An extension to SystemC-A to support mixed-technology systems with distributed components

Chenxu Zhao and Tom J. Kazmierski
School of Electronics and Computer Science
University of Southampton, UK
Email: cz05r,tjk@ecs.soton.ac.uk

***Abstract*— This contribution proposes syntax extensions to SystemC-A that support mixed-technology system modelling where components might exhibit distributed behaviour modelled by partial differential equations. The important need for such extensions arises from the well known modelling difficulties in hardware description languages where complex electronics in a mixed-technology system interfaces with distributed components from different physical domains, e.g. mechanical, magnetic or thermal. A digital MEMS accelerometer with distributed mechanical sensing element is used as a case study to illustrate modelling capabilities offered by the proposed extended syntax of SystemC-A.**

I. Introduction

SystemC-A [1] is a superset of SystemC intended to extend the modelling capabilities of SystemC [2] to the analogue domain. It provides constructs to support user defined ordinary differential and algebraic equations(ODAEs), and analogue components to enable modelling of analogue, mixed-signal and mixed-domain systems from high level of abstraction down to the circuit level. Most powerful features of VHDL-AMS [3] and Verilog-AMS [4] are provided in SystemC-A in addition to a number of extra advantages such as high simulation speed, support for hardware-software co-design and high levels of modeling [1]. SystemC-A is used to model mixed-signal and mixed-physical domain systems such as automotive seating vibration isolation system [1] and ferromagnetic hysteresis [5].

Although major AMS HDLs such as SystemC-A and VHDL-AMS are very powerful and flexible mixed physical domain modeling tools, they face a challenge in modelling mixed-technology microsystem applications such as energy harvesting systems and MEMS sensors. This is because current HDLs only support ODAEs modelling. This limits accurate modelling of systems with distributed effects (mechanical [6], electromagnetic(EM) [7], thermal [8], [9], etc.) which can not be neglected and may even play vital roles. Thus an implementation of PDEs in major HDLs is in demand. Some attempts have already been made to implement PDEs within the existing limits of major AMS-HDLs [10], [11], [8]. Among them, a proposal for syntax extension for VHDL-AMS(named VHDL-AMSP) has been presented [10]. Pending the development of a new standard, a preprocessor has been developed to convert VHDL-AMSP into the existing VHDL-AMS 1076.1 standard automatically which can be simulated using currently available simulators. In this paper, we propose the first full implementation of the PDE extension to SystemC-A where no preprocessor is required.

The proposed extension has particular advantages in mixed physical technology systems that exhibit distributed physical effects. For example, electromechanical Sigma-Delta MEMS sensor designs, e.g. accelerometers and gyroscopes, which are based on incorporation of mechanical sensing elements into $\Sigma\Delta$ modulator control loops, have attracted great research interest [12]. The mechanical sensing element, which is usually modeled by the lumped mass-spring-damper model(a second order ODE), is also a part of the loop filter in these systems. However, the lumped model only can capture the first resonant mode which is not accurate enough as higher order mechanical resonant modes may significantly affect the performance and stability of the $\Sigma\Delta$ control loop [6]. Consequently, it is necessary to improve the accuracy of the mechanical model and use partial rather than ordinary differential equations. In section III the case study of a MEMS accelerometer embdedded in a $\Sigma\Delta$ control loop is modelled in the extended SystemC-A to demonstrate the efficiency of the new syntax. The accelerometer uses distributed sensing elements.

II. SystemC-A syntax extension and implementation

This section describes the new syntax of SystemC-A with which users can define PDEs.

The abstract base class for PDEs is derived from the existing SystemC-A abstract base class $sc\_a\_component$. Both PDEs and their boundary conditions are generated from the new abstract base classes $sc\_a\_PDE\_base$ in the proposed extension. This new abstract base class also inherits the virtual *build* method which is invoked by the SystemC-A analogue kernel at each time step to build the system matrix from contributions of all the components. A sample component class hierarchy with PDE extension is shown in Fig.1. The mechanical component in this example includes user defined

978-3-9810801-7-9/DATE11/©2011 EDAA

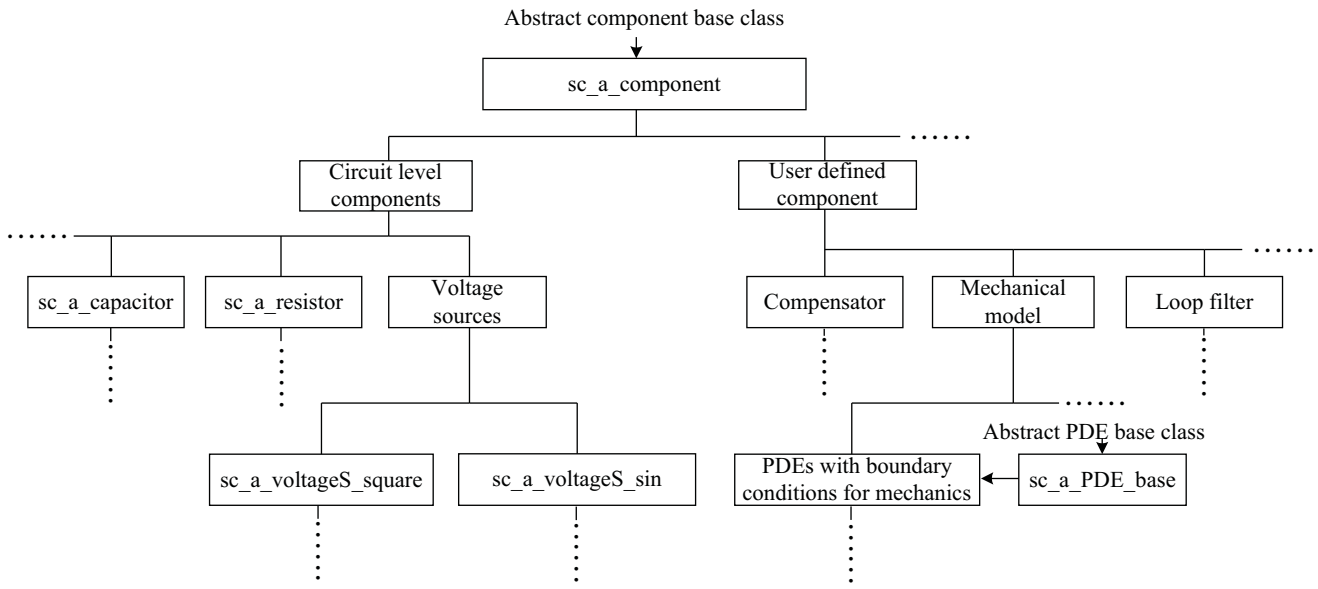

Fig. 1. SystemC-A component hierarchy with PDE extensions.

PDEs and associated boundary conditions which are derived from the PDE base class.

A finite difference approach is used to discretize the PDEs with respect to spatial variables and leave the time derivatives unchanged. Consequently, PDEs are converted to a series of ODEs which can be handled by the existing SystemC-A analogue solver. The modelling flow in SystemC-A with PDE extensions is shown in the Fig.2.

The following example of a simple one dimensional PDE demonstrates the new syntax:

$$\frac{\partial Q(x,t)}{\partial x} + A\frac{\partial Q(x,t)}{\partial t} = B \tag{1}$$

Let the boundary condition be:

$$\frac{\partial^{M+N} Q(x,t)}{\partial x^M \partial t^N} = C; \tag{2}$$

where $Q(x,t)$ is the partial quantity of interest, A is the parameter, B is the excitation, C is the right hand side value of the boundary condition equation, x is a spatial variable and t is time. The boundary condition is defined in Eq.2.

The extended SystemC-A code for this example is:

```
sc_a_PDE_example::sc_a_PDE_example{
  sc_a_PDE_base("PDE_example"){
  PDE_Coordinate_Declaration("Q","x",R,
                             N_node,dx);
 }
}

void sc_a_PDE_example::Build{
 Pdxdt_Boundary(M,N,"Q", x, C);
 for(x=1;x<=N_node;x++)
 {
 PDE(x,-Pdx(1,"Q",x)-A*Pdt(1,"Q",x)+B);
 }
}
```

A. Spatial Coordinate and Partial Quantity

Currently, in SystemC-A, three types of analogue system variables (node, flow, quantity), which are derived from an abstract base class called sc_a_system_variable, are defined. In the proposed PDEs extension, a new type of system variable (Partial Quantity), which is also derived from the abstract base class, is defined as illustrated in Fig.3.

The method $PDE\_coordinate\_Declaration()$ is used for partial quantity definition and spatial coordinate declaration. Multiple coordinate declarations will form a hypercube in the multi-dimensional space. As shown in the example code, the spatial coordinate "x" with the range R is divided into N_node segments and the partial quantity "Q" is discretized and defined inside the function in array format and discretization step size dx(R/N_node) is returned.

The method $PDE\_Quantity()$ is used to read a value of a particular partial quantity. For example,$PDE\_Quantity("Q", x)$ returns the value of Partial Quantity Q at node x. This function's counterpart in SystemC-A is X() which reads the value of a quantity. Similar to the differentiator ($Xdot()$) and integrator ($INTEG()$) operators which can be performed on ordinary quantities, the new methods ($PDE\_Quantity\_dot()$ and $PDE\_Quantity\_INIEG()$) allow performing these two operators on partial quantities.

B. Partial Derivatives

If "Q" is a partial quantity, the function Pdx(N,"Q",x) represents the derivative of "Q" with respect to spatial coordinative at position x. N is an integral number which represents the derivative order. For example, Pdx(4,"Q",x) represents the 4th order partial derivative $\frac{\partial^4 Q}{\partial x^4}$. A partial quantity can also have a derivative with respect to time, using the attribute dt, so item Pdxdt(3,2,"Q",x) represents $\frac{\partial Q^5}{\partial^3 x \partial^2 t}$.

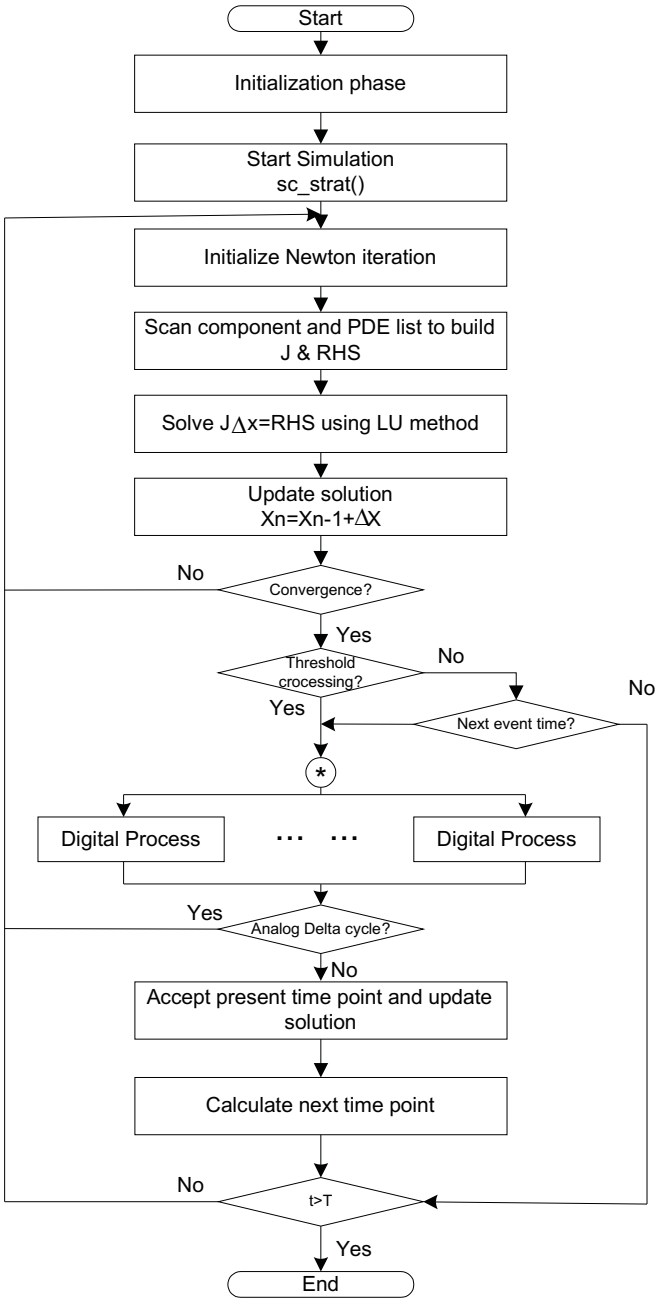

Fig. 2. Simulation cycle with PDE modelling in extended SystemC-A.

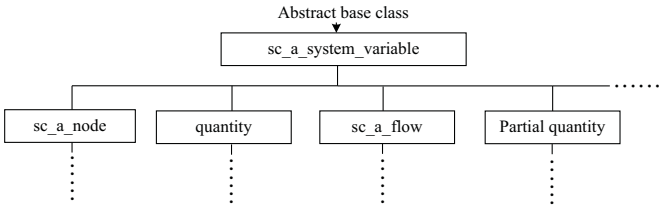

Fig. 3. Analogue system variables

C. Boundary Conditions

Boundary condition are declared by method $Pdxdt\_Boundary()$. As shown in the example code above, M and N determine the order of derivative with respect to coordinate x and time t, x is the specified position where the conditions should apply and C is the right hand side value of the boundary condition equation. As an example, $Pdxdt\_Boundary(1, 0, "Q", 100.0, 0.0)$ represents the first order derivative of Q at the user specified spatial boundary (x=100.0) is equal to 0.0.

D. PDE Formulation Method

Function PDE() realizes the automatic equation formulation of the PDEs to be modelled. This function is required to be implemented in a "for" loop and the number of loops determined by the number of segments(N_node). After providing the RHS vector will be provided in the 2nd term of the PDE() function, Jacobian matrix will be automated generated using a secant finite difference approximation which is defined in terms of system RHS(fi(xj)) and a scalar Δx:

$$J_{ij} = \frac{\partial f_i}{\partial x_j} = \frac{f_i(x_j + \Delta x_j) - f_i(x_j)}{\Delta x_j} \tag{3}$$

Finally, the function matrix is solved in the embedded SystemC-A analog solver.

III. Example: Distributed MEMS accelerometer with $\Sigma\Delta$ control loop

A single axis lateral capacitive MEMS accelerometer with a $\Sigma\Delta$ control loop is used as an example to illustrate the SystemC-A syntax extension. Fig.4 shows the block diagram of this system.

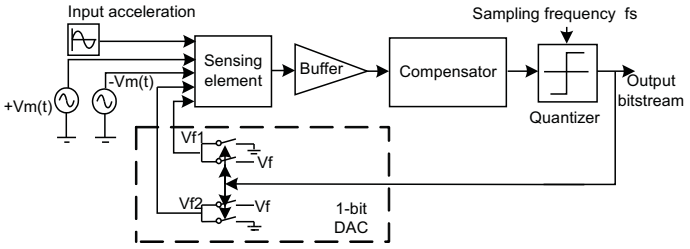

Fig. 4. 2nd order electromechanical Sigma-Delta accelerometer

Conventionally, the mechanical sensing element is used as a loop filter to form the 2nd order electromechanical $\Sigma\Delta$ accelerometer. The mechanical part of a lateral capacitive MEMS accelerometer(Fig.5) is composed of a proof mass, springs and comb fingers. In the lateral capacitive structure, the proof mass is suspended by springs and it is equipped with sense and force comb fingers which are placed between fixed fingers to form a capacitive bridge. The sense fingers moves with the proof mass resulting in a differential imbalance in capacitance which is measured. The electrostatic force acting on the force fingers is used as the feedback signal to pull the proof mass in the desired direction.

Like most conventional modelling approached, the sensing element dynamics in the sense-direction is normally modeled to reflect only one resonant mode by a lumped mass, spring,

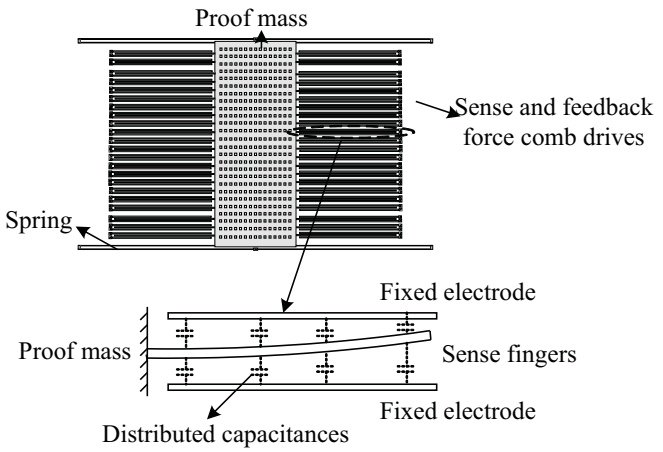

Fig. 5. Mechanical sensing element

and damper. which is represented by a simple 2nd order ordinary differential equation:

$$M\frac{d^2y(t)}{dt^2} + D\frac{dy(t)}{dt} + Ky(t) = Ma_{in}(t) + F_{feedback}(t) \quad (4)$$

where M is the total mass of the structure, D and M are damping and spring coefficients. $a_{in}(t)$ is the input acceleration and $F_{feedback}(t)$ is the feedback force.

Distributed mechanical modeling is important in such MEMS designs with digital control because the dynamics of mechanical components may severely affect the system's performance. It has been well documented that sense fingers in lateral capacitive accelerometers may vibrate due to their own dynamics, thus rendering the feedback excitation ineffective, causing an incorrect output and a failure of the system [6]. This failure scenario cannot be modeled by the conventional mass-damper-spring model 2nd order ordinary differential equation.

The motion of the sense beam can be modeled by the following partial differential equation (PDE):

$$\rho S\frac{\partial^2 y(x,t)}{\partial t^2} + C_D I\frac{\partial^5 y(x,t)}{\partial x^4 \partial t} + EI\frac{\partial^4 y(x,t)}{\partial x^4} = F_e(x,t) \quad (5)$$

where $y(x,t)$, a function of time and position, represents the beam deflection, E, I, C_d, ρ, S are physical properties of the beam: ρ is the material density, S is the cross sectional area ($W_f * T$), where W_f and T are width and thickness of the beam), E is Young's modulus and I is the second moment of area which could be calculated as $I = TW_f^3/12$ and C_D is the internal damping modulus. The product EI is usually regarded as the stiffness . Feedback force is acting on the proof mass, so the sense finger is only deformed by distributed electrostatic force and input acceleration. $F_e(x,t)$ in the equation is the electrostatic force per unit length:

$$F_e(x,t) = \frac{1}{2}\varepsilon W_f[\frac{V_m(t)^2}{(d_0 - y(x,t)^2)} - \frac{V_m(t)^2}{(d_0 + y(x,t)^2)}] \quad (6)$$

where ε is the permittivity of the gap, d_0 is the initial gap between the sense finger and the electrode and $V_m(t)$ is the high frequency carrier voltage.

The boundary conditions at the clamped end and the free end are shown in the following equations:

At the clamped end (x=0):

$$y(0,t) = z(t) \quad (7)$$

$$\theta = \frac{\partial y(0,t)}{\partial x} = 0 \quad (8)$$

At free end (x=l):

$$M = -\frac{\partial^2 y(l,t)}{\partial x^2} = 0 \quad (9)$$

$$Q = -\frac{\partial^3 y(l,t)}{\partial x^3} = 0 \quad (10)$$

where θ, M and Q denote the slope angle, the bending moment and the shear force respectively, l is the length of the finger.

The clamped end of the sense fingers moves with the lumped proof mass whose deflection z(t) could be modelled by a 2nd order differential equation Eq.(4).

The total distributed sense capacitance between the sense fingers and electrodes is:

$$C_1(t) = N_s\varepsilon T\int_0^{L_f}\frac{1}{d_0 - y(x,t)}dx \quad (11)$$

$$C_2(t) = N_s\varepsilon T\int_0^{L_f}\frac{1}{d_0 + y(x,t)}dx \quad (12)$$

Where N_s is the number of sense fingers. The output voltage can be calculated as:

$$V_{out}(t) = \frac{C_1 - C_2}{C_1 + C_2}V_m(t) \quad (13)$$

Where $V_m(t)$ is high frequency carrier voltage applied on the fixed electrode in comb fingers unit.

A. SystemC-A Code for mechanical sensing element

The SystemC-A model of the mechanical sensing element present below provides an example of how the PDEs discussed above are implemented.

```
PDE_sensing::PDE_sensing(char nameC[5],
sc_signal<double>*Output, sc_signal<double>*input,
sc_signal<double>*feedback, sc_signal<double>*Vm){

 sc_a_PDE_base("sensing"){
   PDE_Coordinate_Declaration("Y","x",100e-6,10,dx);
   Deflection=Output;
   ain_sig=input;
   feedback_sig=feedback;
   Vm_sig=Vm;

// Quantities for lumped mass-damper-spring model
   z[1] = new Quantity("z1");
   z[2] = new Quantity("z2");
 }
```

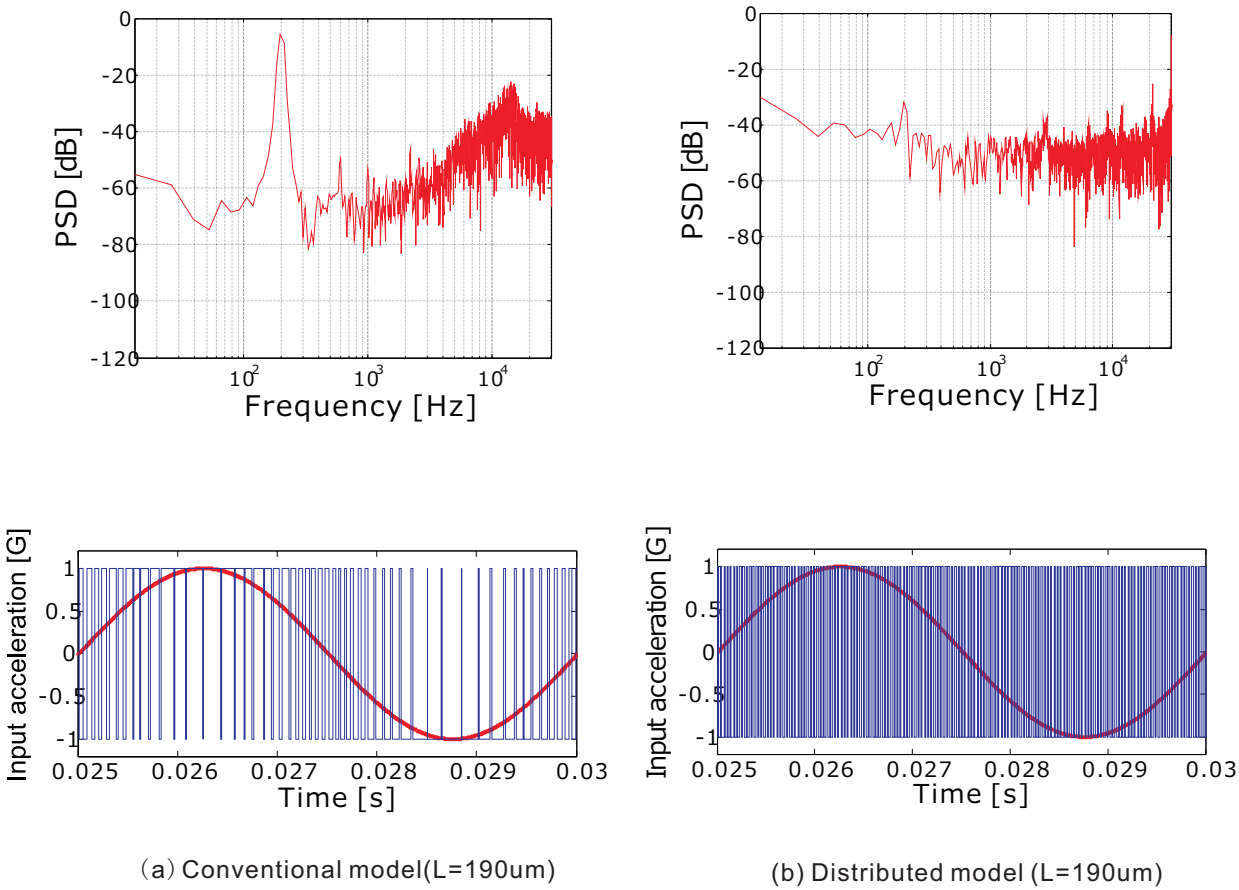

Fig. 6. Failure of the Sigma-Delta control is captured by the distributed model

```
}

void PDE_sensing::build(void){

//Parameters of sensing element
  mass=3.57e-10;
  spring=0.2855;
  damper=1.62e-5;
  d0=1.5e-6;
  ...

//Get input acceleration
 ain=ain_sig->read();
 fin=mass*ain;

//Get feedback force;
Ffeedback=feedback_sig->read();

//Get modulation voltage Vm;
Vm=Vm_sig->read();

//Deflection of proof mass(2n order ODE)
    Zn[1]=X(z[1]);
    Zdotn[1]=  Xdot(z[1]);
    Zn[2]=X(z[2]);
    Zdotn[2]=  Xdot(z[2]);
    Equation(z[1],-Zdotn[1] + Zn[2]);
    Equation(z[2],-mass*Zdotn[2]-(damper)*Zn[2]
    -(spring)*Zn[1]+fin+Ffeedback);

//Boundary conditions
    Pdxdt_Boundary(0,0,0,Zn[1]);
    Pdxdt_Boundary(1,0,0,0);
    Pdxdt_Boundary(2,0,100e-6,0);
    Pdxdt_Boundary(3,0,100e-6,0);

//PDE function
    for(x=1;x<N_node+1;x++)
    {
      Yn[x]=PDE_quantity("Y",x);
    }

     for(x=1;x<N_node;x++)
     {
       PDE(x, -1*ro*A*Pdxdt(0,2,"Y",x)-
       c*Pdxdt(4,1,"Y",x)-E*I*Pdx(4,"Y",x)
       +((ep*A2)/(2))*(Vm*Vm/((d0-Yn[x])
       *(d0-Yn[x]))-Vm*Vm/((d0+Yn[x])*(d0
       +Yn[x])));
     }

//Average deflection of the sense finger
     Y=0.0;
     for(x=1;x<N_node+1;x++){
       Y=Y+Yn[x];
     }
     Y=Y/N_node;

//Write output
     Deflection->write(Y);
}
```

B. Simulation results

In this section, simulation results of the distributed model illustrate the distributed effects of the finger dynamics. The conventional model only contains the dynamics of the lumped proof mass as shown in Eq.(4). This means that the sense fingers and lumped mass move together and the sense fingers do not bend. The frequency is approximately:

$$w_0 = \sqrt{\frac{K}{M}} \tag{14}$$

In reality, the sense fingers bend to the extent that the performance of the Sigma-Delta control loop might be seriously affected. The first two resonant frequencies of a sense finger could be calculated as those of the cantilever beam [6]:

$$\omega_i = \alpha_i^2 \frac{W}{L^2}\sqrt{\frac{E}{12\rho}} \quad i = 1, 2 \tag{15}$$

$$\alpha_1 = 1.875 \quad \alpha_2 = 4.694 \tag{16}$$

Where W and L is the width and length of the sense fingers respectively. First and second resonant frequencies are 31KHz and 209KHz respectively if the finger dimensions are: $L = 190\mu m$, $W = 1\mu m$, $T = 2\mu m$. The effect of the sense finger dynamics are illustrated in Fig.6 which shows the power spectrum density (PSD) of the Sigma-Delta modulator output bitstream and time-domain response when applying 200Hz sine wave input acceleration with 1G($9.8m/s^2$) amplitude and 2^{17}Hz oversampling frequency to both conventional and proposed distributed system.

Here, the SystemC-A simulation carries out the time-domain analysis and in the testbench performs a frequency domain FFT analysis to obtain PSD of the output bitstream. As shown in the Fig.6(b), the failure of the Sigma-Delta control system is captured by the distributed model while the Sigma-Delta control loop still works in the conventional model (Fig6(a)). This is because the lumped model only contains one resonant mode caused by the spring-mass-damper system and its frequency response has no relationship with the dimensions of fingers. Thus, effects of the finger dynamics cannot be captured. In other word, this performance degradation of the Sigma-Delta control loop caused by finger dynamics cannot be modeled by lumped model.

IV. Conclusion

This paper proposes a syntax extension to SystemC-A to provide support for PDE modelling. This is the first full implementation of PDE support in SystemC-A where no preprocessor is required for conversion of user defined PDEs to a series of ODAEs. The proposed PDE extension has particular advantages in modelling of mixed physical-domain systems, especially systems with mechanical parts which frequently exhibit distributed behaviour. Typical systems of this kind are MEMS sensors or kinetic energy harvesters. The distributed effects present in such systems usually can not be neglected, may even play vital roles and be essential to predicting correctly the system's performance. The efficiency of the new syntax has been verified by its application to an electromechanical Sigma-Delta accelerometer with distributed mechanical sensing element where the finger dynamics are modeled using a PDE. The well-known failure of the Sigma-Delta control system caused by the sense finger dynamics is correctly captured by the extended SystemC-A model while the conventional lumped model fails to reflect the true behaviour.

References

[1] H. Al-Junaid and T. Kazmierski. Analogue and mixed-signal extension to systemc. *IEE proc.-Circuit Devices Syst.*, 152(6):682–690, Dec. 2005.

[2] 'Draft Standard SystemC AMS Extensions Language Reference Manual'. Technical report, Open SystemC Initiative (OSCI), 3 Dec. 2008.

[3] 'IEEE Standard VHDL Analog and Mixed-Signal Extensions-Packages for Multiple Energy Domain Support'. Technical report, Design Automation Standards Committee of the IEEE Computer Society, 2005. ISBN: 0-7381-4645-5.

[4] Pecheux F., Lallement C., and Vachoux A. 'VHDL-AMS and Verilog-AMS as alternative hardware description languages for efficient modeling of multidiscipline systems'. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, 24(2):pp.204–225, 2005.

[5] Hessa Al-Junaid, Tom Kazmierski, Peter R. Wilson, and Jerzy Baranowski. Timeless discretization of magnetization slope in the modeling of ferromagnetic hysteresis. *EEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25:2757 – 2764, 2006.

[6] Seeger J.I., Xuesong J., Kraft M., and Boser B.E. 'Sense finger dynamics in a $\Sigma\Delta$ force feedback gyroscope'. *Tech. Digest of Solid State Sensor and Actuator Workshop, Hilton Head Island, USA.*, pages pp.296–299, Jun 2000.

[7] D.A. White and M. Stowell. Full-wave simulation of electromagnetic coupling effects in rf and mixed-signal ics using a time-domain finite-element method. *IEEE Trans. Microwave Theory Tech.*, 52(2):1404–1413, 2004.

[8] E. Normark P.V. Nikitin and C.J.R. Shi. Distributed electrothermal modeling in vhdl-ams. *International Workshop on Behavioral Modeling and Simulation, 2003. BMAS*, pages 128–133, 2003.

[9] X. Huang and H.A. Mantooth. Event-driven electrothermal modelng of mixed-signal circuits. *International Workshop on Behavioral Modeling and Simulation, (BMAS)*, pages 10–15, 2000.

[10] L. Wang, C.Zhao, and T.J.Kazmierski. An extension to vhdl-ams for ams systems with partial differential equations,. *Forum on Specification & Design Languages Conference (FDL)*, 2007.

[11] C.R. Shi P.V. Nikitin and B. Wan. Modeling partial differential equations in vhdl-ams,. *In Systems-on- Chip Conference, 2003. Proceedings. IEEE International*, pages 345–348, 2003.

[12] Dong Y., Kraft M., Gollasch C., and Redman-White W. 'A high-performance accelerometer with a fifth-order Sigma-Delta modulator'. *Journal of Micromechanics and Microengineering*, 2:S22–S29, 2005.