GRACE: Generative Robust Analog Circuit Exploration

Michael A. Terry, Jonathan Marcus, Matthew Farrell, Varun Aggarwal, Una-May O'Reilly

Computer Science and Artificial Intelligence Lab (CSAIL)
Massachusetts Institute of Technology
Cambridge, MA, USA
m_terry@alum.mit.edu, unamay@csail.mit.edu

Abstract. We motivate and describe an analog evolvable hardware design platform named GRACE (i.e. Generative Robust Analog Circuit Exploration). GRACE combines coarse-grained, topological circuit search with intrinsic testing on a Commercial Off-The-Shelf (COTS) field programmable device, the Anadigm AN221E04. It is suited for adaptive, fault tolerant system design as well as CAD flow applications.

1 Introduction

With our Generative Robust Analog Ciruit Exploration (GRACE) tool we are investigating whether it is possible to evolve circuits that can be realized efficiently and in a routine manner. We are focusing upon the domain of analog circuit design. Our decision is motivated by the lack of automation in analog design as compared to digital design. We intend to investigate whether evolvable hardware (EHW) approaches can contribute to the complex, human-intensive activity domain of analog design.

The goal of this paper is to describe how we arrived at GRACE. By combining the exploitation of coarse grained elements with intrinsic testing, we think GRACE sits in an interesting and novel space. It allows a distinctive foray into on-line adaptive and fault tolerant evolvable hardware circuits since it uses a COTS (Commercial-Off-The-Shelf) device and standard components. This should make its results more acceptable to industry. It also allows an economical and time efficient foray into the broad domain of VLSI and CAD with its use of elements that are conversant with human design. We proceed thus: In Section 2 we describe how we reached a decision to select the Anadigm AN221E04 as GRACE's reconfigurable device. In Section 3 we give an overview of GRACE. In Section 4 we describe GRACE's genetic representation of a circuit and its search algorithms. In Section 5 we design a controller for a plant using GRACE to demonstrate its ability to evolve circuits. We conclude with a summary.

2 Choosing GRACE's Reconfigurable Device

For GRACE, the choice of its reconfigurable device was driven by three criteria (see [1] for a related discussion):

1. A desire to work at an abstraction level where the human design principles are *inherent* in the building blocks so that GRACE will derive conventional, human-competent, portable and robust circuits;
2. Availability of a reconfigurable device that matched the project's budget of $5K;
3. Flexibility that would allow design elements to be chosen depending upon the design problem, (i.e. level of hierarchy in the analog design flow).

Among the devices we assessed for our purposes were the class of custom designed Field Programmable Transistor Arrays (FPTA), the Lattice Semiconductor ispPAC10 field programmable analog array (FPAA, e.g. [2]), and the Anadigm FPAA family ([3]).

With respect to Criterion 1, there are open questions regarding the suitability of an FPTA for evolving conventional, human-competent circuits:

1. Can an FPTA be configured to respect certain design principles so that interconnections of the transistor-switched cells and inter-cell topology will constitute circuits that a human engineer will trust? Some of these design principles such as no floating gates could be encoded in the circuit construction and circuit modification functions of an evolutionary algorithm. However, not all expectations/insights (such as parasitic insensitive connections) can be mapped into rules.
2. Can the non-idealities arising from the switching elements in the FPTAs be circumvented to avoid reliability, portability and engineering acceptance issues? The FPTAs require electronic switches that are implemented by transmission gates. This adds parasitic non-linear resistance and capacitance, which results in delay, de-amplification and alteration in frequency domain properties. While some of the evolved circuits to date *use* these non-idealities of switches as an integral feature of the design [4], realistically this makes the evolved design idiosyncratic (i.e. unportable) and unreliable since the behavior of its switches is neither characterizable or replicable.
3. Is there sufficient flexibility for sizing? Industry practice is to explore sizing options as a means of balancing functional goals and specifications. Because it only has one size of transistor, the FPTA-2 ([1])is constrained in this respect.

FPTAs are well suited for the exploration of non-conventional realms of circuit design such as polymorphic circuits, extreme temperature electronics and fault tolerant circuits [5–7]. However, they are not well suited to our desire to explore robust, novel topologies of interpretable and portable circuits.

Circuit synthesis with opamps has straightforward and methodical design rules (which can be easily incorporated in the evolutionary algorithm) to ensure

that the evolved circuit is interpretable and robust. The IsPAC10 and Anadigm FPAA have circuit elements based on opamps. The IsPAC10, see Figure 1 right, consists of 4 programmable analog modules (4 opamps, and 8 input amplifiers total) interconnected with programmable switching networks. Configuration of the IsPAC10 is a proprietary process [2] . The Anadigm AN221E04, see Figure 1 left (and described in more detail in Section 2.1), also provides opamp based circuits as building blocks. It uses switched-capacitor technology which is inherently robust and portable.

With respect to Criterion 2, the cost of an FPTA is beyond $5K. The development board of an IsPAC10 or Anadigm AN221E04 has a cost in the low hundreds of dollars. Integrated with a conventional computer and other signal processing devices, they facilitate a system with cost below $5K.

Wth respect to Criterion 3, each device we assessed offers a different level of circuit element granularity. The FPTAs are very flexible, fine-grained devices The U. of Heidelberg's FPTA ([8], henceforth called FPTA-H) is a switched network of 256 (16 X 16) programmable CMOS transistors (half NMOS and half PMOS) arranged in a checkerboard pattern.

The FPTA-1 designed at JPL ([4, 9]) is composed of 12 cells, where each cell contains 8 CMOS transistors interconnectable via 24 switches. The transistors are fixed size and the switches are electronically programmable. The FPTA-1 appears to have been a prototype device for FPTA-2. The FPTA-2 ([1]) contains an 8X8 matrix of 64 reconfigurable cells, where each cell consists of 14 transistors interconnectable via 44 switches. The transistors are fixed size. Each cell also contains three capacitors, two reconfigurable resistors and photodetectors. It fits into the Evolution-Oriented Reconfigurable Architecture (EORA) and is integrated with a DSP processor running the evolutionary algorithm to form the SABLES (Stand-Alone Board-Level Evolvable System).

FPTA-H is more versatile than FPTA-2 with respect to interconnection and sizing of transistors. In FPTA-H, in general any transistor can connect to any other transistor, while in FPTA-2, transistors are arranged in a particular topology with switches to realize different circuits. Even though the FPTA-2 cell has 44 switches which creates a large space of possible realizable topologies, there are human-conceivable designs which cannot be directly synthesized using it. In FPTA-H, 75 different aspect ratios could be chosen for each transistor, while the FPTA-2 uses fixed length transistors. This flexibility of FPTA-H comes at the cost of space (equivalent to the number of transistors that can be fabricated on the same chip). The FPTA-2 has 3.5 times more transistors on the same chip as compared to FPTA-H. This difference may also be attributed to the fact that FPTA-2 uses 0.18μm process, while FPTA-H uses 0.6μm process.

The IsPAC10 and Anadigm AN221E04 exemplify a tradeoff between flexibility and appropriate building block abstraction. The opamp is a building block that can be combined with passive components to arrive at variety of human-designed circuits such as amplifiers, integrators, differentiators, sum-difference amplifiers, or filters. However, it is not as flexible as a switched transistor array. On the IsPAC10, there is very limited interconnect between a small quantity of

resources. The Anadigm AN221E04 provides a fixed abstraction level of opamp based circuits but supports very flexible interconnection.

After our assessment, we chose the Anadigm AN221E04 over the IsPAC10 or an FPTA. In a nutshell, we have forgone a large degree of flexibility by choosing a fixed abstraction level (of opamp based circuits) in order to ensure robustness, portability and reliability. Nonetheless we are content given that there are a number of analog design problems (such as PID controllers, ADCs and filters) which can be addressed by the given design abstraction. More details of the Anadigm AN221E04 are provided in the next section.

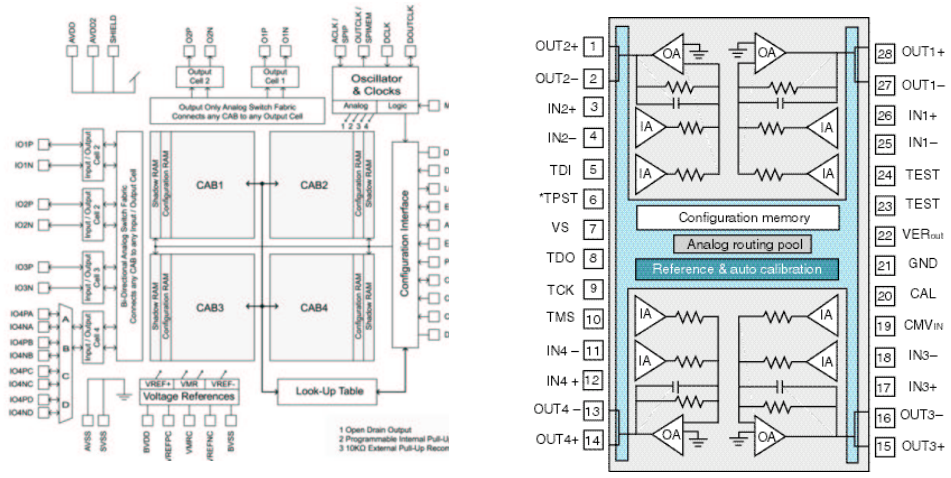

Fig. 1. Reconfigurable FPAA Architectures: Anadigm(Left), Lattice ispPAC10(right)

2.1 The Anadigm FPAA

For detailed description of the Anadigm Vortex family of devices, see [3].

Resources: The Anadigm AN221E04 is an array of CABs (configurable analog blocks), each of which contain two opamps, 8 capacitors, a comparator, and a Successive Approximation Register (SAR) that performs 8-bit analog-to-digital conversion of signals. The device also contains one programmable lookup table that can be used to store information about the generation of arbitrary waveforms, and is shared amongst the CABs. The architecture is illustrated in the left hand block diagram of Figure 1. Any signal can be routed to the I/O pins of the device through 4 programmable I/O interface blocks and two dedicated outputs, each of which can also act as a filter or amplifier. The option for expanding the number of resources is to daisy chain multiple devices.

Configurable Elements: Despite the existence of opamps and switched capacitors, the Anadigm AN221E04 does not support circuit design at this level of granularity. Instead, a circuit must be specified at the abstraction of coarser

grained building blocks termed Configurable Analog Modules (CAMs) that are interconnected by wires. CAMs come predefined by Anadigm. See Table 1 for the set of available CAMs. Among the broad set is a flexible selection of filters, amplifiers and rectifiers that designers frequently use. Each CAM has programmable *options* and parameters. For example, the SumDiff CAM has a set of 4 options which decide upon clock phase, optional use of inputs 3 and 4, and inversion of each input. Its parameters are its two or more gains. To insert a CAM, the GUI must be able to fully allocate its resources from one CAB. To track how many resources are available as a circuit is defined, we reverse engineered the resource allocation strategy of the Anadigm software for GRACE.

Configuration Technology: The Anadigm FPAA uses the 'switched capacitor' technology ([10]). A switched capacitor implements an equivalent resistance by alternately opening and closing the terminals of a capacitor. Macroscopic resistance is controlled by the frequency of switching. This frequency, of course, is limited to the maximum clock frequency. Microscopic resistance is tuned by changing the capacitance value. The disadvantage of switched capacitor technology is that it performs the signal processing in discrete time domain. Thus, it requires anti-aliasing and reconstructions filters. Also, the device can only handle signals whose frequency is half its switching frequency, which is 16MHz maximum. For all blocks of the FPAA, the input and output are valid either for one of the two phases of clock or both phases. This implies a constraint on the connection of components, since a component whose output is valid on phase 1 cannot be connected to a component whose input is sampled at phase 2 of the clock and vice versa. Each internal capacitor in the Anadigm AN221E04 is drawn from a bank of capacitors. Although the software allows for the generation and routing of signals between CAMs at design time, the software only allows *dynamic* reconfiguration of the options and parameters of a circuit, not the reconfiguration of a circuit topology. The actual configuration process and mapping of the configuration bitstream is proprietary. The configuration bitstream is stored in SRAM, which is more reliable than other FPAAs based on EEPROM technology.

Configuration from GRACE: The configuration process of the Anadigm AN221E04 is proprietary. With the assistance of a colleague [11] and through a special agreement with Anadigm, we obtained a non-commerical software package that had been developed to test the GUI during product development. With this package and a Microsoft C++ compiler, GRACE sends designs from its EA module to the GUI by translating them to a series of "build commands" dispatched to the GUI. A subsequent "configure" command downloads the configuration to the device. This takes about a second which is not ideal but not prohibitive either.

While we are restricted to low to medium frequency range, we nonetheless are content. An industry segment also works in this domain due to the use of switched capacitor technology so there an industry target for whom evolutionary techniques may be useful exists.

CAM	CAM	CAM
Voltage Transfer Function	Inverting Differentiator	Divider
Half cycle inverting Gain Stage	Biquadratic Filter	Half Cycle Gain Stage
Half Cycle Sum/Difference Stage	DC Voltage Source	Inverting Gain Stage
Gain Stage: Switchable inputs	Bilinear Filter	Integrator
Gain Stage: Polarity Control	Half Cycle Rectifier	Half Cycle Gain Stage
Gain Stage - Output V Limiting	Inverting Sum Stage	Multiplier
Rectifier with Low Pass Filter	Sample & Hold	Sinewave Oscillator
Transimpedance Amplifier		Waveform Generator

Table 1. Anadigm AN221E04 CAMs

3 GRACE: The system

GRACE is depicted in Figure 2 which shows an adaptive controller on the FPAA that controls a third order plant. The evolutionary algorithm (EA) runs on an Pentium P4 machine. It reconfigures the Anadigm AN221E04 to build new controllers, evaluate their efficiency in controlling the plant and thus guide the search to find better controllers. A summary of the components is given in Table 2.

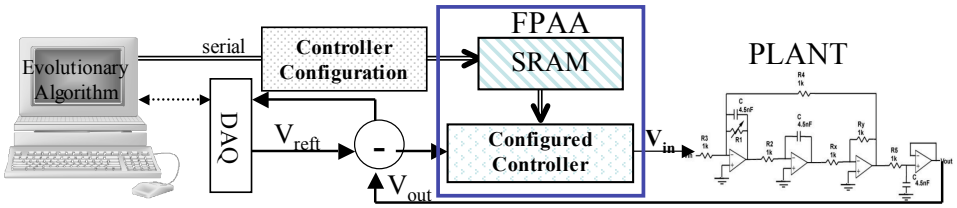

Fig. 2. GRACE Architecture.

The Anadigm AN221E04 is configured by the EA via the serial port. The EA sends inputs to the hardware and extract outputs via National Instrument's PCI-6221 multifunction data acquisition card (DAQ). (The PCI-6221 DAQ board provides up to 80 analog inputs and 4 analog outputs giving GRACE scalability). The DAQ provides both analog to digital and digital to analog conversion with 16-bit resolution (for a voltage range of -10V to 10V). The reference signal to the testbench is specified by the algorithm to the DAQ as a digital waveform. The DAQ converts it to an analog signal and sends it to the testbench. Simultaneously, the DAQ converts the plant's analog output signal to a digital signal for the evolutionary algorithm to compare with the reference signal. Our system actually duplicates the reference signal sent to the controller to be matched with the plant output. This yields a time synchronized comparison between the reference and plant signals.

Component	Specifications	Procured From	Price
Dell Dimension 8400	3.6GHz P4 CPU, 2GB RAM	Dell Computers	$2200 + cost of monitor
FPAA Development Kit	PCB with FPAA and 2 Signal Conditioning Dual-opamps	Anadigm	$200
AnadigmDesigner2	Configuration Software For Win32 Platform	Anadigm	free
AutomationDoc	Documentation for Anadigm GUI Scripting	Anadigm Support	free
NI 6221 DAQ	Multifunction DAQ with analog output, PCI Card	National Instruments	$430
NI Connect Block and Cable	Shielded Connection Block with Cable to Interface to PCI DAQ card	National Instruments	$350

Table 2. GRACE: System Components, specifications, sources and cost.

4 Choosing a Genetic Representation

The genetic representations of the evolvable hardware community have ranged from directly expressing the configuration bitstream to expressing a circuit component representation. A prominent example of the first extreme are the projects by A. Thompson [12] and his co-authors who used the Xilinx 6216. At the other is the "circuit constructing tree" which is a developmental encoding, e.g. [13]. In contrast, in GRACE a subset of the Anadigm CAM's are the functions in the sense of genetic programming. All CAMs with valid output for only one of the clock phases had their outputs connected to a "Sample and Hold" component. The GRACE genome is a cyclic graph (see Figure 3) in which each node is an instance of a CAM and directional links define the topology. A circuit has a variable number of CAMs but we implement the graph as a fixed length vector. Each element of the vector is a structure which specifies a CAM, its options, parameters and input source(s). Each instance of a CAM has a variable number of programmable options and parameters. For example, the SumDiff CAM has 4 options and 2 gain parameters while the simple "Half cycle Gain Stage" has only 2 options and 1 gain parameter. The genome stores in each structure another two vectors of data that the genome-to-circuit translation process interprets as parameters and options. Each vector is a fixed length. If the parameters and attributes of the CAM are fewer than the vector length, the extra values are ignored. Like the redundant nodes and links of the circuit which do not connect input to output, this redundant information is maintained in the genome.

The encoding of coarse grained components in the genome makes GRACE reminiscent of Koza's genetic programming tree representation, e.g. in [14]. The obvious difference is the cyclic graph versus the tree. Another difference is the genome length – fixed in GRACE's case and variable in Koza's. The physical limitation of resource quantities on the device demand that GRACE not evolve a genome that requires more resources than on the device. This is ensured by

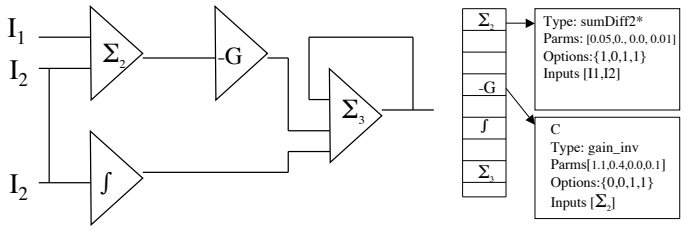

Fig. 3. Left: A circuit in GRACE is a graph. Nodes are components and edges are wires. Right: This graph is stored in a fixed length linear genome. Each object of the genome is a structure describing component, options, parameters and inputs.

the fixed length genome and by the decoding algorithm that maps the genome to a series of build commands. The decoding algorithm makes use of a resource manager to account for resources that will be used on the device as it translates the genome into "build" commands. If it ever encounters a CAM (i.e. node) for which the resources cannot be allocated, it replaces this node with a wire. GRACE's genome is also influenced by Miller's Cartesian Genetic Programming (CGP), [15]. The CGP genome is also a graph mapped to a matrix of varying component with links between and among columns.

The search algorithms: We use the standard generation based processing loop of an EA to conduct topology search. At initialization, a population of random genomes is created. Each genome is mapped to a circuit topology with each instance of a CAM specified using its input list, options and parameter values. Serially each genome is configured on the device and given a test signal. The resulting output signal is captured and evaluated in comparison to a desired output signal. The error is mapped to a genome fitness. After the entire current generation is tested, tournament selection supplies parents for the next generation. Each parent is copied to create an offspring in the next generation. Offspring are mutated before being added to the population of the next generation. Mutation can be applied in two ways to the genome: to a CAM instance by changing its type and, to the input(s) of a CAM by changing a link in the graph.

Given a topology, finding the parameters of the CAM is a numerical optimization problem. Recently, Particle Swarm Optimization (PSO) has emerged as an efficient approach to do fast numerical optimization to find the global optimum [16]. We use PSO to set the parameters of CAMs rather than evolving it together with the topology by the EA. We believe that performing the steps of topology search and component optimization separately makes the problem more tractable for the EA. These two steps of topology search using an EA and component optimization using PSO can be combined in various ways which shall effect the efficiency and speed of the algorithm. For the current set of experiments, we run PSO on each individual in the EA population and assign the best of swarm fitness to the individual. Intuitively, this approach assigns the topology fitness according to its best performance given the most suitable parameter val-

ues. Other approaches which trade speed for efficiency and vice-versa are under study.

5 GRACE in Action: Evolving a Controller

We have initially used GRACE to evolve a controller for the simple first order plant shown in Figure 4 (left). The plant has bandwidth of 338.6Hz and a steady state gain of 2.5. The CAMs in the primitive set for the given problem can be found in Table 3 along with their respective parameters and options. These CAMs are capable of creating any transfer function (realizable given the capacity of the FPAA) including the ubiquitous Proportional-Integral-Differential control. The population size was 15 with tournament selection of size 3 and elitism for 3 individuals. A run was 10 generations with the probability of mutating a CAM 0.45 and a wire 0.45. The PSO ran 6 iterations every generation on every individual with a swarm size of 4.

Fitness Function Though a simple step function would seem to be all that is required to evaluate a controller, we used a more complex signal to ensure that GRACE did not evolve a signal generator regardless of the input. The signal and a candidate circuit's response is shown on the right in Figure 4. The signal has six voltage levels (-1.5V, -0.75V, -0.375, 0.375, 0.75V. and 1.5V) and changes state every 4.16ms. We sampled the signal at 125 KHz. The fitness of a circuit is the weighted sum of squared errors between the circuit's output signal and the test signal. The fitness function weights can be tuned to trade-off criteria of settling time, peak overshoot and steady state error. For instance, more weight to the error in latter part of the step response shall bias the search towards controllers with lower steady state error and shall care less for rise time and peak overshoot. For the current set of experiments, we used the time-weighted least squares, which increases the weights linearly with time. It is postulated in [17], such a fitness function is ideal for judging the efficiency of a controller.

Evolved Solutions The system evolved solutions with high fitness value (validated by visual inspection of generated waveforms) that instantiate various control strategies, for instance, proportional control, integral control or lossy integral control. Evolution also found interesting ways to build solutions, like use of a differentiator in feedback to evolve a lossy integrator and using multiple feedback to realize different gains (including high gain through positive feedback) for proportional control.

Analysis of one of the best-of-run controller showed how evolution can *think* out-of-the-box. Figure 5 shows the controller as seen in the Anadigm GUI on the left and the equivalent simple block diagram on the right. Simple hand-analysis shows that the solution is a filter. The summing-integrator filter topology is a well-known approach to synthesize filters. It is counter-intuitive why a filter would be a good controller. Evolution exploits the high integration-constant (of the order of Mega per second) realizable by the integrator. It evolves a high gain filter with a large bandwidth that has an integrator in both the forward and feedback paths. This effectively behaves like proportional control with a

large gain. The high gain of the P-control reduces the steady-state error thus contributing to high fitness. This solution has not been included in discussion for its usefulness in a real scenario, but due to illustrate the ability of algorithm to synthesize interesting topologies and the capability of evolution to explore realms of unconventional design even when it uses coarse-grained building blocks.

Work is underway to study the solutions generated and use a carefully crafted fitness function to better capture the characteristics of the controller. With an effective fitness function instantiated in the system, we shall determine the usefulness of the circuits evolved and compare them with those evolved on other platforms such as the FPTAs. We also plan to study how variation in the evolutionary algorithm (method/parameters) affects its ability to search for a solution in the given problem domain.

CAM	**Parameter(s)**	**# In**
SumDiff-2*	inputs gain value(s)	2
SumDiff-3*	inputs gain value(s)	3
Inverting Differentiator	diff. constant (us)	1
Integrator	gain	1
Gain Inverter	gain	1
Gain*	gain	1
Wire	0	1

Table 3. CAMs used in the GRACE Function Set for Controller Evolution. Asterisk indicates output is connected to sample and hold block for two clock phase results.

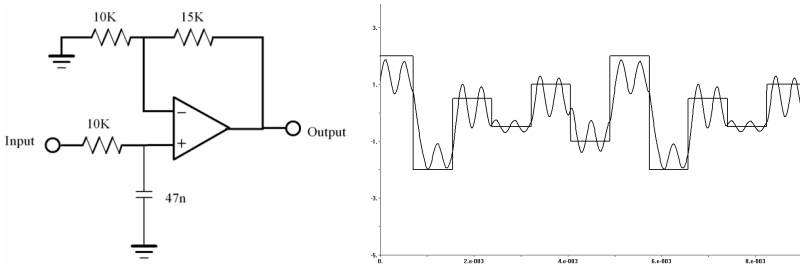

Fig. 4. Left: Plant for evolved controller, Right: Fitness function test signal (square wave) with example circuit's output signal for the controller experiment.

6 Summary

By combining the exploitation of coarse grained elements with intrinsic testing on a COTS device, we think GRACE comprises a distinctive approach to analog EHW. This paper's goal has been to elucidate our decision process in engineering GRACE. We feel our decision to use the Anadigm AN221E04 forges GRACE's

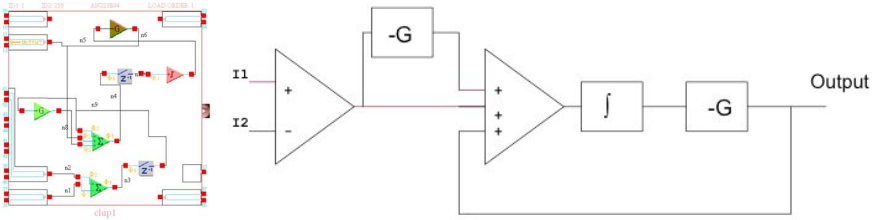

Fig. 5. Left: An evolved filter solution displayed from GUI , Right: Schematic of same solution.

identity. It is a COTS rather than custom device. The proprietary nature of its configuration process can be circumvented for practicality. It uses SRAM to hold a configuration. This makes it suited as a component of an adaptive, fault tolerant system. It exploits switched capacitor technology. This allows its evolved designs to conform with industry specifications and be realizable. This will facilitate the ultimate placement of evolved circuits in the field.

Finally, in using the Anadigm AN221E04, it offers coarse grained elements. Coarse granularity makes GRACE contrast with FPTA approaches by exchanging flexibility with higher level building block abstraction. We think that GRACE enables a parallel set of investigations that will provide interesting comparisons between the non-linear design space of the FPTA and the human oriented, conventional design space. We believe our choices additionally provide us with traction into both adaptive, robust hardware evolution and the more traditional pursuit of analog CAD. This will be the direction of our future research using GRACE.

Acknowledgements

We would like to thank Anadigm, Dimitri Berensen, Garrison Greenwood, David Hunter, Didier Keymeulen, Adrian Stoica, and Eduardo Torres-Jara for their contributions to the development of GRACE.

References

1. Stoica, A., Zebulum, R.S., Keymeulen, D.: Progress and challenges in building evolvable devices. In: Evolvable Hardware. (2001) 33–35
2. Greenwood, G., Hunter, D.: Fault recovery in linear systems via intrinsic evolution. In Zebulum, R., Gwaltney, D., Hornby, G., Keymeulen, D., Lohn, J., Stoica, A., eds.: Proceedings of the NASA/DoD Conference on Evolvable Hardware, Seattle, Washington, IEEE Computer Society (2004) 115–122
3. Anadigm: AN221E04 datasheet: Dynamically reconfigurable fpaa. http://www.anadigm.com/doc/DS030100-U006.pdf (2004)
4. Stoica, A., Zebulum, R., Keymeulen, D., Tawel, R., Daud, T., Thakoor, A.: Reconfigurable vlsi architectures for evolvable hardware: from experimental field programmable transistor arrays to evolution-oriented chips. IEEE Transactions on VLSI Systems, Special Issue on Reconfigurable and Adaptive VLSI Systems **9**(1) (2001) 227–232

5. Stoica, A., Zebulum, R.S., Keymeulen, D.: Polymorphic electronics. In Liu, Y., Tanaka, K., Iwata, M., Higuchi, T., Yasunaga, M., eds.: ICES. Volume 2210 of Lecture Notes in Computer Science., Springer (2001) 291–302
6. Keymeulen, D., Stoica, A., Zebulum, R.: Fault-tolerant evolvable hardware using field programmable transistor arrays. IEEE Transactions on Reliability, Special Issue on Fault-Tolerant VLSI Systems **49**(3) (2000) 305–316
7. Stoica, A., Keymeulen, D., Zebulum, R.S.: Evolvable hardware solutions for extreme temperature electronics. In: Evolvable Hardware, IEEE Computer Society (2001) 93–97
8. Langeheine, J., Becker, J., Fölling, S., Meier, K., Schemmel, J.: Initial studies of a new vlsi field programmable transistor array. In Liu, Y., Tanaka, K., Iwata, M., Higuchi, T., Yasunaga, M., eds.: Evolvable Systems: From Biology to Hardware: Proceedings of 4th International Conference, ICES 2001. Volume 2210 of Lecture Notes in Computer Science., Tokyo, Japan, Springer-Verlag (2001)
9. Stoica, A., Zebulum, R., Ferguson, M., Keymeulen, D., Duong, V.: Evolving circuits in seconds: Experiments with a stand-along board-level evolvable system. In Stoica, A., Lohn, J., Katz, R., Keymeulen, D., Zebulum, R.S., eds.: Proceedings of the NASA/DoD Conferenece on Evolvable Hardware, Alexandria, Virginia, IEEE Computer Society (2002) 129–130
10. Allen, P.E., Sanchez-Sinencio, E.: Switched Capacitor Circuits. VanNostrand Reinhold Company (1984)
11. Berenson, D.: Personal communication. email: January 18, 2005 (2005)
12. Thompson, A.: Hardware Evolution: Automatic design of electronic circuits in reconfigurable hardware by artificial evolution. Springer-Verlag (1998)
13. Koza, J.R., Bennett III, F.H., Andre, D., Keane, M.A.: Four problems for which a computer program evolved by genetic programming is competitive with human performance. In: Proceedings of the 1996 IEEE International Conference on Evolutionary Computation. Volume 1., IEEE Press (1996) 1–10
14. Koza, J.R., Kean, M.A., Streeter, M.J., Mydlowec, W., Yu, J., Lanza, G.: Genetic Programming IV:Routine Human-Competitive Machine Intelligence. Kluwer Academic Publishers (2003)
15. Miller, J.F., Thompson, P.: Cartesian genetic programming. In: Proceedings of Third European Conference on Genetic Programming, Springer-Verlag (2000) 121–132
16. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the Fourth IEEE International Conference on Neural Networks, IEEE Press (1995)
17. Krohling, R.A., Jaschek, H., Rey, J.: Designing PI/PID controllers for a motion control system based on genetic algorithms. In: Proceedings of the 12th IEEE International Symposium on Intelligent Control. (1997) 125–130