A Table-Based Time-Domain Simulation Method for Oversampled Microelectromechanical Systems

Jiangfeng Wu and L. Richard Carley
ECE Department, Carnegie Mellon University, Pittsburgh, PA 15213
{wjf, carley}@ece.cmu.edu

Abstract

This paper describes a table-based behavioral modeling method and an efficient transient simulation method for oversampled MEMS systems. A system is represented by a set of coupled ordinary differential equations (ODEs) numerically described by table-based macromodels. Cubic spline interpolation is used to evaluate the model functions. The instant settling approximation and the explicit ODE solver are employed to enable efficient transient simulation. These methods are used to simulate a delta-sigma MEMS accelerometer and demonstrate high simulation efficiency.

1. Introduction

The development of micro fabrication technology have enabled the integration of miniaturized sensors and actuators with analog and digital circuits to create micro-electro-mechanical systems (MEMS). Complex systems consisting of multiple sensors, actuators and electronics have been developed. Many systems employ closed-loop feedback control to enhance performance. Digital or analog sampled-data circuits are often used in closed-loop systems so that more accurate and complex control algorithms can be implemented. These clocked MEMS systems are usually over sampled and clocked at frequencies much higher than the bandwidth of transducer dynamics to offer high control bandwidth, track the high frequency modes and reduce the quantization noise. Two examples of the oversampled MEMS systems are digital output sensors using delta-sigma modulation [1, 2] and MEMS actuators with digital servo. For oversampled systems, the study of system dynamics requires long, in many cases, clock by clock transient simulation to collect a large number of data.

MEMS systems typically consist of multiple micro devices in multiple physical domains. The simulation of single-domain characteristics of micro devices is usually performed by solving partial differential equations (PDE) with geometry-related boundary conditions. Examples of device-level single-domain simulators include mechanical and thermomechanical simulators based on finite element analysis (FEA), capacitance simulators using finite difference method (FDM) or boundary element method (BEM), and simulators for fluid, electromagnetic field and semiconductor devices. Device-level coupled-domain simulations and dynamic simulations are very costly in computation and currently under intensive research.

For complex systems consisting of multiple components, system-level simulation using behavioral representations of devices, is a more efficient way to study the system dynamics, specially when the transducers are integrated with electronics. In recent years, behavioral macromodels have been created for a variety of micro devices. Based on libraries of macromodels, several system-level MEMS simulators have been developed. These simulators use circuit simulators such as SPICE, or general-purpose simulation tools such as MATLAB, to solve the system equations. Accordingly, the macromodels are represented either in the form of lumped network elements, or in the form of analytical ordinary differential equations (ODE). There are two limitations in these simulation tools. First, the accuracy of the simulation is limited by the systematic model errors of the macromodels. Second, since these tools relies on host simulators to solve the system equations, it is difficult for them to exploit certain properties of the systems to improve the simulation efficiency.

In this paper, a behavioral modeling technique and a system-level simulation technique are described that efficiently model and simulate oversampled complex MEMS systems. A table-based modeling approach [3] is introduced in which models are built on numerical data acquired by device-level simulations, and are evaluated by cubic spline interpolation [6]. Therefore, both the need to derive analytical functions and systematic model errors are eliminated. The main advantage of this approach is that the macromodels are constructed directly from sets of numerical data without the knowledge of the underlying physics, therefore, the model extraction can be easily automated. Furthermore, an automated simulation flow then can be constructed with individual device-level simulations, the table-based model extraction and the system-level simulation. The simulation technique is targeted to oversampled MEMS systems in which some parts of the systems can be treated as discrete-time systems. By exploiting the oversampling property, these systems can be simulated by solving differential equations using clock-driven explicit ODE solvers under instant settling approximation [4], which results in significant improvement in simulation speed.

2. Table-Based Behavioral Modeling

Many continuum physical systems are described by partial differential equations (PDE) of spatial and temporal variables. For complex systems involving multiple physical domains, it is often impossible to dynamically solve the large sets of PDEs of different characteristics. To study complex systems, the systems are partitioned into lightly coupled components. As the separation of spatial variables and temporal variable can often be applied to these components, we are able to simulate the individual components by solving time-independent PDEs at much lower computational cost and then represent the components by nonlinear ordinary differential equations (ODE). The system description is then reduced to a set of coupled ordinary differential equations. With the spatial variables and geometry-related boundary conditions removed from the system description, the dynamic simulations of the systems is realized by solving initial value problems of ODEs, which can be done much more efficiently than solving PDEs. Specifically, with behavioral modeling, a system is divided into several large components, and each component is represented by low-order macromodel, hence, a relatively small set of coupled ODEs are obtained, which enables more efficient yet less accurate simulations.

The behavior of a MEMS device or a MEMS system is described by a set of ODEs in the following form:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= f(\mathbf{x}(t), \mathbf{u}(t), t) \\ \mathbf{y}(t) &= g(\mathbf{x}(t), \mathbf{u}(t), t) \end{aligned} \tag{1}$$

where $\mathbf{u}$ is the input vector, $\mathbf{x}$ represents the vector of state variables, $\mathbf{y}$ is the output vector, function f gives the derivative vector, and g is the output function. For each device, the functions f and g are represented by a set of model functions obtained from device-level simulations or experimental measurements. In many cases, the model functions of MEMS transducers are time invariant, therefore, can be constructed based on static simulations that use methods such as FEA, FDM and BEM, to solve time-independent PDEs with geometry-related boundary conditions. For electronic circuits, macromodels can be built based on circuit simulations. For a system, as the system is represented by an interconnected set of macromodels, the system behavior is described by a collection of model equations of all components, which is a set of coupled ordinary differential equations.

Traditionally, the macromodels are represented either in the form of lumped electrical network elements or directly by analytical functions and equations. These models are usually in functional forms with some physical meanings. The model parameters are estimated by fitting simulated or measured data. And correction terms are occasionally added to take some second-order effects into account. It is sometimes difficult to construct a good analytical model. The model errors of analytical models will cause systematic and global inaccuracies in system-level simulation. And more accurate models often result in complex functional forms which make both the model parameter estimation and the model evaluation during the simulation more expensive in computation. Finally, it is worthwhile to point out that, although an analytical model is valuable for designing the device, it is not necessary for simulating the system.

An alternative approach is to represent the coupled ODEs in a numerical form by using table-based models. Numerical models of single-domain device characteristics, in the form of tables, are built on the data acquired by single-domain simulations or experimental measurements. In this approach, only the order and the form of the ODE, and the state variables need to be determined, the model functions are directly described by sets of numerical data organized in tables. Therefore, the difficulties in deriving analytical functions and the systematic model errors are eliminated altogether. And some nonlinear behaviors that are difficult to describe by analytical functions, such as hysteresis, can be easily represented in tabular form. The construction of these table-based models requires no knowledge of the underlying physics, no human intervention, and considerably less computation than analytical model parameter estimation. The numerical model extraction can be easily automated and be built into general simulation flows as the bridge between the device-level simulation and the system-level simulation.

In this work, the table-based model functions are evaluated by performing multi-dimensional cubic spline interpolations [6]. The cubic spline interpolation is a method that interpolates functions with localized cubic polynomials and globally determined second derivatives to ensure smooth first derivatives and continuous second derivatives everywhere. For simplicity, let us consider an one-dimensional cubic spline interpolation problem. The function value y at variable value x between x_i and x_{i+1} is computed by a cubic polynomial,

$$\begin{aligned}
y &= ay_i + by_{i+1} + cy''_i + dy''_{i+1} \\
a &= \frac{x_{i+1} - x}{x_{i+1} - x_i} \\
b &= \frac{x - x_i}{x_{i+1} - x_i} \\
c &= \frac{1}{6}(a^3 - a)(x_{i+1} - x_i)^2 \\
d &= \frac{1}{6}(b^3 - b)(x_{i+1} - x_i)^2
\end{aligned} \tag{2}$$

As shown in the equation, the second derivatives of the functions must be pre-computed and stored before interpolation. The calculation of the second derivatives is called spline construction. Spline interpolation is well known to have good smoothness and stability property. Because of the low order of the interpolation polynomials, evaluating functions by spline interpolation is faster than evaluating many analytical functions. One may argue that extra efforts are needed to locate the interpolation intervals. However, since most functions vary continuously, the data in two consecutive simulation steps, in most cases, will fall in the same interpolation interval or in the neighboring intervals. Thus, locating the interpolation intervals gives very little overhead in simulation. The second derivatives are calculated by solving a set of linear equations and the equation matrix is tridiagonal, therefore, a list of all second derivatives can be computed efficiently in *O(n)* operations.

For functions with multiple variables, the computational complexity of spline interpolation increases with the number of variables. For example, to interpolate a two-variable function with *n* tabulated values for one variable and *m* tabulated values for the other variable, *m* interpolating operations are first performed on the *n* tabulated values to calculate the intermediate function values, then a list of *m* 2rd derivatives are computed, and finally another interpolating operation is used get the function value. Thus, unlike in one-dimensional interpolation problems, the spline construction now must be performed dynamically during the simulation. Since the reason for the second derivatives to be determined globally is to ensure the interpolation stability to avoid excessive errors, it is often not necessary to construct the spline across the whole table. Therefore, for multi-variable problems, the increase of computational complexity can be reduced by limiting the spline construction and spline interpolation in a fraction of the entire table surrounding the interpolation point.

In this work, the tables of the first and second derivatives are computed in the preprocessing phase of each simulation. The first derivatives are used to solve nonlinear equations to get the initial operating points. They may also be used by implicit solvers to solve the differential equations [6]. The second derivatives are used by the interpolation routine to evaluate the model functions. The derivative tables can also be established as a part of the models during the model extraction.

3. Efficient Transient Simulation

Many MEMS systems employ feedback. For closed-loop systems, digital circuits such as digital signal processors (DSP) and analog sampled-data circuits such as switched capacitor (SC) filters are often used due to their advantages in accuracy and programmability. Several delta-sigma digital output sensors using SC circuits have been reported [1, 2]. And the advancement of DSPs makes it very attractive for the control of MEMS transducers. These clocked systems are often also over sampled, which means the system is clocked at a frequency much higher than the bandwidth of the transducers, in order to provide larger control bandwidth, to control the high-frequency modes, and to reduce the quantization noise. Because of oversampling, the measurement of the system performance requires long transient simulations of many clock cycles to collect large number of sample data. The efficiency of the simulation is extremely important for this type of systems, which is the subject of our investigation.

Time-domain simulation is to solve the initial value problems for coupled ODEs by numerical integration. The major challenge in simulating an oversampled system is that the system is a multi-rate system with the transducers and the clocked control electronics operated at very different frequencies. Nevertheless, several properties of oversampled MEMS systems can be exploited to improve the simulation efficiency. First, the system dynamics is dominated by the transducers and the front-end electronics, while the inner-clock behaviors and settling properties of the clocked control electronics having very little impacts. Second, the system is over sampled so that the clock frequency is much higher than the transducer bandwidth. It is not necessary to accurately simulate the clocked circuit components, such as SC filters and DSPs, within each clock period. Instead, they can be modeled as discrete-time systems by instant settling approximation [4]. The instant settling approximation assumes the output of sampled-data circuit settles instantly at the clock edges and remains constant between neighboring clock edges. The whole system is then divided into two interconnected parts, a set of coupled ODEs that describe the transducers and the front-end electronics, and a discrete-time system that represents the clocked control circuits. Because the system is oversampled, the clock period is larger than the integration timestep needed to solve the ODEs with adequate accuracy. Thus, by evoking numerical integration only at a clock event to limit the number of total integration steps, the simulation time is reduced.

The use of instant settling approximation is justified by the fact that only the data at the sampling time and the signal fed back to the transducer will affect the system behaviors. It is also verified in our simulation example that solving the system ODEs by integrating once every clock cycle is sufficiently accurate for our particular problem. In addition, non-ideal effects in clocked circuits that might affect the system performance, such as the rise and fall time of clocks, timing jitters, and slew-rate limiting, can be represented by behavioral macromodels within the discrete-time system representation.

The speed of the transient simulation is further improved by examining the algorithms that solve the ODEs. Most circuit simulators use implicit methods, such as trapezoidal and Gear, to solve ODEs by a combination of numerical integration and Newton-Raphson iterations. Compared with explicit ODE solvers that do not require Newton iterations, the implicit methods have better stability property, but are slower and may introduce nonconvergence. In behavioral simulation of MEMS systems, unlike in circuit simulations where the system is represented by a large-scale netlist of circuit elements, the behavioral representation of a system consists of a relatively small number of macromodels. With clocked circuit blocks reduced into discrete-time models, most simulation problems are non-stiff problems in nature. All these lead to lower stability requirement for ODE solvers. It is then possible to use explicit methods, such as forward Euler and explicit Runge-Kutta, to solve the system equations. In this work, as spline interpolation on tabulated data is used in simulation, the use of explicit solvers is further justified because spline interpolation itself has very good stability property. By eliminating Newton iterations, the explicit methods provide significant speed improvement and avoid the convergence problem completely [6].

In summary, a complex oversampled system can be simulated efficiently by using the clock-driven explicit method to solve the system ordinary differential equations.

4. Simulation Example

The methods introduced in previous sections are used to simulate a digital-output delta-sigma (Δ–Σ) MEMS accelerometer shown in Figure 1. This system consists of a MEMS transducer and circuit blocks including a readout amplifier, a switched-capacitor filter, an A/D converter, a digital feedback circuit and a digital decimator that produces the high-resolution digital output. A micromechanical resonator structure translates the acceleration into displacement. The displacement is detected and converted into electrical signal by capacitive sensing. The feedback is applied by an electrostatic actuator. The resonant frequency of the transducer is about 8 KHz. And the designed signal bandwidth is smaller than 1 KHz. To reduce quantization noise, the system is clocked at 1 MHz to provide over-sampling-ratio (OSR) greater than 1000. Because of the large OSR, measuring the system performance, such as signal-to-noise ratio (SNR), requires multiple runs of long transient simulations to collect millions of data, which takes very long time for our existing tool based on circuit simulators. This is the primary motivation of this investigation.

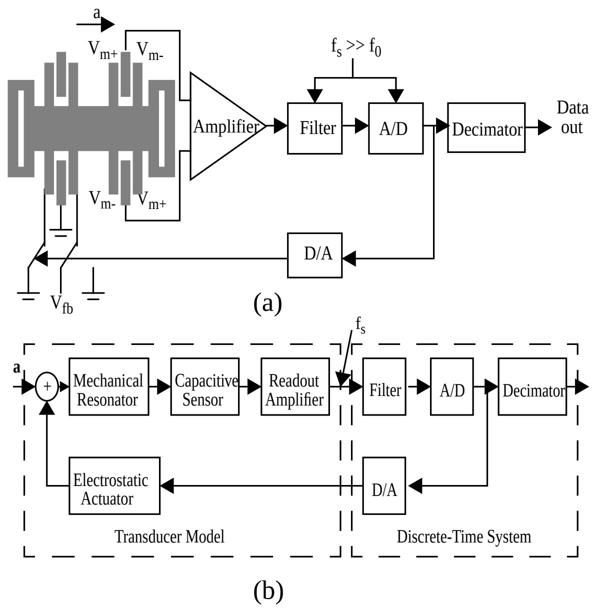


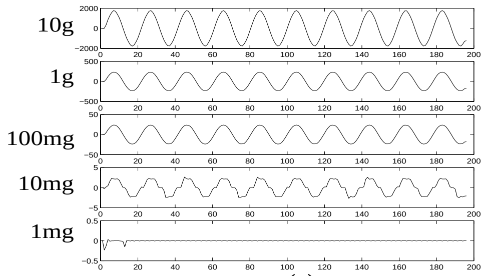
Figure 1: The delta-sigma MEMS accelerometer (a) and its behavioral model representation used in simulation (b).

To model the system, the transducer is partitioned into a micromechanical resonator, a capacitive position sensor, and an electrostatic actuator. The transducer is simulated by Abaqus which is a FEA mechanical simulator, and Raphael which is a FDM capacitance solver. The readout amplifier is simulated by HSPICE. Based on these simulations, table-based macromodels are constructed. Non-ideal effects, such as the cross-axis sensitivity, the electrostatic spring softening, and the parasitic capacitance at the sensor-circuit interface, are included in the models. Each mode of the mechanical resonator is represented by a 2nd-order ODE. Among these modes, only one mode is in the sensing direction, the others give rise to cross-axis sensitivity. If *n* modes are considered in simulation, the transducer is modeled by a nonlinear system of order 2*n*. As anti-alias filtering is performed before the sampling to limit the noise bandwidth, a single-pole first-order model is used to described its low-pass frequency response. Thus, the total order of the system is *2n+1*. During each simulation, the tables of the first and second derivatives are first computed, and the initial operating points are determined by solving the nonlinear equations, then the explicit 4th-order Runge-Kutta method is used to solve the system differential equations. The time derivatives are evaluated by a series of multi-dimensional cubic spline interpolations on the tabulated model functions. The uses of the spline interpolation, the instant settling approximation and the explicit method were first validated on small simulation problems using SIMULINK. Compared to implicit methods and methods using small step size or automatic step control, this clock-driven explicit method gives the same digital output sequences while offering significant speed improvement. A custom simulator written in C++ language has been developed to simulate the real problems.

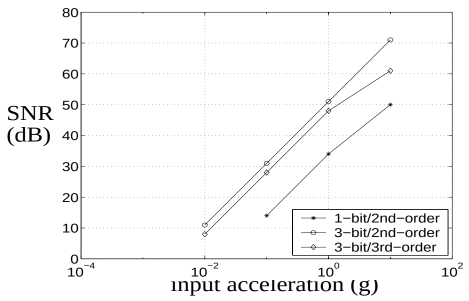
The simulator is used to simulate several Δ–Σ architectures with different quantizers and loop filter structures, including a 2nd-order Δ–Σ modulator with 1-bit A/D and no integrator, a 2nd-order modulator with 3-bit A/D and no integrator, and a 3rd-order modulator with 3-bit A/D and one integrator. The primary goal of the simulations is to determine which modulator architecture provides the lowest quantization noise floor. The signal-to-noise ratio (SNR) is estimated on 1000 decimated data from 1 million simulation cycles. Figure 2 shows some simulation results. In addition, macromodels of the transducer mechanical noise, the circuit noise, and the clock timing jitter can be inserted into the simulation to study their effects on system performance. Our simulator is able to compute 50,000 clock cycles in 280 CPU seconds on a 300 MHz Sun UltraSparcII workstation.

5. Conclusions

In this paper, we presented a table-based behavioral modeling method that uses spline interpolation to evaluate tabulated model functions, and an efficient transient simulation method for oversampled MEMS systems based on behavioral modeling, instant settling approximation and explicit ODE solvers. By combining these techniques, we are able to simulate a complex MEMS accelerometer system very efficiently.



(a)



(b)

Figure 2: Simulation results: (a) simulated waveforms of decimated output; (b) signal-to-noise ratio (SNR) of different modulator architectures.

Acknowledgement

This research was sponsored by the Defense Advanced Research Projects Agency and the AFRL, Air Force Materiel Command, USAF, under agreement F30602-97-2-0323. The United States government has certain rights to this material.

References

[1] G.K. Fedder, "Simulation of Microelectro-mechanical Systems", Doctoral Dissertation, Univ. of California at Berkeley, 1994.

[2] M. Lemkin, "Micro Accelerometer Design with Digital Feedback Control", Doctoral Dissertation, Univ. of California at Berkeley, 1997.

[3] R.J. Bishop, J.J. Paulos, M.B. Steer, and S.H. Ardalan, "Table-Based Simulation of Delta Sigma Modulators", *IEEE Trans. on Circuits and Systems*, vol. 37, pp. 447-451, March 1990.

[4] I. Yusim, "Simulation of Switched-Capacitor and Switched-Current Networks Under Instant Settling Approximation", Doctoral Dissertation, Columbia University, 1999.

[5] L.T. Pillage, R.A. Rohrer and C. Visweswariah, *Electronic Circuit and System Simulation Methods*, McGraw-Hill, 1995.

[6] H.R. Schwarz, with a contribution byJ.Waldvogel, *Numerical Analysis: A Comprehensive Introduction*, Wiley, New York, 1989.