An Evolutionary Platform for Developing Next-Generation Electronic Circuits

James A Hilder
Intelligent Systems Group
University of York
United Kingdom
+44 1904 432 832
jah128@ohm.york.ac.uk

Andy M Tyrrell
Intelligent Systems Group
University of York
United Kingdom
+44 1904 432 340
amt@ohm.york.ac.uk

ABSTRACT
In this paper, a new method for evolving simple electronic circuits is discussed, with the aim of improving the reliability and performance of basic circuit blocks. Next-generation CMOS device models will be used in the simulation of circuits. Circuits are mapped to a grid layout which reflects the appearance of conventional schematic blocks. The performance of the system at designing passive low-pass filters is discussed, with an outline given of the intended future steps, towards the goal of integrating sub 100 *nm* MOSFET models into the circuits.

Categories and Subject Descriptors
I.6.8 [**Simulation**]: Types of simulation – *parallel, visual.* J.2 [**Physical Sciences and Engineering**]: Electronics

General Terms: Algorithms, Design

Keywords: Analogue circuit design, genetic programming, genetic algorithms, SPICE, CMOS

1. INTRODUCTION
This paper describes the basic design and early proof-of-concept results obtained from a new evolutionary software platform which is designed to create basic electronic circuits, both digital and analogue, which may be used as the building blocks for more complex circuits.

The concept of the design is to create circuit layouts in a manner similar to those used in traditional circuit design strategies. Most analogue and discrete digital circuits can be drawn as a schematic on a regularly space grid layout, provided wires can cross and the grid is sufficiently large. Analogue circuits such as filters often fit very well into a grid as they often have repetitive yet simple ladder topologies. Logic gates and other simple digital functions also fit well into this category; as these form the basis of more complex logic structures they also can be drawn in this way, albeit on a much larger scale.

GECCO'07, July 7–11, 2007, London, England, United Kingdom.

The design described in the paper has a matrix of cells which contain either four or five nodes, at the compass points and also the centre if the node contains passive components. The component values and types, including wires and links (and thus circuit topology) are generated using a genetic algorithm, simulated and scored for fitness. The goal of the project is not to generate complex circuits or novel solutions for problems, but instead to examine simple circuits using next-generation *complimentary metal-oxide semiconductor* (CMOS) models with the aims of improving reliability and performance.

The system described will allow for parallel simulation using a number of different evolutionary strategies on a computing grid. The software is written in Java and will use a version of the SPICE simulation package and other circuit and device simulation software, using models developed as part of the UK based, EPSRC and industry funded multiple university *nano-CMOS* project. The next section describes in detail the background of the overall project, detailing the inherent problems of next generation devices and also the complexity of the models involved in simulating them. This is followed by a more detailed overview of the system itself, some early results and timeline and thoughts on future work.

2. NEXT GENERATION CMOS DEVICES
The Intelligent Systems Group at the Department of Electronics, York University is involved in a group partnership involving teams at Glasgow, Edinburgh, Manchester and Southampton Universities and also the National E-Science Centre in Edinburgh. The overall aim of the partnership is to design accurate models for next generation devices using statistically-based modelling methods to evaluate device parameters.

Semiconductor devices are now entering what has been termed the "***nano-CMOS***" era, with the presence of many semiconductor fabrication plants tooled for sub-0.1 μm technology nodes. New semiconductor fabs are currently tooled for the 65 *nm* and 45 *nm* nodes, with current technology roadmaps suggesting 22 *nm* technologies by the year 2015. Next-generation transistor devices are subject to parameter variations which result in unpredictable behaviour using current models. The device variability is caused by both unavoidable intrinsic parameter fluctuations and microscopic differences which occur at many different stages of the silicon manufacturing process (see Figure 1). Adjusting for this device variability and the new architectures which may help alleviate some variability will add significant complexity to the overall design process, and will require the coordination of many teams of device experts and their associated design tools.

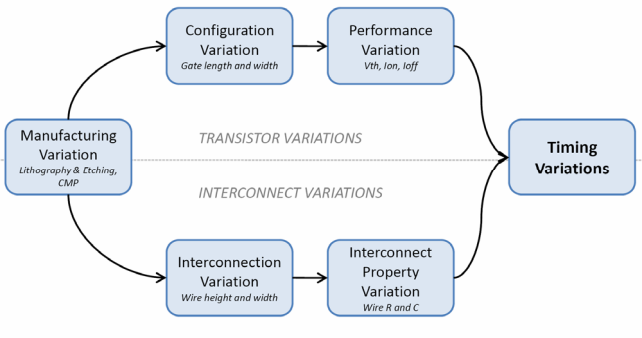

Figure 1. Variations due to Manufacturing Processes

CMOS transistors from the 45 *nm* & 32 *nm* technology nodes will have channel lengths of typically 25 *nm* and 18 *nm*. At these dimensions atomistic and quantum mechanical limits will profoundly influence the delay and power variability seen within devices. As body thickness scales down, effects such as surface-roughness and line-edge roughness introduce significant characteristic mismatches at an individual device scale. Electro-migration erodes interconnect admittance which causes further threshold voltage fluctuations. Other effects such as SOI history effect and negative bias temperature instability also affect the *IV* curves on an individual transistor basis. This variation will have a profound effect when many gates are integrated: one can be certain individual transistors will fail (*perform unsatisfactorily*), so when VLSI circuits of tens or hundreds of millions of transistors are developed measures must be taken to allow for these faults. As the technology node shrinks the magnitude of the problem will increase which may force a dramatic rethink in various aspects of IC design, possibly using a level of redundancy and asynchronous or un-clocked designs [2].

A number of new semiconductor technologies and architectures for producing next generation devices are emerging in manufacture. These all generally have the aims of improving performance and reliability of future CMOS devices. They include the use of thin silicon channels to reduce the short-channel effects such as drain induced barrier lowering, such as *extremely-thin* and *fully-depleted silicon on insulator* (ETSOI & FDSOI) and *silicon-on-nothing* (SON) structures. Double-gate and triple-gate architectures and back-gate transistors add a level of relief from random-doping fluctuations at the expense of increases sensitivity to L_{gate} fluctuations. Ultra-thin bodied SOI devices both improve resistance to certain intrinsic parameter fluctuations and also offer superior electrostatic integrity and reduced junction capacitance thus are likely to be very important in future device advances. The teams at Glasgow University and Manchester University are developing device and compact models incorporating a number of the variations and new technologies; these models will increase in depth and accuracy over the course of the next 36 months [1].

Of key importance to the future of the project will be the use of Grid-computing architectures. This will allow for the secure yet accessible transfer of results and simulations between the partner sites and also offer up power computing resources. These computing resources will likely prove invaluable in later stages of the evolutionary approach as the processing time will increase dramatically as the models increase in complexity. The project is aided by the *Open-Middleware Infrastructure Institute* (OMII), which links together e-Science based activities from Edinburgh, Manchester and Southampton, providing web services with mechanisms allowing the recording of software, a data repository and several development wizards for web related works.

3. EVOLUTIONARY SYSTEM

A number of different strategies for circuit simulation have been created in the past to evolve circuits, a relevant selection are shown in Table 1. Both extrinsic and intrinsic solutions have been designed which follow a general workflow (see Figure 2). Evolvable hardware has been used to design analogue, digital and mixed-mode circuits [12,13,18]. Whilst potentially offering performance and realism benefits unavailable to simulated designs, such an approach is not possible when designing for components that do not yet exist in the real world. Generally SPICE is used as the platform for simulating circuits in extrinsic designs, although other possibilities do exist and may offer speed or possibly accuracy benefits. However, SPICE will form that backbone of this software as the models to be used later and associated software are assembled for this software.

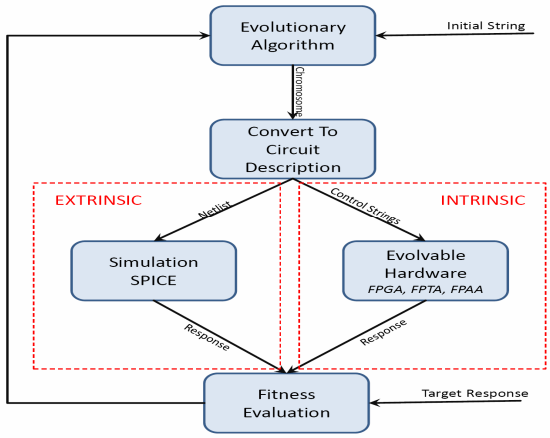

Figure 2. Extrinsic and Intrinsic Circuit Evolution

Table 1. Various examples of EAs in circuit design

Circuit	Implementation	Reference
Logic gates and functions	GP – Boolean logic CGP – Boolean logic Evolvable FPTA chip	Koza *92* [4] Miller *99* [15] Heidelberg *00* [21]
Op-Amps	DARWIN –GA (values) GP using SPICE	Kruiskamp *95* [11] Koza *99* [8]
Frequency Discriminator	Using FPGA chip	Thompson *96* [18]
Low-pass, ladder, bandpass & crossover filters	GP – SPICE GP – SPICE , using a free variable	Koza *96* [5] Koza *99* [6]
20^{th} & 21^{st} Century Patented Circuits	Including low-voltage balun, mixed A-D variable capacitance, voltage-current converter, high-current load circuit and cubic-function generator.	Streeter *04* [16]

The grid-based circuit design system takes a different approach to most designs of circuit evolution in that the chromosome circuit strings are used to describe the values and topologies of components on a grid-based arrangement of cells. This grid-based circuit description is then remapped into a SPICE net-list for fitness evaluation. This approach is intended to increase the likelihood of both the circuits being valid (with a path from input to output), and also of the ability of SPICE to successfully simulate the circuits. It also allows the circuits to be easily rendered in their grid layout, which allows a more clear observation of improvements over analysing the net-lists alone.

3.1 CELL DESCRIPTION

The system to be built describes a circuit using an interconnected matrix of cells. The cells are connected to adjacent cells at the compass positions, and around the perimeter external ports may be present. All these points may be considered as nodes, although they may be inactive or disconnected. Cells can contain different things; they may contain a combination of passive, 2-port components (including wire links), a single transistor (with various wiring arrangements) or a 4-port sub-circuit.

In the first of these cell modes the cell has an extra node at the centre, and four components connect the nodes (see Figure 3). The components can be resistors, capacitors, inductors or *0* Ω links. The matrix shown in Figure 3 is from the initial generation of a run which uses all first-mode cells. The types of component and their values are determined from the string describing the cell, which is encoded such that all possible combinations of these components can be present[1]. Every component can be set to enabled or disabled; note how some components and complete cells are disabled in Figure 3. This cell mode has been fully implemented and some early results of performance are discussed later.

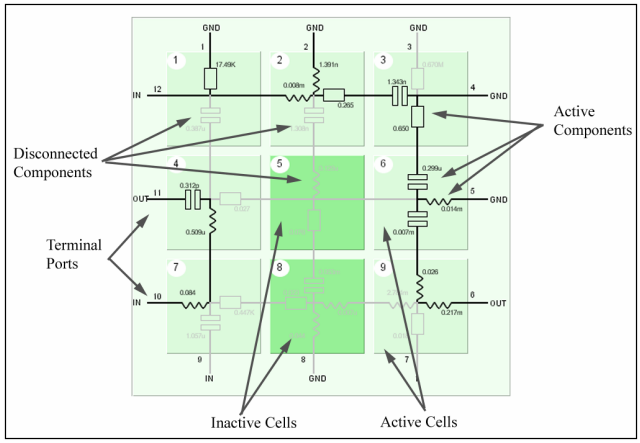

Figure 3. Example 3x3 Matrix of Mode 1 Cells

The second mode is presently being implemented and will be fundamental to the goals of the project. The cell will allow simple (single component) blocks with up to 4 ports. These blocks will typically contain transistors, with either one node disconnected or two nodes joined, however theoretically many other arrangements and components may be included, provided it is possible to simulate them. The initial components will be based on native SPICE transistor models, for BJT and FET type transistors. As with passive components the topology and type of transistor will be described by the cell string, along with a number of parameters. Different parameter sets will be used for various models, ranging from simple transistor type, to L_{gate} and W_{gate} for simple CMOS models to a far more complex (and time consuming) combination of parameters used by different models[2].

The final mode will allow small 4-port sub-circuits from present circuits to be defined as a single cell function, allowing larger circuits to be assembled from smaller functions. A similar feature is already implemented as cells can be cloned as a mutation operation; however, this will allow larger blocks (even complete circuits) to be used, provided they only have to connect with four terminals. The actual format and definition for this mode is yet to be defined, although the basic principle is similar to the use of automatically defined functions.

3.2 GENETIC ALGORITHM

The current model used for the chromosome definition uses fixed-length binary strings. In this arrangement, the first two bytes describe the width and height of the cell matrix and determine the length the chromosome should be. The remainder of the chromosome describes the individual cells[3]. Each cell is defined by a ten-byte string in which the first two bytes describe the topology and the remaining eight bytes determine the component values (each based on a two-byte string). When transistor models are used the eight bytes will define various transistor parameters. The initial population either consists of random chromosome strings, or a population based upon a single string which is mutated.

External ports can be mapped to ground, input and output nodes, to a positive voltage source, or to other external ports. A key reason behind the choice of the grid layout that it allows for the easy rendering of circuit descriptions which allows the evolutionary process to be seen as it works – the various mutation, cloning and cross-over events can be seen on-screen if desired as circuits evolve.

Flexibility has been built into the system from the ground up to allow a number of different selection, mutation and cross-over strategies to be chosen. Both tournament selection, whereby the fittest of *n* individuals selected at random is promoted, and rank-based selection, whereby all individuals are ranked and weighted for selection have been implemented. Mutation strategies include a bit-wise mutation based on the chromosome string, a value-based mutation where component values are adjusted around their present values, cell-based mutation where individual cells are swapped or cloned, and random strategies which can use any of the mutation strategies. The cross-over methods include the swapping of individual cells, or rows, columns and blocks of cells from the two parent, and value averaging cross-over where the component values are distributed between the two parent values. The mutation and cross-over strategies used for each new member can be logged so that the effectiveness of the different strategies can be observed;

[1] It is possible to choose which combinations are allowed; it is planned to be able also weight these for different scenarios.

[2] The BSIM3v3 MOSFET models have over 75 definable parameters, although only a small proportion of these will be set when using this model.

[3] The string is `10(w*h)+2` bytes long in current implementation, with each cell represented in reading order. More than ten bytes may be necessary to describe later transistor parameters.

from the results obtained so far it is noted that the different strategies can all be effective in certain situations.

3.3 FITNESS EVALUATION

The system created uses the Berkeley SPICE simulation software to evaluate the fitness of evolved circuits in the form of net-lists. A number of different skeleton net-lists are available which contain the static nodes for the test circuit, and also the setup instructions for the analyses. For example, the net-list for filters creates a circuit with a single input and output connected to source and load resistors with an AC voltage source connected to the input. It instructs the SPICE simulator to perform an AC sweep analysis. For evaluating logic functions a setup which performs transient analysis on pulsed inputs may be used. It should be possible to calculate many different fitness metrics using an intelligent combination of SPICE analysis and fitness functions.

The present version of the software is using *SpiceOPUS*, a freeware distribution of the Spice package, although this is likely to be changed to allow for more advanced transistor models which are not available in this distribution. To increase the probability of SPICE being able to simulate the circuits a number of small optimizations can be introduced at the conversion to net-list stage. Parallel and series combinations of components can be converted into single components to speed up simulation; SPICE runs tend to increase exponential with node count. SPICE cannot simulate closed capacitive loops a very high values resistor can be included in parallel with all capacitors. Open loops and disconnected section can be removed from the net-list (and also from the chromosome, although the option to leave in place exists).

The results from the SPICE run are then read by a function which compares them to a target value at each point and scores a fitness value based on this. For an AC sweep analysis this will be at every defined frequency step (typically 20 per decade). The target value may specify a '*satisfactory*' range to define a region outside which fitness scores will be penalized; it may also be a '*don't care*' region. The scores are then totalled to provide a final fitness value, with the lowest scores most likely to be promoted to the next generation. It is possible to set end criteria – either when fitness reaches a certain low or generation count reaches a certain number, or let the simulation run endlessly.

The system has been designed such that other fitness functions and input/output sets can be easily implemented and added. The ability to perform DC & transient analysis and to allow multiple, pulsed inputs will be added in the future so that analysis of logic functions can take place. This will require only simple adjustments to the shell of the net-list and fitness calculating function.

3.4 PERFORMANCE

The testing done so far has largely been based on the creation of various filter designs, in particular a low-pass filter which is lossless for frequencies up to 1 *kHz* and fully attenuated for frequencies above 2 *kHz*. This particular filter has been chosen as lots of previous GP results exist for it, by Koza and others. Filter design has been attempted using GP [4,6], bond-graph representations [3], linear representations [14] and CGP [20]. It is a relatively simple test, requiring only passive components, with only a single input and output and providing an easy-to-calculate fitness function. The results obtained to date are not intended to show novel or human-competitive designs in anyway, nor prove the benefits of this design over any other. Instead they are intended to demonstrate a proof-of-concept behind the layout, and also show the design may offer both design flexibility and promising performance.

An AC sweep analysis is performed in SPICE, measuring the output voltage between 1 *Hz* and 100 *kHz* input, with twenty measurement points per decade (a total of 101 measurements). These output voltages are then compared to the target voltages of 1000 *mV* below 1 *kHz* and 0 *mV* above 2 *kHz*. The five points which are between 1 *kHz* and 2 *kHz* are considered "don't care" points and do not influence the fitness score, thus a total of 96 points influence the fitness score. The fitness score is weighted to punish values which fall outside what is deemed an acceptable deviation – this is ± 30 *mV* in the pass region and ± 1 *mV* in the attenuation region. The fitness score is the difference between the target and observed voltage in *mV*, with a weighting factor of 10 if this difference is outside the acceptable limits. Thus a perfect filter will achieve a score of zero and the worst case filter (a 'perfect' high-pass filter) will achieve a score of 960,000[4]. The fitness function and weighting system has been chosen so that results are directly comparable to results from Koza (*and others* [5,20]) work[5].

A number of different short runs were performed using different starting parameters, including different mutation and cross-over strategies and various cell-matrix and population sizes. The best results obtained from these early tests have been based on a **5x5** cell-matrix, with a population size of **100**. The mutation strategy allows chosen allows both component values to be adjusted with a **12%** probability (the amount of mutation is normally distributed from the starting component value with a **12%** standard deviation) and circuit topology can be mutated (using bit-wise mutation) at a **0.6%** probability. A cell-based cross-over where **50%** of the new population contain the dominant parents' cells with one cell copied from the less-dominant parent is used. These settings were observed to have produced the best results after one thousand generations[6]. Of five runs performed with these parameters, three had created circuits with a fitness score of below **10,000** at the end of the final generation. The best fitness score achieved from the five run was **5536**[7].

It is important to note that the circuits developed after 1000 generations fail to meet the fitness criteria (are outside the *satisfactory* region) and do not represent particularly useful filters or recognizable layouts. However, they do perform the function of attenuating high frequencies, with pass and cut regions closely matching those of the target. They also vastly outperform very simple circuits such a single component filters. The best-of-generation fitness chart (see Figure 4) clearly shows the gradual improvement in fitness as component values are adjusted, and also some more distinct steps which generally occur where the circuit topology has mutated in a beneficial way.

---

[4] Assuming a 0 *mV* output below 1 *kHz* and 1000 *mV* output above 2 *kHz*.

[5] Koza's results are calculated using voltage difference thus are a factor of 1000 smaller but otherwise the fitness functions are identical.

[6] This is 100,000 circuit evaluations. In context, Koza's work uses population sizes of this order, such as 320,000.

[7] Equates to 5.536 using Koza's fitness function, as fitness scores are calculated based on *V* difference instead of *mV* difference.

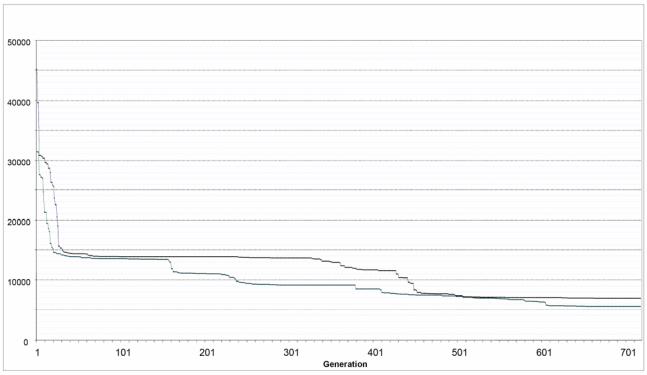

Figure 4. Best of Generation Fitness for 2 Filter Runs

Using the metric of circuits evaluated to achieve a certain fitness score, a simple assessment of the algorithms performance can be gauged. Results published for this circuit problem by Koza achieve a fitness score which would be in single figures using the described fitness function, whereas the best figure in the run described here is of the order of one thousand times worse. However, it is noted that in a typical run described by Koza the fitness score decreases from a first generation best of *c.* **65,000** down to a score of *c.* **20,000** by generation 15. It is not until generation 19 that the best fitness better the typical end-of-run scores observed in the tests here[8], a generation 26 when the fitness rapidly decreases to single figures as a 100% *satisfactory* circuit is found. As each generation in Koza's run evaluates **320,000** circuits, a far greater computational effort in terms of circuits evolved has been exhausted by this stage [5].

Whilst no fully compliant have been created to date using the new system, the performance in creating near-compliant circuits in a reduced time has been demonstrated. It is also observed that this performance outperforms other work based on a CGP implementation on circuit construction, although it is suggested this is not necessarily due to a more efficient algorithm but instead the framework for circuit construction which the cell based layout provides, something that will better demonstrated as more functions are implemented. Indeed it is intended to evaluate the performance of CGP and other algorithms in constructing circuit strings in forthcoming work, as is described in the following section.

4. FUTURE WORK

The software described thus far is still early in its development cycle as key basic elements such as transistor cells are not yet fully implemented. The performance of various parameter settings and different mutation and cross-over schemes will be examined in more detail once simple transistor models have been introduced, then more complex models and transistor parameters can be added. Work to allow improved parallel performance of system is being carried out with goal of grid-enabling the software to make use of more processing threads.

4.1 OTHER ALGORITHMS

Present work involves an adaptation of the way circuits are described to allow for the implementation of a *Cartesian Genetic Programming* (CGP) method of evolving the circuit strings, whereby circuit strings will be represented as an indexed graph. Previous work has demonstrated the effectiveness of this evolutionary strategy in many fields, including a limited amount of research in analogue circuit design. This will require a few subtle changes to the arrangement of the chromosome strings, although the method of converting strings to a grid-based circuit layout (and subsequent SPICE net-list) will remain the same. Other representations and algorithm settings may also be implemented with the goal of finding the best performing setup [15,20].

4.2 TRANSISTOR BASED CIRCUITS

The goals of the project are to develop circuits which can either improve the reliability or the performance of simple functions using the next-generation models. This goal is being gradually worked towards, first with the setting up of the cell structure and system integration, then the development of the transistor-based cells. The first transistor modes to be developed will be based on the SPICE BJT model, with the cell definition describing the topology and particular model. Following this various MOS models will be implemented as deemed useful or necessary to achieve the project goals; these may include Mos6, BSIM1, BSIM2, BSIM3 or BSIM4 models and other high level models. The target circuits for testing will include examples such as inverters, logic gates, multiplexers and memory (SRAM or other) cells, all fundamental building blocks of digital electronics.

5. CONCLUSION

This paper has outlined the work which is taking place at the University of York in developing a system for the simulation and evolution of simple electronic circuits which will use models for next-generation CMOS devices. The basic principles behind the layout of the system and why this design has been chosen have been discussed, and initial results based on the creation of analogue low-pass filters have been observed. The future roadmap of the system, which will gradually incorporate more sophisticated and complex transistor models, has been outlined.

The grid-based layout demonstrated within this paper helps accelerate development on circuits due to the way in which circuit cells are populated and assembled. It is intended in future work to demonstrate the performance of the principle in more detail and use the system to improve the performance and reliability of next generation devices.

6. ACKNOWLEDGMENTS

This work is funded by a UK Engineering and Physical Sciences Research Council (EPSRC) grant EP/E001610/1. The authors would like to acknowledge the contribution of the other partners in this project.

7. REFERENCES

[1] Asenov, Asen. *Random dopant induced threshold voltage lowering and fluctuations in sub 50-nm MOSFETs: a statistical 3D 'atomisitc' study.* Nanotechnology Vol. 10, 1999, pp.153-158

[2] Bernstein, K et al. *High-performance CMOS variability in the 65-nm regime and beyond.* IBM Journal Res. & Dev. Vol. 50, 2006, pp.433-446

[3] Fan, Zhun et al. *A Bond Graph Representation Approach for Automated Analog Filter Design.*

[8] At this point over 6 million circuit evaluations have occurred, 60 times more than in each run of the new system.

[4] Koza, John R., *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, 1992

[5] Koza, John R., *Genetic Programming III: Darwinian Invention and Problem Solving.* Morgan Kaufmann, 1999

[6] Koza, John R. et al, *Genetic Programming IV: Routine Human-Competitive Machine Intelligence.* Springer, 2003

[7] Koza, John R. et al, *Automatic Synthesis of Electrical Circuits Containing a Free Variable Using Genetic Programming.* Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 2000, pp. 477-484

[8] Koza, John R. et al, *Evolution Using Genetic Programming of a Low-Distortion 96 Decibel Operational Amplifier.* Proceedings of the 1997 ACM symposium on Applied computing

[9] Koza, John R. et al, *Use of Automatically Defined Functions and Architecture-Altering Operations In Automated Circuit Synthesis with Genetic Programming*. Proceedings of the First Annual Conference on Genetic Programming, 1996.

[10] Koza, John R. et al, *Automated WYWIWYG Design of Both the Topology and Component Values of Electrical Circuits using Genetic Programming*. Proceedings of the First Annual Conference on Genetic Programming, 1996

[11] Kruiskamp, Wim and Leenarts, Domine, *DARWIN: CMOS op-amp Synthesis by Means of a Genetic Algorithm.* Proceedings of the 32nd ACM/IEEE conference on Design automation, 1995, pp.433-438

[12] Langeheine, Jörg, Meier, Karlheinz and Schemmel, Johannes. *Intrinsic Evolution of Analogue Electronic Circuits Using a CMOS FPTA Chip.* Barcelona: CIMNE, 2003

[13] Layzell, Paul. *Hardware Evolution: On the Nature of Artificially Evolved Electronic Circuits.* [Online: www.infomatics.sussex.ac.uk/users/adrianth/lazwebpage/web/Motherboard/index.html]

[14] Lohn, Jason D. and Colombano, Silvano. *Automated Analog Circuit Synthesis using a Linear Representation* Lecture Notes in Computer Science, Vol 1478, 1998, pp. 125-136.

[15] Miller, Julian F. And Job, Dominic, *Principles in the Evolutionary Design of Digital Circuits – Part I.* Journal of Genetic Programming and Evolvable Machines, Vol 1. No.1, 2000, pp. 8-35.

[16] Noren, Kenneth V and Ross, John. *Analog Circuit Design Using Genetic Algorithms.* 1998 [Online: http://www.mrc.uidaho.edu/~knoren/GAs/B-159_paper.PDF]

[17] Streeter, Matthew J, Keane, Martin A and Koza, John R. *Automatic Synthesis Using Genetic Programming of both the Topology and Sizing for Five Post-2000 Patented Analog and Mixed-Analog Digital Circuits.* Southwest Symposium on Mixed-Signal Design, Feb 2003, pp. 5-10.

[18] Thompson, Adrian, Layzell, Paul and Zebulum, Ricardo. *Explorations in Design Space: Unconventional Electronics Design through Artificial Evolution.* IEE Transactions on Evolutionary Computation, Vol. 3, Issue 3, 1999, pp. 167-196

[19] Terry, Michael A et al. *GRACE: Generative Robust Analog Circuit Exploration.* Lecture Notes In Computer Science, No. 3907, 2006, pp. 332-343.

[20] Wu, Chao. *Evolutionary Design of Analogue Circuits.* Masters Thesis, Electronics Department, University of York, 2005

[21] *Evolvable Hardware.* University of Heidelburg Institute for Physics [Online: http://www.kip.uni-heidelburg.de/vision/projects/eh]