Addressing the Smart Systems Design Challenge: The SMAC Platform

Nicola Bombieri
EDALAB
Verona, ITALY
nicola.bombieri@edalab.it

Dimitris Drogoudis
Agilent Technologies
Gent, Belgium
dimitris_drogoudis@agilent.com

Giuliana Gangemi
STMicroelectronics,
Catania, ITALY
giuliana.gangemi@st.com

Renaud Gillon
ON Semiconductor
Oudenaarde, Belgium
Renaud.Gillon@onsemi.com

Michelangelo Grosso
ST-POLITO s.c.a.r.l.
Torino, ITALY
michelangelo.grosso@st-polito.com

Enrico Macii
Politecnico di Torino
Torino, ITALY
enrico.macii@polito.it

Massimo Poncino[1]
Politecnico di Torino
Torino, ITALY
massimo.poncino@polito.it

Salvatore Rinaudo
STMicroelectronics
Catania, ITALY
salvatore.rinaudo@st.com

Abstract

This article presents the concepts, the organization, and the preliminary application results of SMAC, a smart systems co-design platform. The SMAC platform, which has been developed as Integrated Project (IP) of the 7th ICT Call under the Objective 3.2 "Smart components and Smart Systems integration" addresses the challenges of the integration of heterogeneous and conflicting domains that emerge in the design of smart systems. SMAC includes methodologies and EDA tools enabling multi-disciplinary and multi-scale modelling and design, simulation of multi-domain systems, subsystems and components at different levels of abstraction, system integration and exploration for optimization of functional and non-functional metrics. The article presents the preliminary results obtained by adopting the SMAC platform for the design of a limb tracking smart system.

Keywords—smart systems, co-design, co-simulation, CAD, EDA, analog design, digital design, MEMS, power devices.

---

1. Introduction

The European Technology Platform on Smart Systems Integration (EPoSS) Strategic Research Agenda (SRA) defines Smart Systems as *"intelligent, often miniaturised, technical subsystems with their own and independent functionality evolving from microsystems technology"* [1].

This definition encompasses then a broad class of devices that incorporate functionalities like sensing, actuation, and control, that are usually energy-autonomous and ubiquitously connected. In order to support these functions, they must include sophisticated and heterogeneous components and subsystems such as application-specific sensors and actuators, multiple power sources and/or storage devices, intelligence in the form of power management, baseband computation, digital signal processing, power actuators, and subsystems for various types of wireless connectivity, as conceptually depicted in Figure 1.

---

[1] Contact Author: Massimo Poncino, Dipartimento di Automatica e Informatica, Politecnico di Torino, 10129 Torino, ITALY, massimo.poncino@polito.it.

It is evident from this heterogeneity that Smart Systems leverage a variety of different technologies and different materials. As a matter of fact, using the ITRS (International Technology Roadmap for Semiconductor) terminology, Smart Systems can be regarded as the bridge between the two orthogonal dimensions that describe technology evolution: "More Moore" and "More than Moore". The former dimension spans the traditional CMOS scaling for digital devices, whereas the latter one addresses the issue of evolution via "diversification", and refers to devices whose functionalities do not necessarily scale according to Moore's Law but aim at providing additional value in different ways (Figure 2). The figure exemplifies how merging the capability of maintaining the scaling trend for digital logic and the increased opportunity of diversifying functionality will drive to integrated multiple functionalities on the same silicon support, migrating from system board-level into a System-in-Package (SiP) or System-on-Chip (SoC); in a word, a Smart System.

The challenge in the implementation of Smart Systems goes therefore beyond the design of their individual components (an already difficult task by itself), and rather lies in the co-existence of a multitude of functionalities, technologies, and materials. The widely acknowledged keyword in Smart Systems design is in fact *integration*. There are essentially two dimensions of integration that represent the main obstacle towards mainstream design of Smart Systems: Technological and methodological. As already experienced in specific domains (e.g., in digital and analog design), a solution has been found first for the technological issues. Advanced packaging technologies such as System-in-Package (SiP) and chip stacking (3D IC) with through-silicon vias (TSVs) allow today manufacturers to package all this functionality more densely, combining the various domains depicted in the figure above in a single package. SiP technology works nicely because it allows merging components and subsystems with different processes, and mixed technologies using the state-of-the-art advanced IC packaging technologies with minor impact on the design flow. Therefore, to some extent, technological solutions aimed towards integration are available.

Design methodologies, however, are falling behind: Current design approaches for Smart Systems use separate design tools and ad-hoc methods for transferring the non-digital domain to that of IC design and verification tools, which are more consolidated and fully automated. This solution is clearly sub-optimal and cannot respond to challenges such as time-to-market and request of advanced sensing functionalities. A big step towards effective large-scale design of Smart Systems would be that of changing their design process from an expert methodology to a mainstream (i.e., automated, integrated, reliable, and repeatable) design methodology, so that design costs are reduced, time-to-market is shortened, design of the various domains is no longer confined to teams of specialists inside IDMs and system miniaturization can be achieved with limited risks.

This objective can be reached by defining and implementing a structured design approach that explicitly accounts for integration as a specific constraint, which in the context of the SMAC project consists of a flexible software platform (i.e., the SMAC platform) that includes methodologies and EDA tools enabling multi-disciplinary and multi-scale modelling and design, simulation of multi-domain systems, subsystems and components at all levels of abstraction, system integration and exploration for optimization of functional and non-functional metrics. The key elements of the SMAC platform are:

1. The development of a co-simulation and co-design environment that accounts for the peculiarities of the basic subsystems and components to be integrated;
2. The development of modelling and design techniques, methods and tools that, when added to the platform, will enable multi-domain simulation and optimization at various levels of abstraction and across different technological domains.

SMAC aims at achieving this ambitious objective through a holistic co-design framework, which requires closing several technical and cultural gaps by means of a multidisciplinary approach. In order to do this, the project has required the joint co-operation of research and industry partners, including EDA vendors to ensure the platform usability in realistic, industry-strength design flows and environments, with a direct impact on the industrial exploitation.

Behind the growing interest in Smart Systems, there is a potentially huge and quickly growing market, which is expected to grow in the order of \$200B in 2020 [1], inducing an even larger market of non-hardware services involving all the various devices envisioned in the Internet of Things. Such a market is much larger than those of smart- or feature-phones in terms of number of devices. Over 50 billion devices will be connected to the Internet according to Cisco forecasts, and most of these devices will be Smart Systems. Miniaturized Smart Systems find applications in a broader range of key strategic sectors, including automotive, healthcare, ICT, safety & security, and aerospace.

Also, efficient energy management and environment protection are business sectors in which the utilization of miniaturized Smart Systems may make a difference. The worldwide market for "Monitoring & Control" products and solutions, one of the most important fields of Smart Systems applications, containing solutions for environment, critical infrastructures, manufacturing and process industry, buildings and homes, household appliances, vehicles, logistics & transport or power grids, is around 188B Euro. This value represents 8% of the total ICT expenditures worldwide, and it is identical to the whole semiconductor industry world revenues and approximately twice that of the world mobile phone manufacturers revenues [2].

This article is organized as follows. Section 2 reviews the state of the art in the design and simulation of smart systems. Section 3 presents technical details about the SMAC platform, whose application to a real-life case study from the consumer electronics sector is demonstrated in Section 4. Finally, Section 5 presents a few concluding remarks and an outlook on other activities related to the project's topics and that could require further investigation.

2. State of the art in Smart System Co-Design

Modelling and design capabilities for heterogeneous components and subsystems are today available at specialized design houses and silicon makers in various forms. The *non-electrical parts* (involving micromechanical structures, electromagnetic fields, thermal phenomena, wave propagation, etc.) are designed using Partial Differential Equations (PDE) solvers, like the Finite Element Method (FEM), or, alternatively, schematic-based behavioural libraries. The *analog and RF parts* are designed resorting to the reuse of existing macros by experienced engineers, following a template-based approach. The *digital parts* are designed using highly automated synthesis tools (from high-level synthesis to physical synthesis) following a top-down paradigm. *System design* is supported by block diagram simulation (e.g., MATLAB-SIMULINK) or newer approaches, which allow obtaining a comprehensive view of the entire system, yet through simple models of the subsystems and of the components. Finally, the amount of *software,* implemented in microcontrollers and DSPs, is significantly increasing, in line with the trend that can be observed in embedded-system design.

A clear framework to categorize methodologies and tools, which are adopted in Smart Systems design, can be provided by distinguishing between the different levels of design abstraction: *System level, device level* and *physical level*.

The interactions of the heterogeneous components and subsystems such as MEMS, RF, analog parts, power sources, digital macro-cells, and software with their environment and control electronics can be modelled and simulated at the system level of abstraction. In this context, SystemC [1] is de de-facto reference standard modelling language for system level design of digital systems while Transaction-level Modelling (TLM) [4] is the key paradigm for modelling such digital systems at a high level of abstraction. Because more and more integrated systems have analog components that are tightly coupled with the digital hardware and software, SystemC was extended to AMS modelling (SystemC-AMS). SystemC/SystemC-AMS introduces numerous methodologies for efficient modelling and simulation of different domains.

Circuit simulation programs, such as SPICE and derivatives, are used for device or circuit level design. A netlist describes the circuit elements (e.g., transistors, resistors, capacitors) and their connections, and translates such description into non-linear differential equations to be solved using implicit integration methods, Newton's method and sparse matrix techniques. MEMS libraries for circuit simulators based on parameterized behavioural models and consisting of electromechanical building blocks, such as beams, plates, comb structures, and electrode models [5],[6],[7],[8] are commercially available and are now well established [9]. Recently, 3D visualization of the complex models and simulation results has been added [10]. Latest advancement allows designers to compose MEMS behavioural model based designs in 3D and pass the model as netlist to an EDA platform for IC design [11].

At physical level, power sources, sensors, MEMS and other discrete devices have traditionally been modelled in 3D CAD environments and simulated using 3D field solvers based on the finite element method (FEM) and boundary element method (BEM). Today, a variety of single-physics and multi-physics field solvers is available, ranging from general-purpose tools to those that address MEMS-specific physics such as electrostatic sensing and actuation, piezo-electric effects, and gas damping [11],[12],[13]. In the recent past, substantial improvements have been made in user-friendliness; automatic meshing algorithms, computational efficiency and results database management to link physical models with system level models [14]. Model Order Reduction (MOR), also called Reduced Order Modelling (ROM) [15],[16], is an evolving approach for a fast generation of behavioural models based on existing FEM component models, e.g., generated in commercial FEM solvers. There are different basic approaches for MOR, such like projection-based methods, simulation-based order reduction.

Some attempts have been recently done to combine different domains into a single co-simulation environment. [34] addresses the problem of conflicting constraints in embedded systems, such as energy and time. It proposes a mechanism for supporting design decisions on energy consumption and performance of embedded system applications. The authors show that the estimates obtained through the conceived model are 93% close to the respective measures obtained from the real hardware platform.

[36] presents an HDL design methodology for multistandard RF SoC's, which covers all the design layers from system design, to automatic extraction of the models from circuits and a systematic top level verification. System or block level verification is obtained with models automatically by overnight runs, without the need for extra test benches or designer interaction. This enables short term detection of functional errors or performance losses. The accuracy of the system level simulations show a very good match with the measurement results after fabrication.

Multi-domain co-simulation has been investigated in the context of smart grids [36]. The data acquisition, protection and control of smart grid highly depend on advanced information infrastructures, which makes smart grid a coupled system of power and information networks. In order to assess the influence of the uncertainties within information networks on the performance of the real-time controls in smart grid, [36] presents a co-simulation method based on OpenDSS and OPNET simulators.

Finally, [37] proposes a global co-simulation methodology for concurrent/simultaneous analysis of passive and active analog/digital parts. An original power-signature concept is introduced to model high-speed digital modules temporal and spatial distribution of their power switching activity through specified chip partitions. Dedicated real-world NXP-Philips-Semiconductors active modules mounted on testboard have been designed and measured for validation of the proposed co-simulation methodology. In the paper, full-wave electromagnetic modelling, broadband SPICE compact model extractions and measurement results are successfully compared.

In all these approaches, Smart Systems co-design suffers from multiple and different types of limitations and bottlenecks. The design of a Smart System that deals with embedded systems generated from bare die requires a strong link between different worlds of design tools, such as the board/module level, the electronic circuit design level, and the physical level. The tight integration of such differing systems sets high requirements on the system design flow. While FEM (Finite-Element-Method) simulation allows analysing detailed problems, models for higher abstraction levels are needed to exceed the complexity barrier imposed by the need of dealing with electrical, electro-mechanical and thermal interfaces. Also the reuse of available components usually requires the creation of ad-hoc functional models based on datasheets or device characterization.

The design and simulation frameworks for component/subsystem integration created by system integrators are non-standard and show several limitations:

- *Very fragmented* (stitching of bits and pieces of existing EDA tools and flows from different vendors).
- *Very fragile* (scripting, tool interfacing, data formatting).
- *Expensive* (licensing from different vendors).
- *Not flexible* (changes in system architecture, building blocks, models and workloads often imply significant revision of the framework).
- *Customized to specific applications* (application domains).

The main issues and research objectives for a multi-level design methodology for Smart Systems include several aspects. Some examples are: Coupling of physical effects towards multi-domain modelling approaches, consideration of non-linear effects and structural discontinuities supported by efficient modelling and simulation algorithms, co-design and co-simulation of electronics (e.g., analog, digital) with non-electronics (e.g., MEMS, power sources) over multiple scales, and so on. In particular, co-simulation is required to verify the IC design and to predict yield sensitivity to manufacturing variations. One obvious path is to do the co-simulation in the environment used by the IC designers. This requires that the MEMS designers deliver a behavioural model of the MEMS devices expressed in a suitable hardware description language (HDL) such as Verilog-A or VHDL-AMS. Today, MEMS engineers have very limited ability to deliver behavioural models in these formats. Very often they build the model manually, usually in the form of a look-up table, or generate a Reduced-Order Model (ROM) from finite element analysis.

The lack of IC-compatible parametric MEMS behavioural models is an impediment to design reuse and to the licensing of MEMS IP. The availability of a library of validated MEMS IP design in the environment that an IC designer or system architect is used to work in, would revolutionize the way smart based systems are developed today.

There is very limited ability to deliver MEMS component models for architectural level design. System engineers are usually required to hand craft non-parametric models or look-up tables from publicly available data sheets. System-level tools such as Simulink are often used for functional modelling and simulation. They may also capture continuous-time behaviour, but do not target the design of Embedded-AMS (E-AMS) systems at an architecture-level. HDLs target the design of mixed-signal subsystems close to implementation level, but these languages have limited capabilities to provide efficient HW/SW co-design at high level of abstraction. Existing co-simulation solutions mixing SystemC and Verilog/VHDL-AMS do not provide high enough simulation performances and lack offering a seamless design refinement flow for modelling mixed discrete-event/continuous-time systems and HW/SW systems at architectural level. In conclusion component models are not yet integration-aware.

3. The SMAC Platform

The SMAC platform consists of innovative design methods and tools for next generation's Smart Systems. More specifically, SMAC aims at enabling multi-physics, multi-layer, multi-scale and multi-domain full Smart System simulation and optimization, by providing modelling and design techniques, methods and tools that allow simulation and optimization at various levels of abstraction and across different technology domains. This is key for enabling the design of complex Smart Systems using a top-down approach from system architecture down to implementation and integration of heterogeneous functions and technologies.

3.1. General overview of the platform

Figure 3 shows a conceptual organization and overall usage scenario of the SMAC Platform. Although the requirements and the specifications are independent on any particular tool used, Keysight SystemVue [18] plays an important role in the project. The SystemVue framework is aimed at system-level design and simulation. It was developed to enable system architects and algorithm developers to innovate the physical layer (PHY) of wireless and aerospace/defense communications systems and provides unique value to RF, DSP, and FPGA/ASIC implementers. As a dedicated platform for system level design and signal processing realization, SystemVue replaces general-purpose digital, analog, and math environments. SystemVue "speaks RF", helps reducing PHY development and verification time, and connects to any mainstream EDA flow and, thus, it is an excellent starting point to develop the multi-domain system co-simulation flow.

The dashed box in the figure shows the baseline SystemVue framework, consisting of built-in models, the actual SystemVue tool and its interface to SystemC. Blocks in black (i.e., Component design, Model Abstraction/Refinement, and Abstract tool-independent interface generation) denote the extension to the baseline SystemVue co-simulation environment and represent the specific contributions towards the SMAC platform.

3.1.1. *Component Design*

This block represents the design activity of subsystems and components belonging to the five fundamental domains involved in typical Smart Systems (digital, analog and RF, discrete and power components, MEMS and sensors, and power sources). The component/subsystem design is interfaced with the existing models, as shown by the bidirectional arrow to/from the models. The models can be updated as a result of the design of the various classes of subsystems and components. The design of these components is strongly integration-aware (see the arrows to and from the co-simulation platform labelled "*Integration*"), in the sense that the type and the interaction among components represent additional constraints for the design of individual components. This activity leverages existing or newly developed models that are relevant for some of the domains, such as EM coupling, thermal and multi-physics coupling models (not shown in the figure) and/or the encompassing of new metrics even in more standardized design flows as the digital one, such as temperature, variability, noise, and aging.

3.1.2. *Model Abstraction/Refinement*

When we step into the actual co-simulation platform, a basic addition to the original SystemVue is the *Model Abstraction/ Refinement block*, which supports the generation of models at different levels of abstraction. This block can select one existing model of a component at a given abstraction level (*level A*) and generate a more abstract or more concrete model (*level B*) depending on the desired implementation detail.

As an example, SMAC provides a set of tools to automatically abstract RTL models of digital components up to TLM levels. The automatic abstraction is fundamental to speed-up simulation of large systems. Moreover, the platform provides tools for abstraction of RF components via a network simulator integrated in SystemC, and for the abstraction of OS primitives to execute software in SystemC. This firmware simulation is based on the integration of ISSs into SystemC.

Another abstraction example comes from the analog domain where the target is to use accurate EM simulation data of Antennas and RF-Boards at a higher abstraction level, inside a circuit simulator like ADS [19]. Finally, by using the diverse simulation capabilities of ADS, a co-simulation between data flow (SystemVue) and analogue envelope models is feasible in order to investigate possible effects of the Physical model in digital modulated signals.

3.1.3. *Abstract tool-independent interface generation*

This block represents the most relevant feature of the SMAC platform. SystemVue interfaces to SystemC, with a pre-defined set of interfaces that correspond to a subset of the possible scenarios. The SMAC platform extends this interface by providing co-simulation (i) at various abstraction levels and (ii) at different spatial and temporal scale. The innovative feature of the SMAC platform is that these interfaces are generated automatically. The interface generation module serves the task of inserting into the SystemVue co-simulation engine the correct interface, for each model and abstraction level. It is worth re-emphasizing the awareness of integration issues for both component/subsystem design and the co-simulation itself: The co-simulation platform influences the way the models are built, and the platform accounts for the constraints posed by the models (at different levels of abstraction, depending on the type of components they refer to).

The co-simulation platform can be generalized to a simulation environment that contains a number of subsystems, interactions, stimuli and global (environmental) parameters such as temperature, process, EM field strength, etc. Each subsystem embeds a number of views that correspond to their representation in each abstraction domain. The subsystem interacts with the platform and other subsystems. To ensure that connections and parameters are correctly translated from the subsystem to the platform, a well-defined interface encloses the subsystem.

3.2. *Tool specific-view of the platform*

SMAC is a simulation platform where different tools from different vendors act together to simulate a system, whose components come from different modelling domains as well as different abstraction levels. The SMAC platform relies on a model generation and conversion flow to bring all the models of the system in an appropriate format, by which they can be simulated through either a set of simulation engines (using a co-simulation flow) or through a single simulation engine at the higher possible abstraction level (i.e., System-Level).

The platform supports a set of different modelling formats, including those that are the de-facto reference standard at the state of the art (e.g., VHDL and Verilog for the Digital Domain). The platform includes different simulators (MEMS, FEM, Method of Moments, Continuous Time, Synchronous Data Flow, Timed Synchronous Data Flow, Circuit, Discrete Events, Instruction Set Simulators, Finite Element Analysis) for simulating the system components in each different domain.

In the first platform development step, the several existing tools, which have been identified as most representative by the project consortium, and their interfaces that are mostly used at the state of the art for smart systems co-design *before* the SMAC project have been analysed. Figure **4** shows a representative comprehensive scheme of such an analysis, which underlines the main lacks of the tools that prevent a complete multi-domain and multi-level co-design environment.

Figure **4** underlines, for instance, how, through the use of tools at the state of the art, co-simulation environments can be set up between models belonging to two different domains. Examples are digital Hardware and Analog-RF co-simulation through the native SystemVue tool [18], digital Hardware and embedded Software co-simulation through QeMU [29] and SystemC [30], Analog-RF and embedded Software through SystemVue and MATLAB, and Analog-RF and Discrete and Power Devices through SystemVue and ADS [19]. Nevertheless, such co-simulation environments do not allow a comprehensive co-simulation between more than two domains and do not cover all smart systems domains.

The gaps in the multi-domain design have been formulated as requirements and goals to the SMAC project, and have implied extensions and enhancements to the identified modelling tools.

Figure 5 (i.e., tool interface *after* the SMAC project) shows the main results obtained by the SMAC consortium regarding the tool interface extension. The bold lines underline the contributions of SMAC to fill the gaps, which consist of new co-simulation environments as well as conversion tools. The conversion tools allow translating model descriptions through abstraction levels and through description formats, with the aim of simulating (rather than co-simulating) the whole smart system description. All these contributions are summarized in the next sub-sections.

3.2.1. SystemC-SystemVue co-simulation

The Keysight Technologies' SystemVue ESL platform [18] has been extended to fill the gap between the digital/embedded Software domain modelled and simulated by SystemC and the analog/RF/MEMS domains, which were modelled and simulated through the SystemVue native data flow simulator.

The polymorphic design entry of SystemVue supports model-based design flow (GUI blocks, language-based C++ or math, VHDL). Models are represented as block entities and are connected with each other, forming a block-diagram design. The data flow simulation engine executes each of these models based on a schedule computed at the pre-simulation phase.

Exploiting this feature, a new block model, named SystemCCosim, has been developed inside SystemVue to allow co-simulation with SystemC designs. When executed, the block establishes an Inter-Process Communication (IPC) link with the SystemC process, which runs externally. Through this link, which is based on shared-memory, data are passed back and forth between SystemVue and SystemC processes [20].

3.2.2. Thermal Simulation in ADS

Advanced System Design (ADS), which is a native circuit system simulator [19], has been enhanced with a new electro-thermal simulator based on a full 3D thermal solver natively integrated. It incorporates dynamic temperature effects to improve accuracy in "thermally aware" circuit simulation results.

All devices dissipate electrical power as heat. Heat flow causes devices to operate at different temperatures. Since device electrical properties change with temperature, they need self-consistent powers and temperatures. The electro-thermal analysis provides coupled simulation between a circuit simulator that computes power dissipation and a thermal simulator that computes temperatures. The simulation platform automatically exchanges data between the two engines and iterates to a self-consistent electro-thermal solution. The thermal simulator solves the full physical three-dimensional problem, incorporating all of the geometries in the layout and their thermal properties.

3.2.3. EMPro and ADS Integration

Electromagnetic Professional (EMPro) [21] is an EM simulation software for analysing the 3D EM effects of components such as high-speed and RF IC packages, bondwires, antennas, on-chip and off-chip embedded passives and PCB interconnects. EMPro has been extended to be integrated with ADS. EMPro designs are stored in a cell, which contains multiple views. These views can be used in ADS layout and schematic. A common database allows 3D components built in EMPro to be placed on ADS schematics and layouts directly. EMPro's increased level of integration with ADS is based on a shared database approach. Three-dimensional objects in EMPro can be saved as ADS design database "cells" for use directly in ADS. Direct integration streamlines the design process and gives circuit designers easier access to full 3D EM modelling capabilities. Furthermore the use of OpenAccess standard in ADS and EMPro enhances the effort to provide true interoperability, not just data exchange, among IC design tools through an open standard data API.

ADS RF designers can import EMPro parameterized 3-D components for EM simulation and optimization in ADS. Because the parameterized 3D component is created just once and integrates with both EMPro and ADS, error prone links between today's standalone 3D EM point tools and ADS are eliminated, resulting in faster design and assured accuracy.

Additionally, ADS circuit designers can transfer their layout from ADS into EMPro to accurately determine effects such as conformal bending of a planar antenna around a curved product package or effectiveness of metal shielding.

EMpro's 3D parameterized components such as connectors, solder ball arrays, packages or any custom components can be transferred into ADS as a library for use in 3D EM co-simulation and co-optimization with circuit and system components, enabling evaluation of complete family of structures in one step.

3.2.4. Automated EM – circuit co-simulation in ADS

EM simulation of a complex electronic design is often required to verify correct operation at high frequencies. EM simulators can model the behaviour of the passive piece of the design, such as interconnect. Active components, such as transistors, cannot be modelled in the same way. A layout design contains both active and passive pieces. Setting up a layout for simulation requires separating both pieces. EM simulation of the passive piece is performed first; then, a subsequent circuit simulation combines the resulting EM model with circuit models for the active piece. The final result is an accurate EM/circuit co-simulation of the whole system.

For complex layouts, manually separating the active and passive pieces and stitching both pieces together in a circuit simulation, is a tedious and error prone job. One of the SMAC contributions is automating this job. First, it allows the user to choose whether a piece of the design needs to be EM or circuit simulated. Then, it uses this information to automatically generate the passive piece of layout, including pins and EM ports on all the positions where active components need to be hooked up. The third step is automatically hooking up these active components to the EM model of the passive piece in a circuit simulation. As a result, setting up a proper EM/circuit co-simulation of a complex system becomes a matter of minutes rather than hours or days.

3.2.5. HIF Suite

HIFSuite [22] is a set of tools and file formats for automatic conversion and manipulation of digital models. The main HIFSuite goal is the creation of a manipulation and verification environment, independent from any specific HDL. During the project, the framework has been extended to be suitable for smart system co-design in many directions. To improve the quality of front/back-end tools and precisely define the semantics of HIFSuite descriptions, a HIF normal form has been defined. Such a normal form represents HIF descriptions generated after the parsing phase. Each further manipulation performed by manipulation tools (e.g., A2T abstraction tool, HR hierarchy removal, etc.) is checked to be compliant with the normal form. *VHDL and Verilog* parsers have been re-designed to generate HIF descriptions compliant to the new HIF normal form. The development of a tool (psl2hif) to convert PSL properties into HIF descriptions has been developed. A tool to convert IP-XACT descriptions into HIF code (and vice versa) has been implemented.

The HIFSuite A2T abstraction tool, used to generate both SystemC-TLM and C++ descriptions starting from RTL designs has been improved in several ways. The "dynamic scheduler", which is the default simulation algorithm embedded into the abstracted designs has been optimized to improve the simulation performance. In addition, a static scheduler, which is alternative to the dynamic one, has been developed from scratch. The combination of dynamic and static scheduling leads to faster simulations in designs that have lightweight processes. A2T has been extended to support designs featuring multiple and derived clocks, i.e. clocks whose period is a multiple of the period of a main clock. In this case the main clock is abstracted, and the other derived clocks are treated as signals, thus ensuring the correct elaboration of the design functionality. The C++ code generated by A2T consists of procedures, which performs simulation through steps. This format has been developed to be compliant to SystemVue's requirements and thus can be easily integrated for complex co-simulations.

3.2.6. MEMS+

MEMS+ [23] is an integrated platform for MEMS design and simulation. It has been especially developed in view of interfacing the MEMS design with EDA tools while still allowing simulating the MEMS device at a high degree of accuracy. MEMS+ enables MEMS designers to assemble a behavioural model of a MEMS device in a 3D graphical user interface based on a library of building blocks mainly including models for mechanics, electrostatics and piezoelectrics.

In the context of SMAC, the existing system level simulation possibilities in MEMS+ were not flexible and sufficient enough. The extensions to MEMS+ developed in the SMAC project are described in the following.

The simulator allows the user to conduct basic MEMS device analysis to verify the correctness of a device before exporting it to MATLAB, Simulink, or Cadence Virtuoso for system simulation. The simulator can be used to run several analyses, such as DC analysis (steady-state equilibrium with mechanical and electrostatic forcing), DC Sweep (electrostatic pull-in and lift-off, and contact), Mechanical modal analysis (includes electrostatic spring softening effects), and Small-signal AC (linear frequency response to mechanical and/or electrostatic forcing with Reynolds gas damping effects).

The user can also run a vary analysis, alter operating points, alter variables, and alter sources. The development of this simulator was required in order to allow the majority of the developments needed to interface with the SMAC platform. In addition, it adds a high level of user friendliness for the MEMS designer, who can now run most the MEMS-only relevant simulation in a 3D environment.

Once the 3D model is created in MEMS+ Innovator, it is ready for simulation in MATLAB. MEMS+-for-MATLAB provides a variety of MATLAB scripts that allow model import, manipulation and simulation directly from the MATLAB command line interface. The scripts provided by MEMS+ support DC, DC transfer, Modal, AC, and transient analyses without requiring any additional toolbox. All MATLAB simulation results can be loaded back into the MEMS+ Scene3D module for 2D and 3D viewing. Users can also create a material database, process, and schematic or modify existing ones from a MATLAB command prompt.

The functionality of exporting Verilog-A models from MEMS+ has been developed to integrate MEMS models easily in most circuit simulators accepting this format, especially Keysight ADS and EDALab HIFSuite. The resulting Verilog-A models are currently linear and non-parametric. Parameterization and a certain degree of non-linearity are under investigation.

In order to obtain a Verilog-A model from MEMS+, the user runs a DC (or DCSweep) in the MEMS+ simulator plug-in. After the DC has run, a new entry in the context menu appears called "Export Linear Verilog-A". Within one click, a file browser pops-up to choose a file where to save the Verilog-A model. It is a fully automatic way of generating a Verilog-A model. The model can now be used in the majority of circuit simulators and without any license. It includes the system matrix of the MEMS device but no physical information on the design.

Several improvements have been incorporated in the MEMS+ platform related to co-simulation aspects between the MEMS+ library components and Cadence Spectre [24]. Most of the improvements are related to gaining further speed. Of particular interest is the co-simulation for MEMS microphones, where noise simulation is a true challenge and requires full co-simulation between mechanics, electrostatics, fluidics (damping) and electronics – considering noise sources of different physical origin.

A number of new model capabilities have been added to the MEMS+ component library to allow packaging-die co-simulation and improved modelling of fluidics damping effects.

3.3. The simulation-levels and the design-domains matrix

To characterize and organize the set of simulation tools that are part of the platform, we refer to the matrix of simulation-levels and design-domains as reported in Figure 6. Rows discriminate abstraction levels, while columns discriminate application domains. Not every combination is meaningful: this is represented as a white box. A simulation tool can occur in more than one box, and a box can contain more than one tool. The SMAC platform exploits abstraction/refinement techniques and tools to move descriptions across rows. Automatic interface generators enable (co-)simulation of tools across columns.

Simulation scenarios are depicted as box clusters from Figure 6. Interconnected boxes participate to the (co-)simulation. At most one box is selected for a component from each application domain. Not every domain is required. In case there is more than one component belonging to the same application domain, models at different abstraction levels may be used.

To better clarify the difference between the most abstracted levels:

- The *functional level* describes, in C/C++, the functionality of a component, and it can be used for simulating a component in isolation, since its execution after compilation represents the module behaviour. It can also be used to simulate a set of components of the same domain, since all such components share the same computational model, thus their composite behaviour can be obtained by executing the functionality of all modules.
- The *transactional level* adds a protocol to the functionality of each module to allow the composition of modules of different domains. The protocol allows designers to correctly combine heterogeneous modules to simulate the composite behaviour. A reference example of transactional description is the one standardized by OSCI (now Accellera) for digital hardware and software and implemented in the SystemC TLM library. In SMAC**,** we propose to use the protocols provided by SystemVue to integrate components of different domains. Each module is thus first abstracted up to the functional level and, then, the protocol is added to allow the integration of heterogeneous modules for a simulation in SystemVue.

The matrix of Figure 6 is used in the SMAC project to represent all design and simulation scenarios. This matrix is able to capture all models that compose the SMAC platform. Some examples of mapping between abstraction levels and design models are the following:

- <u>Physical level</u>: a TCAD structure/model of a transistor; a PCB geometry in the EM simulator.
- <u>Device level</u>: the gate-level netlist for a digital block; the SPICE netlist for an analog block; a SPICE transistor model.
- <u>Structural level</u>: the RTL description of a digital block; the TRAPPIST VerilogA model of an analog block.
- <u>Functional level</u>: the abstracted description of a RTL digital block.
- <u>Transactional level</u>: the functional description of an analog module completed with the protocol for the simulation in SystemVue.

3.4. Simulation vs. Co-Simulation in the SMAC platform

The design-domains/simulation-level matrix allows us to correctly differentiate the use of co-simulation from the use of simulation. This can be summarized with the annotated matrix reported in Figure 7. Models of the lowest abstraction levels are represented by different design languages, therefore they must be simulated by using their own simulator (e.g., MATLAB, Modelsim, EMPro, etc.). For this reason, a simulation covering more than one domain can be implemented only by using co-simulation techniques which connect different tools by exchanging simulation data from one tool to another.

Moving to the functional level there is a convergence in the modeling language, that is all models of the different domains are represented in C++. This would in principle allow simulation to be performed among different domains. However, the computational model implemented into each C++ model may be diffent from domain to domain. Thus, a simulation cannot be simply obtained by linking functional C++ models of different domains, but such models must be coherent with respect to the same computation model, which must be able to cover all domains. A universal computational model (*univerCM [25]*) has been introduced in the SMAC project to describe models of all design domains. UniverCM is the core of the language abstraction and transformation tools composing the SMAC platform. If functional C++ models are generated by starting from UniverCM, they can be simply linked together to simulate a design covering more than one domain.

Not all functional models cannot be generated, or are not generated, by using the automatic abstraction tools of SMAC based on U*niverCM*: in this case, we can move to the transactional level where SystemVue is used as general integration methodology to have a simulation covering more then one design domain. In fact, functional C++ models are extended with the protocol at the transactional level and they can thus be connected in SystemVue by using a standard interface defined in the SMAC project. This allows the coherent simulation of functional models based on different computation models, reaching the result of producing the SMAC simulation platform.

4. The SMAC platform evaluation and assessment: the limb tracking case study

The effectiveness of the SMAC platform has been evaluated from the end-users perspective through the design and simulation of several real smart systems *components* from all the considered domains, as well as design and simulation of complete *smart systems*. In this article we present some of the results obtained by using the tools and methods of the SMAC platform to design and simulate the system-level model of limb tracking equipment (i.e., a demonstrator), a wearable smart system for body motion reconstruction based on inertial sensor nodes. The monitoring of limb trajectories in daily life experiences is an attractive topic for many application fields: Fitness, sport and rehabilitation, or, in general, healthcare and wellness. The recent advances in inertial sensors allow a high level of integration of solutions based on Inertial Measurement Units (IMUs) with very low costs and quite good measures reliability. In addition, compared to optical methodologies, they do not require setting up specific infrastructures in the measurement environment and do not lose motion tracking when a body part is hidden from the camera. The usual laboratory setup for performance characterization of inertial sensor nodes includes costly vision-based systems employing a number of cameras, markers on the body and complex motion tracking algorithms (e.g., [28]). The SMAC precise system-level simulation, taking into account the characteristics of the real components deriving from the designers' lower-level representations, allows evaluating the limb tracking accuracy in mission-like conditions while drastically lowering development time and costs for performance improvement and optimization with respect to prototype-based flows.

The studied limb tracking smart system stems from the motion capture system presented in [26] and uses a set of MEMS sensor nodes attached to various body parts to track the relative displacement of limbs. Each node includes 3-axis MEMS gyroscope, accelerometer and magnetometer, and a 32-bit microcontroller (see Figure 8.a).

The microcontroller runs an extended Kalman filtering algorithm [27] to perform the *sensor fusion*, i.e., the integration of data generated from different sensors, in order to provide better angular position estimation. The algorithm integrates the gyroscope reading, i.e., angular velocity, to compute the angular position in dynamic conditions, and corrects the computed values (subject to drift due to the numeric integration) leveraging information about gravity from accelerometers and magnetic field readings from magnetometers. Once an absolute angular position is obtained (with respect to the gravity vector and to the magnetic North) for each body part (e.g., one node on the arm and one on the forearm, and taking into account the elbow movement constraints), the system reconstructs the movement of a person's limbs. The system is conceived to be scalable in terms of number of nodes, thus guaranteeing the possibility to monitor several body sections.

The sensor node is based on the iNEMO M1 System-on-Board (SoB) manufactured by STMicroelectronics (Figure 8.*b*). The main features of the SoB regarding the 3-axis digital gyroscope L3GD20 are the following:

- 3 selectable full scales: ±250, ±500, ±2000 dps.
- 4 different user selectable output data rates (ODR): 95, 190, 380 and 760 Hz.

- 16-bit data output, I2C/SPI digital interface.
- Integrated Low Pass and High Pass filters with selectable cut-off frequencies.
- Embedded FIFO.

The main characteristics of the geomagnetic module LSM303DLHC are the following:

- 3 magnetic field channels and 3 acceleration channels.
- From ±1.3 to ±8.1 gauss magnetic full scale.
- ±2g, ±4g, ±8g, ±16g dynamically selectable acceleration full scale.
- 16-bit data output, I2C digital interface.
- Embedded FIFO.

The SMAC platform has been used to simulate the limb tracking equipment by means of a system-level model, taking into account both the functional and the multi-physical (functional and extra-functional) properties of such a smart system. In particular, the goal of the system simulation was to tune the system components and software (i.e., the Kalman filtering algorithm) to make the tracking more precise, and to render it immune and capable to compensate environmental effects such as temperature and local magnetic fields.

The full set of system requirements is the following:

- At least two sensor nodes, each one equipped with tri-axial gyroscope, tri-axial accelerometer, tri-axial magnetometer.
- Each sensor node has to perform a preliminary sensor fusion based on extended Kalman filtering (with a distributed computing approach to enhance scalability).
- Each sensor node has to compensate temperature effects (between 0 and 50°C) and magnetic disturbance caused by vicinity to ferromagnetic elements.
- The sensors shall transmit real-time data to a central unit able to display their relative positions in space.
- Maximum current consumption of each sensor node shall be less than 100 mA at 3.3V, and be powered by battery.
- The tracking accuracy in terms of computed limb angle in static conditions shall be < 2°.

Figure 9 shows the SMAC platform tools and flows presented in Section 3.2 used for the system-level design of the case study (see the purple arrow and box outlines), which are analysed more in detail in the following.

4.1. Model abstraction and cross-domain constraint management

The system-level model of each sensor node includes MEMS accelerometer, MEMS magnetometer, MEMS gyroscope, and a microcontroller running the sensor fusion algorithms. The analysis and optimization of such a system require modelling techniques merging multi-domain models and constraints. The SMAC platform supported the definition of complex interactions between the system components, including functional and non-functional connections. Among the former ones, there are physical stimuli (sensor kinematics, gravity, magnetic fields) feeding the sensors, electrical signals and data transactions. Non-functional connections is represented by temperature.

The main building blocks have been translated across different domains. MEMS components, which were originally modelled in MEMS+ format, have been translated into VerilogA and, then, translated into SystemC-AMS (see rightmost side of Figure 9).

In addition, abstraction tools of the SMAC platform have been evaluated to abstract accurate and synthesizable RTL models implementing digital components into SystemC TLM models (see rightmost side of Figure 9). As an example, the HIFSuite A2T abstraction tool (see Section 3.2) has been applied to abstract a 32-bit RISC core, which represents the digital counterpart and whose original description was in VHDL, into SystemC TLM. The RTL code of the CPU features 32 32-bit general purpose registers and has a three-stage pipeline. The VHDL RTL description of the core consists of several sub-modules, such as a program counter unit, a memory controller, an instruction opcode decoder, the register bank, a bus multiplexer, an ALU, a shifter, a multiplication and a division unit.

The RTL model implementing the digital component (i.e., the microcontroller) consists of 1,274 lines of VHDL code, with 22 primary inputs and 22 primary outputs, 7 synchronous processes and 94 asynchronous processes.

The generated SystemC TLM description consists of 9,827 lines of code. The TLM abstraction led to an increase of the lines of code due to the fact that the abstraction process adds scheduling routines of concurrent statements besides the functional parts, in order to preserve the event-driven semantics and, thus, the functional equivalence with regard to the starting RTL model. Nonetheless, TLM code generation was almost instantaneous (approx. 1.5 mins).

The RTL simulation and TLM simulation time are 242.33 and 51.67 seconds, respectively. Both RTL and TLM models were simulated and compared with the same set of stimuli. The TLM version allowed reaching an average speed-up of 4.7x. These results underline the efficiency of the SMAC platform abstraction flow, in the context of the digital domain, to speedup the simulation performance by abstracting the description models. This is particularly worth, from an industrial point of view, when the modeled system is complex, the RTL description would slow down the system simulation performance, and a manual RTL-to-TLM abstraction would be time expensive (as in the limb tracking case study).

The abstracted microcontroller and the other block models composing the limb tracking system have been integrated and co-simulated as a whole at system level. The system level modelling and simulation have been performed targeting the following requirements:

- Simulation of multi-domain interaction including mechanical (acceleration and angular velocity), electromagnetic (Earth's magnetic field and disturbance fields) and thermal effects (on the sensor transfer functions), by synthesizing suitable inputs for the sensor models while injecting environmental perturbations representing both EM and temperature noises.
- Integration and customization of the sensor fusion algorithms on each sensor node.
- Modelling of acquisition, elaboration and data transmission latency.

System-level performance has been evaluated in terms of position reconstruction accuracy and latenc. Initial requirements and constraints have also been refined and applied to sub-system boundaries as in the top-down approach described in [31].

The system integrator manages cross-domain and cross-layer constraints by suitably interconnecting functional and non-functional signals between the models. As an example, the environmental temperature, and the possible heating of the electronic parts, affects the transfer functions of the MEMS transducers. This effect is visible in system-level simulation and has been corrected by tuning the compensation algorithm running on the microcontroller.

4.2. The simulation platform

Figure 10 shows the block diagram of the employed accelerometer model for one axis. It includes MEMS, analog and digital parts: Its main composing elements are the mechanical second order Laplace transfer function, which expresses the relationship between applied acceleration (and consequently force F) and the displacement d of the oscillating mass, a non-linear module expressing the capacitance of the plates C as a function of their displacement, the analog front-end (AFE, comprising amplifiers and filters), the analog-to-digital converter (ADC) and the digital front-end (DFE). Mechanical, analog and digital electronic domains are represented.

The model parameters derive from lower-level representations employed by the device designers. They represent the behaviour of a *nominal* device, but it can be easily customized in order to evaluate the effect of manufacturing variability and corner cases or to mimic specific devices.

Beyond the functional multi-physics relations that allow sensing acceleration and expressing it as a digital reading, electrical noise is also included in the model. The temperature value has been taken into account in a limited range (0÷50°C) since its variations dynamically affect the sensor model behaviour.

The mechanical and analog parts, up to the ADC module, have been modelled in SystemC-AMS language using the Timed Data Flow (TDF) model of computation. SystemC TLM has been used to describe the behaviour of the digital front-end including the state machines that manage the device read-out configuration and the programmable registers (full scale selection, output data rate, and digital filters). The complete sensor model is encapsulated in a SystemC wrapper that operates the required SystemC to/from SystemC AMS conversions needed for the instantiation of the sensor modules in the SystemVue environment (SystemC-SystemVue co-simulation).

The magnetometer (sharing its digital interface with the accelerometer within the LSM303DLHC module) and the gyroscope have been modelled in similar ways. Since the objective of the virtual prototype is the enhancement of sensor fusion and limb tracking algorithms, digital communications at the system level are handled as TLM transactions.

The sensor fusion library, which implements an extended Kalman filtering algorithm, is designed as a C language code that runs on the microcontroller. Such a library is derived from the iNEMO Engine Lite developed and distributed as freeware by STMicroelectronics [32]. The algorithm works on quaternions to avoid singularities introduced by Euler angles.

Finally, a data collector unit, which detects the position on the body of each connected sensor, receives the attitude and heading information from each unit and provides an estimate of the relative limb position. Figure 11 shows a block diagram of the complete system.

In order to test the performance of the limb tracking system and to develop and evaluate enhancements and customization of the sensor fusion and limb position estimation algorithms, input stimuli have been synthesized, by reflecting real human movements.

To obtain such data, OpenSim was selected. OpenSim is an open-source platform for modelling, simulating and analysing the neuromusculoskeletal system [33]. The limb movement recordings collected during experiments and simulations and available in the OpenSim online database have been employed as stimuli. The positions in which the sensor nodes have to be located are defined on an OpenSim virtual human body representation (Figure 12). Thanks to the analysis tools, the kinematic properties of the sensor positions were tracked (position and Euler angles) during motion simulation, and saved. For each of the tracked points, an ad-hoc tool generates ideal sensor stimuli for the accelerometer (kinematic acceleration and projections of the gravity vector), magnetic field for the magnetometer (with the possibility of taking into account the Earth's and local magnetic source fields) and angular velocity for the gyroscope.

These values have been fed into the sensor models instantiated in the SystemVue virtual platform implemented through SMAC. During simulation, the environmental temperature has been dynamically controlled and its effects on the sensors transfer functions have been evaluated. The output of local sensor fusion at each node and the tracked limb position estimations have been computed and compared to those generated through OpenSim. This allowed us to evaluate the system performance in terms of errors introduced by the system non-ideality taken into account by the model.

Figure 13 shows the estimation of Euler angles (roll, pitch and yaw *iNEMO* computed from the quaternions) generated by the virtual model running the original sensor fusion algorithm (i.e., iNEMO M1), obtained during a simulated walk at constant temperature and elaborated at a 50Hz frequency. The sensor is placed on the right femur. The figure shows the comparison between such data with the reference data generated by OpenSim (roll, pitch and yaw *ref.*). The figure underlines that the yaw angle estimation provided by the original sensor fusion algorithm does not converge within the initial 3 seconds.

By relying on the SMAC platform models, the sensor fusion algorithm has been customized and improved in different ways. First, sensor calibration has been taken into account to minimize the effect of MEMS and electronic component manufacturing variability, by computing gain and offset values for each sensor axis. Then, the model of the Earth's magnetic field used by the Kalman update step has been improved by considering local declination and inclination. Temperature compensation has been then introduced, by studying the simulated behaviour of each sensor and deriving the best fitting parameters. Finally, the parameters of the Kalman noise covariance matrices have been optimized to further reduce the estimation errors.

To study the effects of magnetic field perturbations, deriving, e.g., from magnets or electric currents, multiphysics models have been developed and introduced in simulation. To minimize the error introduced by such perturbations in the orientation estimation, we introduced dynamically weighted sensor covariance matrices in the Kalman filter: the weights have been tuned proportionally to the error between the expected field modules and the measured values.

The optimization and customization of the software application running in each sensor node allowed us to satisfy each of the requirements initially set. The results of orientation estimation for a sensor node, in the same conditions of the aforementioned experiment, are reported in Figures 14 (angle estimation) and 15 (absolute error). As shown in the figures, after an initial setup, the maximum error in dynamic condition is below 5° on each axis.

Each second of simulation of a single sensor node takes about 30 seconds of real time, on a quad-core Intel Core i7 – 2670QM running at 2.20GHz with 4GB of RAM. It has to be noted that each SystemC module instantiated within SystemVue runs on a separate thread, thus making the system scalable when more than one sensor node needs to be simulated, directly exploiting multi-core parallelism.

5. Conclusions

This article presented SMAC, a smart systems co-design platform. The usual design and optimization methods for heterogeneous smart systems involve separate flows: Each smart system component is designed with specific tools and methods, while the system integrator needs to develop from scratch a high-level uniform system representation (e.g., in SystemC or MATLAB). The optimization of embedded algorithms requires measuring system performance and possibly costly prototype iterations. The multi-domain and multi-layer modeling and simulation methods developed in SMAC improve the state-of-the-art smart system design techniques, defining faster, more accurate, integrate and less expensive design flows. The article summarized the contributions of SMAC in terms of tools, enhancement of existing tools, conversion flows, and methodologies, to allow a complete multi-domain and multi-level co-design environment. The article presented preliminary results obtained by applying the SMAC platform to design and simulate the system-level model of a limb tracking smart system to test the efficiency of the platform to a real and complex case of study.

Acknowledgements

For obvious space reasons we had the limit the authors' list to the project work package leaders and to the partners with the largest involvement in the project. Needless to say, the achievements of the project would not have been possible without all the other partners and the many people (more than 100) involved in SMAC.

References

[1] Strategic Research Agenda of EPoSS – The European Technology Platform on Smart Systems Integration. Pre-Print September 2013 http://www.smart-systems-integration.org/public/documents/publications
[2] J. Pereira (ed.). Monitoring and Control: Today's Market and Its Evolution till 2020. Luxembourg: Office for Official Publications of the European Communities, 2009.
[3] T. Grötker, S. Liao, G. Martin, S. Swan, System Design with SystemC. Springer, 2002, Chapter 8., pp. 131. ISBN 1402070721
[4] L. Cai, D. Gajski, Transaction Level Modelling: An Overview, in proceedings of the Int. Conference on HW/SW Codesign and System Synthesis (CODES-ISSS), Oct. 2003, pp. 19–24.
[5] B. Romanowicz, Y. Ansel, M. Laudon, Ch. Amacker, P. Renaud, A. Vachoux, G. Schröpfer, VHDL-1076.1 modelling examples for microsystem simulation, "Analog and Mixed-Signal Hardware Description Languages" Alain Vachoux (Editor), Kluwer Academic Publishers, July 1997
[6] N. Zhou, J. V. Clark, K. S. J. Pister: Nodal Analysis for MEMS Design Using SUGAR v0.5, Proc. Int. Conf. on Modelling and Simulation of Microsystems, Semiconductors, Sensors and Actuators (MSM), Santa Clara, CA, USA, 1998, 308-313
[7] J. E. Vandemeer, M. S. Kranz, G. K. Fedder, Hierarchical Representation and Simulation of Micromachined Inertial Sensors, Proc. Int. Conf. on Modelling and Simulation of Microsystems, Semiconductors, Sensors and Actuators (MSM), Santa Clara, CA, USA, 1998, 540-545.
[8] G. Lorenz, R. Neul, Network-Type Modelling of Micromachined Sensor Systems. Proc. Int. Conf. on Modelling and Simulation of Microsystems, Semiconductors, Sensors and Actuators, MSM98, Santa Clara, April 1998, pp. 233-238.
[9] G. Lorenz and J. Repke, Design Tools for MEMS, Sensors Applications, Volume 4, Sensors in Automotive Technology, Jossey-Bass Publishers (A Wiley Company), ISBN: 3527295534, p. 58-72
[10] G. Lorenz, M. Kamon, A System-Model-Based Design Environment for 3D Simulation and Animation of Micro-Electro-Mechanical Systems (MEMS), Proceedings of APCOM'07 in conjunction with EPMESC XI, Kyoto, Japan, December 3-6, 2007
[11] G. Schröpfer, G. Lorenz, S. Breit, Novel 3D modelling methods for virtual fabrication and EDA compatible design of MEMS via parametric libraries, Journal of Micromechanics and Microengineering 20 (2010), 064003 (based on invited talk at Micromechanics Europe 2009, Toulouse, France, September 2009)
[12] Frangi, G. Spinola, B. Vigna, On the evaluation of damping in MEMS in the slip–flow regime, Int. J. Numer. Meth. Engng 2006; 68:1031–1051
[13] Frangi, A fast multipole implementation of the qualocation mixed-velocity–traction approach for exterior Stokes flows, J. of Eng.neering Analysis with Boundary Elements 29 (2005), 1039–1046
[14] G. Lorenz, K. Greiner, S. Breit, A Hybrid Modelling Approach for Multi-Physics Systems, IMSTW 2006, June 2006. Edinbugh, UK
[15] Schneider, P.; Reitz, S.; Bastian, J.; Schwarz, P.: Combination of analytical models and order reduction methods for system level modelling of gyroscopes. Nanotech 2005, MSM, Anaheim, Cal., 8.-12. May 2005.
[16] Tamara Bechtold, Model Order Reduction of Electro-Thermal MEMS, PhD-thesis at University of Freiburg, Germany, 2005
[17] A.M. Sabatini, "Quaternion-Based Extended Kalman Filter for Determining Orientation by Inertial and Magnetic Sensing", IEEE Trans. on Biomedical Engineering, 53-7, pp. 1346-1356, July 2006
[18] SystemVue ESL Software – Keysight Technologies. http://www.keysight.com/en/pc-1297131/systemvue-electronic-system-level-esl-design-software?nid=-34264.0&cc=GB&lc=eng
[19] Advanced Design System (ADS) – Keysight Technologies. http://www.keysight.com/en/pc-1297113/advanced-design-system-ads?cc=US&lc=eng
[20] "SystemC Cosimulation," Keysight Technologies, [Online]. Available: http://edadocs.software.keysight.com/display/ads201101/SystemC+Cosimulation.
[21] EMPro 3D EM simulation software – Keysight Technologies. http://www.keysight.com/en/pc-1297143/empro-3d-em-simulation-software
[22] HIFSuite – EDALab s.rl.l. http://hifsuite.com
[23] MEMS+ - Conventor. http://www.coventor.com/mems-solutions/products/mems.
[24] Spectre Circuit Simulator – Cadence. http://www.cadence.com/products/cic/spectre_circuit/pages/default.aspx
[25] L. Di Guglielmo, F. Fummi, G. Pravadelli, F. Stefanni, S.Vinco. UNIVERCM: The UNIversal VERsatile computational model for heterogeneous system integration. IEEE Transactions on Computer (62)1. 2013
[26] Brigante, C.M.N., Abbate, N., Basile, A., Faulisi, A.C., and Sessa, S. 2011. Towards Miniaturization of a MEMS-Based Wearable Motion Capture System. IEEE T. Comput. 58, 8 (2011), 3234-3241.
[27] Sabatini, A.M. Quaternion-Based Extended Kalman Filter for Determining Orientation by Inertial and Magnetic Sensing. IEEE T. Bio-Med. Eng. 53, 7 (July 2006), 1346-1356.
[28] Vicon Motion Systems Ltd. http://www.vicon.com/.
[29] QeMU – Open source processor emulator. http://www.qemu.org
[30] Accellera Systems Initiative: SystemC. http://www.accellera.org/downloads/standards/systemc
[31] M. Crepaldi, M. Grosso, A. Sassone, S. Gallinaro, S. Rinaudo, M. Poncino, E. Macii, D. Demarchi. A Top-Down Constraint-Driven Methodology for Smart System Design. IEEE Circuits and Systems Magazine 14, 1 (2014), 37-57.
[32] iNEMO Engine Software Libraries – STMicroelectronics s.r.l.. http://www.st.com/web/en/catalog/tools/FM147/CL1818/SC1528?sc=inemo-engine
[33] S.L. Delp, F.C. Anderson, A.S. Arnold, P. Loan, A. Habib, C.T. John, E. Guendelman, D.G. Thelen. OpenSim: Open-source Software to Create and Analyze Dynamic Simulations of Movement. IEEE T. Bio-Med. Eng. 54, 11 (2007), 1940-1950.
[34] Callou, G., MacIel, P., Tavares, E., Andrade, E., Nogueira, B., Araujo, C., Cunha, P. Energy consumption and execution time estimation of embedded

system applications (2011) Microprocessors and Microsystems, 35 (4), pp. 426-440.

[35] Atac, A., Chen, Z., Liao, L., Wang, Y., Schleyer, M., Zhang, Y., Wunderlich, R., Heinen, S. An HDL-based system design methodology for multistandard RF SoC's (2014) Proceedings - Design Automation Conference, art. no. 2593122, .

[36] Sun, X., Chen, Y., Liu, J., Huang, S. A co-simulation platform for smart grid considering interaction between information and power systems (2014) 2014 IEEE PES Innovative Smart Grid Technologies Conference, ISGT 2014, art. no. 6816423, .

[37] Wane, S., Boguszewski, G. Global digital-analog co-simulation methodology for power and signal integrity aware design and analysis (2008) Proceedings of the 38th European Microwave Conference, EuMC 2008, art. no. 4751746, pp. 1477-1480.

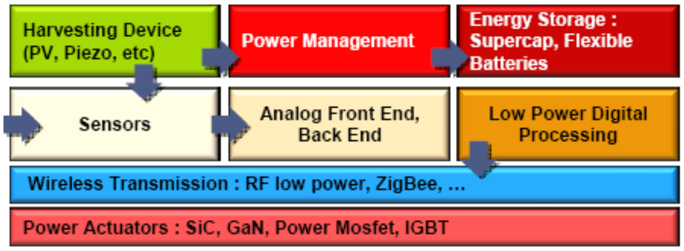


Figure 1. Typical Components of a Smart System.

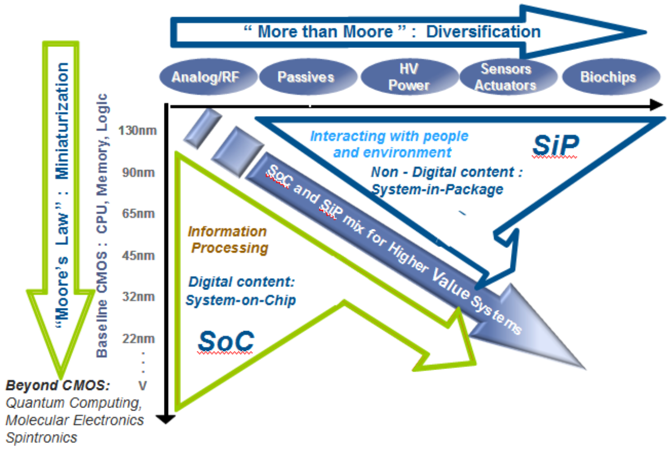


Figure 2. Smart Systems as a bridge between More Moore and More than Moore.

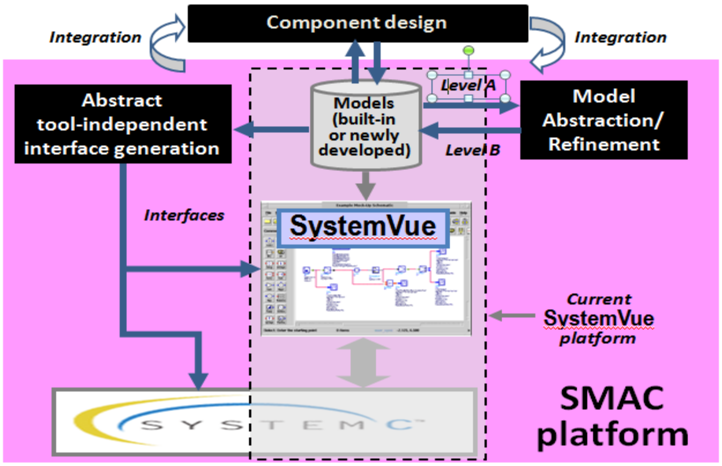
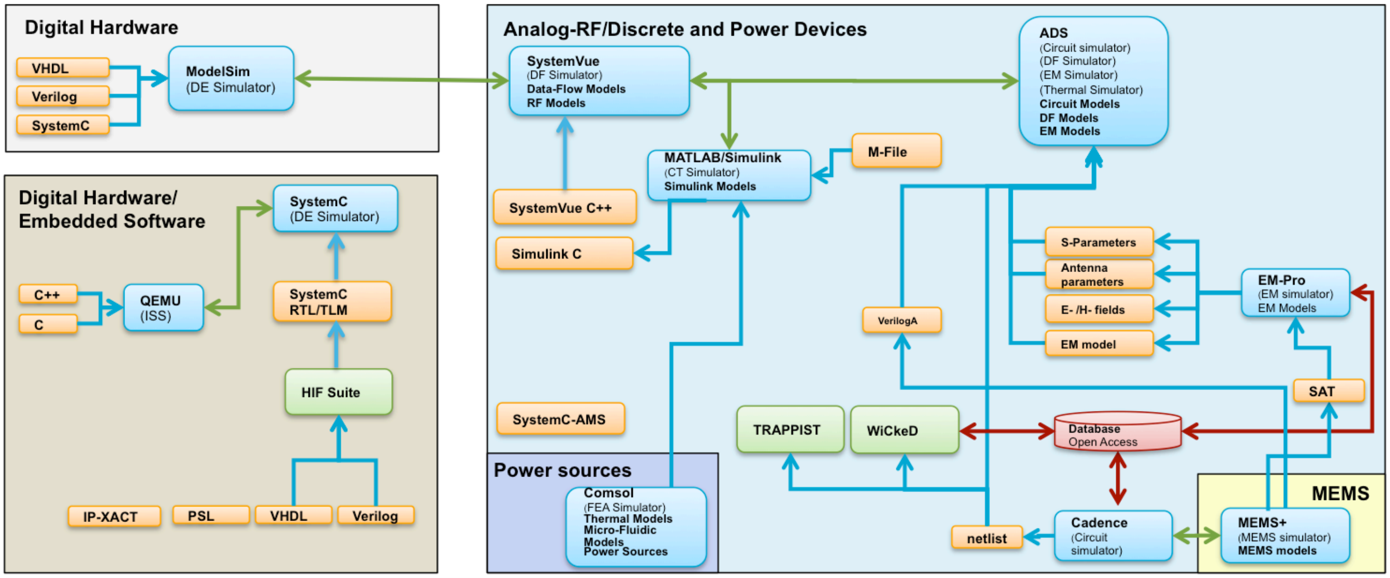
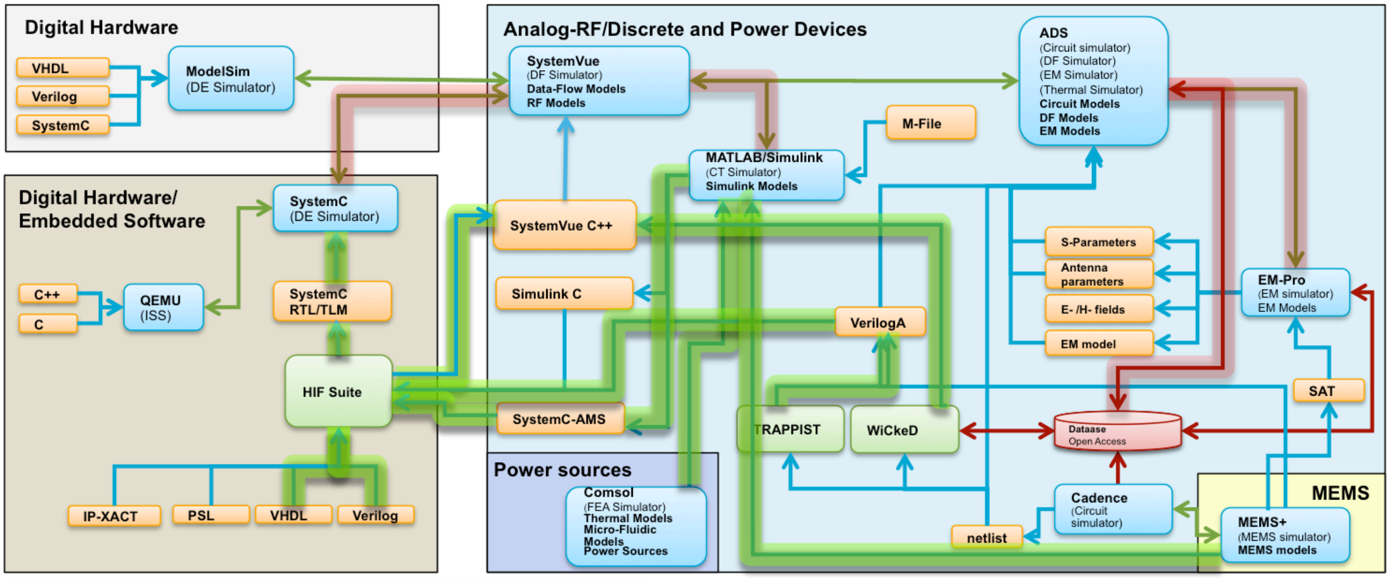


Figure 3. The SMAC platform: Organization and usage scenario.



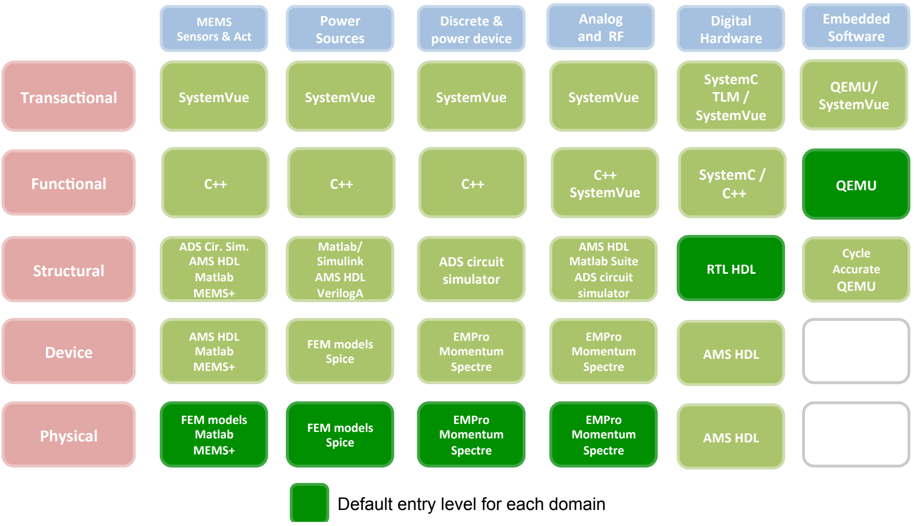
Light blue boxes indicate Simulation Tools
Light green boxes indicate Model Generation Tools
Light orange boxes indicate Modeling Formats
Light red cylinders indicate Database or external Libraries
Bidirectional green lines indicate Co-simulation Links
Unidirectional blue lines indicate Input / Output Models in specific format
Dotted unidirectional blue lines indicate Output Modeling Functions (extending models by parameterization)
Bidirectional red lines indicate access to a Database or external Libraries

Figure 4. Tool interface *before* the SMAC platform.



Light blue boxes indicate Simulation Tools
Light green boxes indicate Model Generation Tools
Light orange boxes indicate Modeling Formats
Light red cylinders indicate Database or external Libraries
Bidirectional green lines indicate Co-simulation Links
Unidirectional blue lines indicate Input / Output Models in specific format
Dotted unidirectional blue lines indicate Output Modeling Functions (extending models by parameterization)
Bidirectional red lines indicate access to a Database or external Libraries

Figure 5. Tool interface *after* the SMAC platform



	MEMS Sensors & Act	Power Sources	Discrete & power device	Analog and RF	Digital Hardware	Embedded Software
Transactional	SystemVue	SystemVue	SystemVue	SystemVue	SystemC TLM / SystemVue	QEMU/ SystemVue
Functional	C++	C++	C++	C++ SystemVue	SystemC / C++	🟩 QEMU
Structural	ADS Cir. Sim. AMS HDL Matlab MEMS+	Matlab/ Simulink AMS HDL VerilogA	ADS circuit simulator	AMS HDL Matlab Suite ADS circuit simulator	🟩 RTL HDL	Cycle Accurate QEMU
Device	AMS HDL Matlab MEMS+	FEM models Spice	EMPro Momentum Spectre	EMPro Momentum Spectre	AMS HDL	
Physical	🟩 FEM models Matlab MEMS+	🟩 FEM models Spice	🟩 EMPro Momentum Spectre	🟩 EMPro Momentum Spectre	AMS HDL	

🟩 Default entry level for each domain

Figure 6. Simulation-Levels / Design-Domains Matrix.

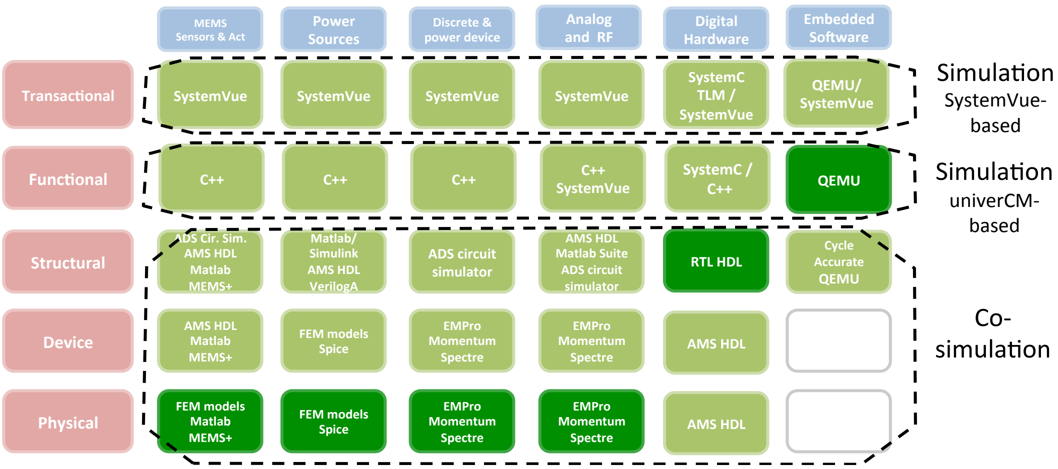
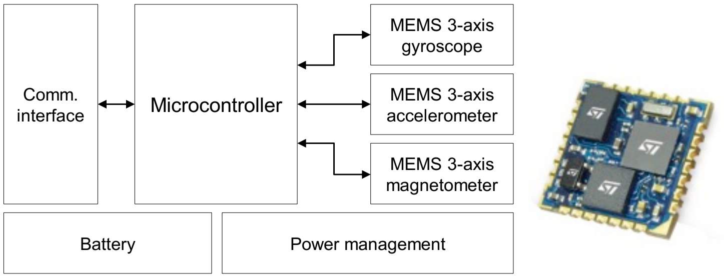


Figure 7. Simulation versus co-simulation.



(*a*) (*b*)

Figure 8. System partitioning of a sensor node (*a*) and an iNEMO M1 System-on-Board (*b*).

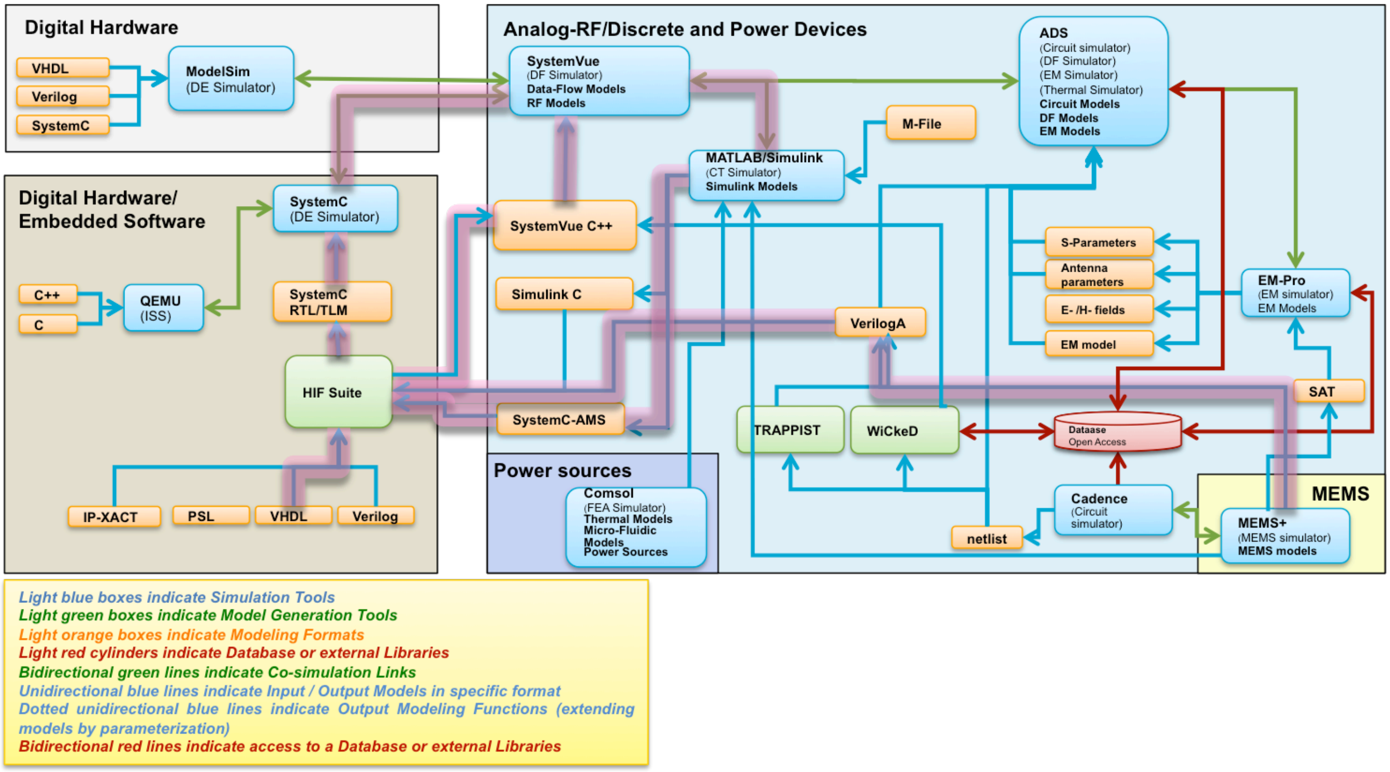


Figure 9. The SMAC platform flows and tools adopted in the limb tracking smart system design

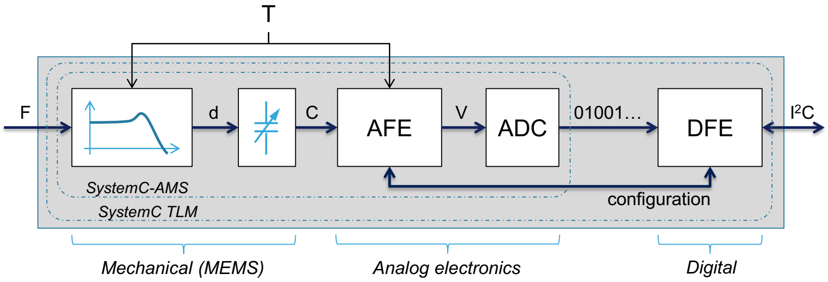


Figure 10. Accelerometer model for one axis.

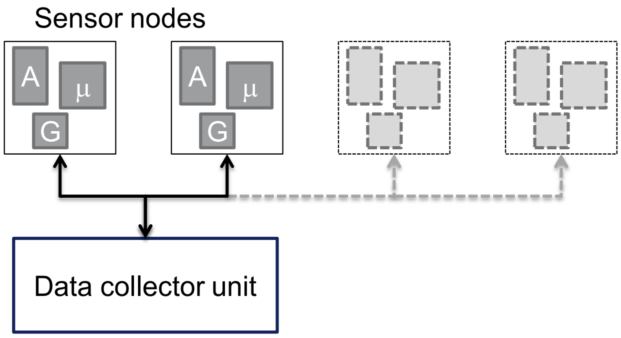
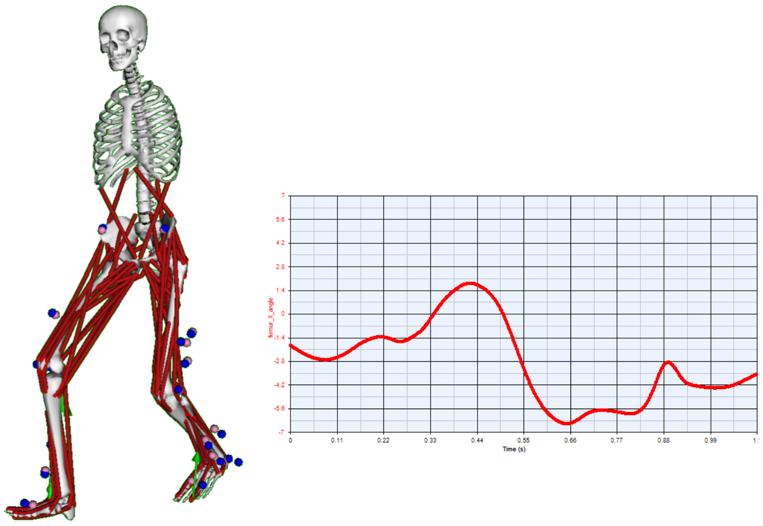


Figure 11. Block diagram of the complete system.
(A: accelerometer/magnetometer, G: gyroscope, µ: microcontroller).



(*a*) (*b*)

Figure 12. Human body representation in the OpenSim environment for leg position tracking (*a*) and simulation data for one of the right femur angles during walk (*b*).

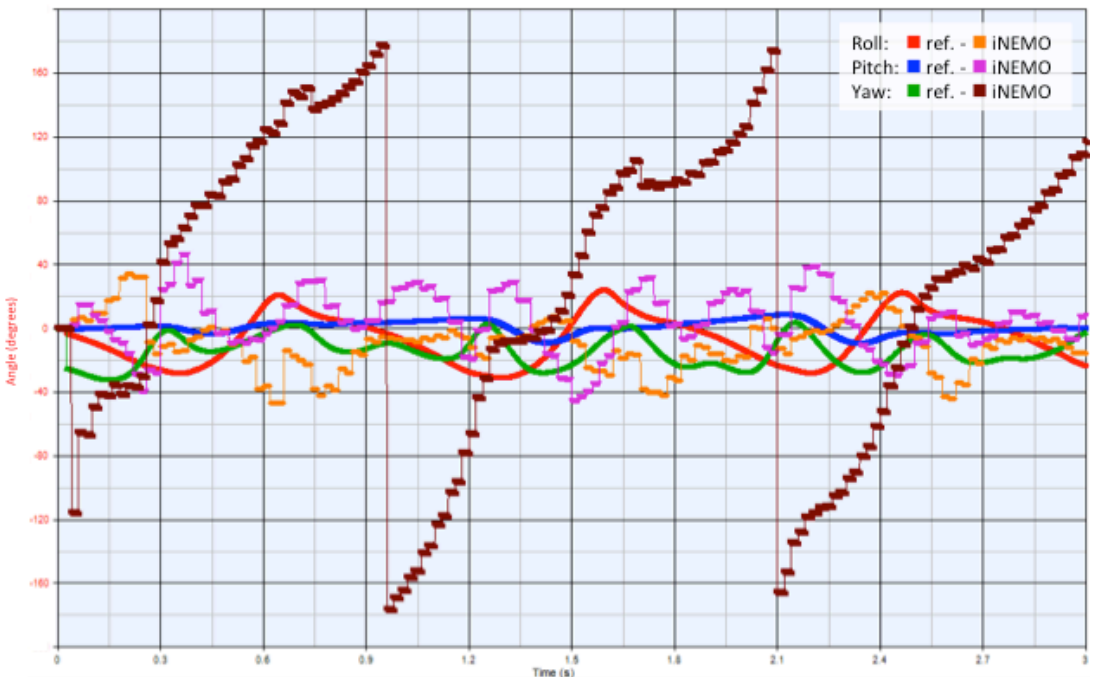


Figure 13. Euler angle estimation in dynamic condition (original algorithm).

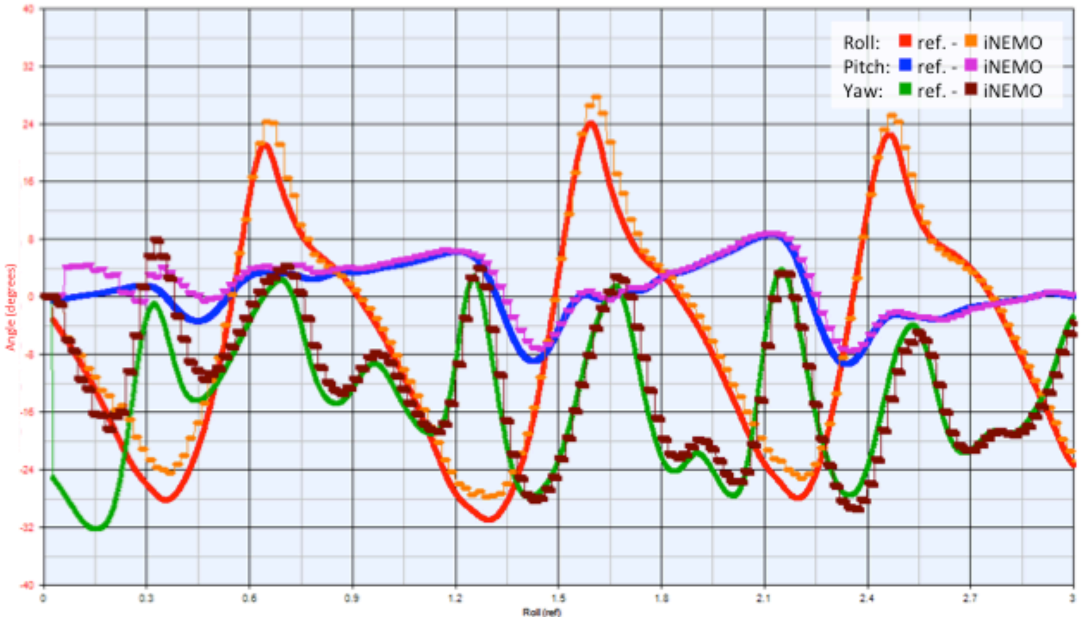


Figure 14. Euler angle estimation in dynamic condition (improved algorithm).

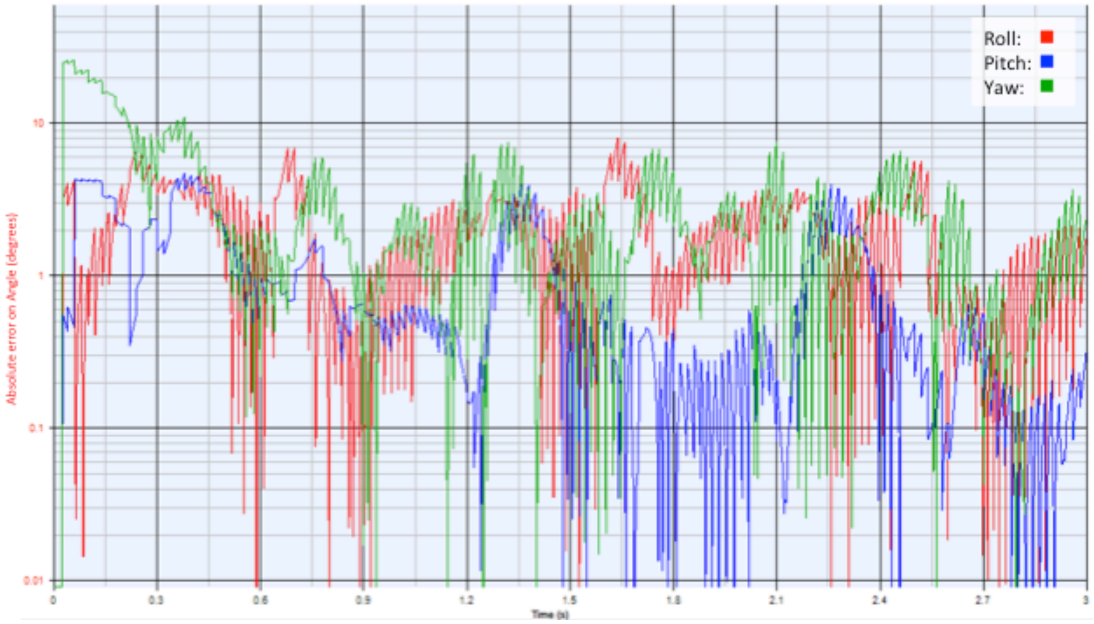


Figure 15. Error on Euler angle estimation (absolute value) in dynamic condition.