Early Power-aware Design Space Exploration for Embedded Systems: MPEG-2 Case Study

Feriel Ben Abdallah*, Chiraz Trabelsi †, Rabie Ben Atitallah † and Mourad Abed†
*Institut Mines-Telecom, Telecom ParisTech, France
Email: feriel.benabdallah@telecom-paristech.fr
†LAMIH, University of Valenciennes, France
Email:(chiraz.trabelsi, rabie.benatitallah, Mourad.Abed)@univ-valenciennes.fr

***Abstract*—Power consumption has become one of the major concerns in embedded systems design, especially for mobile devices, which integrate many applications leading to a high power consumption. In this context, designers have the challenge to identify power issues early in the design flow and to explore the largest possible space of power-efficient solutions. In this paper, we present a Model Driven Engineering (MDE) approach for early power-aware Design Space Exploration (DSE). This approach is based on a high-level modeling of power estimation and dynamic management aspects targeting an automatic generation of the corresponding simulation code. It was implemented in the DSE toolkit TTool by integrating power concepts in its DIPLODOCUS UML profile. The existing C++ simulation code generator was extended in order to integrate power estimation. The main objective of this article is to illustrate the potential of our approach through an MPEG-2 case study. The proposed high-level power modeling concepts were used to target two different platforms for the implementation of an MPEG-2 decoding application. The processor power estimates obtained from simulations were compared to real board measurements. This comparison showed that our MDE approach is capable of obtaining results that can be used to make early power-efficient design decisions.**

I. Introduction

Power consumption is a critical design constraint particularly in embedded devices as a high power consumption reduces the lifetime of battery-operated systems and increases heat dissipation. To address this issue, many power reduction policies have been proposed in the literature. For example, Dynamic Power Management (DPM) [1] policy reduces the power consumption by putting electronic components into power-efficient modes during idle time. The drawback of using this policy is that transitions from idle to running state requires an overhead of time and power to start an incoming task. The Dynamic Voltage/Frequency Scaling (DVFS) policy [2] adjusts at runtime the operating frequency as well as the voltage of electronic components with respect to changing workloads, real-time constraints, etc. One critical problem is that reliable results on power saving policies can only be obtained at low design stages, which might be too late for design consideration [3].

In order to make early power-efficient design decisions, high-level power estimation and optimization would be extremely beneficial. In this context, Model Driven Engineering (MDE) [4] is an interesting approach for power estimation at a high level of abstraction [5]. This methodology allows designers to use models to specify the system functionality and architecture instead of the use of a programming language defining how the system is implemented. MDE is based essentially on metamodels, models and model transformations. It aims at constructing an abstraction layer to avoid dealing with implementation details and to allow model reuse. One of the significant languages used in MDE is the Unified Modeling Language (UML) which is a standard promoted by the OMG (Object Management Group) [6]. A UML profile is a set of stereotypes that add specific information to a UML model in order to describe a system related to a specific domain. Several UML profiles target embedded systems design such as SysML (System Modeling Language) [7] and MARTE (Modeling and Analysis of Real-Time and Embedded systems) [8]. These UML profiles have proved their capability of modeling complex Systems-on-Chip (SoCs) containing both software and hardware components, and many design tools have been developed basing on them [8]. Among UML-based design tools for embedded systems, we cite TTool [9], which supports the UML profile DIPLODOCUS introduced a few years ago to address Design Space Exploration (DSE) for SoC.

Our MDE-based approach [10] aims at enabling designers to make early power-efficient decisions. This approach allows to move from a high-level modeling of power estimation and optimization aspects to an automatic code generation for fast power-aware functional simulations. The implementation of this approach was in the TTool environment by extending the DIPLODOCUS UML profile in order to integrate power aspects. Using the generated functional simulation code, the processor power consumption related to a given application can be estimated early in the design flow. These estimates are then used for DSE in order to make efficient decisions about the system configuration and the used power management technique.

This paper evaluates the efficiency of our approach through an MPEG-2 case study targeting two different platforms. The processor power estimates obtained from simulations were compared to real board measurements. This comparison showed that our MDE approach is capable of obtaining results that can be used to take adequate decisions for an early DSE.

The rest of this paper is organized as follows: Section 2 presents research works related to high level power evaluation techniques. Section 3 describes our power-aware modeling approach for the MPEG-2 case study. Section 4 presents the power results given by the generated functional simulation code and compares them to hardware measurements.

978-1-4799-6890-9/14/$31.00

II. Related work

In this section, we present Computer Aided Design (CAD) tools based on MDE approaches and taking into account the system power consumption at early stage in the design flow.

STORM (Simulation TOol for Real-time Multiprocessor Scheduling) [11] is a CAD tool that allows performance and power consumption evaluation for real-time multi-processor systems. This simulation engine allows the integration of different power reduction techniques (eg. DPM and DVFS, etc.) [12]. The inputs of this tool is the specifications of the hardware and software architectures together with the scheduling policy. In [13], a MDE approach was proposed to generate automatically the STORM inputs. This approach was based on three focal concepts: AADL Modeling, code transformation from AADL to STORM and OS services power/energy evaluation. The power management techniques integrated in STROM are implemented by the OS scheduler. In our work, we propose a modeling approach that can be used to target different implementations of power management techniques (using an OS, a hardware module, etc.). The Gaspard2 [14] environment for MPSoC design, based on the MARTE standard UML profile was adopted in [15] to design power estimators for hardware component models. In this approach, SystemC code (at cycle-accurate bit-accurate level) is automatically generated and used to estimate the power consumption during simulation. Since this approach is built around an instruction set simulator (ISS), it is considered as a slow technique for large DSE. PETS (Power Estimation Tool at System-Level) [16], developed in the context of the OpenPeople project, is another tool that is based on ISS for power estimation at the transactional level. This abstraction level allows faster simulations than the cycle-accurate level. However, it still requires micro-architectural activities (captured using ISS simulations) to estimate power, which is not adequate for a functional simulation as we propose in our work. This tool is based on manually written SystemC simulations and does not offer high-level modeling and code generation.

CAT (consumption Analysis Toolbox), developed in the context of SPICES project, proposes a MDE approach for system-level power and energy consumption estimation using AADL [17]. It provides power characterization of hardware components (processors, buses, memories, hardware accelerators, etc.) [18] and software components (eg. OS services) using the FLPA (Functional Level Power Analysis) [19] methodology. The main limitation of CAT and Gaspard tools is that they do not take into account the dynamic behavior of the system for power/energy optimizations, which is the main concern of our paper.

Arpinen et al. [20] proposed extensions to the MARTE UML profile for dynamic power management. MARTE defines some features to characterize power consumption in embedded systems related to power supply, battery, etc. Arpinen et al. added features allowing the designer to model dynamic power management policies basing on Finite State Machines (FSM). This approach is still at the conceptual level, and no analysis tools have been developed yet to evaluate power dissipation.

On the other hand, DIPLODOCUS/TTool is an advanced DSE environment which considers only functional aspects of embedded systems. This environment lacks the concepts of power consumption definition and analysis. Our MDE-based power-aware approach proposes extensions to DIPLODOCUS/TTool in order to integrate power consumption aspects. The extended tool allows to model CPU power-related characteristics (voltage, frequency, etc.) and dynamic power management policies. The generated functional simulation code offers a fast estimation of the power consumption of a system taking into account dynamic power management. Simulation results for different system configurations can be then used for power-aware DSE.

III. Power-aware modeling of the MPEG-2 case study

In this section, we present the application of our power-aware modeling approach for the MPEG-2 case study using the extended DIPLODOCUS UML profile. This section is organized as follows. First, the case study application is presented. Second, the DIPLODOCUS/TTool system modeling is described for the MPEG-2 case study. Finally, the introduced power concepts to the DIPLODOCUS profile are presented.

A. Brief Overview of the MPEG-2 decoder

An MPEG-2 video stream consists of three frame types: I frame (Intra-coded), P-frame (Predictive-coded) and B-frame (Bi-directionally coded). I frames can be decoded without knowing anything about previous frames (it is the case of the first frame of the video sequence). P frames are decoded using information from previous frames. B frames requires both the previous and the next frames in order to be decoded. To decode each frame several steps are needed: (1) Parsing, (2) Variable Length Decoding, (3) Inverse Quantization, (4) Inverse Discrete Cosine Transformation (IDCT), (5) Reconstruction, (6) Dithering and display. These steps can be divided into steps whose computational complexity depend on the frame type (the first 5 steps) and steps whose computational complexity depend only on he frame size and the frame rate in a given video sequence (the last step). These two types will be called the Frame-Dependent (FD) and Frame-independent (FI) parts in the rest of this paper.

One of the objectives of this paper is to compare the DSE results provided by our modeling approach with the results obtained by hardware experiments. In our case, we propose to validate our approach by comparing it to the work in [21]. The interesting aspect about this work is that it proposes a DVFS technique (Frame-based DVFS), which was applied for two different hardware platforms and significantly reduced the CPU power consumption of the MPEG-2 decoder by more than 50%. Our objective is to compare the results provided by our modeling approach with the results obtained in their hardware experiments. Therefore, our modeling approach will target the same platforms: a low performance StrongARM-1100-based evaluation board from Intel and a high performance XScale-based testbed platform.

B. System modeling using the DIPLODOCUS profile

This profile targets DSE of embedded systems at a high level of abstraction. It separates application and architecture modeling. Its abstractions allow fast simulation and formal analysis techniques. The main objective of this UML profile is to help designers to spot suitable design solutions regarding functionality, performance, silicon area, power consumption metrics, etc. DIPLODOCUS system modeling is based on

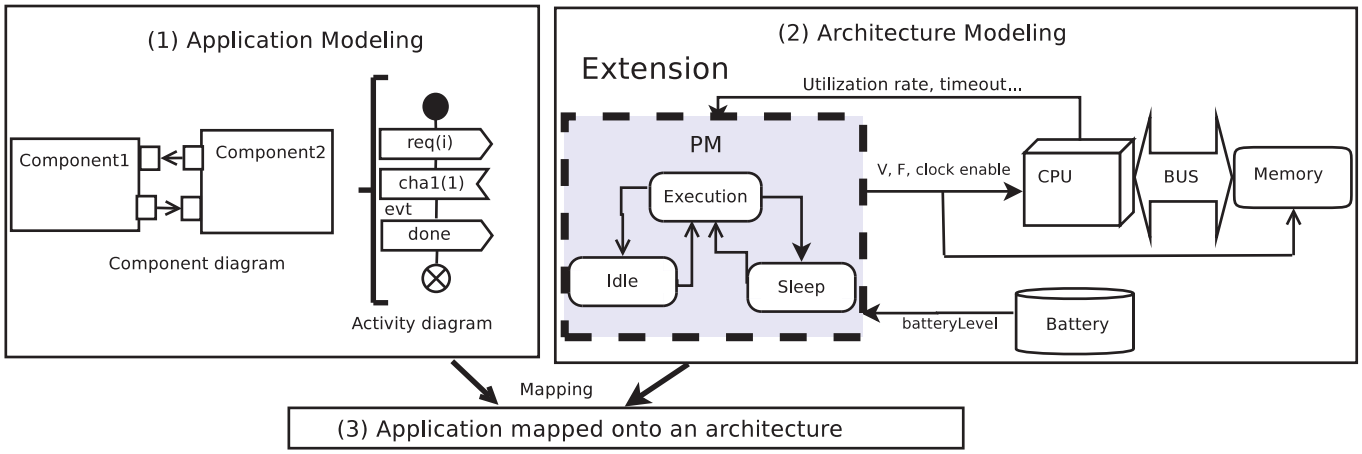

Fig. 1: Overview of system modeling using DIPLODOCUS with integrated power management

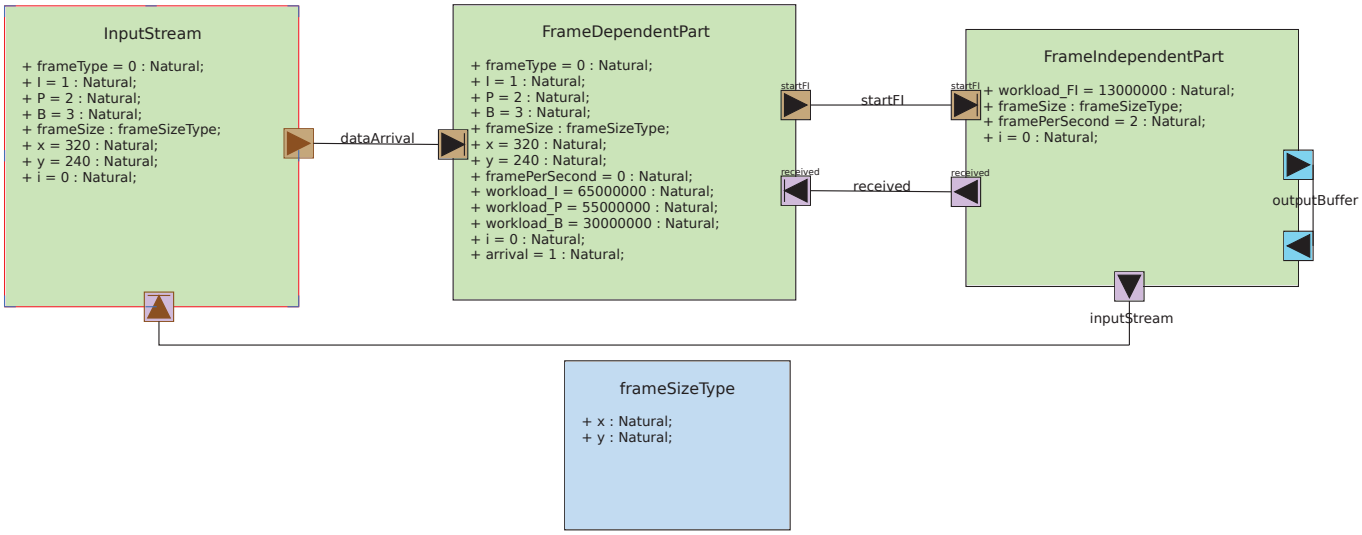

Fig. 2: MPEG-2 task diagram modeling

the Y chart methodology following three main stages related to application, architecture and mapping modeling as shown in Fig. 1. The application is modeled as a network of tasks communicating with channels, events, and requests. Each task behavior is described with a UML activity diagram extended with communication operators and abstract computational complexity operators. The architecture platform is modeled as a set of interconnected parameterizable execution nodes (CPUs, HW accelerators, DSPs), storage nodes (memories), and communication nodes (buses and bridges). The mapping phase is described using a set of interconnected hardware nodes on which tasks, channels, events and requests are placed. More details about DIPLODOCUS modeling concepts can be found in [9].

Application modeling: Application modeling is based on a component based diagram, where each component represents a task of the application. Fig. 2 illustrates the task diagram corresponding to the MPEG-2 case study. The diagram contains three UML components representing the InputStream, FrameDependentPart (FD), FrameIndependentPart (FI) tasks. The InputStream task is in charge of the generation of encoded stream. Each GoP is composed of the following sequence of frames: IBBPBBPBBPBBPBB. The stream is then processed by the FD part followed by the FI part. Each task is detailed using an activity diagram. In DIPLODOCUS/TTool, activity diagrams abstract the actual implementation of application tasks by focusing on activity communications and execution times. For instance, the activity diagram of the FD part is shown in Fig. 3. The behavior of the FD part is abstracted with the computational cost operator *EXECI* with different values for each frame type (using the DIPLOCOCUS choice operator). In this figure, the used EXECI operators are $workload\_I$, $workload\_B$ and $workload\_P$. These operators give the execution time of the FD part in number of cycles according to the frame type. Here, we use a Gaussian function to determine these execution times. The parameters of this function (average and prediction error) are extracted from the hardware results presented in [21]. As shown

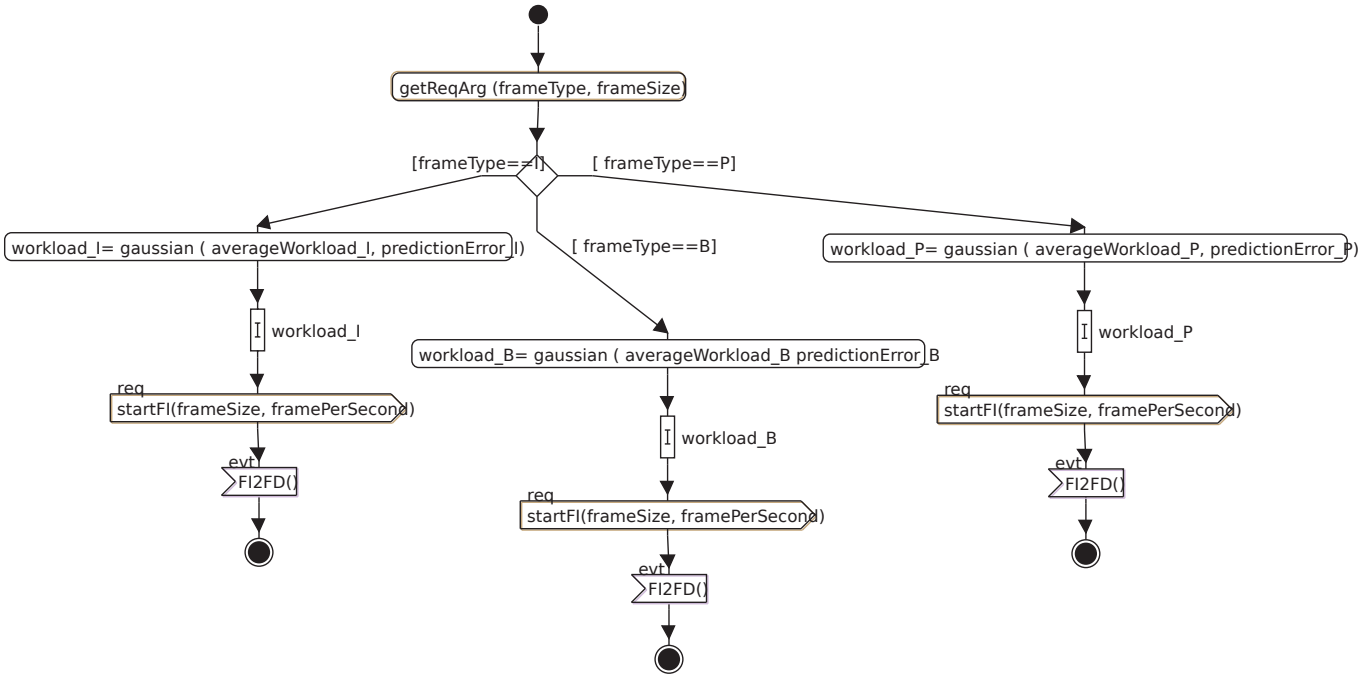

Fig. 3: Activity Diagram of the frame dependent part

in Fig. 3, after executing the FD part (represented by the EXCECI operators), the FI part is called ($startFI()$). This function takes into account the frame size and the required frame rate. When an event (evt) indicating that the FI part is finished, the FD part can be executed for the next frame ($FI2FD()$).

Architecture and mapping modeling: As explained previously, the targeted platforms are StrongARM and XScale platforms. The architecture and mapping modeling for this case study are very simple. They are based on the modeling of one processor for both platforms using the DIPLODOCUS profile. In the mapping phase, all the application tasks are linked to the modeled processor.

C. Power modeling using the extended DIPLODOCUS profile

The objective of the extensions we propose for the DIPLODOCUS profile is to model the power consumption of embedded applications at a high level of abstraction together with dynamic management policies. The generated C++ simulation code can be then used to evaluate different system configurations and power management policies. Fig. 1 shows an example of the integration of the dynamic power management by linking the PM (Power Manager) component to the CPU. The UML concepts introduced for the modeling of power management aspects will be detailed later in this section. These concepts can be used to target different implementations of power management techniques (by an OS service, a hardware module, etc.).

In this paper, we integrate the power models defined in [22] for static and dynamic power consumption. The power consumption as defined in [22] is given by:

$$P = cfV_{dd}^2\alpha + I_{leakage}V_{dd} \quad (1)$$

The first and second parts of the equation represent the dynamic and static power respectively. V_{dd} is the supply voltage, f is the operating frequency, c is the total circuit capacitance, α is the switching activity, and $I_{leakage}$ is the leakage current. For the rest of the paper we use the notation $C = c * \alpha$ to denote the switching capacitance. Equation (1) highlights the CPU power-related characteristics (C, f, V_{dd}) that should be defined in the DIPLODOCUS profile in order to enable power-aware simulation using TTool. The extended profile has also to take into account static power.

Energy-aware CPU: In order to integrate dynamic power consumption in the DIPLODOCUS profile, we propose the *<<energyAwareCPU>>* and *<<CPUFinitePowerStateMachine>>* stereotypes. The first stereotype extends the DIPLODOCUS CPU metaclass. Fig. 4 illustrates the power-aware modeling of the StrongARM platform for the MPEG-2 case study. As shown in this figure, the *<<energyAwareCPU>>* stereotype provides the CPU characteristics. Among them, we find the scheduling policy, the number of cores and the pipeline size. We find also the capacitance (c), which we need for dynamic power estimation. The switching activity (α) is fixed at the mapping phase.

The V_{dd} and f are not defined in the *<<energyAwareCPU>>* stereotype because recent processors support several voltage/frequency levels, which we call operating points (OPs) in this paper. These OPs can be switched at runtime. Therefore, we propose the *<<CPUFinitePowerStateMachine>>* stereotype, which represents these OPs in form of *Power Finite State Machine* (PFSM). Each state (<<PowerState>>) of this FSM corresponds to an OP, or mode of the system, and is characterized by a voltage/frequency couple. Fig. 4 illustrates

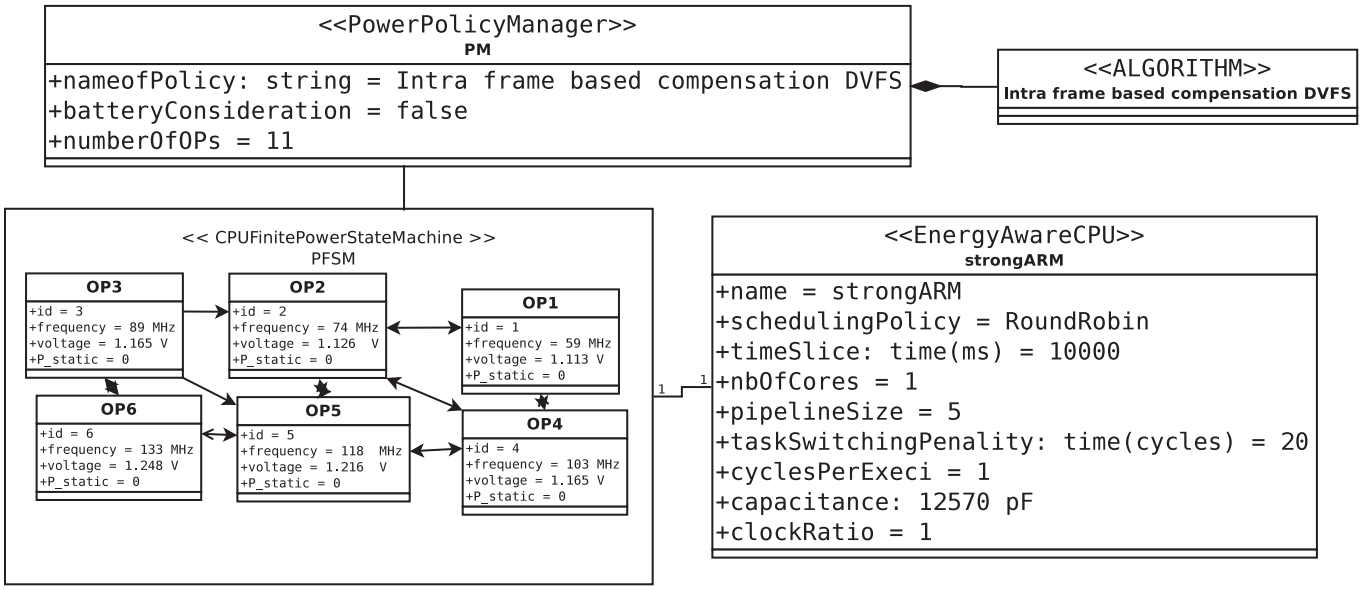

Fig. 4: Power-aware modeling for the StrongARM platform

only some of the modeled OPs for the StrongARM platform for a better readability. Each OP describes also the static power related to the used voltage. In this case study, we neglect the static power since for the StrongARM platform, the dynamic power consumption is the most significant [23]. Transitions between power states are characterized by their durations and power costs (not represented in the figure for better readability).

Dynamic power management policies: As shown in Fig. 4, a PFSM can be used by one or several power management policies (<<PowerPolicyManager>>) in charge of triggering transitions between OPs and selecting the current OP or system mode. In this paper, our objective is to evaluate the impact of dynamic power management on the power consumption. However, our approach can also be used to experiment with different power management policies. A power management policy can be implemented either in hardware or software through a given algorithm (<<Algorithm>>).

In this paper, we propose to use the frame-based DVFS technique described in [21], which significantly reduced the CPU power consumption of the MPEG-2 decoder by more than 50%. Our objective is to compare the results provided by our modeling approach with the results obtained in their hardware experiments. The proposed DVFS technique takes into account the frame type for DVFS. It is based on the prediction of the execution time of the FD part for different frame types and compensating prediction error by adjusting the frequency during the execution. This technique can be implemented in three different ways: 1) the intra-frame compensation algorithm, 2) the inter-frame compensation algorithm, 3) the use both intra- and inter-frame compensation algorithms. In this case study, we use a distinct algorithm for each platform: the intra-frame algorithm for the StrongARM, as shown in Fig. 4, and the inter-frame algorithm for the XScale platform.

For the StrongARM platform, by applying the intra-frame compensation algorithm, the FI time is used as an adjustment variable to compensate prediction errors that may occur during the workload prediction of the FD part. The typical operations performed in the FI part are memory-intensive. Since the memory bus frequency depends on the CPU frequency, the FI time is predictable and depends only on the chosen processor frequency. The FD time is predicted taking into account the type of image (I, P, B) and the average workload of a small group of pictures having the same type. When the measured workload of the FD part is higher than the predicted one, the correction takes place by switching to an OP with a higher frequency and voltage level in order to respect the FI part deadline. When the measured workload of FD part is lower than the predicted one, a considerable power consumption gain can take place by switching to an OP with a lower frequency and voltage level.

The DVFS algorithm used for the XScale platform is the inter-frame compensation algorithm. Since in this platform the memory bus clock and CPU clock speed are decoupled, the FI time does not depend on the CPU frequency and cannot be used as adjustment variable. The operating frequency is set to the minimum allowed value for the FI part in order to save power. The inter-frame technique is applied to the FD part in order to adjust the FD time of a given frame taking into account both its predicted workload and the prediction error related to previous frames.

Since the FD and FI times for the StrongARM platform depend on the CPU frequency, both intra-frame and inter-frame compensation DVFS techniques can be applied. This can be used to explore the three power management algorithms on this platform. However, as explained in [21], in the hardware implementation only intra-frame DVFS technique was used for this platform since there were many time slacks during the FI part due to the low frame rates achieved by this low-performance platform [21]. Therefore, in our case study, we apply only one power management algorithm on each platform.

Frequency (MHz)	voltage (V)
59	1.113
74	1.126
89	1.156
103	1.165
118	1.216
133	1.248
148	1.326
162	1.394
177	1.464
192	1.536
206	1.605
221	1.670

TABLE I: StrongARM operating points

IV. Experimental results

A. Simulation code generation

Using the TTool code generator, application, architecture and mapping models are translated into C++ code. From a high-level representation, the simulation code is generated automatically. For our case study, 1830 and 1701 lines of code were generated automatically for the StrongARM and XScale platforms respectively. This has the benefits of increasing design productivity. The TTool simulator was also extended in order to integrate power management control. In fact, before our extensions, the C++ code used to simulate the CPU did not take into account different frequency modes as many CPUs do. The simulator was extended in order to guarantee its compatibility with the generated power-aware code.

In the generated simulation code, power consumption is estimated using the following equation:

$$P_{average} = \overbrace{C * \frac{\sum_{i=1}^{N}(V_{FD}^i)^2 f_{FD}^i T_{FD}^i + (V_{FI}^i)^2 f_{FI}^i T_{FI}^i}{Simulation\ \ Time}}^{\text{dynamic power}} + \underbrace{\frac{\sum_{i=1}^{N} P_{static\ FD}^i T_{FD}^i + P_{static\ FI}^i T_{FI}^i}{Simulation\ \ Time}}_{\text{static power}} \quad (2)$$

N is the number of displayed frames. C is the switching capacitance. V_{FD}^i, f_{FD}^i, T_{FD}^i, $P_{staticFD}^i$ are the CPU voltage, the CPU frequency, the execution time and the static power related to the processing of the FD part for the i^{th} frame. V_{FI}^i, f_{FI}^i, T_{FI}^i, $P_{staticFI}^i$ are the CPU voltage, the CPU frequency, the execution time and the static power related to the processing of the FI part for the i^{th} frame. T_{FD} and T_{FI} are calculated by multiplying the current operating frequency by the number of cycles of the FD and FI parts respectively. The number of cycles is obtained from the *EXECI* operators modeled in the activity diagrams. The other parameters of the equation are the $capacitance$ of the <<EnergyAwareCPU>>, the α property modeled during the mapping phase, and the $voltage$ and $frequency$ of <<PowerStates>>. In this equation only the switching capacitance (C) is static during simulation. Voltage and frequency are determined at runtime using the DVFS algorithms.

Frequency (MHz)	voltage (V)
333	0.91
400	0.99
466	1.05
533	1.12
600	1.19
666	1.26
733	1.49

TABLE II: XScale operating points

B. Power-aware DSE

The aim of these experiments is to show that our MDE approach is capable of obtaining results that can be used to take adequate DSE decisions. Tables I and II summarize the OPs of the StrongARM and XScale platforms respectively. Simulations were applied to the Siberian Tiger video which is composed of 634 frames. At the modeling step, the frame rate on the StrongARM platform was set to 2 fps (the frame rate is a parameter in activity diagrams, see Fig. 3). The reason for this low frame rate is that frame rates higher than 3 fps (obtained without DVFS at maximum frequency) were not achievable for the StrongARM platform [21]. For this platform, we obtain a simulated execution time of 317 seconds. As for the XScale platform, the used frame rate was equal to 14 fps and the simulated execution time was 46 seconds. These simulation results show that the deadlines (related to the required frame rates) set for both platforms were respected with the usage of DVFS techniques.

Fig. 5 illustrates the variations of frequency and power, during the simulation the of MPEG2 decoding on the StrongARM platform. Using the intra-frame compensation DVFS technique, the average CPU power consumption obtained by simulation for the StrongARM platform is 667 mW as shown in Table III. This represents a power reduction of 30% compared to the power consumption without DVFS. When compared to the power values obtained with real board measurements [21] when using the DVFS technique (see Table III), we can consider that the accuracy of the estimates given by our approach are acceptable (error percentage: 6,5%).

Fig. 6 illustrates the variations of frequency and power, during the simulation the of MPEG2 decoding on the XScale platform. For this platform, the inter-frame compensation DVFS technique is used. This technique allows to adjust the frequency of FD part according to the exceeding cycles of the previous frame. Since the FI execution time does not depend on the frequency in the XScale platform, when using DVFS, the minimum frequency (333 MHz) is always set for the FI part in order to save power. The average CPU power consumption obtained by simulation for the XScale platform is 283 mW as shown in Table III. This represents a power reduction of 68% compared to the power consumption without DVFS. When compared to the power values obtained with real board measurements [21] when using the DVFS technique (see Table III), we can consider the accuracy of the estimates given by our approach are acceptable (error percentage: 16%).

According to the obtained results for both platforms, our DSE can be concluded by deciding to apply dynamic power management since it allowed power reduction while respecting task deadlines. The second decision is that to choose the

	StrongARM platform			XScale platform		
	without DVFS	with DVFS		without DVFS	with DVFS	
		hardware implementation [21]	simulation		hardware implementation [21]	simulation
CPU Power consumption (mW)	954	714	667	883	244	283

TABLE III: Simulation and hardware implementation power consumption results

XScale platform since it gives more advantages performance and power consumption. Certainly, this result is predictable for the platforms chosen in this case study, but our objective here is to show that our approach allows to obtain fast power estimates with an acceptable accuracy and can be used to make early adequate design decisions. In this case study, we applied only one DVFS technique for each platform. However, our approach can be also used to explore more than one DVFS technique for a given platform. At the modeling level, as we explained previously the power state machine describing the CPU operating points can be linked to more than one <<powerPolicyManager>>. The designer can thus link different power management algorithms to the CPU. The generated code can be then used in order to explore different power management techniques.

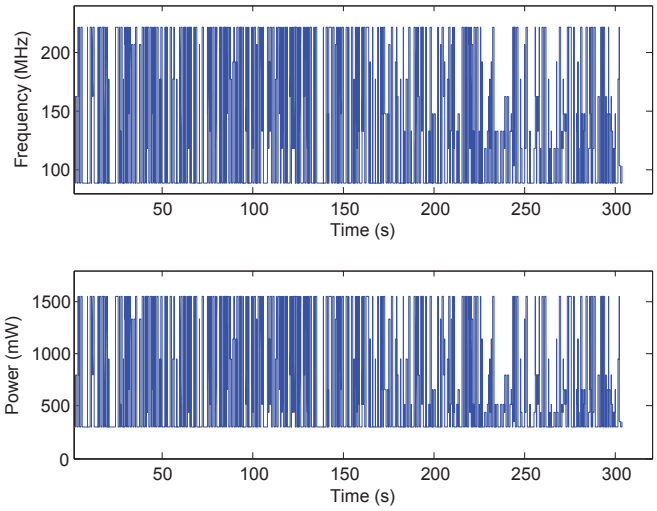

Fig. 5: Simulation of the MPEG-2 decoder on the StrongARM platform using the intra-frame compensation DVFS technique with a frame rate = 2 fps

V. Conclusion and Future Directions

In this paper, we present a Model-Driven approach for early power-aware Design Space Exploration (DSE) of embedded systems. This approach allows a high-level modeling of various aspects related to CPU consumption and dynamic power management targeting an automatic generation of the corresponding simulation code. It was implemented in the DSE toolkit TTool by integrating power concepts in the DIPLODOCUS UML profile. TTool was also extended in

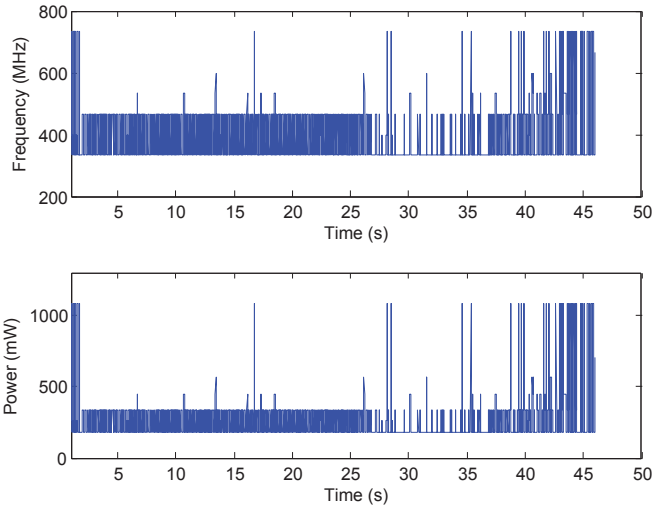

Fig. 6: Simulation of the MPEG-2 decoder on the XScale platform using the inter-frame compensation DVFS technique with a frame rate = 14 fps

order to take into account the introduced concepts in the generated C++ simulation code. The proposed approach was applied for the modeling of the power aspects related to the implementation MPEG-2 decoder on two different platforms namely the StrongARM platform (a low quality performance) and the XScale platform (a high quality performance). The power estimates given by the generated simulation codes were comparable to real board measurements. These results showed that our approach allowed to take efficient design decisions early in the design flow, which permits to avoid a considerable material and design costs. There are many avenues for future work, as we intend to improve the power model specification to enhance the accuracy of our estimations and to target multiprocessor architectures.

References

[1] V. Venkatachalam and M. Franz, "Power reduction techniques for microprocessor systems," *ACM Comput. Surv.*, vol. 37, no. 3, pp. 195–237, Sep. 2005.

[2] K. Choi, W. Lee, R. Soma, and M. Pedram, "Dynamic voltage and frequency scaling under a precise energy model considering variable and fixed components of the system power dissipation," in *IEEE/ACM International Conference on Computer Aided Design*, November 2004.

[3] L. Benini and G. D. Micheli, "System-level power optimization: Techniques and tools," *ACM TRANSACTIONS ON DESIGN AUTOMATION OF ELECTRONIC SYSTEMS*, vol. 5, no. 2, pp. 115–192, 2000.

[4] C. Atkinson and T. Kühne, "Model-driven development: A metamodeling foundation," *IEEE Softw.*, vol. 20, no. 5, pp. 36–41, 2003.

[5] O. Mbarek, A. Khecharem, A. Pegatoquet, and M. Auguin, "Using model driven engineering to reliably accelerate early low power intent exploration for a system-on-chip design," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, ser. SAC '12. New York, NY, USA: ACM, 2012, pp. 1580–1587.

[6] OMG, "Unified Modeling Language (OMG UML)," no. November, 2007.

[7] S. Friedenthal, A. Moore, and R. Steiner, *A Practical Guide to SysML: Systems Modeling Language*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2008.

[8] O. M. G. D. Number and A. Files, "Uml profile for marte : Modeling and analysis of real-time embedded systems," *Engineering*, vol. 15, p. 738, 2009.

[9] L. Apvrille, "Ttool for diplodocus: An environment for design space exploration," in *Proceedings of the 8th international conference on New technologies in distributed systems*, ser. NOTERE '08. New York, NY, USA: ACM, 2008, pp. 28:1–28:4.

[10] F. B. Abdallah and L. Apvrille, "Fast evaluation of power consumption of embedded systems using diplodocus," in *EUROMICRO-SEAA*, 2013, pp. 138–144.

[11] "Storm simulation tool." [Online]. Available: http://storm.rts-software.org

[12] M. K. Bhatti, C. Belleudy, and M. Auguin, "Hybrid power management in real time embedded systems: an interplay of dvfs and dpm techniques," *Real-Time Systems*, vol. 47, no. 2, pp. 143–162, 2011.

[13] B. Ouni, C. Belleudy, and E. Senn, "Accurate energy characterization of os services in embedded systems." *EURASIP J. Emb. Sys.*, vol. 2012, p. 6, 2012.

[14] R. Ben Atitallah, r. Piel, J. Taillard, S. Niar, and J.-L. Dekeyser, "From High Level MPSoC description to SystemC Code Generation," in *International ModEasy'07 Workshop in conjunction with Forum on specification and Design Languages (FDL'07)*, Barcelona, Spain, September 2007.

[15] C. Trabelsi, R. Ben Atitallah, S. Meftali, J.-L. Dekeyser, and A. Jemai, "A Model-Driven Approach for Hybrid Power Estimation in Embedded Systems Design," *EURASIP Journal on Embedded Systems*, vol. 2011, no. 1, p. 569031, 2011.

[16] S.-K. Rethinagiri, O. Palomar, O. Unsal, A. Cristal, R. B. Atitallah, and S. Niar, "Pets: Power and energy estimation tool at system-level," in *15th International Symposium on Quality Electronic Design (ISQED)*, 2014.

[17] E. Senn, S. Douhib, D. Blouin, J. Laurent, S. Turki, and J.-P. Diguet, *Power and Energy Estimations in Model-Based Design*, ser. Lecture Notes in Electrical Engineering, M. Radetzki, Ed. Springer Netherlands, 2009, vol. 36, 10.1007/978-1-4020-9714-0_1.

[18] E. Senn, J. Laurent, E. Juin, and J.-P. Diguet, "Refining power consumption estimations in the component-based aadl design flow." in *FDL*. IEEE, 2008, pp. 173–178.

[19] J. Laurent, E. Senn, N. Julien, and E. Martin, "High Level Energy Estimation for DSP Systems," in *in Proc. Int. Workshop on Power And Timing Modeling, Optimization and Simulation PATMOS*, 2001, pp. 311–316.

[20] T. Arpinen, E. Salminen, T. D. Hmlinen, and M. Hnnikinen, "Marte profile extension for modeling dynamic power management of embedded systems," *Journal of Systems Architecture*, vol. In Press,, pp. 1–11, 2011.

[21] K. Choi, W.-C. Cheng, and M. Pedram, "Frame-based dynamic voltage and frequency scaling for a mpeg decoder," *Journal of Low Power Electronics*, pp. 27–43, 2005.

[22] W. Zhang, J. Williamson, and L. Shang. Springer US, 2011, ch. Power Dissipation, pp. 41–80.

[23] T. Jiang and P. Y. Chiang, "Sense amplifier power and delay characterization for operation under low-vdd and low-voltage clock swing," in *IEEE International Symposium on Circuits and Systems*, ser. ISCAS, 2009.