**Proceedings of the 2007 IEEE Workshop on
Evolvable and Adaptive Hardware (WEAH 2007)**

On Learning to Trust the Unexpected: Toward a Systematic Apologetic for Evolvable Hardware

John C. Gallagher, *Senior Member, IEEE*

Abstract- Evolvable Hardware (EH) has been described as a composition of reconfigurable hardware and a learning algorithm that optimizes configurations according to pre-defined operational requirements. EH, to be most effective, requires its users to surrender significant amounts of design authority to an automated process. Even though EH has shown significant promise in creating novel designs, working engineers often show reluctance in relying on them. Understanding the causes of such reluctance is a necessary step toward constructing cogent arguments for the adoption of EH methods in practice. In this paper, we will attempt to examine some of the roots of observed reluctance and construct some preliminary arguments to counter it.

I. Introduction

Evolvable Hardware (EH) [1-8] is an emerging subspecialty within Evolutionary Computation [9-11] in which one evolves designs for mechanical, computational, or electrical devices. Evolvable Hardware can often be distinguished from related practices in design optimization by the degree to which automated methods are given control over the final device configuration. In design optimization, a human designer selects the basic form of a solution and allows an automated technique to tune and adjust parameters within the boundaries of that solution type. In EH, a human designer surrenders large amounts of control over the form of the solution. In effect, the automated optimizer is allowed to assemble components in ways that admit solutions that cross normally recognized type boundaries or which fall into type categories not commonly part of engineering practice.

Consider the process of designing a controller to suppress vibration in jet engine combustion chambers. A more traditional design optimization approach might entail construction of a model of the combustion chamber, selection of a specific control strategy (E.G. Proportional-Integral-Derivative, Linear Quadratic Regulator, etc.), and then a combination of model-based and/or automated tuning of strategy parameters to ensure the controlled system does not vibrate dangerously while in operation. The parameter tuning would be followed by both model-based and live system verification. Long term stability and corner cases that might not be testable in the live system would be considered via examination of the coupled model/controller system. Contrast this with an EH approach, in which a highly tunable reconfigurable controller substrate would be evolved to better maintain the engine system in a desired state without regard to remaining within the boundaries of any pre-determined control strategy. Of course, though an EH system's ability to color outside the lines certainly opens the door to novelty, it can also severely disadvantage the necessary engineering tasks that come after a candidate design is on the table. How does one verify and validate an evolved device that might not follow any conventional rules? EC and EH researchers tend to value novelty on its own merits. Therefore, the discipline has collectively spent a great deal of time demonstrating that novel and valuable things can be evolved. Working engineers, however, through training and necessity have come to value predictability, reproducibility, and other practical concerns at least as much. Sometimes, however, one can have too much of a good thing. The author has observed in his interactions with traditionally trained controls and aerospace engineers a decided mistrust of EH methods. Ironically, the same engineers that willingly field controllers based on measured set-points and empirically generated lookup tables balk at EH controllers grounded in at least as much hard empirical data. There is no doubt in the author's mind that others have at least occasionally encountered similar attitudes. The point of this paper is not to disparage those attitudes – a healthy core of skepticism and conservatism is necessary for an engineer. The point of this paper is to make a provisional attempt at making explicit the causes of engineering objections. Many in the EH community will read this paper and quite correctly observe that all of it is obvious. That's good news – because it means that reluctance to adopt EH methods is largely a function of poor communication with those outside our community. This can be fixed.

So, how do we fix these perceived communication faults? First, we must realize that EH is well beyond the proof-of-concept stage. There are sufficient results in the literature to demonstrate that EH methods can produce novel solutions that outperform those created using conventional engineering methods. Further demonstrations of this fact would, of course, be welcome and of huge interest to those already convinced – but would do little to address the concerns of the larger world outside our community. Second, we must realize that lack of wider acceptance is likely rooted in deeply held conventions that are learned during an engineer's early training and reinforced repeatedly over his/her career. It would be both foolish and arrogant to presume these conventions are somehow wrong. However, sometimes conventions need to be reassessed upon the coming of disruptive technology. EH certainly qualifies for

The materials and findings in this paper were based upon work supported by the National Science Foundation under Grant No. 0238200.J. Gallagher is with the Computer Science and Engineering and Electrical Engineering Departments at Wright State University, Dayton OH, 45435-0001(phone 937-775-3929; fax 937-775-5133; e-mail john.gallagher@wright.edu)

that. Third, having realized the above two items, we might consider characterizing current engineering convention and determine in what ways EH may be perceived as violating it. Having done that, we hopefully would be in a position to offer justifications and mitigating arguments.

II. Where To Start

If the goal is to help the wider community of engineers to "stop worrying" and accept that evolvable hardware might be useful, then we need to identify the causes of existing impediments and attempt to remove them. There are two basic ways of moving forward. The first way is to examine those EH devices that have already been used to solve real world problems and identify why they were acceptable to their adopters. Study of successful adoptation can unveil justifications and reasons that might be leveraged into other situations. Philosophically, we could consider such study to be an attempt to synthesize *a posteriori* justifications based on the experience of successful cases. The second way is to examine the process of engineering in and of itself and attempt to construct "reasonable objections to EH" based upon proceedural assumptions taken (rightly or not) as axiomatic by engineers. Once we had straw man objections in hand, we could examine the chain of reasoning leading up to their creation and be in a position to argue that they either do not apply to specific EH efforts, or that they do –but that EH does not actually violate them. Philosophically, we could consider such study to be an attempt to synthesize *a priori* justifications based upon first principles.

Both *a priori* and *a posteriori* justifications will rely on specific definitions of "engineering" and "evolvable hardware". To make some progress in attempting both, this paper will provide its author's definitions. It is understood, however, that definitions might vary and with them, the conclusions this paper will eventually arrive upon. However, it is hoped that that even if the conclusions are imperfect, the process by which they were reached at least is seen as sufficienty sound for the reader to make his or her own attempts with his or her own starting definitions.

III. Engineering Process

For purposes of argument, this paper will define *engineering* as the application of scientific and economic principles to the solution of technical problems. Practical engineering processes contain most, if not all, of the following phases which proceed roughly in the order indicated:

a) *analysis phase* in which solution requirements are assessed

b) *specification phase* in which aspects of acceptable solutions are enumerated

c) *design phase* in which one or more candidate designs that satisfy specifications are generated

d) *design evaluation phase* in which one or more candidates are analyzed for suitability – often using models or *a priori* arguments. The goal of this phase is to determine if solutions are even reasonable with respect to the specifications and the constraints of physics and economics. Issues evaluated can include, but are not limited to, efficacy, economy, manufacturability, and ethical acceptability.

e) *design testing phase* in which one or more candidates passing *a priori* evaluation is subjected to empirical (*a posteriori*) verification. Such verification might include stress testing of actual, prototype, or simulated, products. The goal of this phase is to ensure that the products will operate as intended in both normal and unusual environments. Issues tested can include, but are not limited to, efficacy, manufacturability, safety under both normal and exceptional conditions, and operational lifetime.

In practice, this ordering is only roughly followed. Individual engineering teams might revisit earlier stages if later stages show no candidate design to be acceptable. Also, the boundaries between phases can be somewhat fuzzy and the specific means by which each is accomplished can differ from discipline to discipline. Still, this paper will take the position that nearly all engineers adopt some variation on the above process and that asking them to leave out any of them will tend to cause them discomfort and to doubt their designs.

IV. Evolvable Hardware

Specific Evolvable Hardware efforts in the literature are diverse in intended application, goals, underlying reconfigurable hardware, and choice of evolutionary / learning algorithm. For purposes of argument, this paper will presume the following commonalities among EH projects:

a) *Reconfigurable hardware used in EH work is knowingly adopted and/or designed so that individual hardware configurations can cut across the boundaries of conventional solution types.* EH hardware substrates tend to be much more general and expressive than those used in design optimization. For example, one can spin the dials on a PID controller all day, but it never stops being a PID controller. FPTAs [8] could evolve to act as digital or analog circuits and definitely have the ability to evolve on either side of the boundary, or even as some odd hybrid of the two. CTRNNs [12-14] are universal dynamics approximators and could in principle evolve to embody any control law. They might evolve into an expected control type – or they might evolve into something that falls between the cracks of conventional types. Similar observations can be made for nearly all analog electrical circuit based EH. For digital circuit EH, it is clear that at least when evolution is done at the gate level, one could in principle evolve devices that cross

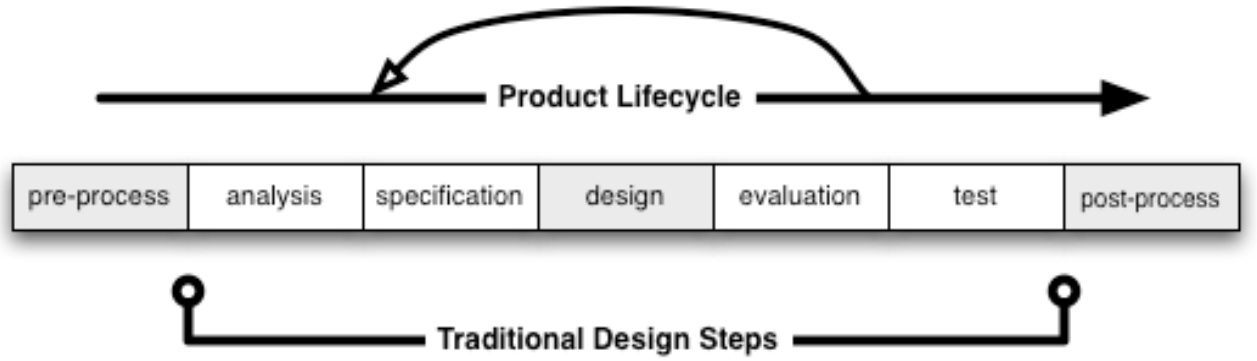

Figure 1: A conventional view of a design process. Gray phases designate areas where EH methods are unlikely to cause concern, yellow phases designate areas where EH methods might cause disquiet.

boundaries of traditional MSI component types (registers, counters, ALUs, etc.).

b) *Adaptive Learning / Evolutionary Algorithms used in EH efforts tend to be weakly constrained in what solutions are allowable and/or acceptable.* EH employs evaluation functions that are some combination of *a priori* and *a posteriori* acceptability tests of fit to a behavioral specification of performance efficacy.

Both of the above are, within the EH community, generally considered to be good things. Hardware substrates that allow out-of-the-box solutions and powerful, but weakly constrained searches, both contribute to an ability to construct novel, perhaps superior, solutions to problems. They also contribute to the ability to deliver systems that can adapt on-the-fly to unexpected circumstances.

V. Engineering Process and EH

The EH community sees benefit in what it does. However, it is the contention of this paper that to the extent that the above mentioned aspects of EH cause real or perceived violations of accepted engineering practice, it will also cause resistance to wider adoption of EH methods. The paper has also contended that we can begin to uncover these real and perceived clashes through both *a priori* examination of the engineering process and how it might clash with EH methods on first principles and *a posteriori* examination of EH case studies that have been found acceptable by traditionally trained engineers.

Figure 1 is a summary of phases of a design process. We have added two phases, "pre-process" and "post-process". These refer respectively to tasks that might take place before or after the traditional design phases. So long as one presumes that pre-process and post-process tasks are independent of the traditional design phases and it can be shown that they don't make "things worse", it is doubtful that any traditional engineer would object. The five phases in the middle can be considered to be the traditional product design/life-cycle. Inclusion of EH methods into green shaded tasks are considered to be unlikely to cause engineers great concern. Inclusion of EH methods into yellow shaded areas are considered to be potentially disturbing to engineers. Let us now look at the intersection of each of the seven defined phases and the two identified central aspects of EH.

VI. EH and Pre-Process Phase

Admittedly, at first blush it might appear a bit strange to speak of engineering tasks that happen before engineering tasks begin. However, because most engineered systems are constructed of parts that were engineered at a lower level of abstraction, we can certainly envision "pre-design" tasks that produce the parts we need to build our product. An obvious example is digital logic gates. Most digital designs rely on pre-engineered gates that satisfy specific performance requirements. In effect, pre-process tasks can be thought of as recursively expanding the whole design process for each part one wants to use in a design. If one were to use EH methods to create novel, but "simple" devices that otherwise do not run afoul of engineering ethic, then one could claim to use EH to produce superior parts to be used by traditional engineers. Of course, all the issues of the next sections would be recursively inherited by any such "pre-process" sub-designs.

VII. EH and The Analysis/Specification Phases

In the analysis phase, a traditional engineer would be attempting to assess and formally record the features required in a solution (requirements). In the specification phase, that engineer would be codifying features need in, and constraints applied to, proposed solutions (specifications). A traditional designer would, after finishing these tasks, enter the design phase and rely on experience and intuition to put forward candidate designs that would be expected to pass the evaluation and test phases. Those that "did the best" in the final phases would be kept, those that did unacceptably well would be discarded.

Design optimization approaches are fundamentally identical, except that some limited parameters are tuned with

respect to tests about how well the design meets specifications and requirements with respect to some subset of evaluation and/or test phase measurements. In effect, design optimization automates a product lifecycle loop from the evaluation and/or test phases back to the design phase – except that the designs are constrained into specific design types/templates that are largely expected to be easily evaluated, easily tested, and predictable in their behavior both in time and across specific instantiations of individual devices. Most importantly for the issues of this section, a design optimization approach often solves satisfying specification by insuring that the generic substrate and all possible tunings of it are in spec. This leaves the tuning algorithm only to deal with optimizing requirement satisfaction.

In that EH methods largely use the same sorts of automated testing of requirements/specifications that is done in design optimization – it's not likely that EH methods would run afoul of most engineers so long as the methods used to elicit an EH objective function were not vastly different than those conventionally used in the field of the application. It would be important, however, to show how a combination of the EH objective function and inherent immutable features of the underlying hardware substrate together map into the full requirements and specifications for a candidate machines. Failing this demonstration would quite correctly raise engineering ire.

VIII. EH and The Design Phase

Oddly, this is perhaps the one phase where introducing EH should not not cause much concern for traditional engineers. Evolutionary Algorithms (EAs) are stochastic processes that over time tend to improve solution quality with respect to some pre-defined objective function. Within the inherent limits of the EA and the expressiveness of the underlying hardware components being combined, one can expect continually improvements in solutions. One can, of course, argue about the efficacy of search, the amount of computational resources needed, the specific nature of the underlying substrate, or any number of details. The basic idea of "EH as idea generator", however, shouldn't be objectionable so long as the EA generated ideas are subjected fully to evaluation and test phases conventionally accepted for the target application field. That's where most of the problems begin.

VIII. EH and The Evaluation and Test Phases

Here be dragons. It is in these phases that many traditionally trained engineers will see EH methodology falling off the edge of the world, or at least falling far enough outside of accepted practice to be less worthy of trust. Consider that, like with design optimization, EH creates an automated loop from evaluation and test back to design. If one desired to be more specific, one could argue that *intrinsic EH* closed an automated loop from *test* to *design* and *extrinsic EH* closed an automated loop from *evaluation* to *design.* Hybrid intrinsic/extrinsic approaches would close a loop including all three of *design*, *test*, and *evaluation*. The difference between EH of any kind (intrinsic, extrinsic, etc.) and design optimization is the extent to which the automatically generated candidates are likely to remain predictable, amenable to evaluation, and amenable to test. Here is where the fundamental aspects of EH can cause problems. Hardware substrates that can jump the tracks among solution types can become impossible (or at least very difficult) to evaluate in the sense defined in section III, item (d). A design optimized LQR will always be an LQR that can be evaluated for theoretical reliability using accepted methods. Hardware that can jump the tracks, or fall in the uncharted spaces between the tracks can confound such analysis. This basic problem can exist independently of intrinsic or extrinsic methodology. A solution constructed of unusual parts that satisfies a loosely specified objective function will not of necessity pass muster in conventional evaluation and test phases. Intrinsic EH can introduce its own unique problems. Presuming the reconfigurable substrate is subject to significant variation from device to device, it would be easy for an EA to exploit specific device peculiarities that are not only difficult to model, but fundamentally unknowable by any direct means. This is not an uncommonly observed problem, and it only exacerbates possible difficulties in maintaining predictability of an evolved design.

IX. EH and the Post-Process

In this phase we would include EH methods that are active after traditional design approaches are exhausted and/or not applicable. In short, this would include any such applications that would attempt fault recovery after catastrophic failure. Once a system has failed, as long as it can be shown that semi-random experimentation can not make things worse, allowing the system to go into an EH learning mode certainly at least gives a chance of recovery at little or no cost. Many "exotic" applications are of this nature.

X. Potential A-Priori Justifications

One could generate a list of possible justifications for the use of EH by examining the problems/opportunities created by the intersections of the "key features" of EH and the elements of the standard design process. Some that can be extracted are:

a) One can safely use EH to create parts to be used in traditional designs as long as the novel parts created are sufficiently simple and/or understandable to survive traditional verification and test in their own right. (Intersection of EH and "pre-process" design phase)

b) The process of creating an objective function entails elicitation of a formal statement of design requirements and specifications. In principle this is not different from,

nor is it any more dangerous than, conventional means of formalizing requirements and specifications. (Intersection of EH and "analysis and specification phases")

c) Allowing an automated process to have a seat at the brainstorming table to generate candidate designs is no more dangerous than having another human at the table, so long as the machine created candidates are as fully vetted with testing and verification. (Intersection of EH and "design phase")

d) Extrinsic EH should not be problematic under the following conditions:

1) Accurate models of hardware components exist

2) The EH system is constrained to not violate model assumptions

3) Systems constructed of the modeled components can be verified through analytical techniques

4) Hardware instantiations of evolved systems are directly testable, or can be inferred to function because of tight coupling between models of and actuality of component behavior and interaction. (Intersection of EH and "Evaluation and Test Phases")

e) EH used as a course of last resort (I.E. to correct problems that persist after any and all possible human engineering efforts have been exhausted) should not be problematic. A system that has a chance of doing something is better than one that has a certainty of doing nothing.

XI. A Whole Lot of Obvious

This paper has not stated anything that most EH practitioners do not already know or have not already considered to at least some degree in the development of their own projects. In fact, there are EH projects in the literature that make either explicit or implicit appeal the above *a-priori* justifications, nearly all of which could have been just as easily extracted via *a posteriori* analysis of the literature. Justification (a) is quite well illustrated by work in evolved antennas [15] and high-temperature logic gates [16]. Justification (d) is implicit in much of Koza's work on evolved electrical circuits [2] as well as in nearly all digital circuit based EH. Justification (e) is implicit in all work related to self-repair and/or adjusting to fabrication faults.

Of course, at least part of the point is that these justifications are obvious, can be made, and have been made. The community can do a great deal toward increasing the acceptability of EH in the minds of line engineers simply by making what is obvious to us clear to them in an organized manner via a systematic analysis of the intersections of stages of a design process and characteristics of EH. Hopefully, that exercise can be completed in more detailed and more application area specific ways as we move into new application domains.

XII. A Whole Lot of Not so Obvious

We can make significant inroads by just making some of the obvious explicit. However, a secondary exercise of potential import would be to carefully examine those EH projects that just violate convention to such a degree that it doesn't seem possible to reconcile them at all to those conventions. At some level, EH is gamed to be able to cross conventional boundaries or solution type. Very often, solution types and or templates exist at least in part because they can be understood via reductionistic method and functional decomposition. For example, there is no inherent reason why microprocessors must be built as parings of data path and microcontroller – generally, however, we do so that we can employ functional decomposition and reductionism to manage complexity. If one were to leverage EH maximally, then she/he would in principle allow the evolution of solutions that might not have reductionistic explanations associated. Many of the previously given justifications will still hold, but any related to the verification/test process will fail. What then?

Traditional engineering has at its core reliance upon and an affinity toward reductionism. It is unlikely that such reliance should be abandoned entire. On the other hand, there are instances where society has come to trust less than completely understood systems in life-critical applications. Seeing eye dogs, for example, receive about four months of training before they are literally entrusted with the lives of their owners. In addition, much of what passes for reductionistic analysis in engineering is anything but. Lookup table based engine controllers, for example, are based on rote lookups of effector efforts that produce desired set-point behaviors. It might appear that the controller is easily understood – after all – it's a lookup table. However, every tuple in the table is an empirically determined bit of "common sense" that in itself is not further reducible. Further examples could be synthesized, the point is that there are plenty of systems that both society in general and engineers in particular accept as useful even though they lack the sorts of reductionistic explanations that engineers have come to covet. This suggests two alternative justification strategies for EH adoption.

a) Find and observe situations in the world where not-completely understood systems are trusted with critical situations. Extract from those situations why people have come to accept the systems. Attempt to recreate those justification methods in the context of EH applications.

b) When attacking a new application area with EH, carefully study existing solutions to identify sub-systems and/or practices that inherently admit empiricism. Attempt to confine EH methods to creating better empirically determined blocks. One could argue that EH could explore empirical strategies far more extensively than could a human. Also, since the EH would be replacing essentially

empirical components anyway, little would be lost in terms of expected reliability and understandability.

XIII. Discussion

It is the author's hope that the better part of the discussion on these topics not appear here, but rather, starts here and is improved upon with the attention and assistance of the community. This brief paper has made a number of observations on why such discussion is necessary and a number of suggestions as to how the discussion could start. First, it was observed that traditionally trained engineers very often feel discomfort in adopting EH. It is believed that at this stage of development of the field, overcoming this discomfort is far more vital than producing additional demonstrations of efficacy. We know that engineers ultimately care about more than mere efficacy. We also know that engineers are human and can, like everyone else, adopt procedure for good reasons, then forget the reasons and convert the guidance of procedure for the shackles of ritual. For both types, those that remain unconvinced due to a perceived lack of attention to issues beyond efficacy and automated novelty, and those that unfortunately have transformed common sense convention into inviolate doctrine, further "look, it works" demonstrations will not be convincing. It is to our benefit to consider how both camps could be convinced. Second, it was suggested that one could attempt both *a priori* and *a posteriori* justifications of EH. This paper attempted both in imperfect and shortened form. However, it is believed that particularly powerful *a-priori* apologetics could be formed by systematically intersecting the steps inherent in traditional engineering and the aspects of EH we hold so dear to uncover places where engineers might feel uncomfortable. This paper accomplished that in a general sense, however, there is no reason a similar process could not be done with more detailed formalized engineering processes used in specific application areas. Once *a priori* justifications are in place, one could mine the EH literature for efforts that are of similar character. These could be used as *a posteriori* justifications that are all the more powerful for having been explicitly tied to new, application specific, design process.

Afterwards, this paper briefly considered that the ultimate form of EH would very likely defy deeply set engineering methods no matter how hard we worked to justify them. It suggested places we could mine for other means of coming to trust systems that are life critical, yet not fully understood in a manner acceptable to line engineers. It is unlikely that many would be convinced to adopt EH in the short term on the weight of such speculations – save in the cases when inherent empericism is already identified in a particular engineering method. However, such speculation should be undertaken as a means of better understanding the boundaries of acceptable and unacceptable technical solution of real world problems.

References

[1] Greenwood, G. and Tyrrell, A. (2005) *Introduction to Evolvable Hardware: A Practical Guide for Designing Self-Adaptive Systems*. In press.

[2] Koza, J.R., Bennett, F.H. III, Andre, D., Keane, M.A., and Dunlap, F. (1997). Automated synthesis of analog electrical circuits by means of genetic programming. *IEEE Transactions of Evolutionary Computation* vol. 1, number 2, pp 109-128

[3] Thompson, A. (1998). On the automatic design of robust electronics through artificial evolution. In *Evolvable Systems : From Biology to Hardware - Second International Conference.*

[4] Lohn, J.D. and Colombano, S.P. (1998). Automated analog circuit synthesis using a linear representation. In *Evolvable Systems : From Biology to Hardware - Second International Conference*

[5] Kajitani, T., Hoshino, T. Nishikawa, D., et.al. (1998). A gate level EHW chip: implementing GA operations and reconfigurable hardware on a single LSI. In *Proceedings of the International Conference on Evolvable Hardware (ICES 1998).*

[6] Higuchi, T., Iwata, M., Keymeulen, D. et. al. (1999). Real-world applications of analog and digital evolvable hardware. *IEEE Transactions on Evolutionary Computation* Vol. 3, No. 3:220-235

[7] Zebulum, R.S., Santini, C.C., Sinohara, H.T., Pacheco, M.A.C, et. al. (2000). A Reconfigurable platform for the automatic synthesis of analog circuits. in *Proceedings of the Second NASA/DoD Workshop on Evolvable Hardware*. IEEE Press.

[8] Stoica, A, Keymeulen, D., Zebulum, R. et. al. (2000). Evolution of analog circuits on field programmable transistor arrays. in *Proceedings of the Second NASA/DoD Workshop on Evolvable Hardware*. IEEE Press.

[9] Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA, Addison-Wesley.

[10] Fogel, D.B., (1991). *System Identification Through Simulated Evolution: A Machine Learning Approach to Modeling*. Needham Heights, MA, Ginn Press

[11] Back, T., Harnmel, U., and Schwefel, H.P. (1997). Evolutionary computation: comments on the history and current state, *IEEE Transactions on Evolutionary Computation*, vol 1, no 1, pp. 3-17.

[12] Gallagher,J.C., Sanjay K. Boddhu and Saranyan Vigraham (2005). A Reconfigurable Continuous Time Recurrent Neural Network for Evolvable Hardware Applications. The Congress on Evolutionary Computation - 2005 Edinburgh, UK, IEEE Press.

[13] Gallagher,J.C. and S. Vigraham (2002). A modified compact genetic algorithm for the intrinsic evolution of continuous time recurrent neural networks. Proc. of the 2002 Genetic and Evolutionary Computation Conference (GECCO).

[14] Saranyan Vigraham, Gallagher,J.C. and Sanjay K. Boddhu (2005). Evolving analog controllers for correcting thermo acoustic instability in real hardware. Proc. of the 2005 Genetic and Evolutionary computation Conference (GECCO).

[15] Lohn, J., Linden, D., Hornby, G, et. al. (2003). Evolutionary Design of an X-Band Antenna for NASA's Space Technology 5 Mission, in the *2003 NASA/DoD Conference on Evolvable Hardware*. IEEE Press. p. 155

[16] Stoica, A.; Keymeulen, D.; Zebulum, R. (2001). Evolvable hardware solutions for extreme temperature electronics, in Evolvable Hardware, 2001. Proceedings. The Third NASA/DoD Workshop on. IEEE Press