Energy-Aware Design of Embedded Software through Modelling and Simulation

José Antonio Esparza Isasa*, Peter Gorm Larsen* and Finn Overgaard Hansen†
Department of Engineering*, Aarhus School of Engineering†
Aarhus University
Aarhus, Denmark
{jaei,pgl}@eng.au.dk*, foh@iha.dk†

***Abstract*—We present a model-driven engineering approach that enables to take energy consumption into account during the development of embedded software. In this approach we address all the constituents of a typical modern embedded solution (mechanics, communication and computation subsystems) through the application of different modelling technologies. This makes it possible to evaluate the implications of different software and system architectures in the system's energy consumption. Additionally it facilitates the exploration of the design space without having to prototype each candidate solution. We also provide details on the application of this approach to the development of a medical grade compression stocking and the benefits this approach has brought to the project currently developing this system.**

I. Introduction

Modern embedded solutions are typically a combination of computing units, communication interfaces and mechanical subsystems and they can operate both autonomously or as part of a network. This makes embedded solutions heterogeneous systems that are very hard to design [1]. In addition many embedded systems are battery powered and therefore they present the added complexity of begin energy efficient while still fulfilling their operational requirements [2].

A possible tool to cope with complexity is the application of abstract modelling. Modelling can be used to represent the system at the highest level of abstraction. These abstract models can be progressively transformed into a concrete system realization [3]. This approach is known as model-driven engineering.

This paper presents a model-driven engineering approach to the design of these complex embedded solutions. This approach makes use of several modelling paradigms in order to represent different aspects of the system and makes special emphasis on the energy performance of different candidate solutions. The modelling activities proposed under this approach are conducted early in the development process and allow design space exploration without requiring physical prototyping. This reduces development time and cost and enables the repeatability of the experimental simulations. Additionally it provides a tool to design quality solutions and to make well-founded design decisions.

The reminder of this article is structured as follows: Section II describes the design approach to energy consumption proposed in this paper. Section III elaborates on HIL and system realization following this approach. Section IV describes how this technique is being applied to a case study and its preliminary results. Sections V and VI present future and related work. Finally, Section VII concludes the paper.

II. An holistic model-driven engineering approach to Embedded Systems design

The approach proposed in here aims at studying the energy consumption in the different subsystems that compose typical embedded solutions. A general representation of such a solution is presented in the SysML block diagram shown in Fig. 1 and its components described below:

- **Embedded Hardware:** represents the electronic hardware that supports the execution of software and possibly other components implementing additional logic in hardware.
- **Embedded Software:** represents the software that controls the operation of the rest of the components in the embedded solution.
- **Mechanics:** represents the mechanical components that are controlled by the system. Interfacing with the mechanical subsystem and the environment from the embedded side is conducted through sensors and actuators.
- **Communications Interface:** represents the communication hardware that makes it possible to establish links with other networked systems.

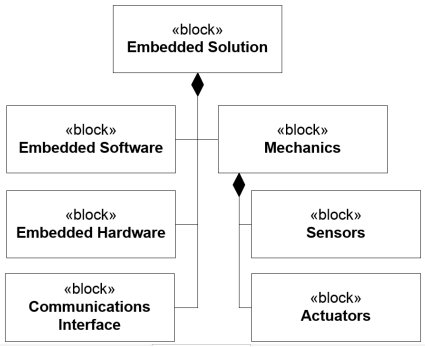

Fig. 1. SysML block representation of a typical embedded solution.

To design an embedded solution comprised by the components presented above so it satisfies its operational requirements is already a challenging task. To design it in

such a way that it is low energy consuming is even more complex. In order to cope with the complexity associated to the design of this kind of systems we proposed the application of model driven engineering techniques that apply System-Level (SL) modelling. SL modelling aims at describing the system under design at the highest level of abstraction and to incorporate progressively system details in order to conduct system analysis. Such a model is gradually transformed into a final implementation. We propose the application of SL design taking into consideration all the energy consuming components in an Embedded Solution in a joint design effort. Additionally we propose Hardware-In-the-Loop as a way to combine executable models with partial system realizations in software and/or hardware running on target. Our approach is said to be holistic because it takes into consideration subsystems that, even though could seem to be unrelated at first sight, they all have an impact in total energy consumption. Therefore, our approach targets the mechanical subsystem, the communication interfaces and the computation logic executed in the embedded software and hardware. All these aspects are addressed in different specific ways in order to obtain energy consumption estimates, that can be used to study the total energy consumption. Additional details on each specific strategy are provided below.

A. Modelling mechanical subsystems

In order to address the design of mechanical subsystems we apply a co-modelling approach in which the engineers use a modelling paradigm able to represent Continuous Time (CT) phenomena and a second one in order to represent Discrete Event (DE) logic [4]. Controlled physical processes (plant) are best represented by using CT abstractions such as differential equations. On the other hand, the control logic that operates the plant is best represented by using logical formalisms.

We proposed a particular way of using this co-modelling approach so it is possible to take into account energy consumption during the design of mechatronic systems [5]. An overview of this approach is presented in Fig. 2. We take as starting point a CT-first methodology [6], in which the modelling starts by focusing on the mechanics of the system and carrying out the control logic modelling afterwards. The CT models are focused on the core functionality that has to be delivered and do not capture heat dissipation due to the conversion efficiency of the electromechanical devices. In case part of the system operation depends on its internal temperature, this could be represented as part of the CT models, since it is a physical process with impact on energy consumption.

Once the system operation has been described, the notion of energy is incorporated into the CT models in a phase that is called models instrumentation. The notion of energy is incorporated only in the CT side since the energy consumption in the DE side is typically negligible if compared with the first one. This instrumentation consist on basically monitoring the variables that have an impact on energy consumption and doing it in a way that does not affect the performance of the models. After this phase it is possible to co-execute the models and produce a number of energy consumption estimates, that can be used to perform trade-off analysis between the different modelled solutions. In case the energy consumption estimations are fed back to the DE model it is possible to use this approach to model energy-aware system operation, meaning that the system can feature different operational models that are switched among depending on the energy consumed. In principle this approach could be applied by using any tool that supports the co-simulation of DE and CT models.

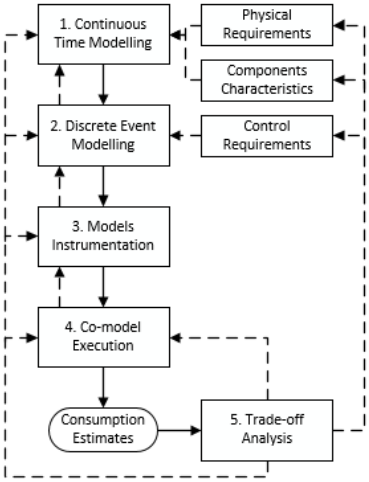

Fig. 2. Overview of the co-simulation based methodology.

In this work we have used the co-execution environment DESTECS/Crescendo [7], [8]. This environment combines the tools Overture [9] and 20-Sim[1]. Overture incorporates an interpreter for the DE modelling language VDM-RT [10]. This language is best suited to represent the control logic supervising the mechanical components and therefore it is used for DE modeling. 20-Sim incorporates a numerical engine that evaluates differential equations. Additionally it supports abstractions built on top of differential equations in the form of bond and block diagrams. This tool is best suited to represent the mechanical side of the system (CT side). The DESTECS environment synchronizes the co-execution of the models providing a common notion of time for both models. Additionally it allows the specification of a number of controlled and monitored variables between the VDM-RT and the 20-sim simulations so the supervision of the plant is possible. Additional details on how DESTECS and the process presented above have been applied in a concrete case study will be provided in Section IV.

B. Modelling computation subsystems

Modern microcontrollers can switch between different operational modes in order to reduce energy consumption. These operational modes temporarily disable the CPU and certain peripherals in order to achieve such a reduction. Switching

[1]20-Sim official website: `http://www.20sim.com`

between operational modes takes time that might have an impact in the real-time performance of the system under study. An example of different power modes can be seen in Fig. 3. In this case the processor features two low power operational states: Hibernate and Sleep. Sleep allows a fast CPU wake-up based on internally or externally generated events. Hibernate requires additional time to wake-up the CPU and it can react only on external events. However it can lower the energy consumption even further. The CPU controls from software how to switch between these modes.

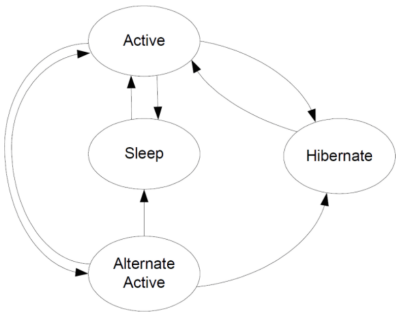

Fig. 3. CPU states in an implementation of a Cortex M3 ARM processor.

In order to explore the application of the different sleep modes, software strategies and architectures required for low power operation we propose the application of the modelling language VDM-RT. This language incorporates the abstraction **`CPU`** that represents an execution environment in which parts of the model can be deployed. Besides representing the computational support it also incorporates a real-time operating system layer. Logic running in VDM-RT CPUs can represent single or multi-threaded software implementations as well as dedicated hardware blocks depending on how they are configured [11]. However, VDM-RT CPUs do not feature the notion of low power consuming states since they are always active and therefore able to perform computations. An initial approach to overcome this limitation was a design pattern structure that regulated the access to the CPU as a resource by the logic running on it depending on the state of a flag [12].

As a way to overcome this situation we have extended the VDM-RT language by adding two new constructs to manage CPU states: **`CPU.sleep()`** and **`CPU.active()`** [13]. The addition of these two new constructs implied the modification of the VDM-RT interpreter and the scheduler built into the Overture platform. In addition to these extensions we proposed specific ways to use the **`sleep`** and **`active`** operation so the models can represent accurately the low power operation of real platforms. These templates show how to model a CPU wake-up based on an interrupt triggered by externally generated events and based on internal sleep timers. Model simulation produces a log file that registers in which states the CPU has been operating and for how long. This information together with the electrical characteristics of the CPU under consideration allows to represent the power consumption of the device over time. The integration of this curve over time results on the total energy consumption on the computational side.

The application of a modelling-based approach to the computational side of the system brings a number of advantages to the design of the software. Besides the obvious case of exploring different sleeping policies for a single CPU without having to prototype them, more complex cases in which several CPUs are involved can be explored. This is especially relevant if the CPUs have to communicate in order to satisfy system requirements.

C. Modelling communication subsystems

The VDM-RT modelling language incorporates the abstraction **`BUS`** that allows to communicate the **`CPU`** processing nodes introduced in the previous section. This abstraction can be used to represent point-to-point communication between CPUs in a static way. Communication performed over VDM-RT BUSes is assumed to be error-less, so any kind of communication problems such as information (packet) loss has to be modelled on top. At this point the BUS abstraction does not incorporate any notion of energy consumption during communication.

1) Modelling network topologies: We propose the application of a design pattern structure to overcome some of these limitations. Our initial approach takes as an example the communication in a wireless context but it could easily be extended to a different one. In Fig. 4 this structure is presented through a UML diagram. The general idea behind this pattern is to create a star topology network in which each networked embedded system is connected to a central component, that runs a simulated transmission medium. This structure is applied in VDM-RT by using CPUs to represent each networked device as well as a central component simulating the medium. Finally buses are communicating each device model with the medium model. In this way there is no direct connection between the individual CPUs representing the devices and any transmission goes through the simulated wireless medium. By using some of the VDM abstractions one can easily establish relations between the CPUs in the simulated wireless medium to represent whether a communication between two nodes is possible or not. Analyzing these "connection maps" during model execution is especially easy due to the expressiveness of the VDM language and it can be accomplished by using map comprehensions. If model simulation time is a concern the connection maps can be translated into a look-up file in which it is explicitly stated the relation between all the networked elements. This structure solves the initial problem of representing a realistic topology of a small scale embedded network in VDM-RT.

2) Introducing the notion of energy consumption: In order to represent the energy consumption we focus on modelling the operational state of the communications interface of the embedded device. We consider operational states the different modes in which the communication interface can be working, typically: transmitting, receiving or deactivated. For each mode the manufacturer provides an average power consumption

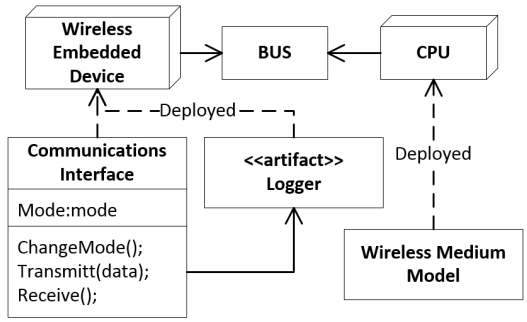

Fig. 4. Design pattern structure to represent wireless communication.

figure that can be used in the VDM-RT models. Changes among these operational modes are logged during model simulation and analyzed when it has been completed. Based on the transitions between the states and for how long the device has stayed on those states one can calculate the evolution of the power consumption over time and hence the total energy consumption during system operation.

Once the notion of energy consumption and the possibility of modelling different network topologies have been facilitated, it is possible to conduct the analysis of communication related problems. Some of these include but are not limited to, routing algorithms, network services, latencies or time synchronization between nodes. All these factors could be analyzed against energy consumption in order to get estimates that would allow an energy aware design of the communication subsystem, including communication software as well as, to some extent, hardware.

One of the advantages of using this structure is the clear separation between the connection map representing the network topology and conditions and the individual networked elements, even though the simulation of both is conducted in the same modelling environment. The main disadvantage of this approach at first sight are the limitations regarding the number of networked elements. We consider this approach valid only for small scale networks. However additional work is necessary to establish its practical limitations.

The approach to communications modelling proposed in this section is conducted only in VDM-RT without involving any other modelling paradigm. A co-simulation approach could be interesting to represent mobile communication nodes or a changes in the environment in which the network is deployed.

III. System realization and Hardware In the Loop

The approach presented above aims at tackling the design problems through modelling and simulation however, at some point, the system has to be realized. Given the fact that a strong emphasis has been placed on the modelling of the system it is desirable that the models created are used as much as possible during the system realization phase. This could include the combination of partial system realizations with models, allowing the co-execution of models with system realizations. The approach that we propose in this work is exemplified in Fig 5. In this diagram we show an initial VDM model of a system that executes a three phases algorithm in which data is acquired, processed and finally an output is provided.

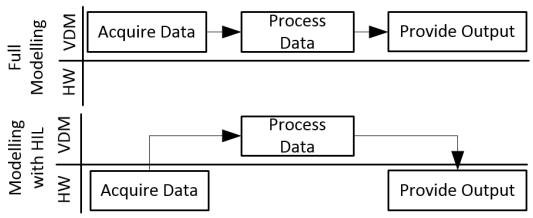

Fig. 5. Overview of the co-simulation based methodology.

We have applied this principle allowing the combination of VDM executable models running in a Workstation with actual components implemented in a Device Under Test (DUT) [14]. These components can be both hardware and software components. An overview of this Hardware In the Loop setup is presented in Fig. 6. In addition to the components mentioned above the system incorporates a Stimuli Provider able to simulate external inputs and a Logic Analyzer able to monitor the evolution of different logical signals. The VDM execution environment is able to interface the Logic Analyzer that can measure the time it takes to execute system realizations running on the DUT. This time figures can be manually incorporated into the VDM model and therefore increasing the fidelity of the model simulation results.

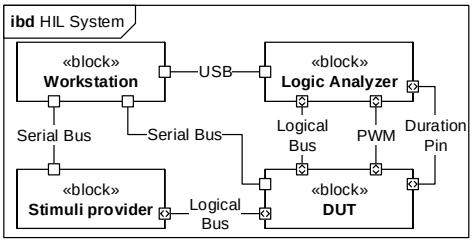

Fig. 6. SysML Internal Block Diagram showing the hardware connections to the DUT.

IV. Application and Preliminary Results

The approach proposed in this work is applied to the development of an intelligent compression stocking to treat leg venous insufficiency. This stocking is shown in Fig. 7 and it is composed of: an inner stocking (1), an inflatable stocking responsible for delivering the required compression levels (2), a pneumatical circuit composed of valves, pumps and a manometer (3), and an embedded system implementing the control logic and interfacing hardware and integrating a Bluetooth-based communication interface (4). This portable device is battery-operated and it is required to work for at least 14 hours. A complete description of this device can be found in [15]. As it is explained above this device is

composed of mechanical, computational and communication subsystems that have to be energy efficient so the device autonomy requirements can be met.

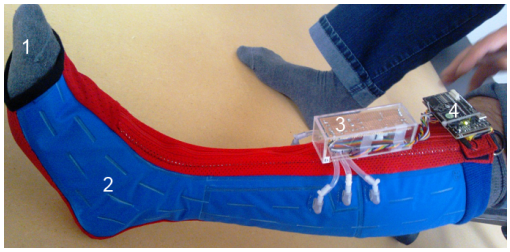

Fig. 7. The medical grade compression stocking.

A. Modelling of the Compression Principle

In order to study the mechanical subsystem we have applied the modelling process presented in Section II-A. The modelling of the mechanical system by itself was already beneficial for the project since it allowed us to gain a thorough understanding of the pneumatics and the physics behind the compression principle. Based on these models we were able to determine that a certain compression strategy was not feasible without having to prototype it. Additionally we were able to model different control software in VDM-RT and co-simulate its performance together with the mechanical models.

The analysis on both control strategies and mechanical pneumatic configurations, provided a number of energy consumption estimates that helped on deciding which system configuration was optimal. These suggestions had an impact on the system realization and introduced improvements on the software that increased its energy efficiency.

B. Modelling of the Embedded Software

We have applied the modelling techniques presented in Section II-B in order to explore two different embedded architectures in a concrete scenario: the regulation. The air pressure level in the air bladders have to be monitored periodically and kept at certain levels so proper compression is delivered to the limb. Through the regulation process the controller reads a manometer, compares the value retrieved against the expected one and depending on this triggers the pump or vents the bladders accordingly. This logic is implemented as a software component and requires the CPU to be active. However and depending on the kind of sensors that are used the CPU can be sleeping for a longer period of time. We have used VDM-RT modelling to study the energy consumption of two different kind of sensor configurations: the first one uses smart sensors that wake up the CPU in case an overpressure event occurs and the second one uses passive sensors that require a poll from the CPU in order to provide a reading. In the first case the CPU presented a lower power consumption than in the second case, since the sensors could run independently from the CPU. In the second case the CPU power consumption was higher because it required periodic wake-ups in order to check the sensors, which were not running independently. These results were expected since this was a simple case, however the purpose of applying the technique in this case was to show the modelling principle in a simple case study.

The predictions provided by these simulations where confirmed by measurements conducted on system realizations for both architectures with a fidelity of up to 95% [13].

C. Modelling of Communication

The modelling of the communication system remains as future work. A complete overview of the different communication scenarios in which this device can operate is presented in [16]. The intention is to model the critical scenarios in which the device is running on batteries. Based on these models we aim at making energy consumption estimations and evaluating computation vs. communication trade-offs.

V. Future work

We are planning to extend the work presented in here so the energy consumption analysis of the communication is also possible. Additionally we are planning to apply the analysis of energy consumption in computation in a more complex situation and possibly combining it with the communication, therefore being able to represent and analyze computation vs. communication trade-offs. We are also aiming at applying some of the modelling techniques presented in here to a second system so it is possible to make a stronger case for the SL energy-aware design approach for embedded solutions.

VI. Related work

The energy consumption problem in todays embedded solutions is well recognized and one of the top research priorities [2]. System Level design is also a well established technique especially in the hardware world, that it is seeing its expansion to other non-computing domains [3]. However and to our knowledge the application of System Level design with the particular intention of addressing the energy consumption problem during the development process through modelling and prototyping has not been formulated previously. Even though energy consumption in all subsystems has not been addressed in a single design effort previously significant work has been conducted in the individual fields of energy consumption in computation, communication and mechatronic systems.

Regarding energy consumption in computation, extensive work has been carried out in order to characterized different layers of abstraction. Some authors propose very accurate characterization of concrete computing platforms by taking into account energy consumption at the micro-architectural and the instruction level [17], [18]. This differs from the work presented in here in the fact that we consider an average power consumption figure within the active state of the CPU in order to obtain a coarse grained estimation over time. Other authors focus on the characterization of energy consumption at the service level [19]. In this case the authors consider the energy consumption at the OS level when the CPU is active. In our work we consider a single energy consumption figure for all

the services provided by the OS, however, in case some of the OS operations result in a longer time having the CPU active, this will be considered under our approach as well even though the services have not been individually characterized. A more comprehensive review of techniques to study power consumption in computation can be found in [20].

As in the computation case, modelling of communication has been conducted at different levels of abstraction, ranging from energy consumption at the communications interface level to higher layers such as routing or application [21], [22]. Our work makes use of more simple power consumption models that, even though they are based on fixed average power consumption estimates are expected to provide sufficient detail to enable trade-off analysis of network algorithms.

Energy consumption in mechatronic systems is typically addressed as a particular application of well-established modelling platforms such as Matlab, Ptolomy or Modellica. Energy consumption has been typically considered just as any other design factor of industrial grade equipment and mechatronic components have not been traditionally considered together with embedded devices. However this situation is changing due to the increasing relevance of Cyber-Physical Systems [2].

VII. Conclusions

This paper has presented a modelling approach to energy-aware design of embedded systems. The preliminary application of this approach to a case study has enabled the exploration of different control algorithms, different hardware and software architectures and different mechanical configurations. This has made it possible to evaluate system performance against energy consumption early during the development process without needing a physical prototype. This work will be complemented in the near future with a study of energy consumption from the communication point of view. A more in-depth description of this work can be found in [23].

We hope that the approach proposed can inspire other researchers working with modelling applied to embedded system development and, to some extent, enact the application of modelling in the design of real embedded solutions.

References

[1] Thomas A. Henzinger and Joseph Sifakis, "The Embedded Systems Design Challenge," in *FM 2006: Formal Methods, 14th International Symposium on Formal Methods, Hamilton, Canada, August 21-27, 2006, Proceedings*, 2006, pp. 1–15.

[2] Banerjee, A. and Venkatasubramanian, K.K. and Mukherjee, T. and Gupta, S. K S, "Ensuring Safety, Security, and Sustainability of Mission-Critical Cyber-Physical Systems," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 283–299, 2012.

[3] S. K. Gupta, T. Mukherjee, G. Varsamopoulos, and A. Banerjee, "Research Directions in Energy-Sustainable CyberPhysical Systems," *Sustainable Computing: Informatics and Systems*, vol. 1, no. 1, pp. 57 – 74, 2011.

[4] J. Fitzgerald, K. Pierce, and P. G. Larsen, *Industry and Research Perspectives on Embedded System Design*. IGI Global, 2014, ch. Collaborative Development of Dependable Cyber-Physical Systems by Co-modelling and Co-simulation.

[5] J. A. E. Isasa, F. O. Hansen, and P. G. Larsen, "Embedded Systems Energy Consumption Analysis Through Co-modelling and Simulation," in *Proceedings of the International Conference on Modeling and Simulation, ICMS 2013*. World Academy of Science, Engineering and Technology, June 2013.

[6] J. F. Broenink and Y. Ni, "Model-Driven Robot-Software Design using Integrated Models and Co-Simulation," in *Proceedings of SAMOS XII*, J. McAllister and S. Bhattacharyya, Eds., jul 2012, pp. 339 – 344.

[7] J. F. Broenink, P. G. Larsen, M. Verhoef, C. Kleijn, D. Jovanovic, K. Pierce, and W. F., "Design Support and Tooling for Dependable Embedded Control Software," in *Proceedings of Serene 2010 International Workshop on Software Engineering for Resilient Systems*. ACM, April 2010, pp. 77–82.

[8] J. Fitzgerald, P. G. Larsen, and M. Verhoef, Eds., *Collaborative Design for Embedded Systems – Co-modelling and Co-simulation*. Springer, 2014.

[9] P. G. Larsen, N. Battle, M. Ferreira, J. Fitzgerald, K. Lausdahl, and M. Verhoef, "The Overture Initiative – Integrating Tools for VDM," *SIGSOFT Softw. Eng. Notes*, vol. 35, no. 1, pp. 1–6, January 2010. [Online]. Available: http://doi.acm.org/10.1145/1668862.1668864

[10] K. Lausdahl, P. G. Larsen, and N. Battle, "A Deterministic Interpreter Simulating A Distributed real time system using VDM," in *Proceedings of the 13th international conference on Formal methods and software engineering*, ser. Lecture Notes in Computer Science, S. Qin and Z. Qiu, Eds., vol. 6991. Berlin, Heidelberg: Springer-Verlag, October 2011, pp. 179–194, ISBN 978-3-642-24558-9. [Online]. Available: http://dl.acm.org/citation.cfm?id=2075089.2075107

[11] J. A. E. Isasa, P. G. Larsen, and K. Bjerge, "Supporting the Partitioning Process in Hardware/Software Co-design with VDM-RT," in *Proceedings of the 10th Overture Workshop 2012*, ser. School of Computing Science, Newcastle University, 2012.

[12] J. A. E. Isasa and P. G. Larsen, "Modelling Different CPU Power States in VDM-RT," in *Proceedings of the 11th Overture Workshop 2013*, ser. Aarhus University, June 2013.

[13] J. A. E. Isasa, P. W. Jørgensen, and C. Ballegaard, "Modelling Energy Consumption in Embedded Systems with VDM-RT," in *Proceedings of the 4th International ABZ conference.*, July 2014.

[14] J. A. E. Isasa, P. W. Jørgensen, and P. G. Larsen, "Hardware In the Loop for VDM-Real Time Modelling of Embedded Systems," in *MODELSWARD 2014, Second International Conference on Model-Driven Engineering and Software Development*, January 2014.

[15] T. F. Jensen, F. O. Hansen, and J. A. E. et al., "ICT-Enabled Medical Compression Stocking for Treatment of Leg-Venous Insufficiency." in *International Conference on Biomedical Electronics and Devices (BIODEVICES 2014)*, March 2014.

[16] F. O. Hansen, T. F. Jensen, and J. A. Esparza, "Distributed ICT Architecture for Developing, Configuring and Monitoring Mobile Embedded Healthcare Systems." in *International Conference on Health Informatics (HEALTHINF 2014)*, March 2014.

[17] Mostafa E.A. Ibrahim and Markus Rupp and Hossam A. H. Fahmy, "A Precise High-Level Power Consumption Model for Embedded Systems Software," *EURASIP Journal on Embedded Systems*, vol. Volume 2011, no. 1, January 2011.

[18] Sheayun Lee and Andreas Ermedahl and Sang Lyul Min, "An Accurate Instruction-Level Energy Consumption Model for Embedded RISC Processors," 2001.

[19] B. Ouni, C. Belleudy, and E. Senn, "Accurate energy characterization of os services in embedded systems," *EURASIP Journal on Embedded Systems*, vol. 2012, no. 1, p. 6, 2012.

[20] Unsal, O.S. and Koren, I., "System-Level Power-aware Design Techniques in Real-Time Systems," *Proceedings of the IEEE*, vol. 91, no. 7, pp. 1055–1069, 2003.

[21] E. Shih, S.-H. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '01. New York, NY, USA: ACM, 2001, pp. 272–287.

[22] R. Min and A. Chandrakasan, "A framework for energy-scalable communication in high-density wireless networks," in *Proceedings of the 2002 International Symposium on Low Power Electronics and Design*, ser. ISLPED '02. New York, NY, USA: ACM, 2002, pp. 36–41.

[23] J. A. E. Isasa, "System-Level Energy Aware Design of Cyber-Physical Systems," Department of Engineering, Aarhus University, Finlandsgade 22, Aarnus N, 8200, Tech. Rep., October 2013, available on-line at http://eng.au.dk/fileadmin/DJF/ENG/PDF-filer/Tekniske_rapporter/ECE-TC-16-samlet.pdf.