# ENHANCED GENETIC ALGORITHM KERNEL APPLIED TO A CIRCUIT-LEVEL OPTIMIZATION E-DESIGN ENVIRONMENT

M. Barros[1,2] J. Silva[3] G. Neves[1] N. Horta[1]

[1]IST/IT - Centre for Microsystems
Av. Rovisco Pais,
1049-001 Lisboa, Portugal

[2]IPT – Inst. Pol. de Tomar
Qt. do Contador – Est. de Serra
2300-313 Tomar, Portugal

[3]INESC-ID
R. Alves Redol, 9,
1000-029 Lisboa, Portugal

## ABSTRACT

This paper presents a circuit/system-level optimization E-Design environment based on an enhanced modified genetic algorithm kernel. The presented approach improves the standard genetic algorithm implementation by including automatic search space decomposition, premature convergence prevention procedures and distributed processing features. In order to cover a broad range of solutions in terms of circuit/system complexity within a realistic execution time both behavioral simulation, using equation-based descriptions, and electrical simulation, using Spice-like simulators, are considered. Moreover, an E-Design front-end allowing an incremental growth of the IC design database, an individual management of each project and the widespread training on the design flow procedures is being developed. Finally, the achieved increase on optimization efficiency, compared to a standard genetic algorithm implementation, as well as the general purpose of the described approach are illustrated by a multi-objective, multi-constraint optimization of some well known circuits.

## 1. INTRODUCTION

The semiconductor market trends indicate a rapid increase of ICs production containing both analog and digital functionalities. Thus, in order to satisfy the need to reduce product development cycles, a significant increase on productivity is required. Therefore, the development of Design Automation (DA) tools became a key factor to improve the effectiveness of ICs design flow by reducing costs and time-to-market. However, and despite some significant research efforts, a full deployment of analog CAD tools into professional environments is still very limited. In this way, the performance improvement on DA tools, addressing key points, such as analog circuit optimization and layout retargeting for different specifications and technologies [1-3], is mandatory in order to fulfill industry requirements.

The work presented in this paper addresses the analog circuit/system optimization problem based on an enhanced modified genetic algorithm (GA) kernel which uses both behavioral simulation, with equation-based descriptions, and electrical simulation, with Spice-like descriptions, for performance analysis purposes. In the following sections the enhanced optimization kernel is discussed, some optimization examples are presented and the E-Design front-end is described.

## 2. GENOM – GA OPTIMIZATION KERNEL

The GENOM optimizer kernel, partially illustrated in fig. 1, implements an enhanced modified GA.

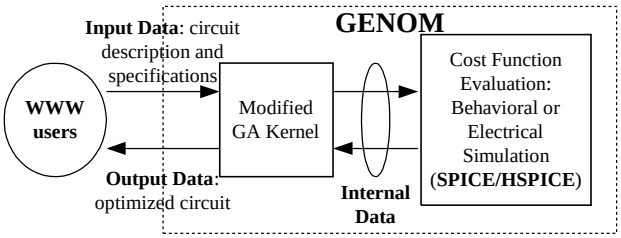


**Figure 1.** E-Design Environment Architecture.

### 2.1. Standard GA Implementation

The GENOM optimizer kernel was derived from a standard GA approach [4] using a continuous parameter representation having in mind the nature of the problem to be addressed, i.e., analog circuit optimization. The complete set of optimization parameters are represented by a chromosome, where each parameter is represented by a gene, i.e., a continuous variable, which is subject to a technological grid constraint. The GA operators such as paring, mating and mutation are initially set to default values and dynamically changed, as part of the innovative solution, according to the selected criterion for search space decomposition and premature convergence prevention. The cost function evaluation is made either by a behavioral or electrical simulation using Spice-like simulators. Finally, the convergence criterion is defined as a set of circuit parameters constraints and performance requirements.

### 2.2. Search Space Decomposition

The search space decomposition [3] was introduced, in order to reduce problem complexity and the number of cost function evaluations, therefore improving GA efficiency particularly when a cluster of workstations is available. The search space decomposition consists of dividing each variable range in $p$ parts, thus welding $p^n$ problem subspaces, where $n$ is the number of optimization variables, illustrated in fig. 2 for $p=2$ and $n=3$.

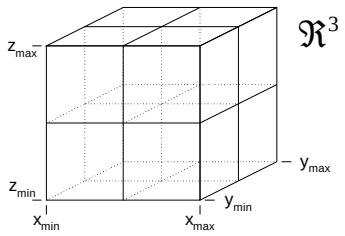


**Figure 2.** Search Space Decomposition.

### 2.3. Premature Convergence Prevention

A premature convergence prevention [3] process was introduced to improve algorithm performance in case chromosomes converge to local minima. This process is implemented by dynamically increasing the mutation rate, whenever the algorithm is in a little evolution period, and, therefore, forcing chromosomes to jump to other search space locations, accounting for solution diversity. Reaching the mutation rate limit means either enlarging the search subspace or outputting the best solution found.

### 2.4. Distributed Processing Implementation

The design objective of a circuit optimizer using stochastic optimization techniques, such us the serial implementation of the modified genetic algorithm described above, usually requires very large search spaces, as well as a considerable amount of memory. Besides that, the fitness function evaluation is usually a highly time-consuming operation. For this reason, a distributed processing solution was added in order to improve performance by reducing the overall processing time. The distributed processing approach uses a standard message passing protocol, MPICH [5], in a non-dedicated cluster of computers interconnected by a high speed network.

The new parallelization method combines the Master-Slave Parallel GA with the Fine-Grain GA methods, illustrated in fig. 3. Basically, the master processor after decomposing the search space in small $p^n$ problem subspaces, as described in section 2.2, assigns to each slave processor the execution of one subspace optimization task. Then, the best chromosomes from each slave processor run are transferred to the global optimization array in the Master processor. However, if the number of available processors is lesser than the number of subspaces, then the master transfers another optimization task to the free slaves, as soon as they finished their tasks. Finally, when all the optimizations sub-tasks are completed and the global optimization array is full with the best overall chromosomes from all search space, the Master processor executes a final global optimization task having those chromosomes as the initial population. The main advantage of this hybrid model resides in increasing the algorithm speed without introducing extra complexity. Besides that, the method does not change the GA search behaviour, so the conclusions for the serial GA Kernel still apply.

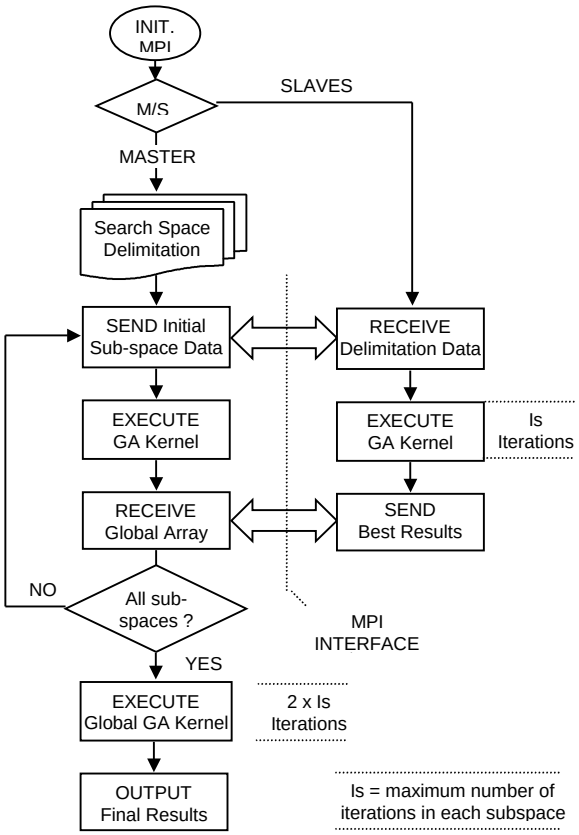


**Figure 3.** Distributed Processing Algorithm.

### *2.5.* Example: 2D Function

In order to evaluate the modifications introduced to the standard GA implementation, several examples with non-trivial mathematical functions have been tested. An example of 2D function, shown in fig. 4, is here considered to illustrate the achieved performance measures. Particularly, four different tests were performed considering respectively the GA standard approach, the GA with search space decomposition, the GA with premature convergence prevention and the GA with both the modifications. The performance measures, presented in table 1, show a considerable increase on efficiency whenever one or both proposed GA modifications are considered. Moreover, the search space decomposition allows a large reduction on CPU time, when using a parallel/distributed processing approach, with the asymptotic limit of $1/n$ CPU time compared to the serial processing approach, where $n$ is the number of used processors.

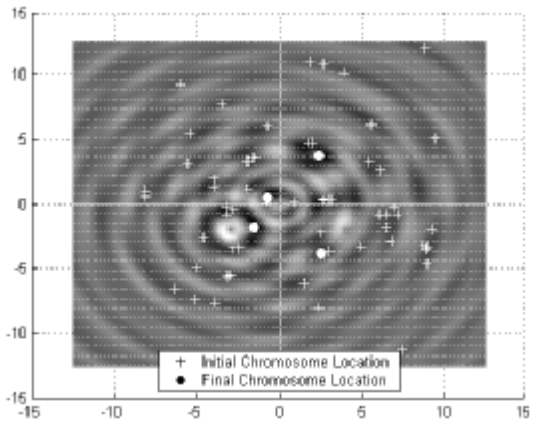


**Figure 4.** Search Space Contour and Chromosome Location.

**Table 1.** GENOM performance measures for 100 executions with a maximum of 500 iterations each.

| Type of Runs | Average no. of iterations | Average no. of cost func. evaluations | Average Minimum |
|---|---|---|---|
| standard | 383 | 9265 | -1.3377 |
| w/ space decomposition*(sd) | 367 | 9042 | -1.3806 |
| w/ premature convergence prevention (pcp) | 175 | 4282 | -1.4150 |
| w/ sd and pcp* | 174 | 4342 | -1.4196 |

*Approximately 1/n CPU time when using parallel processing.

## 3. CIRCUIT OPTIMIZATION EXAMPLES

In order to illustrate the applicability of the discussed optimization algorithm two well known circuit structures were selected and submitted to a retargeting/optimization task using the equation-based approach and the electrical simulation approach for the cost function evaluation.

### 3.1. Equation-based Approach

The equation-based approach may be used either to get further insight on simple circuit structures or to accelerate complex circuit optimization using developed behavioral models. In order to illustrate this approach a simple Sallen & Key Biquad section was selected. First, the design equations for the selected circuit are loaded or hardcoded. Then, circuit specs, such as bandwidth, quality factor and acceptable mismatch are specified as well as the acceptable ranges for circuit parameters. The circuit sizing was achieved in a few seconds, as shown on fig. 5, for the evolution from the initial to the final bode plots and for the performance specs of table 2.

**Table 2.** Performance Specifications and Results.

| Parameter | Specification | Results |
|---|---|---|
| DC gain | > 70 dB | 72.7 dB |
| GBW | > 40 MHz | 40.6 MHz |
| PM | < -75º | -79.02º |

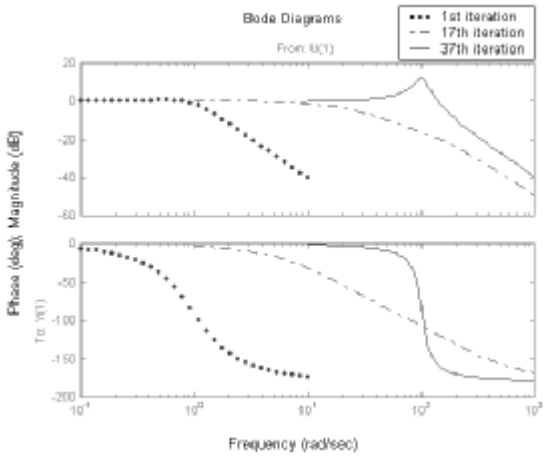


**Figure 5.** Bode Plots Evolution for the S. & K. Biquad.

### 3.2. Simulation-based Approach

The electrical simulation based approach is here illustrated by an optimization example of a two stage amplifier, illustrated in fig. 6, considering that a DC gain of 65 dB and a gain-bandwidth product of 25 MHz were required with a maximum error of 5%. In this experiment, only the bias circuitry transistors were subject to resizing as presented in table 3. The bode plot in fig. 7 illustrates the achievement of the specified design goal. Note that any other circuit parameter or range may have been selected for optimization purposes on designer demand.

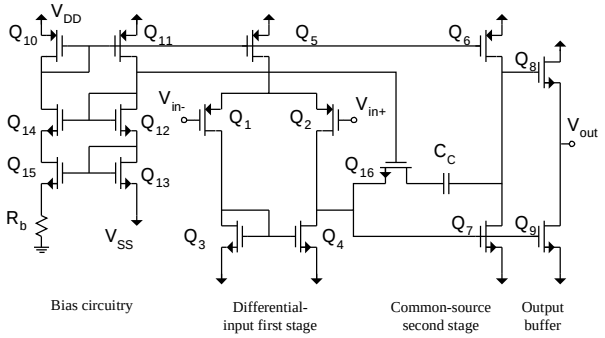


**Figure 6.** Two-stage amplifier.

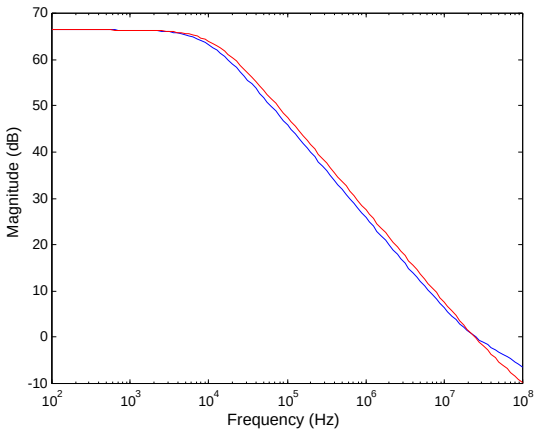


**Figure 7.** Two-stage amplifier bode plots.

**Table 3.** Optimization Parameters Range and Results.

| Name | Corresponding transistors | Range (μm) | Results |
|---|---|---|---|
| w1 | $Q_{10}$ to $Q_{14}$ | [12.5, 37.5] | 12.5 |
| l1 | $Q_{10}$ to $Q_{14}$ | [0.8, 2.4] | 1.0 |
| w2 | $Q_{15}$ | [50, 150] | 141.6 |
| l2 | $Q_{15}$ | [0.8, 2.4] | 1.2 |

## 4. E-DESIGN FRONT-END

An E-Design front-end allowing an incremental growth of the IC design database, an individual management of each project and, simultaneously, the widespread training on the design flow procedures is being developed.

### 4.1. E-Design Front-End Architecture

The E-Design front-end follows a client-server architecture. The E-Design server was developed in PHP and is based on a Linux environment running Apache as the HTTP server. The server unit is responsible for managing all the interaction among databases, archives and requested information from the client side. A circuit/system design and a user databases were developed in order to allow the storage of all the data associated to the available circuit/system structures and keep track of all user designs. The client access to the E-Design front-end is achieved through any WWW browser and after registering the client gains access to the circuit/system design database as well as all options concerning to circuit/system optimization/retargeting.

### 4.2. E-Design Example

In order to briefly illustrate the E-Design environment and the corresponding interface, the focus is here given to the specification of an optimization task for a Folded-Cascode Opamp, as shown in fig. 8. First, the circuit topology is selected from the IC database. Then, a specification page is dynamically generated based on the number and class of optimization variables, performance parameters, technology files, etc. Next, the required design goals must be appropriately specified and, finally, the job submitted to the server. Once the optimization task is submitted the job is launched and the achieved results stored at the authorized user account allowing the download of the circuit schematic, sized netlist, performance values or characteristics, etc. In this particular case, the optimization task converges very easily, naturally, with more complex examples or larger search spaces specified, the time resources associated to the job might increase rapidly. In order to minimize this effect and contribute to the development of the client/designer skills and sensitivity to optimization problems for each circuit entry in the database, an accompanied heuristic is considered, as well as a global description of the optimization techniques used in the background.

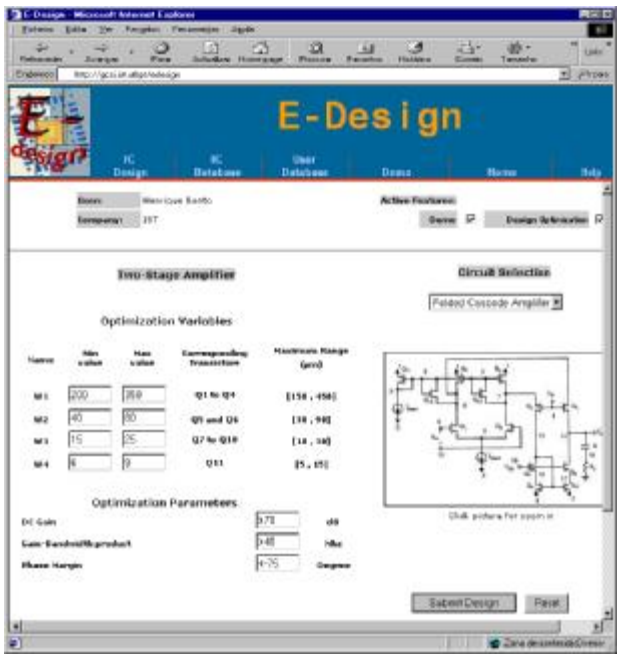


**Figure 8.** E-Design environment.

## 5. CONCLUSIONS

This paper introduces an enhanced genetic algorithm kernel applied to a circuit/system-level optimization E-Design environment. The presented approach, tested for both global minimum location on non trivial mathematical functions and optimization/retargeting of circuit/system-level structures, grants an increase on performance by reducing the number of total iterations and the required overall processing time. Moreover, the implementation of both an equation based and electrical simulation approach for function cost evaluation allows an efficient solution of multi-objective, multi-constraint optimization problems. Finally, an E-Design front-end developed for the optimization environment allows a widespread increase on training and design tasks performed by a general user.

## 6. REFERENCES

[1] R. Phelps, M. Krasnicki, R. Rutembar, L Carley and J. Hellums, “Anaconda: Simulation-Based Synthesis of Analog Circuits Via Stochastic Pattern Search”, IEEE Trans. on CAD, vol. 19, no. 6, pp. 703-717, 2000.

[2] J. Koza, F. Bennett, D. Andre, M. Keane and F. Dunlap, “Automated Synthesis of Analog Electrical Circuits by Means of Genetic Programming”, IEEE Trans. on Evolutionary Computation, vol. 1, no. 2, pp. 109-128, 1997.

[3] J. Silva, N. Horta, “Genom: Circuit-Level Optimizer Based on A Modified Genetic Algorithm Kernel”, in Proc. IEEE International Symposium on Circuits and Systems, May, 2002, pp. 745-748.

[4] Z. Michalewicz, “Genetic Algorithms + Data Structures = Evolution Programs”, Springer-Verlag, 1996.

[5] “MPICH – A Portable Implementation of MPI” in http://www.mcs.anl.gov/mpich