Possibilities and Limitations of Applying Evolvable Hardware to Real-World Applications

Jim Torresen

Department of Informatics
University of Oslo
PO Box 1080 Blindern
N-0316 Oslo, Norway
E-mail: jimtoer@ifi.uio.no

Abstract. Evolvable Hardware (EHW) has been proposed as a new method for designing systems for real-world applications. This paper contains a classification of the published work on this topic. Further, a thorough discussion about the limitations of the present EHW and possible solutions to these are proposed. EHW has been applied to a wide range of applications. However, to solve more complex applications, the evolutionary schemes should be improved.

1 Introduction

Evolvable hardware (EHW) has recently been introduced as a new scheme for designing systems for real-world applications. It was introduced for about seven years ago [1] as a new scheme for designing electronic circuits. In this method, a set of circuits – i.e. circuit representations, are first randomly generated. The behavior of each circuit is evaluated and the best circuits are combined to generate new and hopefully better circuits. The evaluation is according to a behavior initially specified by the user. After a number of generations, the fittest circuit is to behave according to the initial specification. The most commonly used evolutionary algorithm is genetic algorithm (GA) [2]. The algorithm – which follows the steps described above, is illustrated in Figure 1.

In GA, each individual circuit is often named chromosome or genotype and shown by a circled "+" in the figure. A circuit can be represented in several different ways. For digital circuits however, gate level representation is most commonly used. That is, the representation contains a description of what kind of gates are applied and their inter-connections. For FPGA technology, this is normally equal to a subset of the configuration bit string. The most computational demanding part of GA is usually the evaluation of each circuit – typically named fitness computation. This involves inputing data to each circuit and computing the error given by the deviation from the specified correct output.

Randomness is introduced in the selection and thus, not only the fittest circuits are selected as seen in the figure. However, the probability of a circuit being selected for breeding decreases with the fitness score. In breeding, the parameters of the pairwise selected circuits are exchanged to generate – for each

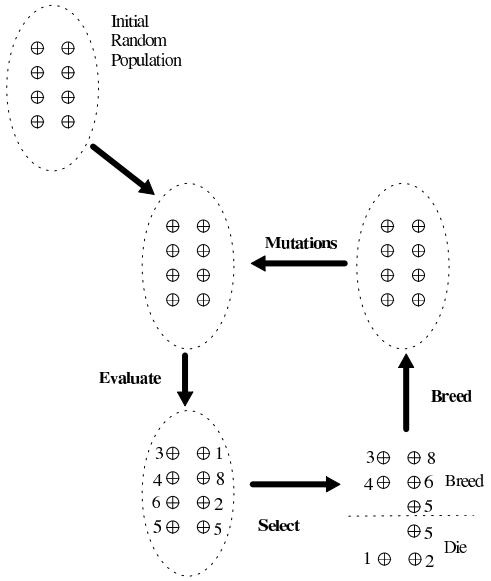

Fig. 1. The genetic algorithm.

couple, two new offsprings – preferably fitter than the parents. Some of the best circuits may as well be directly copied into the next generation. Mutations may also occur and introduce changes in the chromosomes, making them slightly different from what could be obtained by only *combining* parent chromosomes.

A number of industrial applications has arrived based on EHW. These are classified in this paper. The paper further includes a discussion of the properties of these applications and proposal of possible new directions for EHW applied to real-world applications. Much work has been undertaken on various topics related to EHW. Some considers modeling of biological systems without any specific application in mind — e.g. artificial life research. Such studies are not considered in this paper.

The next section contains a classification of EHW research based on a given classification framework. This is followed by a discussion about the limitations and possibilities in using EHW for real-world applications in Section 3. Conclusions are given in Section 4.

2 A Framework for Classifying EHW

EHW research is rapidly diverging. Thus, to understand the EHW field of research, a classification framework would be beneficial. This is presented below. The many degrees of freedom in EHW could be represented in a multi-dimensional space. However, here a list format is prefered.

Evolutionary Algorithms (EA). A set of major algorithms exists:
- **Genetic Algorithm (GA)**
- **Genetic Programming (GP)**
- **Evolutionary Programming (EP)**

The major difference between GA and GP is the chromosome representation. GA organizes the genes in an array, while GP applies a tree of genes. Both schemes apply both crossover and mutation, while EP – which has no contraints on the representation, uses mutation only.

Technology (TE). Technology for the target EHW:
- **Digital**
- **Analog**

Building Block (BB). The evolution of a hardware circuit is based on connecting basic units together. Several levels of complexity in these building blocks are possible:
- **Analog comp. level.** E.g. transistors, resistors, inductors and capacitors.
- **Gate level** E.g. OR and AND gates.
- **Function Level** E.g. sine generators, adders and multipliers.

Target Hardware (THW). In EHW, the goal is to evolve a circuit. The two major alternatives for target hardware available today are:
- **Commercially available devices.** FPGA (Field Programmable Gate Arrays) are most commonly used. They consist of a number of reconfigurable digital gates, which are connected by entering a binary bit string into the device. This string specifies how the gates are connected. Field-Programmable Analog Arrays (FPAA) are available as well. They use the same programming principle as FPGAs, but they consist of reconfigurable analog components instead of digital gates.
- **Custom hardware.** ASIC (Application Specific Integrated Circuit) is a chip fully designed by the user.

Fitness Computation (FC). Degree of fitness computation in hardware:
- **Off-line EHW (OFL).** The evolution is simulated in software, and only the elite chromosome is written to the hardware device (sometimes named extrinsic evolution).
- **On-line EHW (ONL).** The hardware device gets configured for each chromosome for each generation (sometimes named intrinsic evolution).

Evolution (EV). Degree of evolution undertaken in hardware:

- **Off-chip evolution.** The evolutionary algorithm is performed on a separate processor.
- **On-chip evolution.** The evolutionary algorithm is performed on a separate processor incorporated into the chip containing the target EHW.
- **Complete HW evolution.** The evolutionary algorithm is implemented in special hardware – i.e. not running on an processor.

Scope (SC). The scope of evolution:

- **Static evolution.** The evolution is finished before the circuit is put into normal operation. No evolution is applied during normal operation. The evolution is used as a circuit optimizing tool.
- **Dynamic evolution.** Evolution is undertaken while the circuit is in operation and this makes the circuit online adaptable.

Application	EA	TE	BB	THW	FC	EV	SC
Adaptive Equalizer [3]	GA	D	Neuron	Custom	ONL	On-chip	S
Ampl. and Filter Design [4]	GA	A	T/R/L/C	Custom	OFL	Off-chip	S
Analog Circuit Synthesis [5]	GP	A	R/L/C	Custom	OFL	Off-chip	S
Character Recognition [6]	GA	D	Gate	Comm.	OFL	Off-chip	S
Clock Adjustment [7]	GA	D	Gate	Custom	ONL	Off-chip	S
Digital Filter Design [8]	GA	D	Gate	–	OFL	Off-chip	S
IF Filter Tuning [9]	GA	A	Filter	Custom	ONL	Off-chip	S
Image Compression [10]	GA	D	Pixel	Custom	OFL	On-chip	D
Multi-spect. Image Rec. [11]	GA	D	Function	Comm.	OFL	Off-chip	S
Number Recognition [12]	GA	D	Gate	Comm.	OFL	Off-chip	S
Prosthetic Hand [13]	GA	D	Gate	Custom	ONL	Complete	S
Robot Control [14]	GA	D	Gate	Comm.	ONL	Complete	D
Robot Control [15]	GA	D	Gate	Comm.	ONL	Off-chip	S
Sonar Classification [16]	GA	D	Gate	Comm.	OFL	Off-chip	S

Table 1. Characteristics of EHW applied to real-world applications.

Table 1 summarizes the characteristics of the published work on EHW applied to real-world applications. A major part of them are based on digital gate level technology using GA as the evolutionary algorithm. However, promising results are given for analog designs, where evolution is used to find optimal parameters for analog components. The applicability of analog technology is further discussed in Section 3.2. About half of the experiments are based on custom hardware – or simulation of such. It is more common to undertake the fitness evaluation on-chip (ONL) compared to the evolution (On-chip/Complete). This is reasonable, since the fitness evaluation is – as mentioned earlier, the most computational demanding part of the evolution. Many topics are relevant when discussing the EHW applicability. Several of these topics are discussed in the next section.

3 EHW used in System Design

This section presents some of the limitations of EHW as they may be important to explain why EHW is not widely used today. Further, possible promising ways of applying EHW and evolutionary schemes are proposed.

3.1 Application of Evolvable Hardware

There are many real-world applications. Many of them may be solved *without* EHW. Thus, one would like to analyze what properties of applications make them of interest to apply EHW. That is, what a given application requires of an implemented system.

First, the scope of evolution should be determined. It must be decided if it is required that the system is online adaptable during execution or not. To make an online adaptable system, dynamic evolution should be applied. So far, only a few examples exist of dynamic evolution [10, 14]. Dynamic evolution provides a new scheme for designing systems adaptable to changes in the environment as well repairing failures in the system itself. The adaptability feature of EHW would probably be more exploited in future systems, since this feature is normally not found in traditional hardware systems. If the architecture of the system is not changing during normal operation, static evolution is used. This may still be interesting if the evolved circuit is performing *better* than a traditionally designed device. So far, only a few applications [16] using *digital* designed systems have arrived where this has been proved. However, Koza has given many successful examples of analog electric circuit synthesis by genetic programming [5]. Another example is tuning of analog filter circuits [9].

Second, if digital technology is to be used, it must be determined if there are real-time performance constraints in the application. That is, if fast – specially designed, hardware is required to run the application. If there are no such constraints, computer simulations would be a better choice. However, special demands like cost, circuit size, power consumption or reliability could still require special hardware. E.g for high volume products, the cost could be reduced by using evolvable hardware compared to computer simulations on COTS[1] hardware. There are some successful examples showing the benefit of using EHW in real-time systems. This include image compression [10] and an artificial hand controller [17]. Both these are based on custom hardware.

To summarize, the directions for promising use of evolvable hardware could be when applied in online adaptable systems requiring special hardware implementations to run the application successfully. E.g. the embedded systems market is large and would probably benefit from such a technology. Further, for applications where an evolved circuit performs better than a traditionally designed system – in the most important fitness issues, should as well be successful. The important issue is that hardware development in general is more time consuming and expensive than software development. Thus, when implementing an application in special hardware, there should be something to gain from it.

[1] Commercial Off The Shelf.

3.2 Evolvable Hardware Technology

One of the device families used as EHW are the Field Programmable Gate Arrays (FPGAs). Many of the limitations of applying such devices as EHW were resolved by the introduction of the Xilinx XC6200 devices. Whereas the configuration bit string coding is normally kept secret by the manufacturer, this is freely available information for the XC6200. Unfortunately, Xilinx has decided to end the production of these devices.

One area within EHW research is analog design [18]. As the world is analog in nature there would always be a need for interfacing the analog environment. However, in contrast to digital design, most of the analog circuits are still hand-crafted by the experts of analog design [19]. Unfortunately, the number of people with knowledge about analog design has diminished as the vast field of digital design and computer science has appeared. Thus, EHW could be a means for making analog design more accessible. A few years ago there arrived several Field-Programmable Analog Arrays (FPAA). Experiments with online fitness evaluation in these kinds of devices have been undertaken [20, 21]. The simple initial experiments are promising. However, the limited precision of the devices leads to noise that could limit the design of large or high-precision systems. The problem would be how to make such a system behave deterministically. More details about FPGAs and FPAAs applied as EHW are given in [18].

Lack of commercial hardware that can be applied as EHW may explain why as much as half of the works presented in Table 1 is based on custom hardware.

3.3 EHW as a Digital Design Tool

Digital design is an area where software tools move in the direction of providing the designer a high level input interface. The input is usually either schematic drawings and/or hardware description language code. The software performs the automatic synthesis down to the target hardware. The optimization ability of evolutionary schemes could prove to be valuable in design development tools offering input at a higher level than today. The evolutionary method should provide the following features:

1. The development time of a digital system is reduced. One would have to specify the input/output-relations rather than designing the explicit circuit. However, the problem is how to be able to cover every possible input/output relation rather than using an explicit HDL or schematic specification.
2. More complex systems can be designed. So far only small and simple circuits have been evolved. However, as new design schemes – like incremental evolution, are developed it should be possible to evolve complex circuits and systems [22].

In normal digital design, one would normally design the system for every possible combination of the inputs. However, for more complex systems, this is a near impossible task. To prove that a circuit is working correctly one would have to simulate every possible combination of the inputs. If this is an unobtainable

task, one would at least try to review the design to be convinced that the design will not fail at any event. Reviewing an evolved complex system is not an easy task. Moreover, experiments have shown that evolving a circuit by using a limited number of rows in a truth table is extremely difficult [23].

These issues seem to be a bottleneck for applying evolutionary design methods as a substitute to manual digital design techniques. These are today based on using complex system designing tools offering the designer *macro* blocks, which maps effectively onto the target hardware. Few experimental results indicate that evolutionary techniques will outperform traditional digital design in the near future. To make EHW more applicable, the EHW could probably benefit from using macro blocks more complex than those that already have been applied in function level evolution.

3.4 Noise Robustness and Generalization

The gate level version of EHW is basically applying two-level signals. In comparison to neural network modeling using 32-bit floating point values, digital EHW could not normally provide the same noise robustness and generalization [6]. To improve the representation ability of EHW, each signal could be coded by a variable number of bits - using multi-valued logic [24]. This is applied in [13, 16]. If the input patterns to the system are in digital format, it would probably be interesting to investigate an architecture where an increased number of bits is used towards the output of the system. That is, the *number* of bits used for representation signals in each layer increases from input to output. This would correspond to providing more accuracy for the higher levels of the system. Detecting the values of a small number of pixels in a picture could be undertaken with a coarse accuracy compared to detecting larger objects in an image. Another option to improve the signal coding is to include time in the coding approach. That is, to attain the value of a signal, it must be observed for a certain time.

3.5 Evolving Complex Systems

The work described in Section 2 is mainly based on circuits with a limited number of building blocks. Thus, the applications have limited complexity. To solve more complex applications, the limitation in the chromosome string length must be solved [25, 26]. A long string is required for representing a complex system. However, a larger number of evolutionary generations are required as the string increases. Thus, work has been undertaken to try to diminish this limitation. There are several ways of solving this problem:

- Dividing the GA.
- Compressing the chromosome string.
- Increasing the building block complexity.
- Dividing the application.

Various experiments on dividing the GA computation and use parallel processing have been undertaken [27]. The schemes involve fitness computation in

parallel or a partitioned population evolved in parallel. This approach requires that GA finds a solution if it is allowed to compute enough generations. When small applications require weeks of evolution time, there would probably be strict limitations on the systems evolvable even by parallel GA.

One approach to compressing the chromosome string is by using variable length chromosome [12]. Increased building block complexity is e.g. by using higher level functions as building blocks instead of gates. This has been called function level evolution. Most work is based on using fixed functions as building blocks. Results from experiments using this approach are found in [28].

Dividing the application is based on the principle of divide-and-conquer. It was proposed for EHW as a way of incremental evolution of the application [22]. The scheme is called increased complexity evolution, since a system is evolved by evolving smaller sub-systems. Increased building block complexity is also a part of this approach, where the building blocks are becoming more complex as the system complexity increases. Experiments show that the number of generations required for evolution by the new method can be substantially reduced compared to evolving a system directly in one operations [6]. Considerable future work on this topic is anticipated [29]. The result of this will probably show the applicability of EHW to complex real-world applications.

3.6 The Future of EHW

Several applications have arrived based on EHW. This is both in digital and analog target technology. There seem to be two major directions for the future: First, evolution can be applied to tune the parameters of a circuit. Second, the evolution can be applied to make online adaptable real-time systems. However, the evolutionary schemes would have to be improved to overcome the limitations described in this paper. It seems like evolution will be introduced as a substitute to traditional analog design earlier than it is applied in traditional digital design.

4 Conclusions

This paper has contained a study of the characteristics of EHW applied to real-world applications. Further, limitations and possibilities of the EHW approach have been discussed. There seems to be a number of applications using analog target hardware. For the digital based applications, only small systems have been evolvable. The major reason for this seems to be the lack of schemes for evolving complex digital systems. This will be an important future research issue.

Acknowledgements

The author would like to thank the group leader Dr. Higuchi and the researchers in the Evolvable Systems Laboratory, Electrotechnical Laboratory, Japan for inspiring discussions and fruitful comments on my work, during my visit there in January-April 2000.

References

1. T. Higuchi et al. Evolvable hardware: A first step towards building a Darwin machine. In *Proc. of the 2nd Int. Conf. on Simulated Behaviour*, pages 417–424. MIT Press, 1993.
2. D. Goldberg. *Genetic Algorithms in search, optimization, and machine learning.* Addison Wesley, 1989.
3. M. Murakawa et al. The grd chip: Genetic reconfiguration of dsps for neural network processing. *IEEE Transactions on Computers*, 48(6):628–638, June 1999.
4. J.D. Lohn and S.P. Colombano. A circuit representation technique for automated circuit design. *IEEE Trans. on Evolutionary Computation*, 3(3):205–219, September 1999.
5. J. R. Koza et al. *Genetic Programming III.* San Francisco, CA: Morgan Kaufmann Publishers, 1999.
6. J. Torresen. Increased complexity evolution applied to evolvable hardware. In Dagli et al., editors, *Smart Engineering System Design: Neural Networks, Fuzzy Logic , Evolutionary Programming, Data Mining, and Complex Systems, Proc. of ANNIE'99.* ASME Press, November 1999.
7. E. Takahashi et al. An evolvable-hardware-based clock timing architecture towards gigahz digital systems. In *Proc. of the Genetic and Evolutionary Computation Conference*, 1999.
8. J. F. Miller. Digital filter design at gate-level using evolutionary algorithms. In *Proc. of the Genetic and Evolutionary Computation Conference*, 1999.
9. M. Murakawa et al. Analogue ehw chip for intermediate frequency filters. In M. Sipper et al., editors, *Evolvable Systems: From Biology to Hardware. Second Int. Conf., ICES 98*, pages 134–143. Springer-Verlag, 1998. Lecture Notes in Computer Science, vol. 1478.
10. Sakanashi et al. Evolvable hardware chip for high precision printer image compression. In *Proc. of 15th National Conference on Artificial Intelligence (AAAI-98)*, 1998.
11. R. Porter et al. An applications approach to evolvable hardware. In *Proc. of the First NASA/DoD Workshop on Evolvable Hardware*, 1999.
12. M. Iwata et al. A pattern recognition system using evolvable hardware. In *Proc. of Parallel Problem Solving from Nature IV (PPSN IV).* Springer Verlag, LNCS 1141, September 1996.
13. I. Kajitani and other. An evolvable hardware chip and its application as a multi-function prosthetic hand controller. In *Proc. of 16th National Conference on Artificial Intelligence (AAAI-99)*, 1999.
14. D. Keymeulen et al. On-line model-based learning using evolvable hardware for a robotics tracking systems. In *Genetic Programming 1998: Proc. of the Third Annual Conference*, pages 816–823. Morgan Kaufmann, 1998.
15. A. Thompson. Exploration in design space: Unconventional electronics design through artificial evolution. *IEEE Trans. on Evolutionary Computation*, 3(3):171–177, September 1999.
16. M. Yasunaga et al. Evolvable sonar spectrum discrimination chip designed by genetic algorithm. In *Proc. of 1999 IEEE Systems, Man, and Cybernetics Conference (SMC'99)*, 1999.
17. I. Kajitani et al. An evolvable hardware chip for prosthetic hand controller. In *Proc. of MicroNeuro'99*, pages 179 – 186, 1999.

18. J. Torresen. Evolvable hardware — The coming hardware design method? In N. Kasabov and R. Kozma, editors, *Neuro-fuzzy techniques for Intelligent Information Systems*, pages 435 – 449. Physica-Verlag (Springer-Verlag), 1999.
19. O. Aaserud and I.R. Nielsen. Trends in current analog design: A panel debate. *Analog Integrated Circuits and Signal Processing*, 7(1):–, 1995.
20. S. J. Flockton and K. Sheehan. Intrinsic circuit evolution using programmable analogue arrays. In M. Sipper et al., editors, *Evolvable Systems: From Biology to Hardware. Second Int. Conf., ICES 98*, pages 144–153. Springer-Verlag, 1998. Lecture Notes in Computer Science, vol. 1478.
21. R. S. Zebulum. Analog circuits evolution in extrinsic and intrinsic modes. In M. Sipper et al., editors, *Evolvable Systems: From Biology to Hardware. Second Int. Conf., ICES 98*, pages 154–165. Springer-Verlag, 1998. Lecture Notes in Computer Science, vol. 1478.
22. J. Torresen. A divide-and-conquer approach to evolvable hardware. In M. Sipper et al., editors, *Evolvable Systems: From Biology to Hardware. Second Int. Conf., ICES 98*, pages 57–65. Springer-Verlag, 1998. Lecture Notes in Computer Science, vol. 1478.
23. J. F. Miller and P. Thomson. Aspects of digital evolution: Geometry and learning. In M. Sipper et al., editors, *Evolvable Systems: From Biology to Hardware. Second Int. Conf., ICES 98*, pages 25–35. Springer-Verlag, 1998. Lecture Notes in Computer Science, vol. 1478.
24. T. Kalganova et al. Some aspects of an evolvable hardware approach for multiple-valued combinational circuit design. In M. Sipper et al., editors, *Evolvable Systems: From Biology to Hardware. Second Int. Conf., ICES 98*, pages 78–89. Springer-Verlag, 1998. Lecture Notes in Computer Science, vol. 1478.
25. W-P. Lee et al. Learning complex robot behaviours by evolutionary computing with task decomposition. In Andreas Brink and John Demiris, editors, *Learning Robots: Proc. of 6th European Workshop, EWLR-6 Brighton*. Springer, 1997.
26. X. Yao and T. Higuchi. Promises and challenges of evolvable hardware. In T. Higuchi et al., editors, *Evolvable Systems: From Biology to Hardware. First Int. Conf., ICES 96*. Springer-Verlag, 1997. Lecture Notes in Computer Science, vol. 1259.
27. E. Cantu-Paz. A survey of parallel genetic algorithms. *Calculateurs Parallels*, 10(2), 1998. Paris: Hermes.
28. M. Murakawa et al. Hardware evolution at function level. In *Proc. of Parallel Problem Solving from Nature IV (PPSNIV)*. Springer Verlag, LNCS 1141, September 1996.
29. J.R. Koza. Future work and practical applications of genetic programming. In *Handbook of Evolutionary Computation*, page H1.1:3. IOP Publishing Ltd and Oxford University Press, 1997.