Intrinsic Evolution of Quasi DC Solutions for Transistor Level Analog Electronic Circuits Using a CMOS FPTA Chip

Jörg Langeheine, Karlheinz Meier, Johannes Schemmel
Heidelberg University,
Kirchhoff-Institute for Physics,
Schröderstr. 90,
D-69120 Heidelberg, Germany.
langehei@kip.uni-heidelberg.de
http://www.kip.uni-heidelberg.de/vision/projects/evo_tarray.html

Abstract

In this paper the results of a series of intrinsic hardware evolution experiments with a CMOS FPTA chip are presented. The experiments discussed are restricted to the evolution of specified target DC behaviors. In the first series of experiments the evolution of different logic gates, namely NAND, NOR, AND, OR and XOR, is studied. The success rates in evolving the different logic gates are compared to each other. Furthermore the influence of three different methods of presenting the test patterns to the chip is analyzed. In a second series of experiments the evolution of a Gaussian voltage transfer characteristic is tackled. Thereby the influence of the chip area available to the genetic algorithm is studied.

1 Introduction

Analog circuit design most often requires the designer to determine position and geometry of every single transistor in the circuit. Accordingly, there is a great interest in the automation of this design process. CAD tools that provide the transistor sizing of given topologies have just entered the market (e.g. [1], [2]), but still fall short of finding new transistor topologies. While these approaches (more can be found e.g. in [3], [4]) use standard simulation techniques and hence are flexible, technology independent and can be well integrated in the design flow, intrinsic hardware evolution offers other desirable perspectives: Reconfigurable devices could be programmed in the field analogous to FPGAs. The integration of an evolutionary algorithm on the chip could use the actual physical properties of the die it is running on and thus fine tune the behavior of the circuit (cf. [5]). Moreover, a background hardware evolution process could adapt the used circuit to changing environments and thereby provide a higher amount of fault tolerance (e.g. [6], [7]). On the other hand, such a system may be used as a circuit search engine, that may find unknown and/or better solutions to known design problems or finds solutions where no (human) one has been found yet ([8]).

As a first step towards such intrinsic hardware evolution systems a Field Programmable Transistor Array (FPTA) has been built and integrated in an evolution environment ([9], [10], [11]). In order to test and study the system two sorts of evolution experiments have been carried out in which the goal has been to find circuits with a desired DC output characteristic. On one hand the restriction to the DC case simplifies both, problem and analysis and thus is considered a good method to test the system. On the other hand, in conventional electronics a stable DC operating point is necessary for most circuits to work properly, so one has to ensure, that stable DC solutions can be found by the evolution engine. In the literal sense testing a DC solution would require to wait for an infinite time before the measurement of each single data point. Since this clearly is not possible, some randomness must be added to the order of the test signals to prevent the candidate solutions from using any temporal correlation in this input data. The evolved circuits discussed in this paper are referred to as *quasi* DC solutions.

As a first class of test problems the evolution of analog DC characteristics for the logic gates NOR, NAND, AND, OR and XOR has been presented to the evolution system. Logic gates are the key element of digital electronics and thus it is desirable to be able to evolve them. Moreover there are known good solutions in the design space to which the evolution results can be compared. Hardware evolution experiments addressing the evolution of logic gates can be found for example in [12] (AND, OR, XOR), [13] (AND) ,[6] (AND), [14] (AND, OR, XOR), [15] (NAND), [16] (XOR), [7] (XNOR). In contrast to the problem of the well known logic gates, most analog designers would probably not be able to directly write down a solution for a circuit producing a Gaussian voltage transfer characteristic (V-V curve). Therefore it should be interesting to see how the hardware evolution deals with this problem, which is more analog in its nature than the logic gate problem. Compared to other target functions like cubics or cubic roots, the Gaussian shape is not monotonic and should thus be harder to learn for the algorithm. A solution for a Gaussian transconductance can be found in [17], whereas EHW experiments on the evolution of Gaussian DC response circuits can e.g. be found in [18] (V-V curve) or [19] (I-V curve).

2 Evolution System

The evolution system, illustrated in Fig. 1, can be divided into three main parts: The actual FPTA chip serving as the silicon substrate to host the candidate circuits, the software that contains the search algorithm running on a standard PC and a PCI interface card that connects the PC to the FPTA chip. The software uploads the configuration bit strings to be tested to the FPTA chip via the PCI card. In order to generate an analog test pattern at the inputs of the FPTA chip, the input data is written to the FPGA on the PCI interface card. There it is converted into an analog signal by a 16 bit DAC. After applying the analog signal to the FPTA, the output of the FPTA is sampled and converted into a digital signal via a 12 bit ADC. The digital output is then fed back to the search algorithm, which in turn generates the new individuals for the next generation.

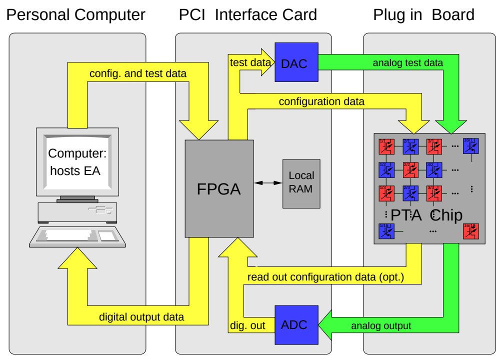


Figure 1. Overview of the evolution system.

2.1 FPTA Chip

The FPTA consists of 16×16 programmable transistor cells. As CMOS transistors come in two flavors, namely N- and PMOS transistors, half of the transistor cells are designed as programmable NMOS transistors and half as programmable PMOS transistors. P- and NMOS transistor cells are arranged in a checkerboard pattern as depicted in Fig. 5.

Each cell contains the programmable transistor itself, a total of 24 SRAM bits to store its configuration, three decoders that allow to connect the three transistor terminals to one of the four cell borders, *vdd* or *gnd*, and six routing switches. A block diagram of the transistor cell is shown in Fig. 2. Width W and Length L of the programmable transistor can be chosen to be $1, 2, \ldots, 15\,\mu\mathrm{m}$ and $0.6, 1, 2, 4, 8\,\mu\mathrm{m}$ respectively. The three terminals *drain*, *gate* and *source* of the programmable transistor can be connected to either of the four cell borders named after the four cardinal points, *vdd* or *gnd*. The only means of routing signals through the chip is given by the six routing switches that connect the four cell borders with each other. Thus in some cases it is not possible to use a transistor cell for routing *and* as a transistor.

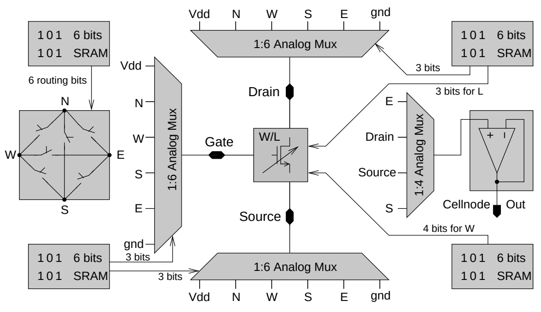


Figure 2. Block diagram of one transistor cell.

The operational amplifier drawn at the right hand side of Fig. 2 is intended to measure the voltages at either of the three terminals of the programmable transistor and the currents through the terminals *drain* and *source* in order to analyze the evolved circuits. More details on the FPTA can be found in [9], [10] and [11].

2.2 Genetic Algorithm

2.2.1 Algorithm

A genetic algorithm was used for the evolution experiments presented in this paper. The rank based selection scheme is depicted in Fig. 3. From the old generation the new one is derived in the following manner: First, after the old population is sorted, the best *reproduction fraction* $\times$ *generation size* individuals are reproduced. Second an individual of the best *crossover fraction* individuals is chosen randomly and mutated. Third, a crossover partner for the mutated genome is chosen randomly from the best *crossover fraction* individuals of the old generation. These two parents then produce

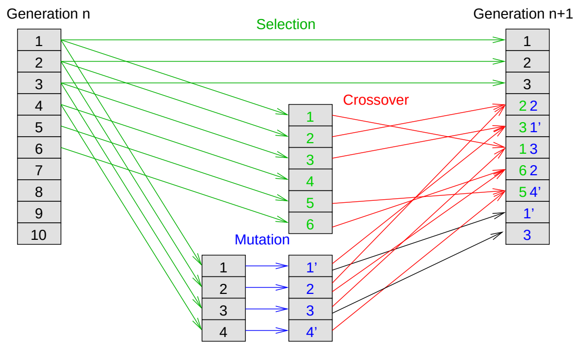


Figure 3. Visualization of the GA.

one child genome for the next generation. Steps two and three are repeated until the new generation is filled with a total of *generation size* genomes. The GA parameter values used are listed in Table. 1.

GA Parameter	Logic Gates	Gaussian curve
generation size	50	50
reproduction fraction	0.2	0.2
mutation fraction	0.4	0.4
crossover fraction	0.6	0.6
crossover rate	40 %	40 %
mutation rate	3 %	3 %
edge length of used array	5	$4 \ldots 11$
crossover block size	3	$3 \ldots 6$
number of generations	5000	10000

Table 1. Genetic algorithm parameters used throughout the presented experiments.

2.2.2 Representation and Genetic Operators

The Genome representation used by the GA reflects the structure of the FPTA: The genome is divided into 256 cells that contain the configuration information for each transistor cell. Consequently, each cell contains an entry as well for the width as for the length of the programmable transistor, an entry for each of the routing connections of the transistors terminals and six entries for the routing switches. As described in section 2.2.1 two genetic operators are used for the GA: Mutation and Crossover.

The Mutation operator acts on entries of the cells: It changes every entry in every cell of the genome to a new random value with a probability given by the *mutation rate*.

The crossover operator works only on the cell level, but does not touch single entries of a cell. The operation of the crossover operator is illustrated in Fig. 4. Each of the squares symbolizes a cell of the genome corresponding to the according transistor cell in the FPTA. Once two

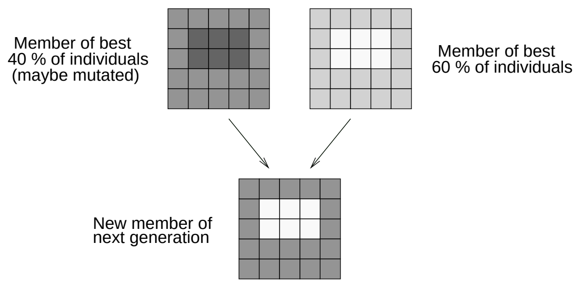


Figure 4. Crossover operation.

crossover partners are designated to create offspring, the size and position of a rectangle of cells to be exchanged is randomly chosen. The child genome is then assembled by taking this rectangle of cells from one parent (the one that already was object to the mutation operation) and the rest of the cells from the other one. The maximum size of the edges of the exchanged rectangles is controlled by the *Crossover block size*.

2.2.3 Fitness/Error Function

Instead of maximizing a fitness function, the genetic algorithm tries to minimize the error of the candidate circuit. The error function

$$\text{RMS Error} = \sqrt{\frac{\sum_{i=1}^{512}(V_{tar}(i) - V_{out}(i))^2}{512}} \times 1000 \quad (1)$$

calculates the root mean square error per measured input data point with respect to the given target function $V_{tar}(i)$ in mV. $V_{out}(i)$ denominates the measured output voltage for the given input stimulus. In general $V_{tar}(i)$ is given by the desired transfer function and depends on the two input variables V_{in1} and V_{in2}:

$$V_{tar} = V_{tar}(V_{in1}, V_{in2})\,. \quad (2)$$

3 Evolution Experiments

3.1 Experimental Setup

The electrical setup for the intrinsic fitness evaluation of the test candidates during the evolution is shown in Fig. 5. All experiments are restricted to a fraction of the whole transistor cell array that is located in the lower right corner of the chip, which is indicated by the square shaded in darker gray in Fig. 5. Throughout the remainder of the paper the size of this square will be described by its edge length. The two inputs are applied to the south side of the chip and the output voltage is measured on its east side. Both, in- and outputs are fixed to the positions indicated in Fig. 5

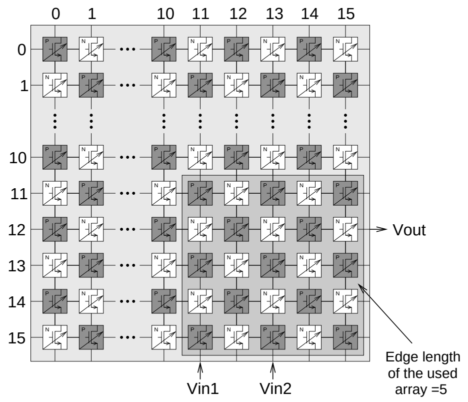


Figure 5. Test setup for the evolution experiments.

throughout all the experiments presented here. This setup is chosen to keep the routing effort necessary for the genetic algorithm to a reasonable amount. For each evolution run, a fixed number of generations is used (5000 for the logic gates, 10000 for the evolution of the Gaussian DC curve). The best individual of the last generation of the evolution and its last fitness are taken as a result.

For the presented experiments the evolution system achieves a throughput rate of about 118 individuals per second. Since in both series of experiments a total of 512 DC points are measured for each individual the average time per measurement must be less than $15\,\mu$s. On the one hand this constrains the successfully evolved circuits to settle to the desired output values in less than these $15\,\mu$s, on the other hand this reflects the quasi DC character of the measurements.

3.2 Evolution of the DC Behavior of Logic Gates

Target Function For all logic gates evolution experiments the desired output voltage V_{tar} is set to be 0 or $5\,$V depending on the inputs and the logic function sought. The threshold for an input to represent a logic zero is set to $2\,$V and the threshold for a logic one to $3\,$V. In- and output definitions are listed in table 2. On the one hand it is nei-

Input low	$< 2\,$V
Input high	$> 3\,$V
Output low	$0\,$V
Output high	$5\,$V

Table 2. Definition of the logic levels for in- and output voltages.

ther necessary nor useful to specify the behavior of a logic gate in the intermediate input range, where it has to switch its output. On the other hand, this choice creates a selection pressure towards solutions, which provide a minimum of amplification. This would not be the case, if the circuits were merely tested at 0 and $5\,$V .

Fitness Evaluation The candidate solutions produced by the GA are tested in the following way: One of the two input channels is set to 8 different voltages. For each of these 8 voltages, 64 different voltages are applied to the other input resulting in 512 voltage pairs. All applied voltages exclude the range between 2 and $3\,$V in accordance with the definition of the threshold voltages for the inputs. For each of the logic gates to be evolved a total of 90 evolution runs are carried out. The first 30 runs are done with V_{in1} statically held at 8 input voltages evenly distributed and linearly spaced in the two valid input range subintervals, while V_{in2} is swept from 0 to $5\,$V (again excluding the range between 2 and $3\,$V). The second 30 evolution runs are done in the same fashion, except that the 64 input voltages for V_{in2} now are chosen randomly for each V_{in1} input voltage. The input test pattern for the last thirty evolution runs differs from the second 30 runs only in that the roles of V_{in1} and V_{in2} are chosen randomly before the test of each individual. The test conditions for all 90 runs are summed up in Table. 3

Run number	V_{in1}	V_{in2}
$1\ldots30$	static	forward sweep
$31\ldots60$	static	random values
$61\ldots90$	static/rand. values	static/rand. values

Table 3. Test paradigms for the fitness evaluation for the evolution of the logic gates.

Verification Tests The functionality of the evolved circuits is verified for different reasons. First, it can not be taken for granted that an evolved circuit achieves the same fitness result a couple of days after it is evolved, because some of the environmental conditions as for example the die temperature may have changed. Second, it is thereby possible to analyze the effect of the fitness evaluation method on the functionality of the evolved circuits. Third, it is interesting to test how well the evolved circuits perform on another chip in another evolution system. The verification test patterns are similar to the test patterns used during the evolution in that for each of 8 different input voltages presented to one input, 64 different voltages are applied to the other one. The voltage for the 'sweep' input is then either swept forward from 0 to $5\,$V, swept backward from 5 to $0\,$V or 64 times chosen randomly. To test if the output of the evolved circuit reacts symmetrically with regard to the two inputs, the roles of the two inputs V_{in1} and V_{in2} are interchanged and the three tests above are repeated. Finally, the same procedure was used to test the evolved circuits on a second chip, hosted by a different PCI interface card in a different computer. Together with the error value obtained from the evaluation during the evolution experiment a total of 13 fitness results are thus obtained. All verification test methods are summarized in Table 4. The error values from the verification measurements used in the remainder of this paper are the averages over 100 consecutive measurements.

Number	V_{in1}	V_{in2}	Chip
1	lowest error from last generation		
2	static	forward sweep	1
3		backward sweep	
4		backward sweep	
5	forward sweep	static	
6	backward sweep		
7	random values		
8	static	forward sweep	2
9		backward sweep	
10		backward sweep	
11	forward sweep	static	
12	backward sweep		
13	random values		

Table 4. Different test modes.

3.3 Evolution of DC V-V Gaussian Circuits

For the evolution experiments aiming at a Gaussian shaped V-V curve, only the input V_{in2} is used and V_{in1} fixed to $0\,\mathrm{V}$. The input voltage V_{in2} is 512 times chosen randomly within the full power supply range for the evaluation of each individual. The desired output voltage is given by

$$V_{tar} = 1\,\mathrm{V} + 3\,\mathrm{V} \cdot \exp^{(\frac{(V_{in2} - 2.5\,\mathrm{V})}{1\,\mathrm{V}})^2} . \qquad (3)$$

In order to study the influence of the resources available to the genetic algorithm, 10 evolution runs are done for edge lengths of the used chip array (see Fig. 5) varying from 4 to 11. Each run is stopped after 10000 generations and the best individual of the last generation is taken as the best circuit solution.

4 Evolution Results

4.1 Evolution of the DC Behavior of Logic Gates

The fitness scale used is given by the error function in Eq. 1. It is defined as the root mean square error per input data point given in mV. The highest error value that can be obtained for the logic gates is thus $5000\,\mathrm{mV}$ for circuits that do exactly the opposite of the desired behavior. On the other hand, error values in the order of 100 mV can be caused by non perfect calibrations of the analog circuitry involved in the measurements.

4.1.1 Comparison of the Results for Different Gates

The error values obtained from all 90 runs for all 5 gates are plotted in error histograms in Fig. 6. For this plot only the error values obtained at the end of each evolution run (test method number 1 in Table 4) are used. The results in Fig. 6

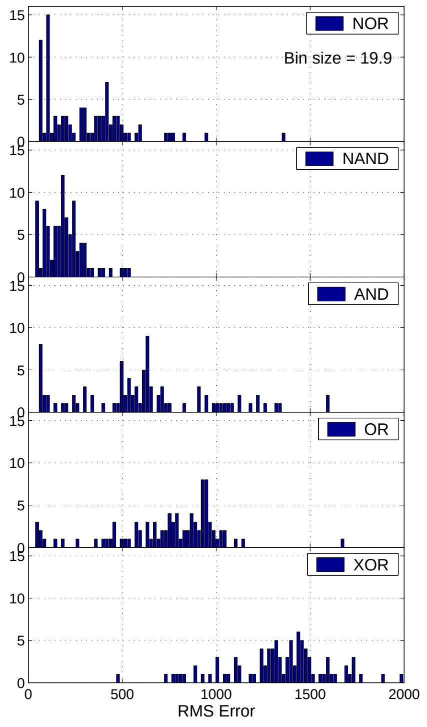


Figure 6. Results for the evolution of the DC behavior of different logic gates.

suggest that the evolution experiments were quite successful in finding solutions for the NOR and the NAND gate behavior, but had more difficulties in finding good solutions for the AND and OR problems. While for these four gates solutions with an error value less than $100\,\mathrm{mV}$ were found by the GA, this did not happen for the XOR problem, where the lowest error was observed to be $470\,\mathrm{mV}$.

The evolved gates are compared to the standard text book implementations shown in Fig. 7. It has to be noted that these solutions possess many features that are neither covered during the evolution nor the verification tests, as e.g. their dynamic behavior or their static power consumption. Thus a comparison of their DC behavior to the one exhibited by the evolved gates misses some points of their design goal. Nevertheless they are considered to be a good reference, because their quality of static behavior is sufficient to use them as building blocks for the design of logic circuits. Moreover, as was pointed out in section 3.1 the evolved circuits must at least manage to settle to their desired output in the order of $10\,\mu\mathrm{s}$, that is, they also have to satisfy a minimum of dynamic requirements (Information on the time scales inherent to the FPTA chip can be found in [9]).

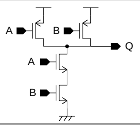
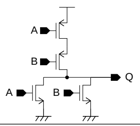
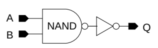
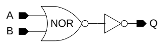
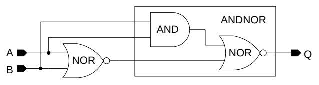
Gate	Implementation	Transistor Count
NOR		4
NAND		4
AND		6
OR		6
XOR		10

Figure 7. Typical CMOS implementation of the 5 logic gates NOR, NAND, AND, OR and XOR.

From Fig. 7 it can be seen that the complexity of these text book solutions increases from NOR and NAND to AND and OR to the XOR, indicated for example by the number of necessary transistors they are built of. A comparison between this complexity and the evolution results of Fig. 6 suggests that the difficulty in learning the according gate functionality corresponds to the difficulty in implementing it in CMOS technology.

The analysis of the results from the evolution exper-

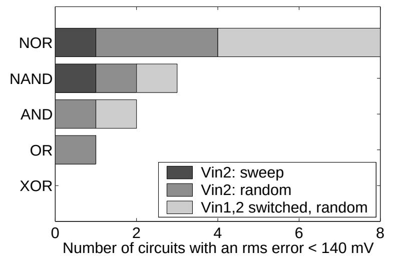


Figure 8. Number of evolved circuits that achieved an overall error smaller than 140 mV.

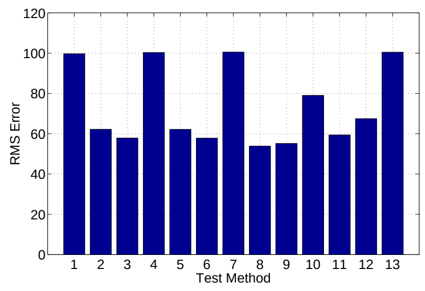


Figure 9. Error values for the best evolved NOR gate for all 13 different test methods. The test method numbers refer to Table 4.

iments done with V_{in2} being swept forward yields that many circuits successfully solving the problem for this test method perform poorly when one of the other testing methods listed in Table 4 is used. Hence, the real symmetric DC logic gate functionality was tested using the test schemes in Table 4. In order to get a feeling for the yield of the evolution experiments, the number of solutions that exhibit an error lower than $140\,\mathrm{mV}$ in all of the 13 tests is plotted in Fig. 8. The choice of a threshold of 140 is somewhat arbitrary, but an error below $140\,\mathrm{mV}$ seems realistic for a circuit that matches the desired output characteristic well. This is partially due to imperfections in the analog test circuitry mentioned above. As an example the error values obtained from the 13 different tests listed in Table 4 are shown in Fig. 9 for the best of the evolved NOR gates, whose DC characteristics are plotted in the upper left corner of Fig. 10.

The set of the applied test methods ensures that the successful circuits work on two different chips, indicating a minimum of robustness against environmental changes (cf. [20] for a definition of robustness in the context of hardware evolution). From Fig. 8 it can be observed that the naive fitness testing method used for the first 30 evolution runs (cf. Table 3) does not prevent the genetic algorithm from finding solutions generalizable to all testing methods. The effect of the evolution method is studied in more detail in section 4.1.2.

The performance of the best solutions of each of the 5 series of evolution experiments are plotted on the left side in Fig. 10, where 'best solutions' refers to those individuals whose maximum error value obtained from all 13 tests was the lowest in comparison to the according maximum error values from all the other best individuals obtained from the 90 evolution runs. For these measurements the voltage V_{in1} was varied in $0.5\,\mathrm{V}$ steps and V_{in2} swept from 0 to $5\,\mathrm{V}$ in $100\,\mathrm{mV}$ steps, now including the range from 2 to $3\,\mathrm{V}$. The error value in the title of each plot refers to the error achieved for test method 2 (see Table 4). The plots in the right column of Fig. 10 represent simulation results ob-

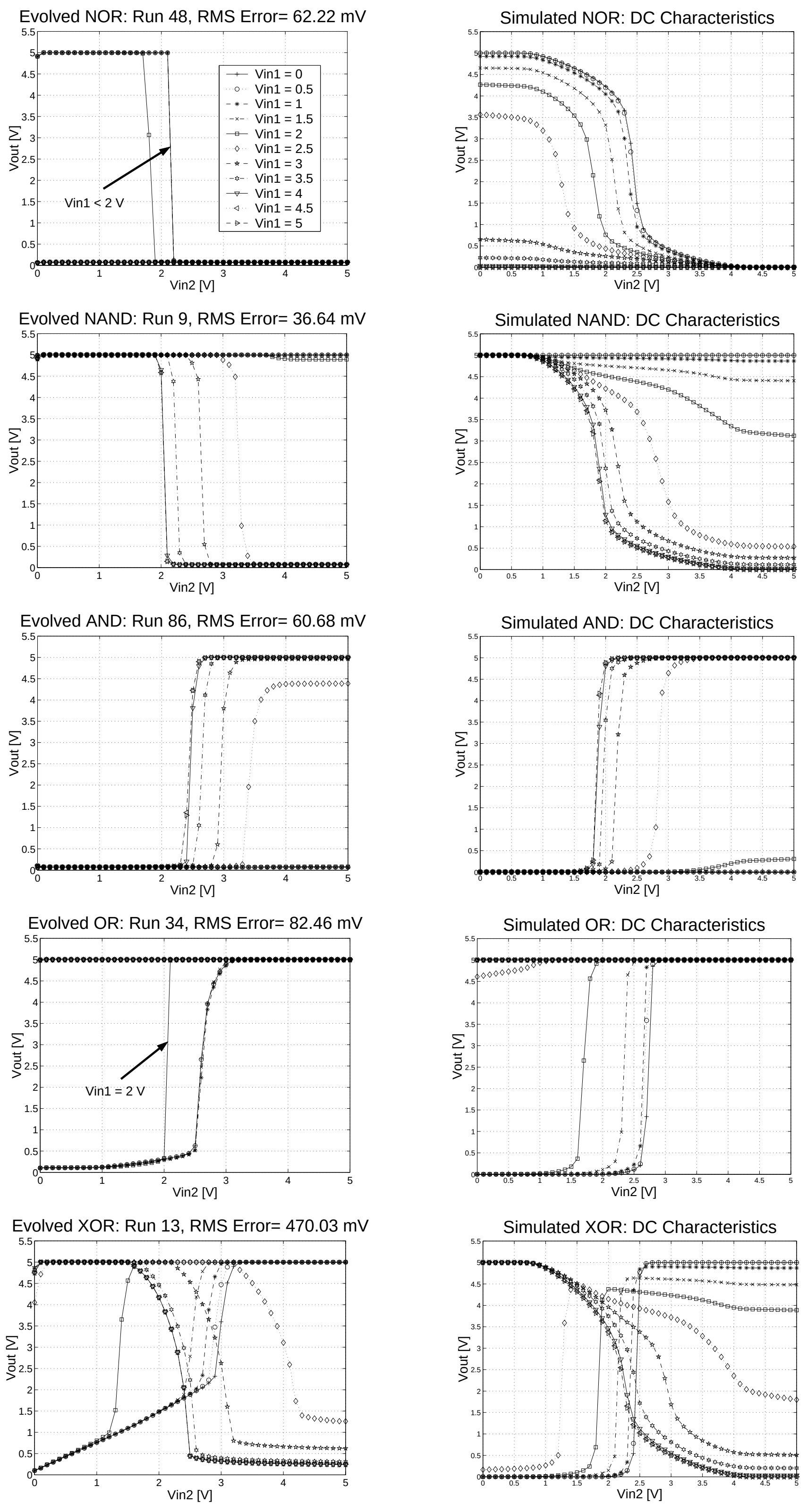


Figure 10. **Left: Measured performance of the best evolved gates. Right: Simulation results for the gates shown in Fig. 7. The legend shown in the plot in the upper left corner is used for all 10 plots.**

tained by simulating the circuits from Fig. 7 in the process the chip was fabricated in (for additional information on the FPTA chip the reader is referred to [9] or [10]).

It should be noted again that the simulated CMOS gates are not designed to meet the specifications posed to the ones evolved, but exhibit a bunch of advantageous features that are not considered here. However, a comparison of the DC characteristics of the evolved circuits and the standard CMOS gates yields the following insights: The circuits found for the NOR, AND, NAND and maybe even the one found for the OR problem exhibit DC characteristics that match the quality of the simulated text book solutions. This is not the case for the circuit found for the XOR problem, as was expected from its error value. The DC characteristics of the evolved AND gate look very similar to the simulation results. The DC behavior of the evolved NOR and NAND show higher amplification as their textbook solution counterparts. Furthermore the transition region of these evolved gates is narrower than the one of the simulated gates. This may indicate that the desired DC behavior of the target curve was more ambitious than necessary. This may in turn have mislead the genetic algorithm and thus may have decreased the yield of sufficiently good solutions.

As an example for the evolved circuits the one for the NOR gate is shown in Fig. 11. A straightforward explanation of the circuit has not been found yet, but it is planned to analyze the evolved circuits using simulations.

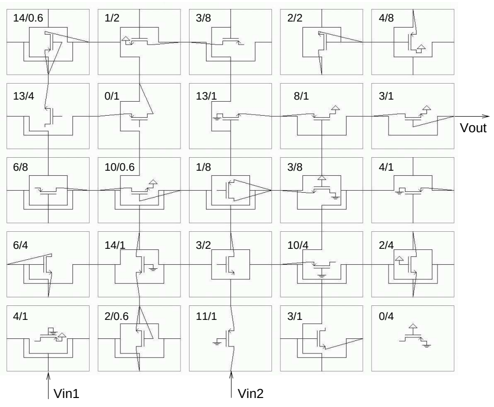


Figure 11. Circuit diagram of the best evolved NOR gate. The numbers in the left corner of each cell denote the W/L ratios of the transistors.

4.1.2 Dependence on the Evolution Method: NOR Gate

In order to analyze the impact of the three different testing paradigms all of the 30 evolution runs belonging to one testing paradigm are averaged for each of the 13 fitness measurements. The result for the evolution runs of the NOR gate are illustrated in Fig. 12.

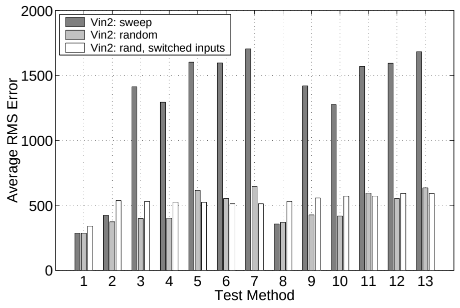


Figure 12. Mean error of all the evolved NOR circuits for each evolution method versus the 13 test methods. The test method numbers refer to Table 4.

The bars for the forward sweep testing paradigm are only low for exactly this test mode and otherwise in an error range suggesting that the according circuits, or at least most of them, since the error values are averaged over 30 runs, are useless under all the other testing conditions. Consequently, the genetic algorithm must have produced circuit solutions that do not only rely on the two different input voltages, but some other information inherent to the order of the presented test voltages. In contrast, the error averages for which the non static input voltage was chosen randomly and the static and the non static inputs changed roles randomly during the evolution are almost constant for test methods 2 to 13. The bars for the evolution runs without the interchange of input roles, exhibit better error values for the measurements done with the same input order used during their evolution. In conclusion, this data indicates that for the evolution experiments presented here it is necessary to include all input scenarios the circuit is to be able to cope with into the fitness evaluation during the evolution experiment.

4.2 Evolution of a Gaussian Output Voltage Characteristic

Since it is not clear what kind of circuits the artificial evolution finds that possess the desired Gaussian output characteristic, it's not possible to predict how many of the transistor cells will be necessary to solve the problem. Therefore the resources available to the chip were varied as described in section 3.3. In Fig. 13 the error of the best of the ten evolution runs done per edge length of the sub array available to the genetic algorithm is plotted versus the respective edge length. As far as can be inferred from the small number of runs carried out, the lowest error of the

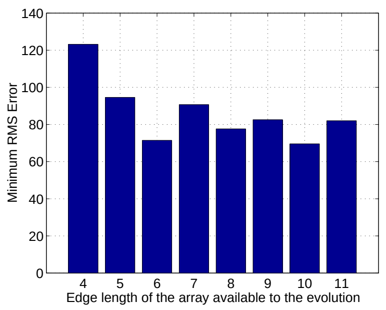


Figure 13. Error of the best individual versus the edge length of the chip area used.

evolved circuits is almost independent of the chip area available to the algorithm for higher edge lengths, but is higher for edge lengths of about 4 or 5. It has to be noted that this result was obtained with a number of 10000 generations. It may very well be that the algorithm could make better use of the higher number of transistor cells if the number of generations was higher. The DC curves obtained from the best individuals of the 9 series of runs are shown in Fig. 14.

In general the measured points of all the curves are located close to the desired output curve defined by Eq. 3. On the other hand, the obtained curves do not exactly look like Gaussian curves. Moreover they do not look smooth over the whole plotting range, but rather as if they were put together from several curve pieces. Usually most sub circuits, as for example current mirrors or differential pairs cannot operate properly over the full power supply range. Thus the only possibility to match the desired output behavior is to construct a circuit that does a piecewise approximation.

5 Conclusions and Outlook

Two different kinds of intrinsic hardware evolution experiments performed on a CMOS Field Programmable Transistor Array have been presented. The results show that the proposed evolution system is capable of finding quasi DC solutions for simple analog circuit design tasks. For the DC functionality of the four logic gates NOR, NAND, AND and OR perfect DC solutions have been found, while a perfect DC XOR behavior could not be obtained. The best successful circuits of each experiment are found to work on two different chips and thus are believed to be portable to different dice of the FPTA chip. The analysis of the effect of using different input test patterns during the evolution recommends to carefully design these input test patterns general and randomly enough as to prevent the search algorithm from using any information not belonging to the posed problem. The successful evolution of Gaussian voltage characteristics demonstrates that it is feasible to find circuits approximating basic mathematical functions with the evolution system presented.

Future experiments including the dynamic behavior as an objective for the artificial evolution are planned. Furthermore it will be interesting to see, if it is possible to simulate and analyze the successfully evolved circuits. After improving the test system the evolution of more complex circuits using bigger portions of the FPTA chip shall be tackled.

6 Acknowledgment

This work is supported by the Ministerium für Wissenschaft, Forschung und Kunst, Baden-Württemberg, Stuttgart, Germany.

References

[1] E. Henning, R. Sommer, and L. Charlack. An automated approach for sizing complex analog circuits in a simulation-based flow. Conference and Exhibition on Design Automation & Test in Europe (DATE 2002), Mar. 2002.

[2] A. H. Shah, S. Dugalleix, and F. Lemery. Technology migration of a high performance CMOS amplifier using an automated fron-to-back analog design flow. Conference and Exhibition on Design Automation & Test in Europe (DATE 2002), Mar. 2002.

[3] R. Phelps, M. J. Krasnicki, R. A. Rutenbar, L. R. Carley, and J. R. Hellums. A case study of synthesis for industrial-scale analog IP: Redesign of the equalizer/filter frontend for an ADSL CODEC. In *Proc. ACM/IEEE Design Automation Conference*, pages 1–6, Los Angeles, CA, USA, June 2000.

[4] M. J. Krasnicki, R. Phelps, R. A. Rutenbar, and L. R. Carley. MAELSTROM: Efficient simulation-based synthesis for custom analog cells. In *Proc. ACM/IEEE Design Automation Conference*, pages 945–950, New Orleans, LA, USA, June 1999.

[5] M. Murakawa, S. Yoshizawa, A. Toshio, S. Suzuki, K. Takasuka, M. Iwata, and T. Higuchi. Analogue EHW chip for intermediate frequency filters. In *Proc. ICES 1998, LNCS 1478*, pages 134–143, Lausanne, Switzerland, Sept. 1998. Springer Verlag.

[6] A. Stoica, D. Keymeulen, and R. S. Zebulum. Evolvable hardware solutions for extreme temperature electronics. In *Proc. of the Third NASA/DOD Workshop on Evolvable Hardware*, pages 93–97, Long Beach, CA, USA, July 2001. IEEE Press.

[7] D. Keymeulen, A. Stoica, R. Zebulum, Y. Jin, and V. Duong. Fault-tolerant approaches based on evolvable hardware and using a reconfigurable electronic devices. In *Proc. of the IEEE Int. Integrated Reliability Workshop*, pages 32–39, Lake Tahoe, CA, USA, Oct. 2000. IEEE Press.

[8] A. Thompson, P. Layzell, and R. S. Zebulum. Explorations in design space: Unconventional electronics design through artificial evolution. *IEEE Trans. on Evolutionary Computation*, 3:167–196, Sept. 1999.

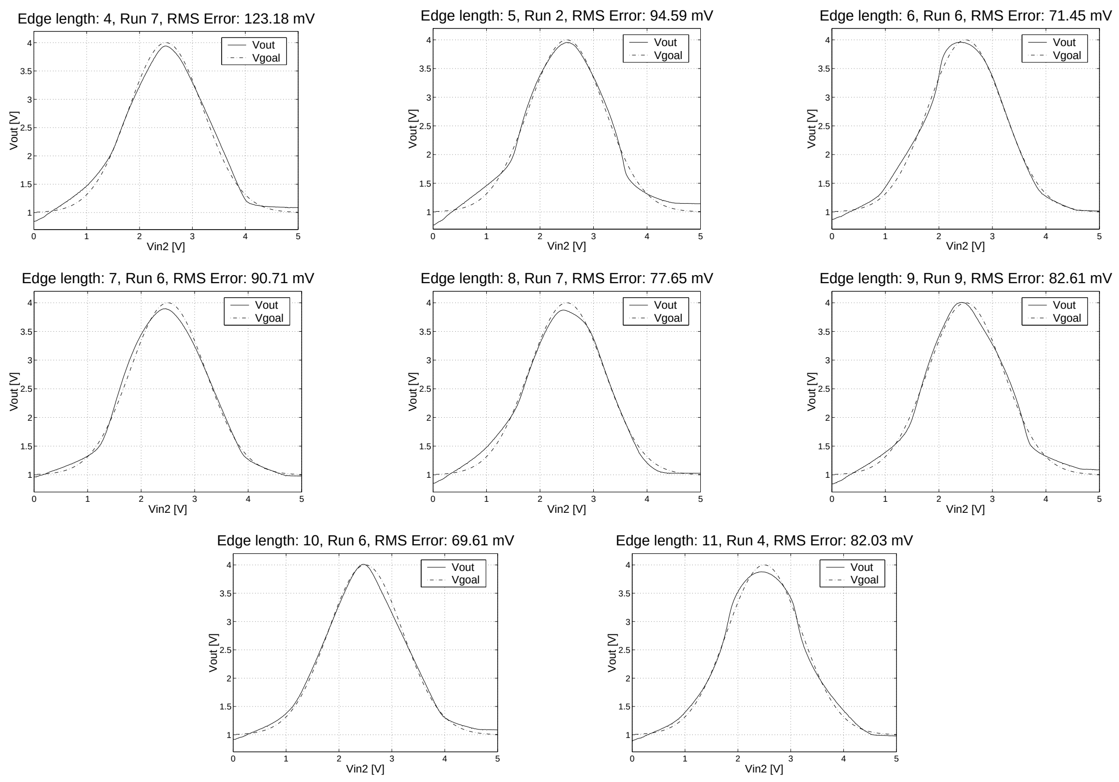


Figure 14. Output characteristic of the best Gaussian circuit for each available array size.

[9] J. Langeheine, J. Becker, S. F¨olling, K. Meier, and J. Schemmel. Initial studies of a new VLSI field programmable transistor array. In *Proc. ICES 2001, LNCS 2210*, pages 62–73, Tokio, Japan, Oct. 2001. Springer Verlag.

[10] J. Langeheine, J. Becker, S. F¨olling, K. Meier, and J. Schemmel. A CMOS FPTA chip for intrinsic hardware evolution of analog electronic circuits. In *Proc. of the Third NASA/DOD Workshop on Evolvable Hardware*, pages 172–175, Long Beach, CA, USA, July 2001. IEEE Press.

[11] J. Langeheine, S. F¨olling, K. Meier, and J. Schemmel. Towards a silicon primordial soup: A fast approach to hardware evolution with a VLSI transistor array. In *Proc. ICES 2000, LNCS 1801*, pages 123–132, Edinburgh, UK, Apr. 2001. Springer Verlag.

[12] R. Zebulum, M. Vellasco, and M. Pacheco. Evolutionary design of logic gates. In *Proc. of the Workshop in Evolutionary Design, AID98*, pages 12–17. Lisbon, July 1998.

[13] A. Stoica, R. S. Zebulum, and D. Keymeulen. Mixtrinsic evolution. In *Proc. ICES 2000, LNCS 1801*, pages 208–217, Edinburgh, UK, Apr. 2001. Springer Verlag.

[14] A. Stoica, R. S. Zebulum, and D. Keymeulen. Polymorphic electronics. In *Proc. ICES 2001, LNCS 2210*, pages 291–302, Tokio, Japan, Oct. 2001. Springer Verlag.

[15] F. H. Bennett III, J. R. Koza, M. A. Keane, J. Yu, W. Mydlowec, and O. Stiffelman. Evolution by means of genetic programming of analog circuits that perform digital functions. In *Proc. of the Genetic and Evolutionary Computation Conference*, pages 1477–1483, Orlando, Florida, USA, July 1999. Morgan Kaufmann.

[16] C. C. Santini, R. S. Zebulum, M. A. C. Pacheco, M. M. R. Vellasco, and M. H. Szwarcman. Evolutionary experiments with a fine-grained reconfigurable architecture for analog and digital CMOS circuits. In *Proc. of the Third NASA/DOD Workshop on Evolvable Hardware*, pages 36–43, Long Beach, CA, USA, July 2001. IEEE Press.

[17] S.-Y. Lin, R.-J. Huang, and T.-D. Chiueh. A tunable gaussian/square function computation circuit for analog neural networks. *IEEE Transs on Circuits and Systems II: Analog and Digital Signal Processing*, 45(3):441–446, Mar. 1998.

[18] A. Stoica, D. Keymeulen, R. S. Zebulum, A. Thakoor, T. Daud, G. Klimeck, Y. Jin, R. Tawel, and V. Duong. Evolution of analog circuits on field programmable transistor arrays. In *Proc. of the Second NASA/DOD Workshop on Evolvable Hardware*, pages 99–108, Palo Alto, CA, USA, July 2000. IEEE Press.

[19] A. Stoica, G. Klimeck, C. Salazar-Lazaro, D. Keymeulen, and A. Thakoor. Evolutionary design of electronic devices and circuits. In *Proc. of the Congress on Evolutionary Computation*, pages 1271–1278, Washington DC, USA, July 1999. IEEE Press.

[20] A. Thompson. On the automatic design of robust electronics through artificial evolution. In *Proc. ICES 1998, LNCS 1478*, pages 13–24, Lausanne, Switzerland, Sept. 1998. Springer Verlag.