures (going to gate level as the lowest level of granularity). From the EHW perspective, it is interesting to have *programmable granularity*, allowing the sampling of novel architectures together with the possibility of implementing standard ones. The optimal choice of elementary block type and granularity is task dependent. At least for experimental work in EHW, it appears a good choice to build reconfigurable hardware based on elements of the lowest level of granularity. Virtual higher-level building blocks can be considered by imposing programming constraints. An example of this would entail forcing groups of elementary cells to act as a whole (e.g. certain parts of their configuration bitstrings with the interconnections for the N transistors implementing a NAND would be frozen). Ideally, the "virtual blocks" for evolution should be automatically defined/clustered during evolution. EORA should be also *transparent architectures*, allowing the analysis and simulation of the evolved circuits. They should also be robust enough not to be *damaged* by any configuration existent in the search space, potentially sampled by evolution. Finally EORA should allow evolution of both analog and digital functions.

An EORA SOC architecture is suggested in Figure 3. The main components are a Field Programmable Transistor Array and a Genetic Processor. The idea of a field programmable transistor array was introduced in [10] as a first step toward EORA. The FPTA is a concept design for hardware reconfigurable at transistor level. As both analog and digital CMOS circuits ultimately rely on functions implemented with transistors, the FPTA appears as a versatile platform for the synthesis of both analog and digital (and mixed-signal) circuits. The architecture is cellular, and has similarities with other cellular architectures as encountered in FPGAs (e.g. Xilinx X6200 family) or cellular neural networks. One key distinguishing characteristic relates to the definition of the elementary cell. The architecture is largely a "sea of transistors" with interconnections implemented by other transistors acting as signal passing devices (gray-level switches).

Figure 4 illustrates an FPTA cell consisting of 8 transistors and 24 programmable switches. The status of the switches (ON or OFF) determines a circuit topology and consequently a specific response. Thus, the topology can be considered as a function of switch states, and can be represented by a binary sequence, such as "1011…", where, by convention one can assign 1 to a switch turned ON and 0 to a switch turned OFF. Programming the switches ON and OFF defines a circuit for which the effects of non-zero, finite impedance of the switches can be neglected in the first approximation (for low frequency circuits).

Figure 3 EORA system on a chip.



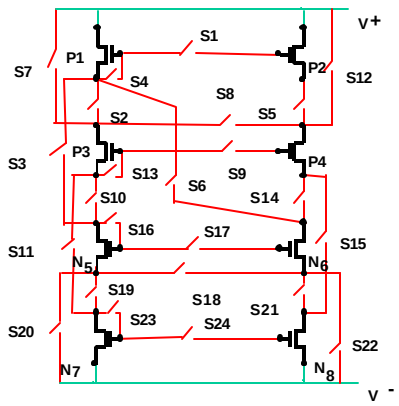


Figure 4. Cell of the Field Programmable Transistor Array

5. Evolving reconfigurable circuits for fuzzy logics

This section illustrates the evolutionary design of infinitesimal multi-valued logic circuits, more precisely circuits for fuzzy logics. The objective is to determine circuit implementations for conjunctions and disjunctions for fuzzy logics. In such logics, conjunction and disjunction are usually interpreted by a *T-norm* and by its dual *T-conorm* (*S-norm*) respectively. A function **T**: [0,1] x [0,1] => [0,1] is called a triangular norm (T-norm for short) if it satisfies the following conditions:

?? associativity (**T**(x,**T**(y,z)) = **T**(**T**(x,y),z)),
?? commutativity (**T**(x,y) = **T**(y,x)),
?? monotonicity (**T**(x,y) = **T**(x,z), whenever y = z), and
?? boundary condition (**T**(x,1) = x).

A function **S**: [0,1] x [0,1] => [0,1] is called a triangular conorm (T-conorm or S-norm for short) if it satisfies the conditions of associativity, commutativity, monotonicity, and the boundary condition **S**(x,0) = x. **S** and **T** are corresponding (or pairs) if they comply with De Morgan's laws. Frank's parametric T-norms and T-conorms (also refered to as fundamental T-norms/conorms in [11]) were the selected choice for modeling the logical connectives. The family of Frank T-norms is given by

$$T_s(x,y) = \begin{cases} \mathrm{MIN}(x,y) & \text{if } (s = 0) \\ x \cdot y & \text{if } (s = 1) \\ \log_s\left[1 + \dfrac{(s^x - 1).(s^y - 1)}{s - 1}\right] & \text{if } (0 < s < ?), s ? 1 \\ \mathrm{MAX}(0, x + y - 1) & \text{if } (s = ?) \end{cases} \quad (1)$$

The family of Frank S-norms is given by

$$S^s(x,y) = \begin{cases} \mathrm{MAX}(x,y) & \text{if } (s = 0) \\ x + y - x.y & \text{if } (s = 1) \\ 1 - \log_s\left[1 + \dfrac{(s^{1-x} - 1).(s^{1-y} - 1)}{s - 1}\right] & \text{if } ((0 < s < ?), s ? 1) \\ \mathrm{MIN}(1, x + y) & \text{if } (s = ?) \end{cases} \quad (2)$$

Electronic circuits implementing the above equations can be used in implementations of fuzzy logic computations or in implementing fuzzy S-T neurons. One interesting application made possible in this implementation is the selection of the most appropriate s-parameter for the application at hand. Examples of the influence of various T-norms and S-norms in fuzzy control and automated reasoning applications can be found in [12] and [13], and for learning in fuzzy neurons in [14].

The following preliminary results illustrate the possibility of evolving circuits that implement T and S for various values of the parameter *s*. The circuits were powered at 5V and the signal excursion was chosen between 1V (for logical level "0") and 4V (for logical level "1"). Intermediary values were in linear correspondence, i.e. 2.5V corresponds to logic level 0.5, etc. The experiments were performed both in software (SPICE simulations) and in hardware using 2 FPTA cells. The experiments used a population size of 128 individuals, and were performed for 400 generations (with uniform crossover, 70% crossover rate, 4% mutation rate, tournament selection).

Figures 5, 6, and 7 show the response of circuits targeting the implementation of fundamental T-norms for s=0, s=1, and s=100 respectively. The diamond symbol (?) marks points of simulated/measured response of the evolved circuit, while the cross symbol (+) marks the points of an ideal/target response for the given inputs. The output (T) is mapped on the vertical axis; values on axis are in Volts. The circuit for T-norm with s=100 is shown mapped on two FPTA cells in Figure 8. Figure 9 shows the response of the circuit implementing the fundamental S-norm for s=100. Figure 10 shows the diagonal cut for the same S-norm. In these examples, the Mean Absolute Percentage Error (MAPE) to the target response ranged from 3.6% to a maximum of 9%.

All these responses relate to circuits evolved in software. For comparison, the response of a circuit evolved in hardware (for s=1) is shown in Figure 11. In this case we limited the voltage levels to the range of 2V to 3V. The MAPE to the correct solution stayed in 3.72%. Two FPTA cells were used in this hardware experiment.

The convergence toward the solution for the extrinsic experiments can be seen in Figure 12, where a function of

the error of the best individual is plotted across the number of generations.

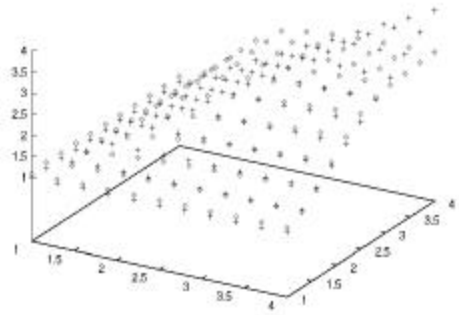


Figure 5 Simulated response of a circuit implementing the fundamental T-norm for s=0 (?). Target characteristic shown with (+). x,y axis are for inputs, z (vertical) is the output, T. Axes are in Volts.

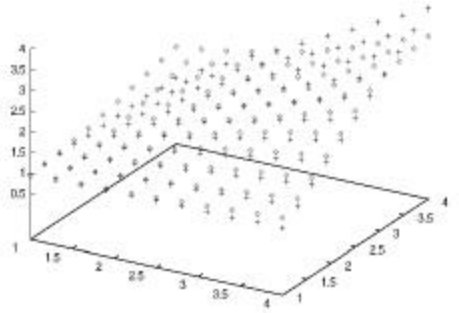


Figure 6 Response of a circuit implementing the fundamental T-norm for s=1 (?). Target characteristic shown with (+).

Finally, another extrinsic experiment was performed, in which we allowed parametric optimization, i.e, the optimization of the width and length (W and L) of the transistor channels after the topology evolution. Figure 13 shows the result for a T-norm with s = 1. Given a set of nine different W/L ratios for each MOS transistor, the GA searches for the best sizing for an evolved circuit topology achieved beforehand. The performance is improved compared to the one without parametric optimization (Figure 6). Both circuits achieved a MAPE around 8%, but the one with parametric optimization presented an integrated squared error around 3, against a value around 8 for the other circuit. The reason is that the optimized circuit reduced the high approximation errors observed in the extremities of the response shown in Figure 6.

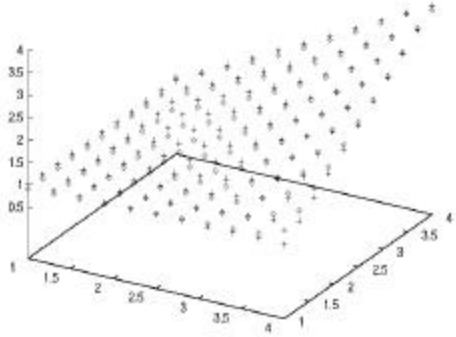


Figure 7 Response of a circuit implementing the fundamental T-norm for s=100 (?). Target characteristic shown with (+).

Another way to increase the approximation power is to allow more resources, e.g. allow resources from more than 2 cells. This is similar to increasing the approximation power of neural networks when extra neurons are added.

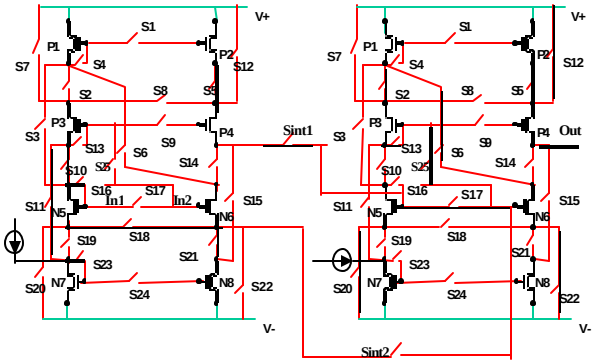


Figure 8 Evolved circuit implementing the fundamental T-norm for s=100 (with the response in Figure 7).

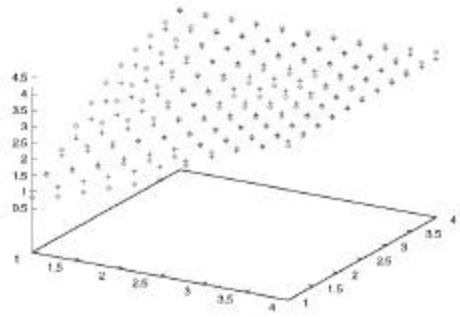


Figure 9 Response of a circuit implementing the fundamental S-norm for s=100 (?). Target characteristic shown with (+).

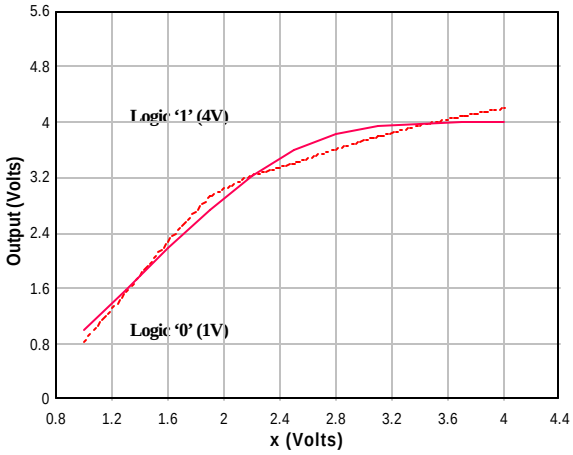


Figure 10 Diagonal cut for the response in Figure 10. Circuit implementing the fundamental S-norm for s=100. Target characteristic shown in full line.

These results are preliminary and are presented mainly to illustrate some aspects of the application of EHW to the synthesis of electronic circuits implementing combinatorial fuzzy logic functions. No comparison with any state-of-the-art design tools is made, and of course, the performance of (computer-assisted) human solutions could exceed the performance of the totally automated solutions illustrated here. However, to the authors' best knowledge, complete automated design for most of the circuits presented here is not available in any other tool.

We can, nonetheless, pick the T-norm with *s* equal to 1 as an example for comparison with human designs. This particular circuit performs analog multiplication, for which we can find many circuits in the literature [15]. Analog multipliers using bipolar technology are often made up of 6 bipolar transistors (excluding biasing circuitry), whereas typical CMOS multipliers consist of 19 MOS transistors (four stages). The hardware evolved multiplier presented in this section uses a total of 16 MOS transistors (two cells), being then competitive with the CMOS human design in terms of amount of hardware needed.

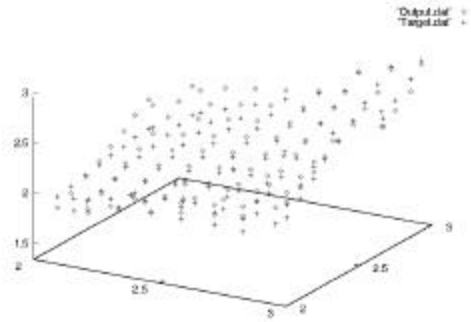


Figure 11 Measured response of a hardware-evolved circuit implementing the fundamental T-norm for s=1(?). Target characteristic shown with (+).

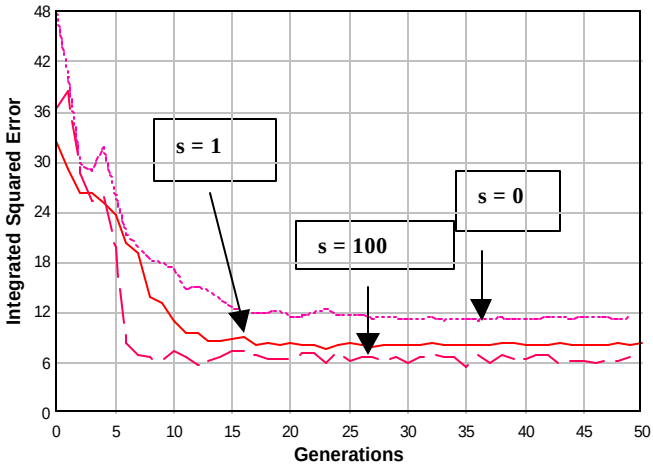


Figure 12 Decreasing error between best individual in each generation and target circuit, for the three software evolved circuits, with s=0, s=1, s=100.

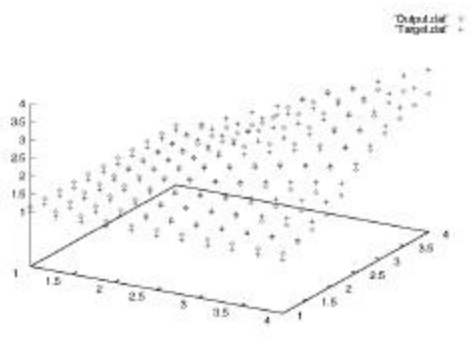


Figure 13 Parametric optimization experiments: Response of a circuit implementing the fundamental T-norm for s=1 (?). Target characteristic shown with (+).

6. Toward Evolvable Space Systems

EHW can bring two key benefits to spacecraft survivability. Firstly, EHW can help preserving existing functions, in conditions where hardware is subject to faults, aging, temperature drifts and radiation, etc. The environmental conditions, in particular the extreme temperatures and radiation effects can have catastrophic impacts on the spacecraft. Interstellar missions or extended missions to other planets in our solar system, with lifetimes in excess of 100 years, are great challenges on the on-board electronics. Secondly, new functions can be generated when needed (more precisely, new hardware configurations can be synthesized to provide required functionality). Figure 14 illustrates these ideas.

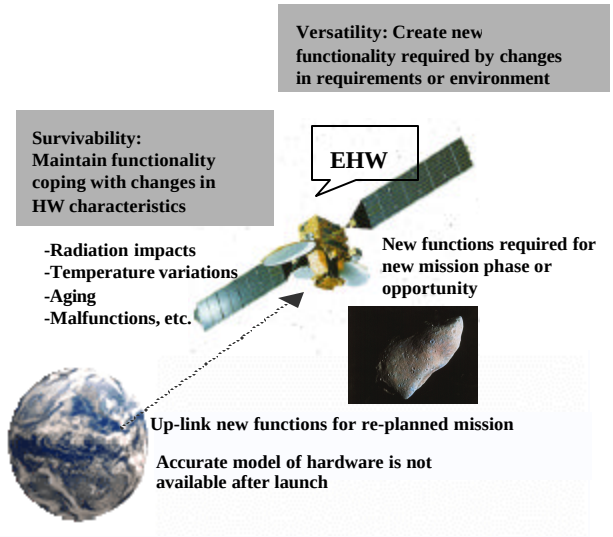


Figure 14 EHW can contribute to increase spacecraft survivability and flexibility.

Previous sections of this paper illustrated how EHW can be used to automatically synthesize circuits implementing new functions. This section summarizes a fault-tolerance experiment presented in detail in [16]. The experiment shows how EHW can recover functionality after being lost due to faults, by finding new circuit configurations that circumvent the faults. The experiment targeted the evolution of circuits implementing a digital XNOR and an analog multiplier response.

A certain quality threshold was set. When the performance decreased below the threshold (e.g. when a fault was injected), the evolution process restarted the search for a new circuit configuration, taking into account the previous circuit configurations in the population. Faults were injected by disconnecting external wires between FPTAs. At that time, a lowering of performance (but not a complete failure) was observed. The reason for the graceful degradation is that the population of circuits obtained by the evolution process contains mutants insensitive to faults. When the fault was injected, the GA restarted with the population of its last run, which included the currently affected by fault and some of its mutants. While starting with a random population took about the same time as finding a solution in the first place, starting with the last available population led to recovery in about 1/3 of the time while the circuit performance recovered to 90%.

Another potential area for EHW refers to the field of extreme temperature electronics. We present a preliminary study on the effect of changing the operating temperature for evolved circuits. We choose the evolutionary synthesis of a computational circuit presenting a DC transfer function in the shape of a Gaussian. Figure 15 depicts the change in the DC response for a circuit evolved at 27° Celsius. It can be verified from this figure that the effect of increasing temperature is to attenuate the Gaussian amplitude (for this particular circuit).

The circuit behavior can be recovered if we start a new evolutionary process, this time setting the PSPICE simulator temperature to a higher value, as it would be if the circuit operated in an extreme environment with high temperature. Figure 16 shows the response of an evolved circuit setting the temperature to 150^oC. It can be seen that evolution produced a new circuit that recovered from the amplitude attenuation observed for the other circuit.

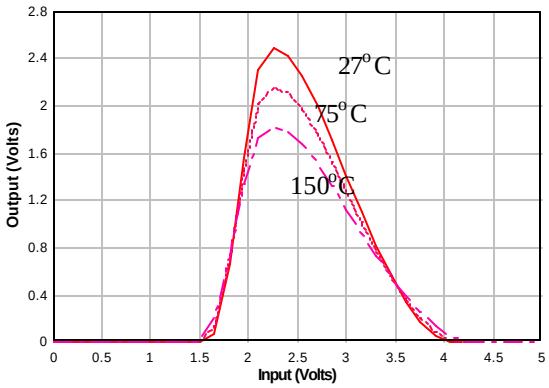


Figure 15 – Effect of increasing the temperature for an evolved circuit.

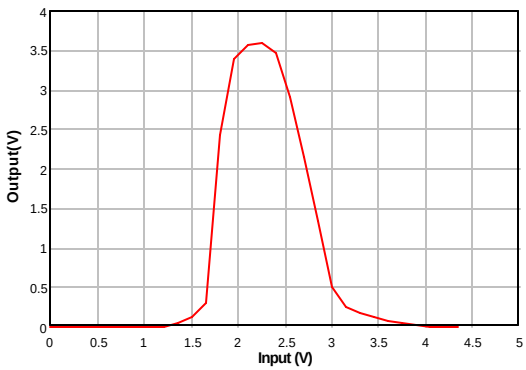


Figure 16 – Gaussian circuit evolved for 150°C.

The evolutionary approach for recovering the circuit behavior in extreme temperatures is very promising for space applications [17].

Evolution of space electronics can be seen as a first step toward evolvable space systems. Evolvable hardware can be extended to include on-board sensors, antennas, mechanical and optical subsystem reconfigurable flight hardware. This has the potential to largely enhance the capabilities of future space systems.

7. New FPTA Chip

A new version of the FPTA chip is currently being designed. Figure 17 shows a block diagram of the future chip, which is organized as a 6x6 matrix of cells. In the new design, the cells are divided into three categories according to their relative position in the array: Boundary Cells (B); Intermediate Cells (I) and Central Cells (C). These three cell categories are similar to the topology depicted in Figure 4.

There are 20 boundary cells, each of which receives one external input. A total of 20 external inputs (I0 to I19) can be applied to the chip via 20 pins. These inputs will be buffered and then applied to the boundary cells. There are 12 intermediate cells whose inputs are connected to the boundary cells outputs. Finally, there are four central cells for which a more flexible interconnection pattern is allowed (thick lines in the figure).

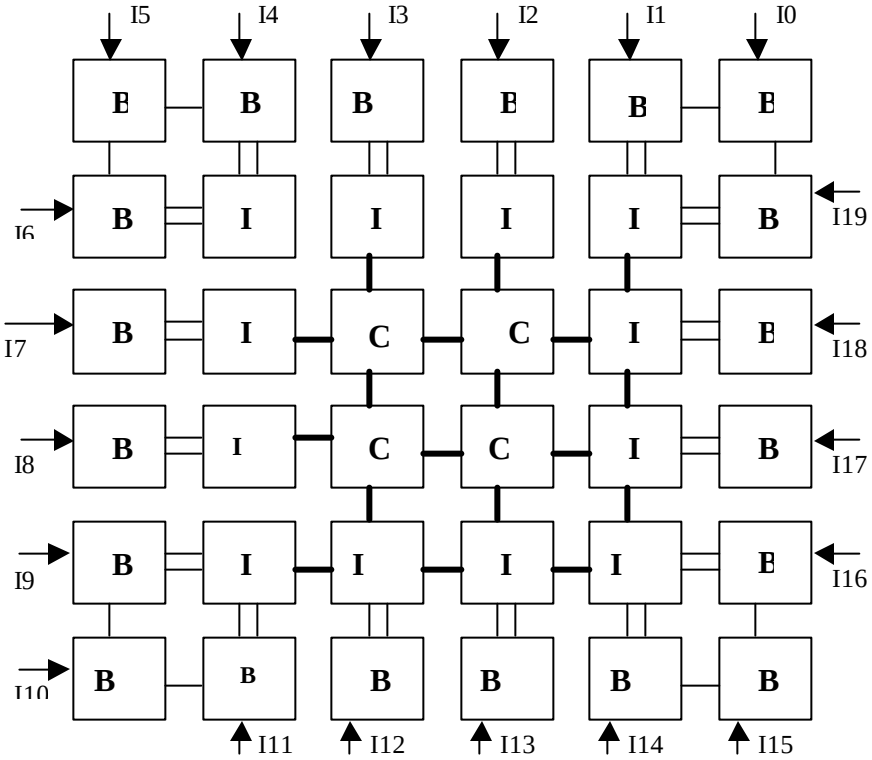


Figure 17 – Block diagram of the array of reconfigurable cells for the new FPTA

Each central cell can connect to its West, East, Central and North neighbors through analog multiplexers (not shown in the figure). We can therefore observe that the density of interconnections increase as we go towards the center of the cell.

The output of each of the 36 cells will be multiplexed and connected to 10 output buffers that deliver the chip outputs.

Finally, in order to allow some degree of parametric optimization, different cells will present different W/L ratios for their MOS transistors.

Another important feature of the central cells is the fact that they will be the only cells in the chip with capacitance resources. On chip capacitances will embed the new FPTA with more flexibility for filtering applications. Figure 18 displays the schematic of one central cell. According to the schematic of this figure, four capacitors are connected between the drain and the gate of transistors 3, 4, 5 and 6 of the cell. This positioning allows better exploration of the Miller effect, which results in reduced sized capacitors. In addition, one programmable capacitance will also be connected to the input. A programmable capacitance is a parallel array of capacitances (about 3 of them) in series with switches. By programming the switches, we can set the overall capacitor value. This configuration was validated in simulated experiments for the evolution of bandpass filters

8. Conclusion

This paper presented some highlights in the history of the field of evolvable hardware and a possible path for its evolution in the future. It presented an effort of building evolution-oriented devices and demonstrated how electronic circuits can be automatically synthesized, on-the-chip, to produce a desired functionality. It illustrated the aspects of using evolvable hardware for the design of unconventional circuits such as combinatorial circuits for fuzzy logics. It addressed the benefits evolvable hardware may bring in flexibility and survivability of future space hardware, by showing that evolution can circumvent faults and recover functionality lost due to an increase in temperature.

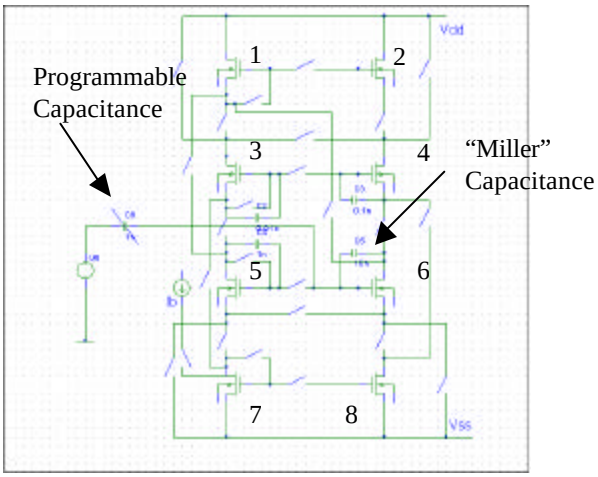


Figure 18 – Schematic of the central cells.

Acknowledgements

The work described in this paper was performed at the Center for Integrated Space Microsystems, Jet Propulsion Laboratory, California Institute of Technology and was sponsored by the National Aeronautics and Space Administration and the Defense Advanced Research Projects Agency.

References

[1] A. Stoica, D. Keymeulen, R. Tawel, C. Lazaro and Wei-te Li. "Evolutionary experiments with a fine-grained reconfigurable architecture for analog and digital CMOS circuits." In [9], 77-84

[2] R. Zebulum, A. Stoica and D. Keymeulen, "A flexible model of a CMOS field programmable transistor array targeted for hardware evolution", Third Int. Conference on Evolvable Systems: From Biology to Hardware (ICES2000), Edinburgh, April 17-19, 2000, 274-283.

[3] T. Higuchi, T. Niwa, T. Tanaka, H. Iba, H. de Garis, and T. Furuya, " Evolving hardware with genetic learning: A first step towards building a Darwin machine". *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*. In Meyer, Jean-Arcady, Roitblat, Herbert L. and Wilson, Stewart W. (editors). pp 417 - 424. 1993. Cambridge, MA: The MIT Press.

[4] A. Thompson, "An evolved circuit, intrinsic in silicon, entwined in physics", In: *International Conference on Evolvable Systems*. Springer-Verlag Lecture Notes in Computer Science, 1996, 390-405

[5] J. Koza, F.H. Bennett, D. Andre, and M.A Keane, "Automated WYWIWYG design of both the topology and component values of analog electrical circuits using genetic programming", *Proceedings of Genetic Programming Conference*, Stanford, CA, 1996, 28-31.

[6] M. Murakawa, S. Yoshizawa, T. Adachi, S. Suzuki, K. Takasuka, M. Iwata, T. Higuchi, "Analogue EHW chip for

intermediate frequency filters", *Proceedings of the Second International Conference on Evolvable Systems: From Biology to Hardware (ICES98)*, Lausanne, Swiss, September 1998. M.Sipper, D. Mange and A. Pérez-Uribe (editors), vol. 1478, LNCS, Springer-Verlag, 134-143.

[7] E. Takahashi, M. Murakawa, K. Toda, T. Higuchi, "An evolvable-hardware-based clock timing architecture towards GigaHz digital systems", Proceedings of the Genetic and Evolutionary Computation Conferences (GECCO), Vol. 2 July, 1999, 1204-1210.

[8] M. Sipper, D. Mange, A. Perez-Uribe (Eds.), "Evolvable systems: from biology to hardware" *Proc. of the Second International Conference, ICES 98*, Lausanne, Switzerland, Springer-Verlag Lecture Notes in Computer Science, 1998.

[9] A. Stoica, D. Keymeulen and J. Lohn (Eds.) *Proc. of the First NASA/DoD Workshop on Evolvable Hardware*, July 19-21, 1999, Pasadena, CA IEEE Computer Society Press.

[10] A. Stoica, "Toward evolvable hardware chips: experiments with a programmable transistor array." Proceedings of 7th International Conference on Microelectronics for Neural, Fuzzy and Bio-Inspired Systems, Granada, Spain, April 7-9, IEEE Comp Sci. Press, 1999, 156-162.

[11] D. Butnariu and E. P. Klement, "Triangular norm-based measures and games with fuzzy coalitions", Kluwer Academics, 1993.

[12] M. M. Gupta and J. Qi, "Design of fuzzy logic controllers based on generalized t-operators", Fuzzy Sets and Systems, Vol. 40, 1991, 473-389.

[13] M.M. Gupta and J. Qi, "Theory of T-norms and fuzzy inference methods", Fuzzy Sets and Systems, Vol. 40, 1991, 431-450.

[14] A. Stoica, "Synaptic and somatic operators for fuzzy neurons: which T-norms to choose?" In Proc. of 1996 Biennial Conference of the North American Fuzzy Information Processing Society - NAFIPS, Berkeley, CA, June 19-22, 55-58.

[15] – A. D. Johns and K. Martin, "Analog Integrated Circuit Design", John Wiley & Sons Inc. 1997.

[16] D. Keymeulen, A. Stoica, R. Zebulum. Fault-Tolerant Evolvable Hardware using Field Programmable Transistor Arrays. In IEEE Transactions on Reliability, Special Issue on Fault-Tolerant VLSI Sytsems, vol. 49, No. 2, IEEE Press.

[17] – Proceedings of the NASA/JPL Conference on Electronics for Extreme Environments, Feb. 9-11, 1999; Pasadena, CA. In http://extremeelectronics.jpl.nasa.gov/conference".