Automated Symbolic Model Reduction for Mechatronical Systems

Jochen Broz, Christoph Clauss, Thomas Halfmann, Patrick Lang, Roland Martin, Peter Schwarz

Abstract—A new modelling approach for mechatronical systems using symbolic methods based on the EDA tool Analog Insydes is presented. The netlist-based modelling language has been extended for the handling of multi-domain and vector-type through and across variables. Additionally, a library of symbolic models for a basic set of mechanical components has been implemented. With this approach, an automated setup of symbolic model equations in terms of a differential-algebraic system of equations starting from a netlist description is possible. This allows the application of DAE solvers for numerical simulation as well as the application of symbolic model reduction methods of mechatronical systems.

I. INTRODUCTION

With the decreasing structural size going along with expanding complexity of technical systems there is an emerging demand for new design methods and modelling support. This becomes even more important because of the increasing heterogeneity of technical systems. In particular, for the design of mechatronical systems this leads to the following problem: There are established tools for the design of the mechanical or electronic parts like FEM, multi-body, or circuit simulators, but those are usually specialized on their physical domain. Therefore, the consideration of interactions between the mechanical and electronic components is extraordinary laborious. These interactions often have to be taken into account because the assembly of independently optimized sub-components usually does not lead to an optimal system. On the other side, considering coupling effects results in new challenges in many aspects, which lead from the need of designers competence in multiple disciplines (electronic and mechanical engineering, physics and mathematics) up to the high complexity of the mathematical models which demand the employment of adapted simulation tools.

Such a software has to be capable of dealing with the multi-physics aspects of mechatronical systems. There are several suitable modelling languages like VHDL-AMS [1] or Modelica [2] and corresponding simulators like AdvanceMSTM or DymolaTM are available. They allow a modularized modelling of the complete system or parts of it with arbitrary accuracy. But the modelling process of heterogeneous systems is very time consuming and moreover the resulting mathematical models become very complex even for comparatively small systems, posing numerical problems with respect to robustness, efficiency, and stability.

Today the simulation of industrial-sized systems lies beyond the limits of this approach. In order to reduce the numerical effort, model reduction techniques become of interest.

In this paper, we present a modelling approach which is based on symbolic methods and can be adapted to multi-physics systems due to its general mathematical principle. This includes an automatic generation of behavioral models as well as model reduction for mechatronical components. Such models allow due to their reduced complexity an interactive processing and a more efficient simulation of the overall system. This may even enable the application of optimization and control methods.

The basic principles of this modelling approach, the basic idea of model reduction, and the tool which has been used, are described in II. In order to apply these methods and tools to mechatronical systems, symbolic device models of mechanical components are needed. Such models are described in III. In IV the new approach is demonstrated on some example applications. Finally, in V a summary and an outlook to future work is given.

II. SYMBOLIC MODELLING APPROACH

A. Background

The symbolic modelling principle originates from the field of analog circuit design where the motivation has been to gain a deeper circuit understanding by interpretation of analytic formulas. In this context dedicated techniques for linear as well as for nonlinear applications have been developed [3]. Starting from a so-called netlist description – the topological representation of an analog circuit, which includes information about the connection graph of the circuit components and corresponding device models and parameters – it is possible to formulate a mathematical equation system which in general is a nonlinear differential-algebraic equation system (DAE system). The equations consist of Kirchhoff’s current and voltage laws as well as the circuit element characteristics given by the corresponding current-voltage relations. The equation system can be set up automatically using standard formulation techniques, e.g. Modified Nodal Analysis (MNA) or Sparse Tableau Analysis (STA). The decisive property of such an equation system is that it can be analytically parameterized in the system parameters, e.g. using the resistor value $R_1$ instead of the numerical value 10Ω which motivated the term symbolic.

Due to the symbolic formulation, the equation system is valid, not only for one dedicated parameter set, but for a complete class of models with arbitrary parameter values. Using computer-algebra methods it is possible to analytically investigate the behavior of the corresponding system.
B. Analog Insydes

The above described symbolic methods are integrated in the software package Analog Insydes ([4], www.analog-insydes.de), which is an add-on to the computer-algebra system Mathematica [5]. The tool includes functionality for analysis, modelling, and optimization of linear and nonlinear circuits of industrial size.

Analog Insydes is based on a hierarchical netlist-description language which allows to automatically set up symbolic circuit equations. Besides standard electrical engineering analysis and visualization methods, dedicated model reduction methods (see II-D) are available within Analog Insydes. Moreover, the tool has been integrated into industrial design frameworks and thus includes interface functionality for exchanging data with commercial circuit simulators like Eldo™, PSpice™, Saber™, or Spectre™.

C. Behavioral device models

Due to its origin Analog Insydes comes with a pre-defined device model library for analog electronic components only. For the modelling of mechatronical systems a device model library containing corresponding mechanical components is required. In III the implemented devices and their physical properties are described in detail.

All device model implementations make use of the standard Analog Insydes modelling language in terms of a behavioral model description. This approach allows for modelling nearly arbitrary element characteristics by directly specifying the corresponding device equations which in general may be a nonlinear DAE system. The Analog Insydes model definition is based on a port branch concept which is illustrated in Fig. 1 considering a nonlinear junction diode of the electronic domain. The diode has the two terminals anode and cathode denoted by the identifiers \( A \) and \( C \), respectively. The current-voltage relation is given by the following device equation for the branch voltage \( V_D \) and the branch current \( I_D \)

\[
I_D = I_S \cdot (e^{\frac{V_D}{V_t}} - 1) ,
\]

where \( I_S \) and \( V_t \) denote the saturation current and the thermal voltage, respectively.

The ports of different components are interconnected at network nodes. Behavioral models are defined in terms of port branches, where each unique pair of port identifiers \((port_1, port_2)\) introduces a port branch between the model ports \( port_1 \) and \( port_2 \) with a positive reference direction from \( port_1 \) to \( port_2 \). The associated port variables in the behavioral model equations can be referred by means of special keywords, which are \( \text{Voltage}[port_1, port_2] \) and \( \text{Current}[port_1, port_2] \) for the electronic domain. For the diode example the Analog Insydes format for the device equation (1) reads as

\[
\text{Current}[A, C] = I_S \cdot (e^{\frac{\text{Voltage}[A, C]}{V_t}} - 1) .
\]

Note that an alternative concept to the port branch concept is the currents into the ports and the port voltages approach. But this method is not well suited for other analysis methods than MNA. Due to the fact that Analog Insydes supports different analysis methods the port branch concept has been implemented.

D. Symbolic model reduction

When setting up symbolic equations for electronic circuits, one encounters a problem: Even for comparatively small systems, the resulting solutions are of such a high complexity, that their application for automated model generation or even system understanding is impossible. Thus, in order to reduce the complexity of the symbolic expression, it has to be simplified.

In general, the term symbolic simplification or symbolic approximation refers to a whole family of hybrid symbolic/numeric algorithms for expression simplification. These techniques require more quantified information about the investigated circuit than manual simplifications but yield compact expressions with predictable error in a fully automated way. In manual circuit analysis the decisions on which expressions to keep and which ones to discard are based on vague and only qualitative assumptions (e.g. \( R_1 \ll R_2 \)) that do not allow for assigning precise error figures to simplified expressions. For automating the designer’s behavior within a computer program one needs exact figures to simplify an expression because qualitative relations between elements are not sufficient for determining the importance of a term especially if the expression to be simplified consists of non-trivial combinations of symbols.

The basic idea behind the simplification algorithms – in both the linear and nonlinear case – can be outlined as follows [6]: starting with a symbolic equation system \( F \) describing the system behavior, the user chooses one or more numerical reference solutions \( f_i \) as well as an error bound \( \varepsilon \). The algorithms then apply symbolic simplifications to the system (e.g. the deletion of an entire expression in a sum) and solve this simplified system numerically. The hereby obtained solutions \( \tilde{f}_i \) are compared to the reference solutions using an appropriate error norm: \( \delta_i = \| \tilde{f}_i - f_i \| \). If the error bound is exceeded, i.e. \( \min \delta_i > \varepsilon \), the simplification is undone. This is repeated until no more simplifications are possible without a violation of the error bound and the simplified symbolic system \( \tilde{F} \) is returned.

The order in which to simplify terms from the equation system is one of the crucial points: It is quite clear that those terms should be simplified first which have only a minor influence on the output behavior. Terms with a large influence should not be removed at all. To achieve a maximum number

\[
\begin{align*}
U_A & \quad \rightarrow & \quad U_C \\
I_A & \quad \rightarrow & \quad I_C \\
\text{Voltage}[A,C] & \quad \rightarrow & \quad \text{Current}[A,C]
\end{align*}
\]

Fig. 1. Diode quantities and branch definition
of simplifications and to avoid unnecessary modifications an optimized order, the so called ranking, is used. For this, a ranking algorithm is needed which predicts the influence on the output a modification would cause. As the number of possible simplifications is very large it is inconvenient to exactly compute the influence and therefore estimation methods have to be used. The design of a good ranking algorithm is a trade off between accurate error prediction and computational effort.

The simplification algorithms assure that the numerical behavior (with respect to the chosen references \( f_i \)) of the simplified system coincides with that of the original system within the user-given error bound. Depending on the analysis task, the reference solutions \( f_i \) can for example be a numerical transfer function, its poles and zeros, or a time-dependent solution.

Basically, Analog Insydes supports two types of simplification methods ([3], [7]): Simplification before generation methods (SBG) simplify the system equations before solving the system for certain output variables. Simplification after generation methods (SAG), which usually are applied in the linear case, simplify the solution of the equation system directly.

E. Design flow integration

Following, we describe the design flow for employing the herein introduced symbolic modelling approach for mechatronical systems which is illustrated in Fig. 2. Starting with the equation system for the mechatronical network, three cases of further applications exist. The overall intention is to generate behavioral models suited for a couple of different simulator languages.

- **Numerical processing** (center section of Fig. 2):
  The system matrices are extracted. They are necessary for generating the behavioral model of the system. Moreover, they can be used in unreduced form which means that the order of the system remains unchanged.

But this approach is suitable for small systems only. On the other hand the system matrices can be reduced, applying appropriate order reduction techniques, for instance ENOR ([8], [9]). Both alternatives are already realized in the modelling flow.

- **Fully symbolic model reduction** (left section of Fig. 2): Analog Insydes is especially designed for this application. The reduced equation system generated with this method consists of a set of equations containing all dominant system parameters symbolically and can be customized for further symbolic computations in Analog Insydes.

- **Mixed numerical/symbolic model reduction** (right section of Fig. 2):
  This application is also realized within Analog Insydes. The resulting model depends only on a few selected controllable parameters, that influence the system behavior. This is the most important feature of this kind of model reduction and is especially suited for optimization purposes.

III. SYMBOLIC DEVICE MODELS OF MECHANICAL COMPONENTS

Mechanics is an important application area of the symbolic modelling approach. Elastic strongly linked (micro-) mechanical components are considered where force effects are restricted to cause only small displacements. Moreover, deformations like transversal contractions are out of scope.

A. Modelling methodology

Mechanical components are decomposed into finite elements. Having the analytical description for a set of basic elements, one can automatically generate the equations modelling the system behavior by exploiting the network-like composition of interconnected elements. This method which is well established in MEMS simulation ([10], [11]) is also the starting point for the symbolic modelling approach.

A library of basic mechanical components is defined, which consists at present of a beam element, and fixed/nonfixed mechanical sources including an anchor element which is a special kind of source. Additionally, each element can be parameterized.

B. Connections and frame of reference

The basic elements are connected at their mechanical ports. The connections specify the interaction of the elements. Due to spacial modelling twelve variables exist at each mechanical port. These variables can be divided into translatory and rotatory ones according to Table I.

<table>
<thead>
<tr>
<th>port</th>
<th>across variables</th>
<th>through variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>translatory</td>
<td>( u_x, u_y, u_z )</td>
<td>( F_x, F_y, F_z )</td>
</tr>
<tr>
<td>rotatory</td>
<td>( \phi_x, \phi_y, \phi_z )</td>
<td>( M_x, M_y, M_z )</td>
</tr>
</tbody>
</table>
\(x, y, z\) denote the spacial coordinates, \(F_i\) the forces, \(u_i\) the displacements, \(M_i\) the torques, and \(\varphi_i\) the rotation angles according to the direction of the coordinates \(i\).

Because of the continuity of displacement at the connections of mechanical elements all corresponding across variables are equal to each other, and due to d’Alembert’s principle the through variables are summed up to be zero. This corresponds to the Kirchhoff laws which apply for all components. A local coordinate system is defined for each element (e.g. beam, Fig. 3) in its quiescent state. Using a matrix of transformation and rotation, which depends on geometrical parameters of the element, the coordinates of the global and the local system can be converted into each other [12].

\[
\begin{align*}
    u_{x1} &= u_{x2}, \quad u_{y1} = u_{y2}, \quad u_{z1} = u_{z2} \\
    F_{x1} + F_{x2} &= 0, \quad F_{y1} + F_{y2} &= 0, \quad F_{z1} + F_{z2} &= 0 \\
    \varphi_{x1} &= \varphi_{x2}, \quad \varphi_{y1} = \varphi_{y2}, \quad \varphi_{z1} = \varphi_{z2} \\
    M_{x1} + M_{x2} &= 0, \quad M_{y1} + M_{y2} &= 0, \quad M_{z1} + M_{z2} &= 0.
\end{align*}
\]

In general, one has to distinguish between the global coordinate system and the local coordinate system. The global coordinate system is the fixed spacial coordinate system which is valid for all components. A local coordinate system is defined for each element (e.g. beam, Fig. 3) in its quiescent state. Using a matrix of transformation and rotation, which depends on geometrical parameters of the element, the coordinates of the global and the local system can be converted into each other [12].

### C. Extended netlist-description language

For the modelling of mechanical systems a couple of extensions to the Analog Insydes netlist-description language had to be considered. When applying the Analog Insydes port branch concept introduced in II-C to a mechanical device, one observes that already for a simple two-port component one already obtains six corresponding model branches. This is illustrated in Fig. 3 for the beam element. In comparison to a two-port electronical component where one has one current quantity (through variable) and one voltage quantity (across variable) within the branch, the mechanical component has six through variables (three force quantities \(F_i\) and three torque quantities \(M_i\)) and six across variables (three displacement quantities \(u_i\) and three rotation angle quantities \(\varphi_i\)). Therefore, the so far scalar-type Analog Insydes ports have been extended to the more general case of vector-type ports. Within a netlist description this is achieved by adding a NetlistAttributes section. This new language object is valid for the whole netlist object within it has been defined. One simply has to specify the dimension of a corresponding netlist node using the NodeDimensions keyword. The syntax is as follows

```latex
\begin{verbatim}
NetlistAttributes[
    NodeDimensions -> {<node1> -> 6, ...
    ...},
    NodePositions -> {<node1> -> {<x1>, <y1>, <z1>}, ...
\}
\end{verbatim}
```

where `<node1>` denotes the name of the mechanical node and `{<x1>, <y1>, <z1>}` is the corresponding numerical coordinate vector, respectively. The coordinate information specified by the NodePositions keyword is useful for automatically computing the geometrical parameters of the mechanical component. Note that with the Default -> 1 setting one can simply define all remaining nodes in the netlist description as being scalar type without stating them explicitly which is very useful when having multi-physics applications. An example netlist description is illustrated in Fig. 5.

### D. Beam element equations

One of the basic elements of the mechanical component library is the beam element (see Fig. 3) which can be used to compose varied beam constructions. The beam element has two ports denoted by the indices 1 and 2, and the parameters width (\(w\)), height (\(h\)), density (\(\rho\)), modulus of elasticity (\(E\)), shear modulus (\(G\)), and damping (\(d\)). Using the coordinates \(x_1, y_1, z_1\) and \(x_2, y_2, z_2\) of the beam ends, the length (\(l\)) can be calculated.

In the local coordinate system the beam equations [11] are

\[
\mathbf{F} = -(\mathbf{M} \dot{\mathbf{u}} + \mathbf{D} \mathbf{u} + \mathbf{K} \mathbf{u})
\]

where

\[
\mathbf{u} = \begin{bmatrix} u_{x1} & u_{x2} & \varphi_{x1} & \varphi_{x2} & u_{y1} & u_{y2} & \varphi_{y1} & \varphi_{y2} & u_{z1} & u_{z2} & \varphi_{z1} & \varphi_{z2} \end{bmatrix}^T
\]

\[
\mathbf{F} = \begin{bmatrix} F_{x1} & F_{x2} & M_{x1} & M_{x2} & M_{y1} & M_{y2} & M_{z1} & M_{z2} \end{bmatrix}^T
\]
M is the mass matrix

\[
M = \rho \cdot V \cdot \begin{bmatrix}
m_1 & 0 & 0 & 0 & 0 & m_2 & 0 & 0 & 0 & 0 \\
0 & m_1 & 0 & m_3 & 0 & m_4 & 0 & m_5 & 0 & m_6 \\
0 & 0 & m_1 & 0 & m_3 & 0 & m_4 & 0 & m_5 & 0 \\
0 & 0 & 0 & m_1 & 0 & m_3 & 0 & m_4 & 0 & m_5 \\
0 & 0 & 0 & 0 & m_1 & 0 & m_3 & 0 & m_4 & 0 \\
0 & 0 & 0 & 0 & 0 & m_1 & 0 & m_3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & m_1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & m_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_1 
\end{bmatrix}
\]

with \( m_1 = 1/3 \), \( m_2 = 1/6 \), \( m_3 = 13/35 \), \( m_4 = 111/210 \), \( m_5 = 9/70 \), \( m_6 = 13/420 \), \( m_7 = J_p/(3wh) \), \( m_8 = J_p/(6wh) \), \( m_9 = l^2/105 \), \( m_{10} = l^2/140 \), and \( V = whl \).

K is the stiffness matrix

\[
K = \begin{bmatrix}
k_1 & 0 & 0 & 0 & 0 & -k_1 & 0 & 0 & 0 & 0 \\
0 & k_1 & 0 & 0 & 0 & 0 & -k_1 & 0 & 0 & 0 \\
0 & 0 & k_1 & 0 & 0 & k_1 & 0 & 0 & k_1 & 0 \\
0 & 0 & 0 & k_1 & 0 & 0 & k_1 & 0 & 0 & k_1 \\
0 & 0 & 0 & 0 & k_1 & 0 & 0 & k_1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & k_1 & 0 & 0 & k_1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & k_1 & 0 & 0 & k_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & k_1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k_1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k_1 
\end{bmatrix}
\]

with \( k_1 = Ewh/l \), \( k_2 = 12EJ_n/l^3 \), \( k_3 = 6EJ_n/l^2 \), \( k_4 = 12EJ_m/l^3 \), \( k_5 = 6EJ_m/l^2 \), \( k_6 = GJ_l/l \), \( k_7 = 4EJ_n/l \), \( k_8 = 2EJ_m/l \), \( k_9 = 4EJ_n/l \), and \( k_{10} = 2EJ_m/l \).

D is the damping matrix

\[
D = \text{diag}(d \ d \ d \ d \ d \ d \ d \ d \ d \ d \ d) \ .
\]

The formulas for the moments of inertia \( J_i \) are

\[
J_m = w^3h/12 \ , \ J_n = wh^3/12 \ , \ \ J_l = \frac{2(wh)^3}{7(w^2+h^2)} \ , \ J_p = J_m + J_n \ .
\]

This is the complete set of equations for a beam element. In the simpler case of a massless, non-twistable beam without damping \((G = 0, \rho = 0, \text{and } d = 0)\) the matrices M and D vanish. The equation for such a beam reduces to

\[
F = -Ku \ .
\]

This simplified beam element is also included in the library of basic mechanical components.

E. Aggregation to system model equations

Assuming a mechanical component is composed of connected beam elements, then the following steps are necessary in order to generate the model equations for this component

- transform the equations of each component from the local to the global coordinate system,
- collect all component equations,
- add equations for d’Alembert’s principle and continuity of displacement for all connections.

These steps can be performed automatically employing Analog Insydes in a symbolic or semi-symbolic way.

IV. APPLICATION EXAMPLES

In this section we demonstrate the symbolic modelling and approximation approach using an acceleration sensor. In IV-A a symbolic model reduction techniques for the dynamic behavior of a mechatronical model are shown. The example makes use of a simple model for the mechanical parts, in order to show, how mechanical and electronical degrees of freedom can be coupled within Analog Insydes.

A more detailed model for the mechanical parts is considered in IV-B. It makes use of the symbolic models for the micromechanical components as described in III. Using symbolic approximation techniques for the low-frequency behavior, an approximated symbolic equation giving the force constant as a function of material properties and geometrical parameters is derived in an automated way, showing the capabilities of the symbolic modelling approach. The combination of this detailed mechanical model and the electronical components building a mechatronical system will be subject to future work.

A. Acceleration sensor

As an example for a multi-physical system we consider an acceleration sensor [13] consisting of mechanical and electronical parts (see Fig. 4). The sensor consists of three parallel conducting plates which form two serial capacities \( C_1 \) and \( C_2 \). The central plate can be moved from its balanced position (center if \( R_A = R_B \)) resulting in a Hook’s force with constant \( K \). In case of an acceleration, the central plate moves away from its central position resulting in changes of the capacities between the electronical connectors \( E_V/E_0 \) and \( E_2/E_0 \). This yields a potential drop \( V_{cm} \) for the central plate with respect to the potential in the idle state.

The acceleration sensor has one mechanical and three electronical ports (the center of mass and each plate). The mechanical port has the vector variables displacement \( u \) and force \( F \). Besides the external accelerating force \( F \) there are internal forces acting on the system. The internal forces result from electrostatics, Hook’s law, and damping

\[
F_{int} = \frac{Q_1^2 - Q_2^2}{2Ae_0} - Kx - D\dot{x} \ .
\]
Here, $Q_1$ and $Q_2$ are the charges of the plates $E_1$ and $E_2$, $A$ is the plate area, $\epsilon_0$ the dielectric constant, and $D$ the damping constant. The force acts along the axial direction and accelerates the central plate with mass $m_{\text{dyn}}$

$$F_{\text{int}} = m_{\text{dyn}} \left( \ddot{x} + e \cdot \ddot{u} \right).$$

(16)

Here, $x$ is the local displacement of the central plate from the idle position and $u$ the global displacement of the acceleration sensor. Forces acting on the static mass $m_{\text{stat}}$ are the external force $F$ and internal force $F_{\text{int}} e$, giving the equation of motion

$$F - F_{\text{int}} e = m_{\text{stat}} \ddot{u}.$$  

(17)

The charges $Q_1$ and $Q_2$ depend on the node voltages $V_0, V_1,$ and $V_2$ at the electrical connection ports $E_0, E_1,$ and $E_2$

$$Q_1 = C_1 (V_1 - V_0)$$  

(18)

$$Q_2 = C_2 (V_2 - V_0)$$  

(19)

where the capacities depend on the plate distances (idle distance $d_0$)

$$C_1 = \epsilon \frac{A}{d_0 - x}$$  

(20)

$$C_2 = \epsilon \frac{A}{d_0 + x}.$$  

(21)

The branch currents $E_1$ to $E_0$ and $E_2$ to $E_0$ are given by

$$I_{i,0} = \frac{Q_i}{C_i}, \quad i \in \{1, 2\}.$$  

(22)

Because we use six-dimensional port variables (displacements and rotation angles), we add trivial angular equations with moment of inertia $\theta$ and torque $M$

$$\ddot{\varphi}_i = \theta M_i, \quad i \in \{1, 2, 3\}.$$  

(23)

Changes of the orientation of the sensor are not considered in this example.

Equations (15)-(23) are implemented as an Analog Insydes model called AccelerationSensor. Fig. 5 shows its usage within the netlist description for the system of Fig. 4. The acceleration sensor as is oriented in $(0, 1, 0)$ direction. Its connections are specified by $<$node$>$ $->$ $<$port$>$ mappings. The electronical ports $E0, E1,$ and $E2$ are connected at nodes $e0, e1,$ and $e2$ with the circuit. The mechanical port $M$ is connected at node $m$ to the point mass and the accelerating force. At this node a time-dependent force acts towards $(0, -1, 0)$, accelerating the system starting at $t = 10$ ms. The corresponding numerical values for all model parameters are listed in Fig. 6. All component parameters are instantiated by the postfix $<$comp$>$, where $<$comp$>$ specifies the corresponding component name as given in the netlist. For all values which are not set within the netlist, default values are used instead.

This notation is also used for the DAE system (Fig. 7), which has been generated automatically from the netlist description. It consists of 39 equations for 39 variables, which are named using a similar convention: e.g. the local displacement $x$ of the central plate of as is named x$\text{as}[t]$ and the third component of the displacement in node $m$ is denoted by u$\text{m}[t]$. Additionally, there are 16 initial conditions for the 14 mechanical variables (location and rotation, velocity and angular speed, displacement and velocity of the central plate) as well as the charges on plates $E_1$ and $E_2$.

For performing the approximation as described in II-D, $V_{\text{out}}$ has been chosen as output variable. The maximum approximation error has been set to $20 \text{ mV (20%) for these settings, the highlighted expressions have been identified to be not relevant for the dynamics of } V_{\text{out}}.$ Furthermore, automated exact algebraic manipulation finally leads to the DAE system shown in Fig. 8.

The system has been reduced to a set of five equations only. Note, that the reduction to the $y$ direction (only displacement variable u$\text{m}[t]$ is left) and removal of all rotational degrees of freedom has been done completely automatic. Additional approximations, e.g. removal of MDYN$\text{as}$ x$\text{as} \cdot t$ in the dynamics of the central plate, have been applied automatically. Finally, Fig. 9 illustrates a comparison of $V_{\text{out}}$ for the original and simplified DAE system. Starting at $t = 10 \text{ ms}$ the force accelerates the point mass. In the reduced model, the acceleration of the system only depends on the point mass (M$\text{mass}$), neglecting the mass of the sensor (MSTAT$\text{as} +$MDYN$\text{as}$) which is less than two percent of the point mass. This finally results in the slightly increased absolute value for the stationary voltage $V_{\text{out}}$. The difference in the dynamic behavior for $t$ near $10 \text{ ms}$ is mainly due to neglecting the inertia of the central plate.

![Fig. 5. Multi-physics netlist including acceleration sensor](image)

![Fig. 6. Numerical values for all system parameters (given in SI units)](image)
B. Mechanical example

The mechanical model used in the previous section is a simple model, concentrating on the basic principles of an acceleration sensor. Here, we concentrate on a more detailed model for the mechanical parts. The aim is to derive a symbolic expression, giving the force constant as a function of the material and geometric properties. Fig. 10 shows a sketch of the structure for a typical micromechanical acceleration sensor (e.g. [14]). We consider the displacement $u$ in C3 as a function of the force $F$.

With these considerations it is sufficient to consider the dark gray parts only. The corresponding model consists of 14 beam elements (three different kinds), two anchors and a force $F$ acting at node C3 in y direction. All beam elements have the same material properties ($\rho$, $E$, and $G$) and width $w$. Length and height of the two central beam elements (C3 to C4 type) are $l_B$ and $h_B$. Height and length of the other beam elements are $h_A$, $l_A$ (C1 to R1 type), and $l_A$ (R1 to R2 type). The system can be described by a netlist, which makes use of the beam element models, described in III-D.

Here, we are using the low-frequency behavior for deriving an approximated symbolic expression for the force-displacement relation in C3. Vibrational analysis is usually referred to AC analysis or small-signal analysis in the electronic domain, where the necessary functionality is already implemented within Analog Insydes, thus all the following steps are already automated:

- **Equation setup**

The full transient model of the system is created. This second order DAE system describes the dynamics of the 264 variables in the general case.
- **Algebraic simplification**
  Using variable elimination and removal of independent blocks, the system can be reduced. In this example this leads to a second order DAE system in 39 variables.

- **Laplace transformation**
  In the nonlinear case, the system is linearized wrt. an operating point. Then, a Laplace transformation is performed. In this example, the initial DAE system is already linear in the variables. We specified the force (in y direction) as harmonically exciting input parameter. This results in a linear equation system with the Laplace frequency $s$ as an additional parameter.

- **Define analysis task**
  For the original system, the transfer function $u(s)$ is computed numerically. The result is shown in Fig. 11. We are interested in the low-frequency behavior. The transfer function remains almost constant below 2 kHz. We choose a maximum error of 10% for $u$ at $s = 2\pi f$ where $f = 10$ Hz.

- **Symbolic approximation**
  Neglectable terms within the symbolic matrix of the equation system are automatically identified and removed. Independent blocks of the remaining equation system are eliminated. This leads to a system of only 10 symbolic linear equations.

- **Symbolic solution**
  This simplified system can be solved symbolically

  \[ u = \frac{2h_B^3h_A^1 + h_A^2h_B^1A_2 + 2h_A^3h_B}{4Eh_B^1h_Bw} \]  

  (24)

  The numerical validation for this approximation in comparison to the original model is shown in Fig. 11. Of course, for the high-frequency behavior the approximation is not valid, because only the low-frequency behavior has been checked in the approximation step.

V. CONCLUSIONS AND FUTURE WORK

In this paper a new modelling approach for mechatronical systems is presented, which is based on symbolic methods. Such methods have already been successfully applied to electronic systems and corresponding modelling tools are available. In order to transfer the methodology to mechatronical systems, especially the multi-domain aspects, i.e. vector-valued variables and the need for adapted behavioral models for basic mechanical components have been considered. The capabilities of this new approach have been demonstrated by means of models for acceleration sensors. Symbolic model reduction techniques have been employed for the modelling of the time-dependent as well as for the frequency behavior of mechatronical systems.

Since these initial investigations show already promising results, future work could include the following aspects:

- extension of the mechanics library (e.g. plate elements),
- extension of the described methods to other physical domains (e.g. magnetics, electrostatics) in order to handle strongly coupled multi-domain systems,
- investigations on the limits of the method concerning both the number of degrees of freedom and the number of symbolic parameters,
- adaption of the behavioral model flow with respect to the generation of symbolic behavioral models,
- application of semi-symbolic models for optimization purposes within simulation environments.

VI. ACKNOWLEDGMENTS

The authors would like to thank the Fraunhofer society for the financial support within the MEF project 662410 (Modellierungsunterstüzung für den Entwurf komplexer mechatronischer Systeme mittels symbolischer Reduktionsverfahren).

REFERENCES


Fig. 11. Comparison of original (solid) and simplified model (dashed)