Power-Management High-Level Synthesis

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 management is an integral part of almost every new system design. It enables to keep the power under constrains, implementing such power-reduction techniques as power gating, multi-voltage design, or voltage and frequency scaling. Due to the complexity of modern designs, the system level of abstraction is adopted as a design starting point. However, the power management is not yet fully adopted at such abstraction level. In the previous research, we have proposed the abstract power-management specification, simplifying its adoption by an order of magnitude. This paper targets the power-management high-level synthesis, closing thus the gap between the system-level power management and its standard form at lower abstraction levels. Such design automation enables to reduce a number of human errors, potentially introduced by manual design. The presented experimental results validate the proposed approach.**

Keywords—design automation; high-level synthesis; low power; power management; specification; system level

I. Introduction

Power density in highly integrated CMOS (Complementary Metal-Oxide Semiconductor) circuits presents serious concerns about reliability of the devices. Because of the limited battery capacity in mobile devices, chip packaging and cooling aspects, or simply the energy consumption, the power needs to be managed in every new chip design [1].

Power management has been established as an application method for various power-reduction techniques that have been developed to alleviate the power problem. These techniques include, for example, clock gating, power gating, voltage scaling, or frequency scaling. In complex systems, the power management integration is rather difficult, and therefore systematic low-power design flows have been standardized. Two key industrial standards are CPF (Common Power Format) [2] and UPF (Unified Power Format) [3]. These standards enable to introduce low-level details (e.g. power-supply ports, supply nets, power-management elements) into a functional HDL (Hardware Description Language) model, and thus enable to verify power management at the RTL (Register-Transfer Level). They have soon gained popularity and are supported by the EDA (Electronic Design Automation) industry today. Using these standard formats, a designer can split the system into several power domains. Power domain is a set of system blocks operating in the same power state (the same supply voltage and operation frequency). Power states are specified using special circuitry – the power-management elements, such as power switches (enabling to power down the domain or to switch its power-supply net), level shifters (adjusting the voltage level of logic signals between domains), isolation, or retention cells. Thus, a power state is defined by a unique combination of control-signals values for power-management elements inside a certain power domain. The designer specifies allowed voltages for supply ports and nets, and for a verification purpose, creates a power-state table, specifying allowed combinations of states among these elements.

Since the complexity of modern designs caused adoption of the system level (ESL) in the design process (as suggested by [4]), the abstraction offered by the current low-power standards is not sufficient. Therefore, the research around extension of low-power design flows offered by these standards to the system level is gaining attention.

A. Related Work

The method [5] augments a transaction-level model with abstract UPF-based power-management concepts. It requires annotating power information to the system-level model in order to proceed with power-architecture exploration. Downsides of this method are that the RTL implementation is manual, the power annotation is time consuming, and architecture exploration does not take into account other important parameters, such as area or performance. Another method [6] automatically fills the power information in the system-level model based on the low-level simulation and technology libraries. Thus, it is highly dependent on a design-reuse concept. This method enables to generate a standard UPF specification, and therefore lower-level verification is easily accomplished using the existing EDA tools. However, the power management offered by this method contains a similar amount of details as the UPF standard; therefore, the abstraction in the system-level model is not sufficient.

Similar methods [7-9] also offer system-level power modelling, supporting the power management. Although they enable power-architecture exploration, they also do not take into account other design parameters (area, performance). Moreover, these methods are not based on standard low-power concepts; therefore, the equivalency between ESL and RTL power management is somewhat difficult to verify. A method [10] puts high-level synthesis to the forefront. A clock-gating ESL specification is passed to the high-level synthesizer, which automatically inserts clock-gating cells into the RTL

This work was partially supported by the Slovak Science Grant Agency (VEGA 1/1008/12 and VEGA 1/0616/14), Slovak University of Technology Bratislava (ASRSS) and COST Action IC 1103 MEDIAN.

model. The exact location and activation of clock gating has to be specified in the abstract model, what requires a manual effort. Another disadvantage is that this method does not support other efficient power-management techniques.

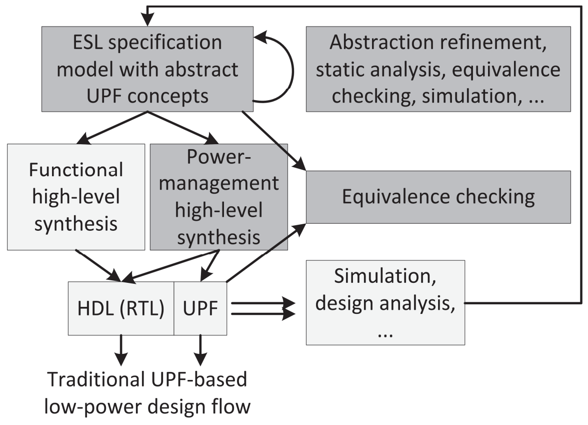
B. *Paper Overview*

Based on the identified benefits and drawbacks of the existing methods for ESL power management, we have proposed a novel low-power design methodology [11, 12]. The methodology provides an abstract power-management specification based on UPF concepts (similarly to [5]), utilizes UPF specification at the lower levels to enable verification using existing EDA tools (similarly to [6]), and offers high-level synthesis for multi-parameters trade-off and RTL design analysis for more accurate results (similarly to [10]). Fig. 1 provides an overview of the proposed design flow (the contributing parts are shown in dark-grey color).

This paper presents the power-management high-level synthesis process augmenting the functional high-level synthesis used in the industry today. Based on the abstract power-management specification at the ESL, the standard UPF specification is generated at the RTL as well as the power-management unit driving the control signals for power-management elements. Section II provides an overview of the existing power-reduction techniques. Section III introduces the basic principles of the proposed abstract power-management specification. Section IV describes the developed power-management high-level synthesis and Section V presents verification of the synthesized aspects. Before the conclusion, the experimental results are provided.

II. Power-Management/Reduction Techniques

Many power-reduction techniques have been developed [2], but only some of them can be used for power management during the runtime. Power management has the ability to change the power states of the system components and thus enables to save the power or to temporarily increase the system performance. Power management as a way to design a low-power device should implement such an algorithm that gives the components just enough power to perform a certain task and powers them down when not needed. Moreover, all unnecessary switching activities should be prevented.



ESL specification model with abstract UPF concepts
Abstraction refinement, static analysis, equivalence checking, simulation, ...
Functional high-level synthesis
Power-management high-level synthesis
Equivalence checking
HDL (RTL)
UPF
Simulation, design analysis, ...
Traditional UPF-based low-power design flow

Fig. 1. Overview of the proposed low-power design flow.

- *Clock gating* – disables the clock signal to stop loading the same value in some register. From the power-management perspective, it can be used to temporarily stop the operation of a synchronous block of the system. In such a way, the dynamic power consumed by unnecessary switching is saved.
- *Power gating* – shuts down the power to a system block when not needed. It saves both the static and dynamic power, but it takes time to power down and power up the block.
- *Operand isolation* – isolates unneeded datapath elements when not required, reducing unnecessary switching and thus saving dynamic power. Used as a power-management technique, it can be used to isolate inputs of an idle system block.
- *Voltage scaling* – enables to operate a system block at various voltage levels – e.g. a high level for high performance and a low level for power saving. Usually, each power domain is scaling its own supply voltage depending on its current requirements.
- *Frequency scaling* – enables to change the clock frequency of a synchronous system block. Higher frequency consumes more dynamic power, but enables to perform a task faster. This technique is usually combined with the voltage scaling, temporarily lowering the performance (task is not time-critical) to save power.
- *Substrate biasing* – enables to temporarily bias a substrate and thus raise the threshold voltage. This technique reduces the leakage current and thus reduces the static power.
- *Multiple voltages, multiple thresholds, gate sizing, logic restructuring, pin swapping* – these techniques are automatically used by a synthesis tool to select a suitable combination of library cells with various parameters to meet preset constraints.
- *Memory partitioning, bus segmentation, hardware acceleration* – these techniques target architectural decisions. Unfortunately, no component can be considered the best for each system, thus the architecture exploration process is commonly taken to select the optimal components.

Some of these techniques require special elements to be added to the design. When powering a system block down, its outputs should be isolated to prevent floating signals to corrupt data in the powered blocks. For such purpose, various isolation cells can be used – from a simple AND gate to a latch-based isolation elements. The communication between the blocks operating at different voltage levels should be level-shifted. Level shifters are added at the inputs of the receiving block. There exist cells that work as both, the isolators and level shifters. For gating the clock signal, clock-gating logic is used. It is similar to the isolation, but the timing impact should be carefully considered when dealing with the clock signal. To operate a block at different voltage levels, a power switch is needed to switch between power-supply networks or to simply shut the power off. Often, the registers state is required to be retained when powering-off some system block. Thus, some retention cells are used.

III. Abstract Power-Management Specification

The abstract power-management specification is also using power states, but in slightly different manner. An abstract power state is not representing control-signals values for power-management elements, since these are not present in the ESL specification. It represents a voltage-frequency pair (i.e. a performance level) of the power domain, to which it is assigned. All power-reduction techniques applicable by means of power management can be introduced by specific power states. The abstract power states include the following:

- *NORMAL* – system blocks in the power domain are operating at the basic voltage and frequency level.
- *HOLD* – all blocks in the domain stop their operation, but stay powered.
- *DIFF_LEVEL#* – blocks of the power domain in this state operate at the voltage and/or frequency level different from the basic one; # is an ordinal number – enables to specify multiple such levels.
- *OFF* – the whole domain is powered off.
- *OFF_RET* – registers values of system blocks in the power domain are retained, while the domain is powered off.

These states specify that certain architectural power-reduction techniques will be integrated into the corresponding domains at lower abstraction levels. The *NORMAL* state means that no explicit architectural power-reduction technique will be applied. The *HOLD* state represents the clock gating and the operand isolation – any switching activity at the power-domain blocks inputs is prevented. The *DIFF_LEVEL* states mean that multiple performance levels are used in the power domain. These enable voltage and frequency scaling and a multi-voltage design. The *OFF* and *OFF_RET* states refer to the application of power gating without and with state retention.

The system is usually set to a specific operating mode in order to execute some task – power domains are in specific power states for such a mode. It is useful to specify possible power modes – i.e. to specify combinations of power states among power domains. Such a practice reduces the verification overhead – only the possible power modes need to be verified. A designer can then easily switch the system power mode to the most feasible one for some specific task, without explicitly specifying which power-domain states need to be changed.

Thus, the abstract power-management specification involves a specification of power domains with a list of possible power states, specification of performance levels for active power states (*NORMAL* and *DIFF_LEVEL*), assignment of system blocks to power domains, specification of possible power modes of the whole system, and switching between the specified power modes. For an illustration, a sample of abstract power-management specification is provided in Fig. 2. We may notice a special *POWER_MODE* variable, representing the current power mode. It needs to be initialized to a certain specified power mode. The power-mode switching is specified by assigning another specified power mode to this variable in the functional design.

```
power_domain1 (off,normal,diff_level1);
diff_level1 (1 V, 50 MHz);
system_block instance1(power_domain1);
power_mode1 (off,hold);
POWER_MODE = power_mode1;

if (control_condition)
  POWER_MODE = power_mode2;
else
  POWER_MODE = power_mode1;
```

Fig. 2. A sample of abstract power-management specificaiton.

This abstract power-management specification is sufficient for high-level synthesis to generate a standard UPF specification. The required power-management elements (isolators, level shifters, power switches, retention elements) can be implicitly deduced from the abstract specification and system-blocks relations. The high-level synthesis has to comply the following rules.

- Inputs of blocks in a power domain in the *HOLD* state coming from outside of the domain have to be isolated.
- Communication between blocks in different power domains that are operating at different voltage levels has to be level-shifted.
- Communication between blocks in different power domains that are operating at different frequencies has to be synchronized.
- The clock signal to system blocks in a power domain that is powered-down has to be stopped.
- Outputs of blocks in a powered-down power domain have to be isolated.
- 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 multiple power states with different voltage levels has to be switchable.

The offered abstraction simplifies the specification, thus reducing the time required for its description. A designer does not have to keep in mind these rules – they should be followed by the high-level synthesis process.

IV. Power-Management Synthesis

The power-management synthesis consists of two distinguished processes. The first one is the synthesis of power-management specification to the UPF standard form. The other one is the synthesis of a controller, called the power-management unit, generating control signals for power-management support logic – power switches, isolation cells, and retention cells.

A. Power-Intent Specification Synthesis

During this process, power intent is extracted from the abstract power-management specification and subsequently analyzed in order to determine which power-management elements need to be specified in the UPF.

The first step is to create power domains in the UPF specification. Since the power domains are specified already in the ESL abstract specification, this step represents their rewriting in the UPF style. During creation of power domains, system blocks are assigned to them according the ESL specification. Beside the specified power domains, there is a top domain created, representing a main power domain of the whole system. Components that are not explicitly assigned to any power domain belong to the top domain.

The second synthesis step creates power-supply ports according to the required voltages in the design. These are determined based on performance-level assignments in the abstract specification. The next step is to create required supply nets and connect them to supply ports. How and which supply nets are created is determined based on power-domains need for power switch. If a power-domain's supply needs to be switchable, it requires a dedicated net. Otherwise, it can reuse a top-domain supply net. As a part of this step, primary power and ground nets are assigned to each power domain. After the supply network is created, the power-management elements can be specified. Firstly, we create a power switch for each power domain with switchable supply. Whether a power domain needs a switchable supply depends on the abstract power states, in which it can operate. If there are at least two voltage states (a power state with a specific voltage level, including off state) specified for a domain, it needs to have a switchable supply net.

The next step is to create the required isolation in the design. Based on the analysis of abstract power states, the power domains with the *HOLD*, *OFF* and *OFF_RET* states are pinpointed. If a power domain can be in *HOLD*, input isolation is created. If a power domain can be powered down, output isolation is created to prevent floating signals. However, in order to reduce unnecessary switching activity in powered down domain, its inputs are also isolated. The abstract specification of performance levels enables to identify communication between power domains operating at different voltages. This communication is regulated using level shifters. Level shifters are placed at the domain boundary. Input level shifters are located inside the power domain, output level shifters are located inside a parent domain (the top domain). In each power domain that contains *OFF_RET* in abstract specification, the retention is set. It retains the value of all registers inside system blocks, located in such a domain.

As the last step, voltage states of created ports are specified. These ports include the created supply ports and the power-switches output ports. These states are determined based on abstract power states specified for power domains, to which primary supply nets these ports connects. An important part of this step is to create a power-state table (PST) based on the power modes, specified at ESL. One of the differences is that the states are specified for individual ports instead of power domains. Another difference is that these states represent voltage states, not the power states. The last difference is specification of additional voltage modes (analogously to voltage states) that are required for correct power management functionality. For example, if a power domain changes its state from *NORMAL* to *OFF*, it has to be firstly isolated and only after that it can be powered down. Thus, for each transition between power modes in the abstract specification, additional intermediate power modes are required (beside the specified power modes). These additional power modes are translated to voltage modes and if any of them is not present in the PST yet, it is added to the UPF specification.

The generated UPF specification for power switches, isolation, and retention contains the control signals. Isolation and retention requires only one signal for each element, but power switch can require multiple signals – it depends on how many voltages the power switch has to switch between. These control signals are driven by the power-management unit.

B. *Power-Management Unit Synthesis*

The power-mode changes are modelled in the functional ESL design using the *POWER_MODE* variable. After the extraction of power-intent information from the functional model, the *POWER_MODE* variable is changed to the enumerated type. Enumerators of this variable are the identifiers of the specified power modes. The actual values, which these enumerators represent, are unsigned integer values. It means that the value of the first specified power mode is 0, the next power mode is 1, and so on. An existing functional high-level synthesizer (e.g. Catapult or C-to-Silicon Compiler) generates an RTL implementation of the ESL specification. The *POWER_MODE* variable is preserved or it is synthesized into a register, depending on the support of enumerated type in the target language. Whether it remains an enumerated-type variable or it is a register, *POWER_MODE* drives the input of the power-management unit. This value is passed to a power-mode determination entity (see Fig. 3), in which a control-signal based encoding of the power mode is determined. Such a target power mode is passed to a power-state machine, handling the power-mode transition through a sequence of intermediate power modes.

Since the power-intent specification synthesis generates the required power-management elements in UPF, we are aware of the required control signals to be driven by the power-management unit. Thus, there is no problem to create an encoder that is encoding the power states of some power mode into the combinations of control signals. The problematic part of the power-management unit is the power-state machine. It has to implement the transitions between all possible power modes, including the intermediate modes. The intermediate power modes are transparent to the power-mode determination entity, i.e. these cannot come as a target power mode at the power-state machine inputs. Using these intermediate power modes, the power-state machine creates the correct control sequences, following the rules below.

- A block has to stop its operation before it is powered-down.
- Inputs and outputs of a block have to be isolated before it is powered-down.
- The state of a block is retained before the block is powered-down, if the retention is required.

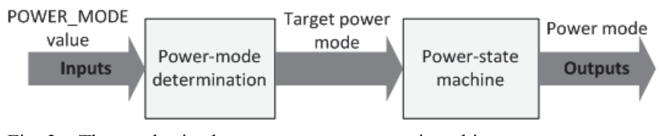


Fig. 3. The synthesized power-management unit architecture.

- A block has to power-up before its state is restored, in case the state was retained.
- A block has to power-up before the isolation is disabled.
- Isolation for a block is activated before the state retention, in case the retention is required.
- The state of a block is restored before the isolation is disabled.
- If the power state of a block is changing from a high-performance state to a low-performance state, the frequency has to be lowered prior to the voltage.
- If the power state of a block is changing from a low-performance state to a high-performance state, the voltage has to be increased prior to the frequency.

The abstract state machine, specified by a designer through power-mode changes in the abstract specification, is thus transformed into a detailed power-state machine containing additional (intermediate) power modes and generating control signals for power-management executive logic. Beside for the power-management elements, the control signals are also driven for a clock-frequency generator, enabling the frequency scaling for system blocks in power domains.

V. Synthesized Power-Management Verification

The abstract power-management specification is already verified at the ESL for presence of syntactical, semantical, and basic structural errors, ensuring its completeness and consistency with the functional specification. Although the automation of the proposed power-management high-level synthesis prevents human-errors introduction, the generated UPF specification should be verified whether the power intent was preserved. Therefore, we have developed an equivalence-checking method between the generated UPF and the abstract power-management specifications. In addition to the power-intent verification, the synthesized power-management unit should be also verified whether it generates correct control sequences and does not violate any of the rules, stated in the previous section. For this purpose, we use the assertion-based verification integrated into the RTL functional verification process.

A. Equivalence Checking

The power intent is extracted from the generated UPF and verified whether corresponds to the ESL specification. Since UPF contains only voltage aspects of the power management a common representation is used for the comparison. This representation includes a list of power domains, a list of power modes, and lists of system blocks in each power domain. Power domains in the common representation are given by their identifiers and voltage states they can reach. Power modes are represented analogously. Thus, the abstract power states are translated to voltage states according to the performance-level assignments. Translation of the UPF to the common representation is achieved using a quasi-reverse process to the high-level synthesis. The first step involves comparison of power domains. The common representation of information extracted from the ESL specification has to contain all UPF-extracted power domains except for the top domain. The lists of assigned system blocks to each power domain have to be the same. At last, the power states of power domains and power modes are compared. Each ESL-extracted voltage state has to be present in the UPF-extracted common representation. It means that each voltage state given by the ESL power management is possible in the generated RTL power management. Error messages, produced by this verification process, drive a designer to the error source, speeding-up the debugging process.

B. Assertion-Based Verification

We also propose another synthesis process – a synthesis of assertions, checking whether the control sequences are correct during runtime (simulation). The generation of these assertions uses state-space exploration method. They ensure that the control-sequences rules are not violated and if they are, assertions pinpoint what went wrong. The generated assertions also provide a suitable coverage-measurement support, driving a designer to create test stimuli covering unverified power modes or transitions. In order to speed-up the RTL verification preparation, a power-aware test-bench skeleton is created, containing the generated assertions and required UPF constructs. There is also a simple pseudorandom-verification approach used, randomly switching *POWER_MODE* value, and thus testing various power-mode transitions. However, a designer should create a proper test-bench for functional verification of the whole synthesized RTL model, not just the power-management unit.

C. Power-Aware Verification

The proposed high-level synthesis generates the standard UPF power-intent specification and the power-management description in the VHDL standard language. Thus, existing EDA tools should be used for full power-aware verification. A formal power-aware static analysis should be used to validate the generated UPF specification and a power-aware simulation should be used to verify the correct functionality of the power-managed system. This step is also used for power analysis, determining whether the power constraints have been met. If they are not, the previously proposed methodology assumes that the abstract power-management specification is modified and the currently proposed power management high-level synthesis is used to rapidly create another power architecture at the RTL. It enables relatively easy power-architecture exploration.

VI. Experimental Results

To validate the proposed synthesis method, we have performed an experiment with 10 randomly generated samples of abstract power-management specifications with various complexities. The goal was to synthesize these samples through the proposed high-level synthesis processes and validate the generated RTL power-management aspects in the existing professional EDA tool. For this purpose, we have used Modelsim SE 10.2c simulation environment, especially its UPF static analysis and power-aware simulation option.

The parameters of the generated samples are provided in Table I. They involve the number of power modes in the

TABLE I. PARAMETERS OF THE GENERATED SAMPLES

#	Power Modes	Power Domains	Power States	Blocks	ESL Power Management
1	2	1	2	3	313
2	3	2	2	2.5	500
3	3	4	1.75	3	751
4	10	3	3	2	862
5	5	3	4	2	643
6	10	5	2.4	2	1275
7	7	4	3.5	2	953
8	7	5	3	2	1051
9	5	10	2.1	3	1939
10	3	10	2.2	2	1402

TABLE II. THE OBSERVED PARAMETERS OF SYNTHESIZED SAMPLES

#	UPF	VM	PMU	PM	Control Signals	SVA	Coverage Statements	Directive Coverage
1	1680	2	3300	4	3	3863	10	100 %
2	2706	3	4452	5	4	4749	17	100 %
3	4658	5	8658	12	7	9488	29	100 %
4	5557	9	63304	49	10	17779	81	100 %
5	6035	15	35205	46	13	25912	103	98 %
6	7443	15	199866	106	13	48111	146	89 %
7	8778	28	142992	114	16	52330	175	81.1 %
8	9399	25	131478	101	17	45150	156	96.1 %
9	12232	27	214122	132	19	91620	188	82.4 %
10	13397	25	67691	65	21	50911	126	99.2 %

abstract specification, the number of power domains, the average number of power states per domains, and the average number of system blocks per domain. The last column in the table represents the number of characters required for the abstract power-management specification for individual samples. Some interesting observed parameters and results after high-level synthesis are provided in Table II. The second column (*UPF*) represents the number of characters required for the synthesized UPF specifications. *VM* refers to the number of voltage modes, generated in UPF. *PMU* represents the number of characters required for description of the generated power-management units. *PM* refers to the number of generated power modes, which include additional intermediate modes. The next column represents the number of control signals, required for individual samples. *SVA* refers to the number of characters required for description of the generated assertions. The next column represents the number of generated explicit coverage statements. The last column provides the achieved coverage using the automatically generated test-benches with pseudorandom power-mode switching in 10 ms simulation runtime. All of the synthesized UPF samples have successfully passed through the proposed equivalence checking and have been statically analyzed by the Modelsim power-aware static checks without any error. The generated power-management units have been exercised during short simulation runtime, achieving high coverage of the power modes. No assertion has been violated, meaning that all control-sequences rules have been followed during simulation. The compilation in Modelsim also proved that the generated power-management information is syntactically and semantically correct. The provided results illustrate that many potential human errors are prevented by this process and a lot of time is saved, considering an amount of automatically generated data. There are up to thirteen thousand characters generated for UPF specification, up to two hundred thousand characters for power-management unit, and up to ninety thousand characters for assertions. The manual effort would require days or even weeks (compared to the seconds) for description of the correct power management at the RTL.

VII. CONCLUSIONS AND FURTHER WORK

In this paper, we have proposed the power-management high-level synthesis. Based on the ESL abstract specification, it synthesizes an RTL power-management model, consisting of the standard UPF specification and the functional description of the power-management unit. The proposed verification steps ensure that the initial power intent is preserved after the synthesis. The proposed synthesis also generates many power-management assertions, usable in RTL functional verification. This automated process significantly speeds-up low-power systems development and avoids many potential human errors. Our further work will involve automated partitioning of system blocks into power domains with power-states assignment. It could enable a complete abstraction from power-reduction techniques, applicable through power management.

REFERENCES

[1] M. Keating, D. Flynn, R. Aitken, A. Gibbons and K. Shi, Low Power Methodology: For System-on-Chip Design, Springer, 2007.

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

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

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

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] D. Macko, K. Jelemenská and P. Čičák, "Power-Management Specification in SystemC," in 2015 IEEE 18th International Symposium on Design and Diagnostics of Electronic Circuits & Systems, 2015, pp. 259-262.