Analyzing and Simulation of MEMS in VHDL-AMS Based on Reduced Order FE-Models

Michael Schlegel, Fouad Bennini, Jan Mehner, Göran Herrmann, Dietmar Müller, Wolfram Dötzel
Department of Electrical Engineering and Information Technology,
Chemnitz University of Technology,
Chemnitz, Germany
michael.schlegel@infotech.tu-chemnitz.de
bennini@infotech.tu-chemnitz.de

Abstract

This paper deals with the computer-aided generation of Reduced Order Macromodels (ROM) [1] for system level simulation using VHDL-AMS. The application of ROM on system level will be demonstrated by two surface micromachined vibration sensor arrays [2] which are intended for wear state recognition on highly stressed machine tools. The paper will discuss how to export the macromodel from FEM data into VHDL-AMS and it presents a new approach called „Multi Architecture Modeling" [3] which simplifies the integration of macromodels in the system environment. Furthermore, the performance of the macromodels for analyzing and simulating on system level will be shown and compared with abstract models [4] and measurements.

Keywords

reduced order modeling, system model, VHDL-AMS, sensor array

INTRODUCTION

Before constructing a prototype, a simulation of the whole system is necessary to check the functionality of the individual components and their interaction. For MEMS, a design environment is needed which allows simulation of different physical domains including coupling effects. VHDL-AMS is a flexible system description language suited to handle such requirements. Additionally, it allows to describe and simulate the system at different levels of abstraction.

System level modeling and simulation have become state of the art in MEMS design due to the increasing system complexity. But high level models have the disadvantage that they often cause a considerable error in simulation. This error occurs because physical phenomena must be simplified to be expressed in an equation. VHDL-AMS or similar simulation systems can only solve differential algebraic equations and no partial differential equations. If the partial differential equations cannot be solved analytically then an approximated solution must be used which increases the error additionally. But using accurate models from the physical level (e. g. FEM models) is very cumbersome for daily tasks on system level.

Parameter reduction of FEM models (reduced order modeling) is an approach filling this gap in the design flow. Using a shape function method to generate ROM of linear systems becomes state of the art [5], [8]. Expanding these methods for non-linear cases is presented in [1]. For a most effective usage of these methods in system design it is required to export the ROM to VHDL-AMS with an interface matching the system structure.

THE SYSTEM „VIBRATION SENSOR ARRAY"

The measurement system (Figure 1.) consists of a sensor array, analog signal processing, digital system control and a fuzzy pattern classifier to analyze the vibrations. For more details see [4].

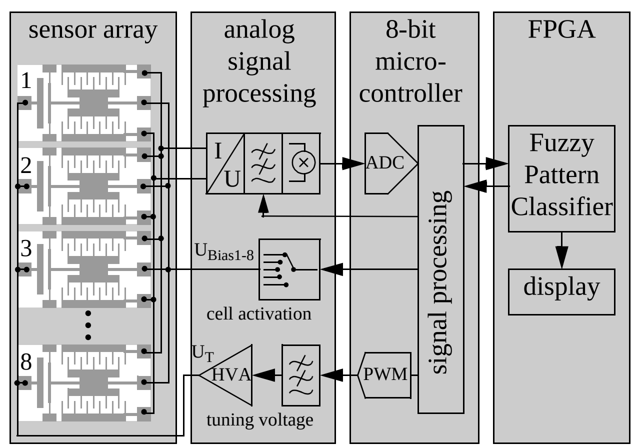
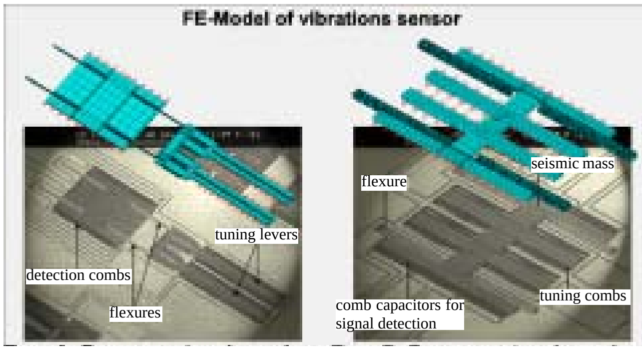


Figure 1. System structure of the measurement system

The center of the measuring system is the vibration sensor array. This surface micromachined vibration sensor consists of 8 laterally moving mass-spring resonators and is intended to work as a frequency selective device at its resonance. Using the stress stiffening effect in the flexures (type A), the resonance frequency is tuned by a voltage (V_{tun}) controlled tensile force, which is electrostatically generated in pairwise arranged lever systems. Vibration detection is effected electrostatically by the symmetrical comb capacitor attached to the proof mass were a DC voltage can be induced.

Type B has a similar structure. But in opposition to type A the tuning is performed by an additional curved comb structure at the mass. This structure behaves like a spring with negative coefficient, so the resonance frequency can be tuned directly.

Figure 2. shows both types of sensors.



Type A: Frequency tuning using stress-stiffening

Type B: Frequency tuning using electrostatic softening

Figure 2. Photograph of the vibration sensor

In [4] abstract system models are discussed, based on simplified physical behavior. These simplifications cause a simulation error of up to 10 %. This is accurate enough to check the global system functionality. But for more significant simulation results, e. g., to test the fuzzy pattern classifier, a more accurate sensor model is required.

REDUCED ORDER MODELING

Reduced order modeling using modal basis functions was originally developed by [8] and has been continuously improved by several authors. In [1] we have shown that the approach is able to cover electrostatic structural interactions of arbitrary 2D and 3D structures, allowing for multiple electrodes, geometrical non-linearities and initial pre-stress conditions while supporting static, transient and harmonic analyses.

The basic idea of the ROM is to approximate the deformation state of the finite element model by a series of weighted modal shape functions (eigenmodes)

$$u_i(t, x_i, y_i, z_i) \approx u_{eq} + \sum_{i=1}^{m} q_j(t) \cdot \Phi_j(x_i, y_i, z_i) \qquad \text{(eq. 1)}$$

where u_i are the time dependent nodal displacements of the FE-model and ϕ_j the eigenmodes which are scaled by time dependent modal amplitude q_j. In general, (eq. 1) describes a coordinate transformation of finite element displacement coordinates (local coordinate) to modal coordinates of the macromodel (global coordinate). The deformation state of the structure given by n nodal displacements u_i ($i=1,2,\ldots,n$) is now represented by a linear combination of m modes weighted by their amplitudes q_j ($j=1,2,\ldots,m$) where $m << n$.

The governing equation of motion describing the ROM of electrostatic actuated MEMS structures in modal coordinates is given by

$$m_j\ddot{q}_j + 2\xi_j\omega_j m_j\dot{q}_j + \frac{\partial}{\partial q_j}W_{st}(q_1, \ldots, q_m)$$
$$= \frac{1}{2}\sum_r \frac{\partial}{\partial q_j}C_{ks}(q_1, \ldots, q_m) \cdot (V_k - V_s)^2 + \sum_{i=1}^{n} \Phi_j^i \cdot F_i \qquad \text{(eq. 2)}$$

where m_j is the modal mass, ω_j the eigenfrequency, ξ_j the linear modal damping ratio, W_{st} the modal strain energy function, C_{ks} the modal capacity-stroke function, r the number of capacities involved for microsystems with multiple electrodes, V the electrode voltage applied and F_i a local force acting at the i-th node. The current I at each electrode k is defined by:

$$I_k = \frac{\partial Q_k}{\partial t} = \sum_r \left(C_{ks} \cdot \left(\frac{\partial V_k}{\partial t} - \frac{\partial V_s}{\partial t}\right) + \frac{\partial C_{ks}}{\partial t} \cdot (V_k - V_s) \right) \qquad \text{(eq. 3)}$$

An essential prerequisite to establish (eq. 2) and (eq. 3) are proper modal strain energy and capacity-stroke functions. Both are derived from a series of FE runs at various deflection states in the operating range. The acquired data are fitted to polynomial functions in order to compute the local derivatives, which describe force and stiffness terms. As a matter of fact, shape function methods can be applied to non-linear systems, too [9]. Geometric non-linearities, as for instance, stress-stiffening, can be regarded if the modal stiffness is computed from the first derivative of the strain energy function with respect to the modal amplitudes. Capacitance-stroke functions provide non-linear coupling between each eigenmode and the electrical quantities (i.e. electrostatic modal forces, electrical current) if stroke is understood as modal amplitude. Damping parameters are assigned to each eigenmode. Modal representations of MEMS are very efficient since just one equation per mode and one equation per involved conductor is necessary to describe the coupled system entirely. The approach will be demonstrated at the example of one cell of the micromechanical vibration sensor array from type A.

The first step of the ROM generation is to determine which modes are really significant, and to estimate a proper amplitude range for each mode. Several criteria can be applied, for instance, the lowest eigenmodes of a modal analysis, modes in operating direction, or modes, which contribute to the deflection state at a typical test load.

Applying a unit tuning voltage (V_{tun}) and an acceleration in operating direction as test loads to one cell of the array reveals that its motion is dominated by the detection mode of the proof mass (mode 1), the tuning mode of the pairwise arranged lever systems (mode 5) and a higher mode (mode 8) which corrects the displacement of mode 5 caused by the imposed acceleration. Next, the dependencies of the strain energy W_{st} and of two capacities (for detection and tuning) on the modal amplitudes are described by polynomial function fitted. The necessary data points are obtained by imposing each eigenmode with varying amplitude on the mechanical model for the non-linear strain energy and on an electrostatic space model for capacitance. This process is computationally expensive but has to be done just once. The result is a black-box model that can be applied to any load situation. In the concept of the modal superposition method, each eigenmode represents a single independent resonator with modal mass m_i and modal damping d_i.

Figure 3. shows the harmonic response of the proof mass excited by acceleration in detection direction and for several tuning voltages. The results were compared with the coupled field analysis with ANSYS. It was clearly shown that the ROM is able to capture correctly the dynamic behavior of the full FE-Model and the stress stiffening effect (error less than 2 %). Analogously, the ROM of micromechanical vibration cells from type B were generated only with two eigenmodes, and the errors are less than 1 %.

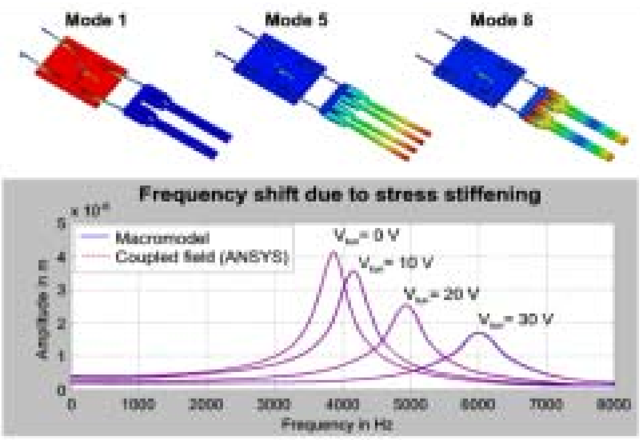


Figure 3. Harmonic response of a micromechanical vibration cell from type A for several tuning voltages

APPLICATION OF ROM IN SYSTEM DESIGN WITH VHDL-AMS

The ultimate goal of the ROM is to obtain an accurate black-box model of the microsystem's behavior. Interface objects are limited to the voltage-current relationship at each electrode, essential inputs such as external loads (e.g., gravitation, pressure) and significant outputs (e.g., a subset of displacements at characteristic points of the model).

Problems during VHDL-AMS export

The export of the ROM to VHDL-AMS is performed in two steps. At first, an initialisation file containing all necessary information of the macromodel, such as the fitted polynomial coefficients and orders, is generated. Then, the source code in VHDL-AMS is generated automatically. Thereby a set of differential algebraic equations (DAE) with non-constant, non-linear coefficients emerges. The DAE can be mapped to the *simultaneous statements* of VHDL-AMS where non-conservative nodes (`QUANTITY`) represent forces, displacements, velocities and accelerations.

The main problem of exporting the ROM in VHDL-AMS is to express the fitted functions of the non-linear strain energy and of the capacities which are part of the coefficients of the DAE (derivatives of the energy dW and of the capacities dCdet, dClev). The algorithm consists of a number of sequential calculations. So this algorithm must be calculated within the VHDL-AMS sequential structures `PROCESS`/`PROCEDURAL` or `FUNCTION`/`PROCEDURE`. A `PROCESS` cannot be used because it is a digital statement. The analog counterpart of a `PROCESS` is the `PROCEDURAL` statement. But this statement was not implemented in the VHDL-AMS simulators, which were available at the beginning of this work. So this algorithm must be implemented inside a VHDL-AMS `FUNCTION` or `PROCEDURE`. The algorithm returns more than one result, so first an attempt was made to use a `PROCEDURE` which can return any number of values. But a `PROCEDURE` can only be called as *sequential statement* within the VHDL-AMS sequential structures described above. The so called *concurrent procedure call* is not really a concurrent call. In this case, a `PROCESS` is placed around the procedure call. This `PROCESS` contains a `WAIT` statement which interrupts the calculation of the `PROCESS` until a digital event occurs. The parameters of the procedure call are analog `QUANTITIES`. Therefore, a digital event never occurs and the coefficients are only calculated at simulation start and not during the simulation.

For these reasons the calculation of the coefficients (dW, dCdet, dClev) must take place in a `FUNCTION`. The return values are aggregated in a vector which is the return value of the `FUNCTION`. The vector is arranged in the following form:

```
return_value(0):= function value f(x),
return_value(1):= first derivative df/dx,
return_value(2):= first derivative df/dy,
return_value(3):= first derivative df/dz
```

For first testing purposes the free simulator hAMSter from SIMEC/Ansoft was used. But the first version did not support vectors. Former versions of AdvanceMS from Mentor Graphics had similar restrictions. A workaround was using a number of `FUNCTIONs` each with a scalar return value. But this costs simulation time and makes the VHDL-AMS source code unnecessarily complex. Fortunately, later versions of these simulators supported vectors so this workaround was no longer necessary.

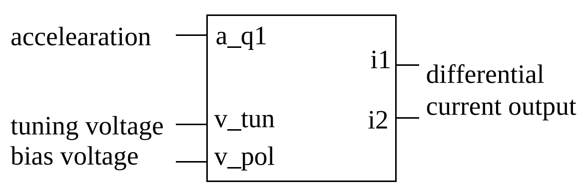


Figure 4. Black-box view of the ROM

```
entity VIBSENS is
   port (-- acceleration
         terminal a_q1:    translational_f_s;
         -- differential output current
         terminal i1, i2:        electrical;
         --bias and tuning voltage
         terminal v_pol, v_tun:  electrical);
end entity VIBSENS;

architecture ROM of VIBSENS is
   ...
   subtype fkt_ret1_type is real_vector(0 to 3);

   -- fitted polynomial function
   function ROMOPER(
      constant x, y, z: in real:=0.0;
      constant p_coe, p_ord: in real_vector;)
   return fkt_ret1_type is
      ...
   begin
      ...
      ret_val(0):= ff; -- function value f(x)
      ret_val(1):= dff1; -- first derivative df/dx
      ret_val(2):= dff2; -- first derivative df/dy
      ret_val(3):= dff3; -- first derivative df/dz
      return ret_val;
   end function ROMOPER;
begin
   ...
   dW    == ROMOPER(q1,q2,q3,w_coe,w_ord);
   dClev == ROMOPER(q1,q2,q3,c0i_coe,c0i_ord);
   ...
   fq1   == MASS(1)*(q1'dot'dot+acc1) +
            DAMP1*q1'dot -
            dCdet(1)*(vcon1)**2/2.0 -
            dClev(1)*(vcon2)**2/2.0 +
            dW(1);
   ...
   i_tun == dClev(0)*(vcon2'dot) +
            dClev(1)*q1'dot*(vcon2) +
            dClev(0)*(vcon2'dot) +
            dClev(2)*q2'dot*(vcon2) +
            dClev(0)*(vcon2'dot) +
            dClev(3)*q3'dot*(vcon2);
...
end architecture ROM;
```

The DAE, representing the movement, e. g., in x-, y- and z-direction, are described separately because the available simulators do not support a description in matrix notation properly, as is known from MATLAB, for instance. If this feature is implemented in the VHDL-AMS simulators the exported VHDL-Models will become more compact and clearer.

Using the ROM

Due to manufacturing problems the sensor array using tuning by stress-stiffening effect could not be used. So the sensor with tuning by electrostatic softening was employed. The export of its ROM to VHDL-AMS is described above. The next step is to insert this ROM into the system environment. Because of a lower abstraction level of the ROM compared with the abstract sensor model, it might happen that the abstract sensor model uses an interface other than that of the ROM although the inputs and outputs of the sensor are the same. This will cause a lot of error prone work to adapt the system interface to the ROM interface. To avoid this, it must be assured that the ROM can use the same interface as the abstract model, wherefore a new methodical approach called „Multi Architecture Modeling“ was created.

The Design Method „Multi Architecture Modeling“

Component models generally have interfaces to their environment. The abstraction level of these interfaces depends on the abstraction level of the model. For a system developed within the scope of a top down system design it may have been necessary until now to modify the interfaces of the system and component models at every design step of a component model. One solution of this problem is Multi Architecture Modeling (MAM). The main idea of MAM is to use the interface, which will become necessary at a lower abstraction level (e. g., ROM) already at a high abstraction level (e. g., abstract sensor model). This avoids a lot of work because abstract models of a component can be replaced by detailed models or vice versa without any modification to the interface or the surrounding models.

This approach was applied in the development of the sensor models. This means:

1. to examine which inputs and outputs, including power supply, the sensor will need,
2. which types of interfaces (conservative/non-conservative analog nodes, digital nodes) the ROM will need,
3. which data types/natures these interfaces will need,
4. development of the abstract sensor model with this collected information,
5. checking the functionality of the sensor in the system context,
6. development of the ROM from the sensor's FEM model observing the information collected in steps 1 to 3,
7. to insert the ROM in the system model,
8. simulation of the complete system.

Without MAM, steps 1 to 3 would drop out and step 4 would be easier. But then step 7 would cause more work than needed for steps 1 to 4 because the system model must be adapted. If an error occurs after this adaptation it is not sure whether it was caused by the ROM or the adaptation. If MAM is used and the system model with the ROM is not working although it does with the abstract model then the error must be inside the ROM and the system simulation can be continued with the abstract model until a working ROM is available.

More details on „Multi Architecture Modeling“ can be found in [3], [6] and [7].

Simulation of the Sensor ROM

By using MAM, application of the ROM in the system context is very easy. First, the eight models for the individual resonators were tested separately to verify the method of reduced order modeling and the VHDL-AMS export. The results of these tests allowed to improve to the method of reduced order modeling and VHDL-AMS export. At the end of these tests the eight individual models were joined to the sensor ROM. Then the abstract model was replaced by the ROM simply by changing the name of the architecture which is to be instantiated in the VHDL-AMS system model. No further work was necessary.

With the macromodel a complete measurement cycle has been simulated.

RESULTS

The following simulation has been done on a SUN ULTRA60 workstation with UltraSPARC-II 296 MHz CPU. An AC-simulation with a tuning voltage of 35 V and a stimulation magnitude of 0,6 µm led to the following results:

Table 1. Results of AC simulation

	abstract, analytical model	ROM	measured
natural frequency	1,8 kHz	1,7 kHz	1,7 kHz
output voltage	0,6 V	0,8 V	0,5 V
simulation time	7 sec.	2 sec.	

It was a surprise that the more accurate ROM can be simulated faster than the abstract model. But this can be explained simply: The abstract model contains a set of simple equations. The ROM contains less but more complex equations than the abstract model. Obviously, the smaller numbers of complex equations can be simulated faster than larger numbers of simple equations.

The results seem to show that the abstract model is more accurate than the ROM. But this effect occurs due to manufacturing tolerances of the real sensor. If the ROM is compared to the FEM Model the error will be less than 1 %.

The simulation results of a transient simulation for 50 ms of the sensor and analog signal processing are shown in Table 2.

Table 2. Results of transient simulation

	abstract model	ROM
tuning voltage for a tuned natural frequency of cell 1 at 1 kHz	36,7 V	36,0 V
output voltage	1,2 V	1,1 V
simulation time	15 min.	7 min.

Again it can be seen that the simulation with the macromodel is faster than the simulation with the abstract model. The ROM is also more accurate.

By simulation of these system parts some side effects were determined, e. g., a „snapping“ of the resonators at high tuning voltages which was taken into account when assembling the prototype.

By using „Multi-Architecture-Modeling“ the system can be reconfigured for testing purposes between abstract model and ROM in less than 5 seconds.

At last a simulation of the whole system was performed to select stimuli patterns for the fuzzy pattern classifier. This simulation was done on a SUN Blade2 with 900 MHz CPU and 4 GByte RAM. It took about 1 hour and 30 minutes. A simulation plot can be seen in Figure 5. The first curve displays a sector of the mechanical stimulation of the sensor array containing a constant spectrum of frequencies representing machine vibrations and interferences. The second curve shows a sector of the response of the sensor. The next two curves display the applied tuning voltage and the output signal of the analog signal processing during a complete measurement cycle. The last curve shows the data sampled by the microcontroller which are transferred to the fuzzy pattern classifier. The differently grey backgrounds of the curves on the right show which sensor cell of the array is activated.

A comparison of these results to an FEM simulation is not possible because such complex systems cannot be simulated with FEM. In comparison to the manufactured system the simulation shows an error of approximately 3 % in frequency measurement and 40 % in amplitude measurement. These errors are due to manufacturing tolerances. Regarding the high amplitude measurement error, it must be noted that the sensor works in a resonant mode, so even small manufacturing tolerances may cause big measurement errors. These errors cannot be predicted, neither by VHDL-AMS nor by FEM simulation. With the results of the system simulation it has been possible to select significant patterns for digital signal processing using the fuzzy pattern algorithm. But due to the manufacturing tolerances, it is not possible to use the simulation results to train the classification algorithm.

CONCLUSION AND OUTLOOK

Within this work it was shown that the application of reduced order modelling in system design is a powerful approach to higher accuracy and simulation speed. By using the reduced order model instead of the abstract model the simulation error has been reduced from 10 % to below 1 % and the simulation speed has been doubled.

The application of the Multi Architecture Modelling approach within this context assures an easy, fast and safe exchange of abstract and reduced order models.

The approach of ROM including VHDL-AMS export is supported by a self developed EDA-Tool in combination with a commercial FEM simulator. In the future, this approach will be able to avoid a lot man work for developing abstract, less accurate models.

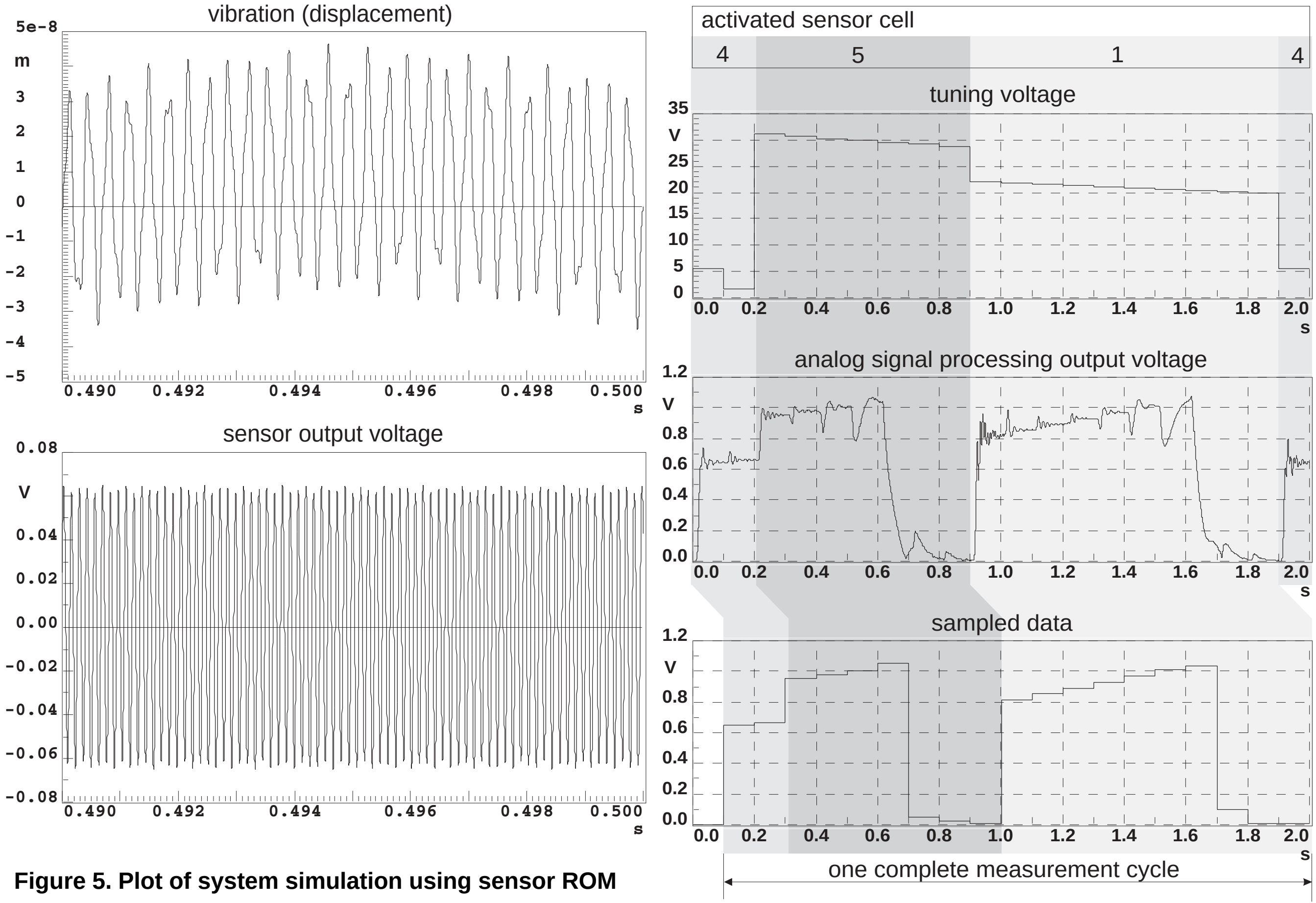


Figure 5. Plot of system simulation using sensor ROM

ACKNOWLEDGEMENTS

The work presented here has been done within the projects A1 „Component Design“ and A2 „System Design“ of the SFB 379 (collaborative research center), which is funded by the German Science Foundation (DFG).

REFERENCES

[1] Bennini, F.; Mehner, J.; Dötzel, W.: *Computational Methods for Reduced Order Modeling of Coupled Domain Simulations*. 11 International Conference on Solid State Sensors and Actuators (Transducers 01), Germany, 2001

[2] Mehner, J.; Scheibner, D.; Wibbeler, J.: *Silicon Vibration Sensor Arrays with Electrically Tunable Band Selectivity*. MICRO SYSTEM Technologies 2001, Düsseldorf, Germany, 2001

[3] Schlegel, M.; Herrmann, G.; Müller, D.: *Application of „Multi Architecture Modeling“ design method in system level MEMS simulation*. DTIP 2003, Mandelieu-La Napoule, France, 2003

[4] Schlegel, M.; Herrmann, G.; Müller, D.: *A system level model in VHDL-AMS for a micromechanic vibration sensor array*. IEEE International Conference on Sensors 2002, Orlando, USA, 2002

[5] Reitz, S.; Bastian, J.; Haase, J.; Schneider, P.; Schwarz, P.: *System Level Modeling of Microsystems using Order Reduction Methods*. DTIP 2002, Cannes, France, 2002

[6] Schlegel, M.; Herrmann, G.; Müller, D.: *Multi-Architecture-Modeling: Entwurfsmethode für Mixed-Signal- und Multi-Domain-Systemsimulation.* GI/ITG/GMM-Workshop: Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen, Tübingen, Germany, 2002

[7] Schlegel, M.; Herrmann, G.; Müller, D.: *Multi Architecture Modeling Desing Method for Mixed Signal and Multi Domain System Simulation - First Solutions*. International MEMS Workshop, Singapore, 2001

[8] Gabbay, L.D.: *Computer-aided Macromodeling for MEMS*. Ph.D. dissertation, MIT Cambridge, MA, USA, 1998

[9] Mehner, J. E. ; Gabbay, L.D.; Senturia, S.D.: *Computer-Aided Generation of Nonlinear Reduced-Order Dynamic Macromodels*. Journal of Microelectromechanical Systems, June 2000, pp 270-277