Power-Management Specification in SystemC

Dominik Macko, Katarína Jelemenská, Pavel Čičák
Faculty of Informatics and Information Technologies
Slovak University of Technology
Bratislava, Slovakia
dominik.macko@stuba.sk, katarina.jelemenska@stuba.sk, pavel.cicak@stuba.sk

***Abstract*—Power consumption is the greatest concern in current highly-integrated hardware-system design. The power reduction is targeted mostly through power management, implementing such techniques as clock gating, power gating, or voltage and frequency scaling. Due to growing complexity, the start-point in the design has moved from the register-transfer level to the system level. However, the power management lacks the abstraction needed for the system level. Also, different power-management techniques are specified differently, complicating the specification even more. This paper targets the unified specification of power-management techniques early in the design flow. SystemC is used for describing the system functionality along with the power management. Efficiency of the proposed approach is illustrated by comparison of the unified power-management specification and the standardized approach.**

Keywords—design; low power; power control; power management; power reduction; system level

I. Introduction

Power in digital systems has become the dominant problem in modern designs. Due to a high integration of the circuits in a small chip area, the power density is rising. It causes concerns not only in battery-operating devices, but due to the packaging and cooling costs it influences any hardware design.

In time, many power-reduction techniques have been developed. Straightforward techniques, such as fine-grained clock gating, logic restructuring, pin swapping, or gate sizing, are applied automatically by modern synthesis tools to meet some preset constraints [1]. Some design techniques that are not originally created for power reduction might help to reduce it, e.g. advanced design for test techniques [2] or reduction of multiplexer trees [3]. Due to a high power dissipation of clock signal, there also exist techniques to change a synchronous system to an asynchronous one, such as the one used in [4]. However, the advanced techniques, working with multiple voltage levels, are not so easily adopted. Usually, the functional HDL (Hardware Description Language) model is augmented by a specification of power-related aspects in some additional form, such as the UPF (Unified Power Format) [5].

This standardized format enables a designer to split the system into several power domains, assign power supplies to these domains, and specify the allowed power states of the components inside these domains. The power states are specified using the power-management elements, such as power switches, level shifters, isolation, or retention cells, which are inevitable for the components to correctly operate in these power states. Thus, a power state is defined by a unique combination of control-signals values for power-management elements inside a certain power domain. All of these power-related aspects could not be described in usual HDL. Therefore, the UPF files stand alongside the HDL files, as illustrated in Fig. 1. Both the HDL and UPF file-sets are used in the verification process.

Since the complexity of hardware designs grows, the system level is used as a starting point in a design process (as suggested by [6]). However, the power-intent specification in UPF is not feasible for the system level. The specification at such a high abstraction level should be as simple as possible. All the necessary information should be specified in one model, but in such a way that the power-related information can be easily separated from the functional model.

This paper introduces the power-management specification into the SystemC model that unifies the way the power-management techniques are used in a design. We propose an abstraction from the low-level power-related information, such as supply networks and power-management elements. All the necessary components are automatically added to the design during high-level synthesis. In the next section, the related work regarding the utilization of the system level in low-power design is described. Section III introduces an abstract power-management specification in SystemC and before the conclusion, some experimental results are provided.

II. Related Work

The PwARCH framework [7] utilizes transaction-level modelling (SystemC with TLM) augmented by a UPF-like power management to explore various power architectures. It represents a golden model to a design team for the RTL implementation. The key idea lies in TLM power estimation. A similar method is proposed in [8]. The power-data model used in this method is filled by information obtained from lower-level power estimations and technology libraries. Thus, it is

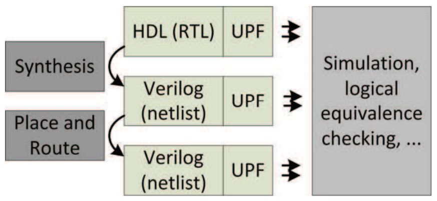

Fig. 1 UPF-based low-power design.

This work was partially supported by the Slovak Science Grant Agency (VEGA 1/1008/12 and VEGA 1/0616/14), and Slovak University of Technology in Bratislava.

highly dependent on the design reuse approach. Similar methods, described in [9-11], are also based on the system-level power modelling, enabling the power-architectures exploration. Although these methods support modelling of power management, they are not based on UPF standard or its concepts. Therefore, the equivalency between the system-level model and its RTL implementation is difficult to show. Another method [12] uses ESL simulation traces to estimate power at the RTL. There is also a method for a clock-gating specification at the ESL described. The manual macro-based specification in a C model is passed to the high-level synthesizer, which automatically inserts clock-gating cells into the RTL model. In addition to the manual pinpointing of the clock-gating location, another disadvantage is not-using other usable power-management techniques.

Based on the observations obtained from the analysis of the existing methods, we combine their advantages and try to eliminate their disadvantages when using the system level for low-power design, especially the utilization of power management in specification stages. The proposed method is based on our previous research, described in [13], which integrates abstract power-management into our proprietary specification model. To integrate the method into the existing industrial design flows, we have decided to modify it to be usable in SystemC modelling.

III. SystemC Power-Management Specification

Although the UPF is a great help for designers to design low-power systems, it is not feasible for the system level. The specification in separated languages and styles (HDL and UPF) complicates the specification. The designer has to keep many system aspects in mind, what increases a probability of introducing an error to the design. There is a need for higher abstraction to simplify the specification. However, the existing methods either still keep the functional and power-management specification separated, or unify the specification style without a proper abstraction of power management. Therefore, we propose a novel method, extending the SystemC modelling to include abstract power-management specification. The methodology using this method is based on the high-level synthesis process and its utilization for power estimation. It enables a designer to obtain fast and more-accurate information about system power consumption. Together with an easy modification of power-management aspects, it enables power-architecture exploration.

A. *Abstract Power-Management Concepts*

In order to use a high-level synthesis to obtain power consumption of the system (or its component), synthesized model has to be standardized in the industry. Many RTL power estimators are using UPF power-intent specification alongside an HDL design. Thus, the abstract power management should be based on the UPF concepts. However, not all power-management techniques can be specified in UPF. Some of them are used as a part of the functional description. Therefore, the abstract power management should unify the specification of power-related aspects.

TABLE I. Abstract Power States

Power State	Description
NORMAL	The block is operating at the basic voltage level and clock frequency.
HOLD	The block stops its operation but stays powered.
DIFF_LEVEL#	The block operates at the voltage and/or frequency level different from the basic one; # represents a number enabling specification of multiple different levels of operation.
OFF	The block is powered down, i.e. its supply is shut off.
OFF_RET	The block is powered down, but its state is retained.

All of the power-reduction techniques applied through power-management architecture can be introduced by a set of power states. These power states are assigned to power domains, which are suitable forms of grouping the related blocks together. The possible power states are described in Table I. The *NORMAL* state represents operation of a block, when no explicit architectural power-reduction technique is used. The *HOLD* state is used to apply the clock gating and operand isolation techniques. It represents a situation when all block inputs are isolated – values switching is prevented. The collection of states marked as *DIFF_LEVEL* is used for the application of multiple performance levels. It enables the voltage and frequency scaling techniques, even the usage of multiple voltages in the design. The last two power states, *OFF* and *OFF_RET*, specify the power-gating technique without and with state retention.

Usually, the system is set to a certain operating mode to perform some specific task, i.e. power domains work in certain power states. It is feasible to specify allowed power modes in order to reduce possible power-mode transitions and enable a more efficient usage of power-management elements. The designer can easily switch the power mode, without explicitly specifying which power domain should change its power state.

Following several rules, the provided information should be enough to specify power management at the system level. All power-management elements (isolators, level shifters, power switches, retention elements) can be introduced into the design implicitly, based on the abstract power-management specification and the relations among system blocks. These rules are as follows.

- Inputs of a power domain in the HOLD state have to be isolated to prevent unnecessary switching.
- The communication between blocks in power domains operating at different voltages has to be level-shifted.
- The communication between blocks operating at different frequencies has to be synchronized to prevent metastability and data loss.
- The clock signal to a power domain in the OFF or OFF_RET state has to be stopped.
- Outputs of a power domain in the OFF or OFF_RET state have to be isolated to prevent floating values.
- Power-supply network of a power domain in the OFF or OFF_RET state has to be switched off.
- Power-supply network of a power domain operating in several power states, representing different voltage levels, has to be switchable to change voltage source.

The implicit introduction of power-management elements enables abstraction at the system level, which simplifies the specification. The designer focuses only on specifying which blocks belong to which power domain, what power states each power domain can operate at, and what are the power modes the system can reach.

B. SystemC Modelling

SystemC is a standardized C++ class library [14], which is supported by many tools to design systems. Therefore, the specification of power management should not disrupt the SystemC compatibility with existing tools, nor it should influence the correct functionality. The easiest solution is to specify additional information in a commentary. However, such a solution is error-prone. Inspired by [12], we use a macro-based specification. Since a macro has to be predefined, any compatible compiler reveals syntactical errors caused by a wrong macro usage. The proposed power-management macros are grouped into an extension library. The problem of this approach lies in a dynamic nature of the abstract power management. The designer specifies names for power modes, names for power domains, the number of power states in individual power domains, and assignments of component instances to power domains. All of these cannot be statically predefined, and therefore some sort of modelling is required – even though we target a specification, not a functional modelling.

In Fig. 2, a part of the extension library is illustrated. Firstly, macros representing the possible power states are

```
#define NORMAL "normal"
#define HOLD "hold"
#define DIFF_LEVEL(i) "diff_level"#i
#define OFF "off"
#define OFF_RET "off_ret"
#define PM(...) PowerMode(__VA_ARGS__, NULL)
#define PD(...) PowerDomain(__VA_ARGS__,NULL)
#define SetLevel(state, voltage, frequency)
static PowerMode POWER_MODE(NULL);

class PowerMode
{
    std::vector<std::string> states;
public: PowerMode(const char* state, ...);
};
PowerMode::PowerMode(const char* state, ...)
{
    va_list args;
    va_start(args, state);
    for (va_start(args, state); state != NULL; state = va_arg(args,
const char*)){ this->states.push_back(state); }
    va_end(args);
}
class PowerDomain
{
    std::vector<std::string> states;
    std::vector<std::string> components;
public: PowerDomain(const char* state, ...);
    void AddComponent(std::string component);
};
```

Fig. 2 The SystemC power-management extension.

predefined. One can notice a difference in specification of *DIFF_LEVEL* power state. The number specifying an actual power state is passed to the macro as an argument, because it cannot be predefined in a static way. All the power states are translated to a string form, which simplifies run-time verification. A C++ class definition is used to model power modes and power domains. These classes contain vectors preserving power states, specified by a designer. For the power domain, specified states are the possible operating power states. For the power mode, they are an allowed combination of power states for power domains. The first state belong to the first specified power domain, the second state to the second domain, and so on. The power domain class contains another vector, storing assigned identifiers of components instances. The method *AddComponent* is then used to fill the list. The constructors of these two classes can take a variable number of string arguments; thus, the designer is not constrained in design complexity. To prevent a memory violation to occur due to a "variadic" function of the constructors, a *NULL* argument has to be the last one. To alleviate a need for the designer to keep this constraint in mind, we predefine two additional macros, *PM* and *PD*. They can be used instead of the actual constructors of those classes, without the *NULL* argument. The constructor of the *PowerDomain* class is analogous to the power mode constructor. To keep the current power mode of the system, a global object needs to be created. It is an instance of the *PowerMode* class with a static name of *POWER_MODE*. This name has to be static in order a high-level synthesis tool to recognize it. Since there are no implicit power domains and power modes, this instance has no states assigned. It has to be initialized by the designer. The main purpose of this instance is to be used in the functional model to specify power-mode changes. There is also a macro *SetLevel* defined. It is used for a specification of actual voltage and frequency values. These are needed to determine whether two power domains operate at the same voltage or frequency levels. These relations influence a high-level synthesis process and therefore are also verified at the system level. These voltage-frequency pairs have to be specified for *NORMAL* and *DIFF_LEVEL* states. The actual voltages are also needed to estimate power at the RTL. The specification at the system level prevents a need for a designer to modify the synthesized UPF power-intent specification.

C. Early Power-Management Verification

Syntax of a specified power-management can be automatically verified by any C++ compiler, supporting the SystemC library, at the compilation time. The predefined language and libraries keywords drive the designer to the correct specification. The modelling introduced into the extension library ensures that only a valid specified power mode is assigned to the *POWER_MODE* variable and that the instances can be assigned only to the existing power domains. The run-time checks, implemented in the traditional conditional manner, verify whether only the valid power states are used in the specification. If the designer uses the predefined macros, this error should not appear. These checks also reveal a redundant specification of some power state in some power domain. Since the power management is dependent on the identifiers used by a designer, more-sophisticated consistency errors are checked for through a static analysis. For example, it can reveal that some instance is assigned to multiple power domains, some specified power state is not used in any power mode, some power domain does not contain any active power state or no voltage-frequency pair is assigned to it, and many other issues. The static analysis produces warnings and errors messages, which notify the designer where the inconsistency occurred and what modification could solve the problem.

Such a combination of verification checks helps the designer to create a correct power-management specification early in the design process. After the high-level synthesis, the existing power-aware verification methods can be used.

IV. Experimental Results

It is difficult to compare the proposed approach of power-management specification and the standardized one. To provide an illustration how the specification is simplified, we compare the proposed SystemC power-management specification to the equivalent UPF form (generated by the high-level power-management synthesizer). The comparison is based on the number of characters needed to describe the power-intent aspects in the design. To provide a relatively accurate illustration, we have pseudo-randomly generated several thousands of samples, scaling the parameters of the number of power domains, the average number of power states in power domains, the number of power modes, the average number of instances in power domains, and the number of inter-instances communications. Since the power management is dependent on what states are assigned to the power domains, we generate five samples for each set of parameters and average the results. A relative comparison of the system-level power management to the UPF specification is given in Fig. 3. The results show that the specification of power-management aspects in the proposed SystemC extension is about five times less complex in average than the UPF specification at the RTL. In the experiment, the simplification is scaling from 2.6 to 7.2 times (in terms of the number of characters reduction).

V. Conclusions and Further Work

The UPF specification of power-management aspects in the design is unsuitable for the system level of abstraction. We have proposed a method extending SystemC modelling in order to include abstract power-management concepts, such as power domains, power states, and power modes. The method unifies the functional and power-management specification style into one specification model, which simplifies the specification. The power-reduction techniques, such as power gating, clock gating, voltage and frequency scaling, and multi-voltage design, can be applied at the system level in an abstract form and architectural manner. The low-power design flow based on the proposed method only supplements the existing UPF-based flow; thus, other power-reduction techniques, not applicable at the system level, can be used in later design stages (e.g. multiple thresholds, gate sizing, or logic restructuring). The simplified power-management specification is provided with the verification checks, ensuring the specification is correct and consistent with the functional model.

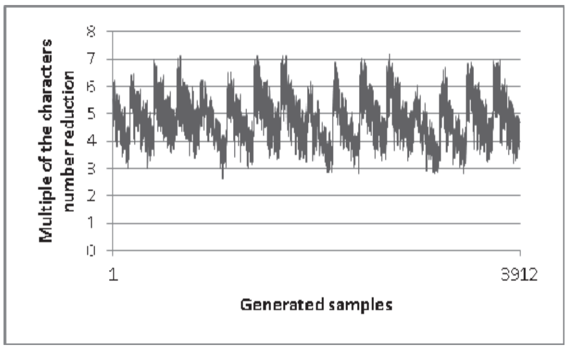

Fig. 3 The proposed SystemC power-management specification compared to the UPF specification.

The further work includes automatic splitting of the system into power domains, along with power states assignment. It will enable to completely abstract from the power-reduction techniques applied through power management.

References

[1] Power Forward Initiative, A practical guide to low power design: User experience with CPF. Power Forward, 2012.

[2] M. Siebert and E. Gramatová, “Delay Fault Coverage Increasing in Digital Circuits,” in DSD, IEEE, 2013, pp. 475-478.

[3] M. Maruniak and P. Pištek, “Binary decision diagram optimization method based on multiplexer reduction methods,” in ICSSE, IEEE, 2013, pp. 395-399.

[4] M. Šimlaštík, V. Stopjaková, L. Majer, and P. Malík, “Clockless Implementation of LEON2 for Low-Power Applications,” in IEEE Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2007, pp. 215-218.

[5] IEEE standard for design and verification of low power integrated circuits. IEEE, 2013. IEEE Std 1801-2013.

[6] The international technology roadmap for semiconductors: Design. ITRS, 2011 edition, 2011.

[7] O. Mbarek, A. Pegatoquet, and M. Auguin, "Using unified power format standard concepts for power-aware design and verification of systems-on-chip at transaction level," IET Circuits, Devices & Systems, vol. 6, no. 5, pp. 287-296, 2012.

[8] J. Karmann and W. Ecker, “The semantic of the power intent format UPF: Consistent power modeling from system level to implementation,” in 2013 23rd international workshop on power and timing modeling, optimization and simulation (PATMOS), 2013, pp. 45-50.

[9] Y. Xu, R. Rosales, B. Wang, M. Streubühr, R. Hasholzner, C. Haubelt, and J. Teich, "A very fast and quasi-accurate power-state-based system-level power modeling methodology," in ARCS'12 Proceedings of the 25th international conference on architecture of computing systems, 2012, pp. 37-49.

[10] H. Lebreton and P. Vivet, "Power modeling in SystemC at transaction level, Application to a DVFS architecture," in IEEE Computer society annual symposium on VLSI, 2008, pp. 463-466.

[11] T. Bouhadiba, M. Moy, and F. Maraninchi, "System-level modeling of energy in TLM for early validation of power and thermal management," in DATE '13 Proceedings of the conference on design, automation and test in Europe, 2013, pp. 1609-1614.

[12] S. Ahuja, High level power estimation and reduction techniques for power aware hardware design, Faculty of the Virginia Polytechnic Institute and State University, 2010. Dissertation thesis.

[13] D. Macko and K. Jelemenská, "Managing digital-system power at the system level," in IEEE Africon 2013 Sustainable Engineering for a Better Future, 2013, pp. 179-183.

[14] IEEE standard for standard SystemC language reference manual. IEEE, 2012. IEEE Std 1666-2011.