Excerpt from the Proceedings of the 2011 COMSOL Conference in Stuttgart

Advanced Applications of an Automated Generative Tool for MEMS Design Based on COMSOL Multiphysics

Francesca Bolognini[*,1]
[1]University of Cambridge Engineering Department,
Trumpington street, CB2 1PZ Cambridge, UK
fb252@cam.ac.uk

Abstract: This work presents a different use of COMSOL, not only as a simulation platform for engineering tasks, but also as an integrated component of a computational AI tool framework used to automate designs creation. CNS-Burst is a computational synthesis method used to create solutions to an assigned design task. This search method, iteratively applied, finds design alternatives and represents them as geometrical objects. In this context, COMSOL is integrated in the method and used to evaluate the performance of the design solutions found. Geometrical objects are transformed in COMSOL objects for analysis and simulations. In this work we have looked at case studies for which industry is interested in finding innovative solutions in a short timeframe. We also looked at a design field that could be accurately analysed using COMSOL. The design field here examined are MEMS. In particular, the case study chosen are sandwich resonators. This paper can be of interest not only to see how COMSOL has been used in this innovative academic project, but also to explore how MEMS design innovation is investigated.

Keywords: Computational Design Synthesis, Automated design, simulation-based design, optimisation, MEMS, COMSOL from Matlab command.

1. Introduction

This work presents the use of COMSOL as part of a computational platform oriented at automating the creation of design solutions for an assigned design task. This research is one of the many academic efforts in the field that goes under the name of Computational Design Synthesis (CDS). Computational synthesis methods have been around since few decades now and a vast research background is slowly changing into initial industrial applications in many fields of engineering [1]. The ultimate goal of computational synthesis is to give support to designers in the creative phase of the design process, i.e. automatically generating designs through the computational simulation of designers' creative effort. The automation of this phase of the design process (synthesis) includes many of the activities that designers carry on in common practice, and of course comprises some elements of AI. The advantage of creating an automated generative design tool is the possibility to boost innovation creating solutions that go beyond designers' insight, regardless of their experience and bias. Another advantage is the possibility to speed up the design process.
Computational synthesis methods often find an obstacle to their development in the difficulty of integrating in the automated search of solutions external packages for the analysis/simulation of the designs generated. The main difficulty is that the analysis has to be performed automatically and simultaneously with the generation of solutions, so that they can be assessed as valid or non-valid and, in case, kept or discarded. In particular, the method developed in this work set a step ahead in the research field, as a complete and complex analysis tool (COMSOL) used by professional engineers is integrated in the search. This choice guarantees accurate and reliable solutions ready to be manufactured. What is of interest here is how the integration is performed and till what point COMSOL is a necessary tool for the success of the method. COMSOL has been chosen for its completeness and flexibility as an analysis tool. It is also important to see how the method is applied to MEMS, the case study of choice. MEMS are particularly indicated, due to the industry interest they raise and to continuous market's request for innovative MEMS solutions. In fact MEMS design is still performed by hand through complex iterative processes. For this reasons, many academic attempts have been carried out to automate MEMS design process. The one presented here has been successfully completed and, at the same time, presents a new type of microresonators called 'sandwich', which are of interest to a major European electronic company.

2. The Method

The aim of computational synthesis methods is not just to find the best possible

solution to a design problem: their emphasis is rather on creating design solutions that are, possibly, new design alternatives. Synthesis as a method contrasts with traditional optimisation in that the goal of synthesis is more broadly to capture, emulate and/or utilise design decisions made by human designers during the creative process. Computational synthesis methods are complex products that do not just automate the optimisation of solutions. Their scope goes much beyond that, extending the concept of search of optimal designs with methods for creating solutions to propose to the optimisation. There is no exact formulation to implement synthesis methods, but there is agreement in the computational synthesis community that they can be considered as a set of distinct activities [2]. Hence, in order to develop such tools, a systematic approach is needed. The framework in Figure 1 provides the sequence of necessary steps for transforming the design process into an algorithmic solution. This is the sequential framework of activities used for the implementation of the method used in this work (CNS-Burst). The steps of this framework are fundamental and necessary milestones of an automated synthesis process.

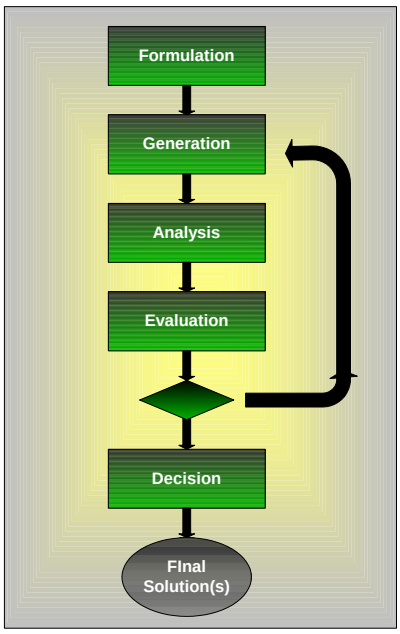

Figure 1. Systematic Approach to Computational synthesis: steps of CNS-Burst method.

The first step is the definition of the task, which includes the mathematical formulation of the problem and search objectives. The search for solutions can start from an initial design, which can be an existing well-known solution or an estimated idea of what the solution is going to be like. The search can also start from a basic component of the design, or even from nothing in some cases.
The next step is the generation of a novel solution: new ideas and past knowledge are recalled at this stage, together with some insights on how to modify design components and behaviours that do not meet the desired objectives. Elements of optimisation and artificial intelligence are used at this stage. In order to come out with the best possible solution, designers usually do not stop their search after their first intuition. Behaviour and design objectives for each generated solution are evaluated and compared with existing solutions, in order to verify whether the new design has come any closer to the desired objectives of the search (Analysis). Solutions that match design objectives are kept in an archive for future comparison or just to form a pool of results to be used by designers. This generate-and-test mechanism (iterative cycle in Figure 1) will be repeated until the solutions gets close enough to the objectives of the search or until time limits for the search are reached.
According to the framework just described, it is understandable how the implementation of an automated synthesis method must follow a general architecture. A synthesis method can be thought of as a set of activities that follow this framework. Each of these activities is constituted by specific components (or set of components) that, in this particular work, will be called ‘modules’.

All the components of this architecture are integrated into a general algorithm that directs the synthesis process. The modules are linked together through the main algorithm that directs the search and calls the required different modules following the logic expressed by the framework in Figure 1. The synthesis task is formulated as a design optimisation task consisting of design parameters, constraints and objectives.

In order to find feasible and optimised designs, synthesis techniques are built on a search algorithm, integrated with an evaluation method. The generated designs are evaluated according to desired design performance criteria, stated as objectives of the search. The evaluation might require complex analysis to

simulate design behaviour, which is often executed through external software embedded in the search code. Analysis tools must be able to perform rapid simulation of the designs' behaviour and pass to the search code accurate feedback on the designs' performance. Also, the design simulation must be performed automatically after each design generation. These requirements often encounter limitations due to the impossibility of finding commercial packages that are able to perform simulation in the design domain. The choice may often be limited to packages that are not optimal for the design performance being analysed or do not offer the level of accuracy needed. A second problem in integrating external tools is the difficulty of supporting multiple design criteria based on multidisciplinary considerations, hence the necessity of finding compatible multidisciplinary analysis packages. Another important problem is due to the impossibility of integrating analysis tools easily and time-efficiently. The necessity to perform accurate analysis in order to obtain solutions as ready as possible to be post-processed, led to the use of COMSOL. In particular, the COMSOL toolbox used in this work is the MEMS toolbox for 3D eigenfrequency and static analysis.

4. Use of COMSOL Multiphysics

For the purpose of this work, the first problem in the integration process has been to automatically translate CNS-Burst design objects created by the search generation modules into COMSOL objects files (.m files), containing the COMSOL geometry of the design and ready to be analysed using the MEMS analysis package.
A second problem encountered has been to visualise the simulation behaviour instantaneously every time a new design was created and pass feedbacks results automatically to the search. These problems have been solved using COMSOL from API, i.e. not through a user interface. COMSOL API is written in Matlab. The CNS-Burst search method, also implemented in Matlab, is able to call COMSOL functions from the Matlab command line, once the entire path of COMSOL functions is loaded into Matlab environment. The direct plug-in of an external package into the search code is a far more complex and time-consuming method than using batch files, but allows the immediate integration of feedbacks into the search.

4.1 Translation of Geometrical Objects into Simulation Objects

This section describes how the design generated by the search code (the geometrical model represented by CNS design representation) is translated into a simulation model (COMSOL object) to be evaluated by COMSOL. In order to visualise the solutions created, CNS-Burst method uses a representation module that make s use of a network of parts (called primitives) connected through nodes (see Figure 2).

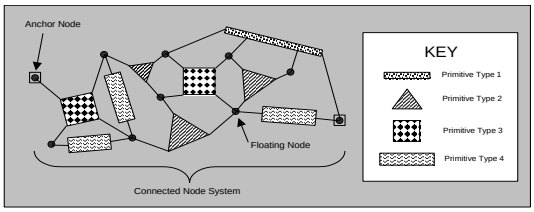

Figure 2. CNS-Design structure formed by primitives and nodes.

Any CNS design is constituted by a set primitives and nodes and matrices (the NodeList matrix and the Connectivity matrix), as described in Figure 3a. The primitive object provides a description of the primitive in terms of:

- Name of the primitive (Name)
- Maximum and minimum number of instances of the primitive (MaxInstances, MinInstances)
- Parameters (for example dimensions) and maximum and minimum values of the parameters (ParamLimits).

The nodes forming a design can be of different nature. Matlab node objects (NodeDefinitions) are described by the following fields (Figure 3.10a):

- Type of node (Name), e.g. anchor, floating, roller, force applied, etc.
- Number of a node type that can be added to/removed from the design (CanAdd, CanRemove)
- Properties of the node that can be changed (PropsChangeable)
- Maximum and minimum value of the properties (PropLimits)
- Maximum and minimum number of primitives connected to that node (MaxConnects, MinConnects).

The NodeList matrix is a list of all the nodes forming the design, with description of their type (as defined in NodeDefinitions) and position (Figure 3a). The Connectivity matrix states connections between nodes and primitives (Figure 3a). Each of its columns is representative of a primitive and lists its starting and ending nodes (connections).

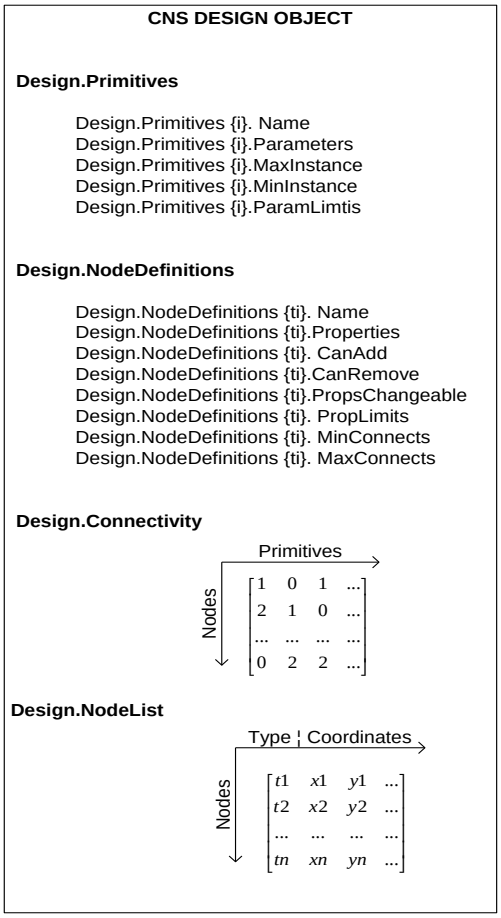

Figure 3a. CNS Design (geometrical model).

The simulation model analysed by COMSOL is a COMSOL object (Figure 3b), i.e. a Matlab file containing information on:

- A list of all the building blocks (primitives in CNS representation) forming a design, and correspondent description (including the type of building blocks and their dimensions)
- The position of each building block. In COMSOL a building block's position in the design reference system is defined by the coordinate of a point, an axis and its orientation (angle) in a reference system.
- Constraints and boundary conditions for each building block.

The search algorithm creates new design solutions in the form of CNS (geometrical model). A translation routine transforms the connected-node system in COMSOL object (simulation model). The COMSOL object is embedded in the code and the information regarding primitives are passed to it in the form of inputs of the file.

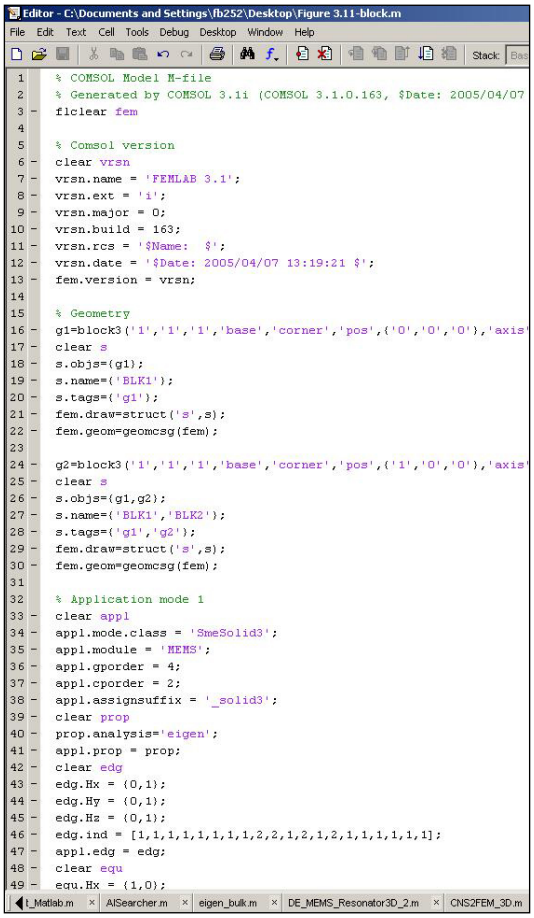

Figure 3b. COMSOL object (simulation model).

The translation routine consists in extracting this information from the CNS design description and transforming them in inputs for the COMSOL file.

The information passed in input to the COMSOL file are extracted from the CNS design as follow (Figure 4):

- Name and dimensions of primitive objects are transformed into type and dimensions of COMSOL blocks. The translation routine transforms information on each primitive of the connected-node system in a COMSOL geometry part, reading the name of the primitive (beam, disk, etc.) and

transforming it in the correspondent COMSOL shape (e.g. block, cylinder, etc).

- For each primitive, the entries of the NodeList and Connectivity matrixes are transformed into the position of the corresponding building block in COMSOL, and passed in input to passed in input to the COMSOL object file. The coordinates of the nodes and their connectivity are transformed in coordinates and orientation angles for the correspondent COMSOL block in the COMSOL design reference system. The nodes defining a primitive are read in the columns of the connectivity matrix. The coordinates of these nodes are extracted from the NodeList matrix. The orientation of the COMSOL blocks in the reference system is found through a transformation matrix, using the coordinates of the connection nodes.

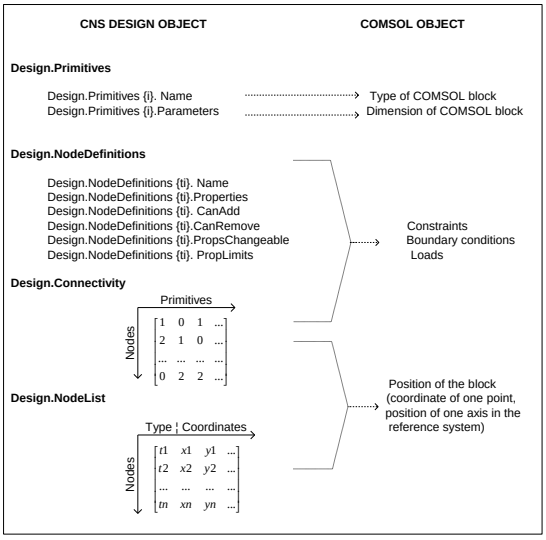

Figure 4. Correspondence of features in CNS objects and COMSOL objects.

Particular attention deserves the translation of constraints, loads and boundary conditions for the primitives. This is due to the fact that in CNS representations these characteristics of the design relate exclusively to nodes (only nodes can be anchored, for example), while COMSOL uses different sorts of constraints/loads/boundary conditions for physical parts, according to their geometry and to designers' needs.

Once the necessary information are passed in input into the compiled COMSOL object, a COMSOL meshing function is called automatically from Matlab command line to mesh the simulation design and analyse its behaviour. Finally, the translation of the geometrical model into a simulation model and the subsequent meshing of the simulation model are a completely automated procedure, repeated at each design generation.

5. Application of the method: Sandwich Microresonators

The case study used for the application of CNS-Burst are Sandwich microresonators. These resonators were first proposed as a new resonator topology in 2005 and have since then raised industry's interest [3]. An example of sandwich resonator is shown in Figure 5. Sandwich resonators are called this way because the resonant structure is sandwiched between two electrode beams. A typical structure is a regular one, where the sandwiched beams are arrayed in parallel in the vertical direction (Figure 5). The resonator is anchored at two edges of a central beam that runs through the length of the structure.

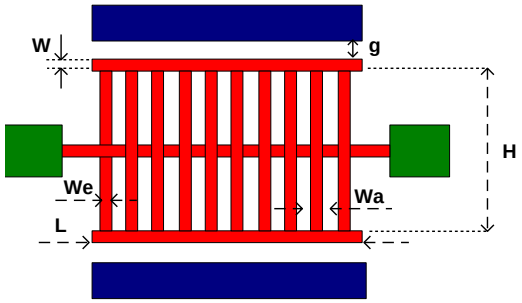

Figure 5. Sandwich resonator (in green: anchors; in blue: electrodes; in red; resonant structure).

The introduction of this new type of silicon resonator comes from the necessity to address some technical challenges, one of which is concerned with the value of the equivalent motional resistance Rm. This key parameter determines the signal to noise ratio and power dissipation of a reference oscillator incorporating the microresonator as a timing element [3]. The topology of the resonator and the coupling of mechanical and electrical domains have a strong influence on Rm. Sandwich resonators, compared to other resonant structures, were seen to better meet the requirement of minimal motional resistance for the same operating frequency. The sandwich structure is also advantageous for designers that perform design calculations by hand. The geometry is simple and the

behaviour of deformed beams is predictable. The primary frequency mode of interest is the bulk in-plane one, which involves in-phase longitudinal extension associated with the array beams (Figure 6). This mode can be driven using an electrostatic parallel-plate excitation mechanism where two electrodes are arranged parallel to the exterior beams (shaded blue in Figure 5). More complex geometries for sandwich resonators would be difficult to examine, especially because of their unknown out-of-plane modes and complex detection of frequency modes of interest. For this reason, innovative structures remain largely unexplored, although accurate and efficient, leaving unexplored the possibility to reach more accurate resonant frequencies. The case study presented here is very complex even for expert designers. The design of these devices has been so far executed by hand analysis. The objective of this case study is to find a resonator with the required resonant frequency and resonant mode. The resonant mode in question is the sandwich-bulk mode, as the structure vibrates in plane in the direction traversal to the axis of the resonator (Figure 6).

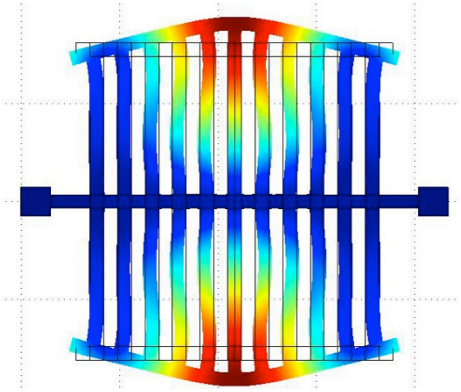

Figure 6. Sandwich bulk mode shape (in colour mode shape resonant mode).

5.1 Results of a Standard Search with Three Design Objectives

The goal for this case study is to generate a structure that resonates in the desired mode and for a desired range of frequencies. Figure 7 shows the design area A=HxL sandwiched between the electrode beams, where new topologies can be synthesised as an alternative to the typical arrayed ones. The design area A is fixed and so are its boundaries. This section reports the optimisation model used for the application of CNS-Burst to the sandwich resonator topology optimisation task.

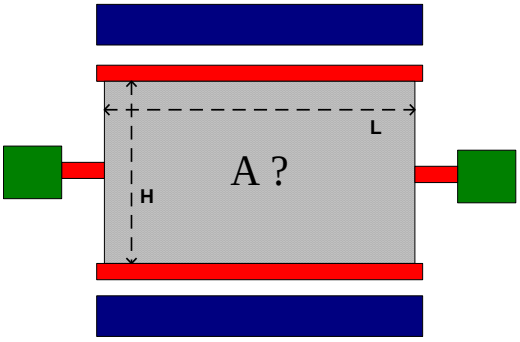

Figure 7. Sandwich resonator: the topology optimisation task.

The design objectives considered are:

- A target operational frequency of 25 MHz (constraint-satisfaction problem formulated as a soft constraint)
- A minimal motional resistance Rm (formulated as a minimisation problem). As for the motional resistance, it has been mentioned above that Rm is the parameter that justifies the recent interest in sandwich resonators. The motional resistance of sandwich resonators can be analytically calculated as [4]:

$$\frac{R_{m-longitudinal}}{R_{m-sandwich}} = \frac{\frac{\pi\sqrt{E\rho}g^4}{8\varepsilon_0 Q V_{DC} T}\frac{1}{W_e^2}}{\frac{\pi\sqrt{E\rho}g^4}{8\varepsilon_0 Q V_{DC} T}\frac{W_e}{nW_a^2}} = \frac{nW_a^2}{W_e^2} \quad (1)$$

where Wa and We are the width of the array beams and the width of the additional exterior layer (i.e. double the half distance between axes of two consecutive beams, Figure 5), n is the number of beam members of the sandwich, E is the Young's Modulus, ρ is the material density, ε0 is the dielectric constant, Q is the quality factor, VDC in the operational DC voltage, T the thickness of the structure, g the gap between resonator and electrode.

- A maximum quality factor Q (formulated as a minimisation problem). The quality factor Q is a measure of the energy dissipated per cycle in the resonator. Q can be expressed as follows:

$$Q = \alpha \frac{E_{total}}{E_{anchor}} \quad (2)$$

where *Etotal* represents the total strain energy of the entire structure (resonant parts

plus anchors) for a certain mode shape and *Eanchor* is the strain energy present in the anchors for the same mode. The strain energy in any given part of the structure is calculated using the COMSOL-MEMS analysis package.

The optimisation model for this design task is:

$$\min\left\{|\Delta f|, R_m, \frac{1}{Q}\right\},$$
$$S.t. \quad 0 \le x_j \le 161\mu m, 0 \le y_j \le 155\mu m$$
$$\textit{Design Constraints for Nodes/Primitives} \quad (3)$$

where $\Delta f = (f0 - f)$, i is the number of beams used to form the structure, j is the number of nodes, (xj, yj) are the coordinates of a node. The minimum width (wi) of the beam elements equal to 1µm is due to fabrication constraints. The three design objectives have a complex computational representation and have never been defined in such details in any other work on MEMS synthesis. Figure 8a shows a geometrical design solutions obtained by the synthesis method. Figure 8b shows the same solutions transformed into COMSOL object, used to perform behavioural analysis of the solution found. Figure 9 presents some of the original and innovative structures obtained with the search.

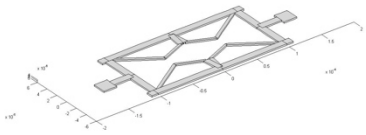

Figure 8a. Design solution (geometrical model).

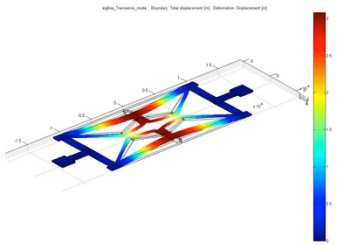

Figure 8b. Design Solution (COMSOL model).

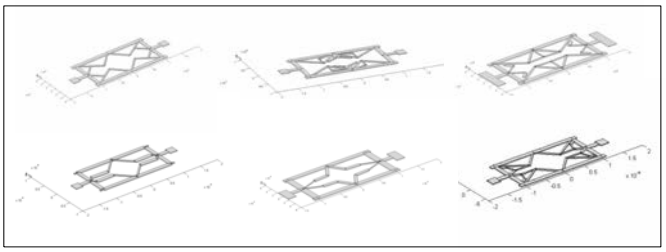

Figure 9. Design solutions obtained with CNS-Burst.

6. Conclusions

This work has presented a particular use of COMSOL as a component of a CDS method. The successful results in obtaining innovative designs through the application of the method are also due to the introduction of COMSOL as part of the evaluation and simulation module. While COMSOL accuracy allowed precision of results, its flexibility allowed its direct and straightforward integration in the computational design process. This work confirms COMSOL uniqueness as an analysis and simulation package.

7. References

1. Bolognini, F., 'An Integrated Simulation-based Generative Design Method for Microelectromechanical Systems', *PhD Thesis*, University of Cambridge (2009)
2. Cagan, J., 2005 CAGAN, J, CAMPBELL, M.I., FINGER, S., TOMIYAMA, T., (2005), *A Framework for Computational Design Synthesis: Model and Applications*. Journal of Computing and Information Science in Engineering, 5, 171-181.
3. YAN, J., SESHIA, A. A., STEENEKEN, P. G. & VAN BEEK, J. T. M. (2006), *A Silicon MEMS Bulk Mode 'Sandwich' Resonator*. 17th European MME. Southampton, UK.
4. YAN, J. (2007), *Micro/Nano-Electromechanical Resonators for Signal Processing*. PhD Thesis, Engineering Department. Cambridge, University of Cambridge.
5. LEE, J. E.-Y., YAN, J. & SESHIA, A. A. (2008), *Quality Factor Enhancement of Bulk AcousticResonators Through Anchor Geometry Design*. Eurosensors XXII. Dresden, Germany.

8. Acknowledgements

The author is grateful to EPSRC UK.