Power Modeling and Analysis in Early Design Phases

Bernhard Fischer, Christian Cech, Hannes Muhr
CT RTC ELE ELD-AT
Siemens AG Österreich
Vienna, Austria
{bernhard.bf.fischer, christian.cech, hannes.muhr}@siemens.com

***Abstract*—Low power consumption of electronic devices has been an important requirement for many cyber-physical systems in field. Today, power dissipation is often estimated by spreadsheet-based power analysis. A leading-edge high-level power analysis method has the objective of providing high confidence levels in early design stages, where power design decisions have severe impact. This work examines and compares three high-level power analysis approaches (spreadsheet-based, Synopsys Platform Architect MCO, and DOCEA Aceplorer) by an industrial use case.**

The first chapter introduces into general power analysis concepts. Chapter two presents different power analysis methods and tools that are applied on an industrial signal-processing use case described in chapter three. Chapter four compares these tools and methods by selected criteria such as power modeling effort and quality of results. Chapter five concludes about the investigated methods.

Keywords—power analysis; power modeling; early design phase; electronic system level; comparison; industrial use case

I. INTRODUCTION

Nowadays requirements on Systems-on-Chip go beyond performance, correctness and size (area). Energy efficiency and its impact on system design plays a major role for many kinds of devices such as wireless and autonomous systems (e.g. wireless sensor networks), and others where cooling capability is limited and expensive. For these reasons, an elaborated power analysis method is crucial for successful system design.

An electronic system design flow starts with requirements engineering leading to a system specification. This specification serves as a base for further refinements with high-level models and simulation with virtual prototypes on the electronic system level (e.g. SystemC). Such virtual prototypes offer several advantages such as early software development start, early verification and early architecture exploration. Through simulation with parameter sweeps corner cases can be found that could have high impact on design decisions.

978-3-9815370-2-4/DATE14/©2014 EDAA

Electronic system level modeling is followed by consecutive design steps, from RTL down to the physical level. Generally, the further the design flow is advanced the smaller is the uncertainty for reaching a valid solution, all for the cost of additional design efforts. Design changes are less costly when conducted in early design phases. Therefore, power analysis that takes place early (e.g. specification phase) and offers sufficient accuracy is an asset.

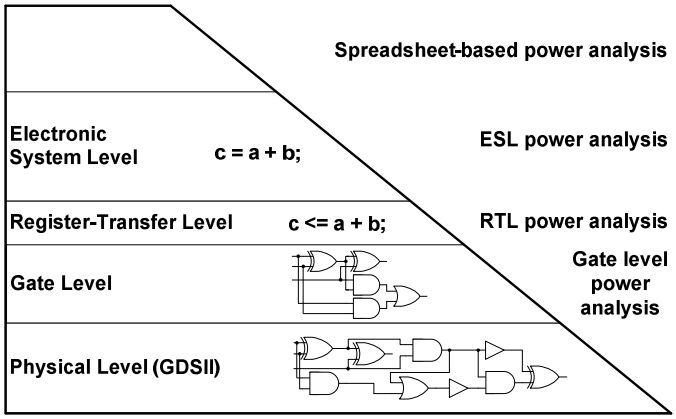

Figure 1: Basic Hardware Design Flow

Power modeling and analysis tools are available for each abstraction level (see Figure 1). Power modeling effort and simulation time increase towards the physical level. For example, a simple Adder component at register transfer level (RTL) is expanded into gates and wires at the gate level. Additionally, after the place & route phase the wiring will have changed, as well as extra gates (e.g. buffers) might have been introduced. These steps insert a lot of detail and increase the design complexity. On the electronic system level power saving potentials between *40-70%* can be expected, *10-25%* on the RTL, and smaller than *10%* on the gate level [8].

What to expect from power analysis? In the best case, power analysis gives reasonably accurate power estimations and takes only short simulation time. It can only be as good as the information provided – power models with elaborated power

state machines and reasonably accurate parameter values can lead to useful results. Important prerequisites to power modeling are well-defined use cases that contain timing behavior next to at least coarse-grained functional behavior. Of course, implementing additional power analysis features can slow down simulations. However, simulation performance is usually not that much of an issue for power estimations on the electronic system level.

Power analysis approaches based on statistical power data without simulation (e.g. spreadsheet-based power calculation) have a major drawback. Power estimations from calculations with empirical data provided by IP datasheets, or with data from low-level IP simulation (e.g. gate-level power analysis) are mostly averaged values for best-case, average-case, or worst-case scenarios. Gaining an oversight about which architecture setup causes the most optimized (according to the requirements) power dissipation is hardly possible. Power estimation methods that just apply such data in simple calculations are useful for rough power estimations, but cannot show the dynamic power behavior of components over time. Power analysis, based on a simulation environment with proper monitoring capabilities, can dissect the power behavior of components in detail. Knowing the power behavior over time gives the ability to identify time intervals with power peaks – enabling focused system architecture optimization. Worst-case power dissipation can be identified and treated, as well as average power dissipation might be enhanced by exploring other architecture and configuration scenarios.

II. HIGH-LEVEL POWER MODELING TOOLS AND METHODS

The expressiveness of power analysis methods depends on the information detail residing inside models, such as:

(Non-)Functional specification for modeling a system's information and activation flow (activation sequence, task dependencies, execution order, etc.). Power state machines rely on that information, since they change their states accordingly to system functions.

Temporal system behavior. Timing information for the activation of system states has to be specified, e.g. in order to define the timing properties of tasks.

Design units implemented on a reasonable abstraction level for power modeling (e.g. SystemC with TLM sockets instead of clock-synchronous designs).

Power state machine specification. Power state machines have to be specified on a reasonable abstraction level. Power models should be kept simple, e.g. possibly make them only dependent on a single source.

Component-specific power dissipation (static, dynamic) has to be obtained either from vendor datasheets or other statistics for customized designs (e.g. data from component reuse, experience, or low-level experiments).

A. Spreadsheet-based Power Analysis

Spreadsheet-based power estimation methods are usually intended for calculating worst-case, average-case and best-case power dissipation for a given HW architecture with a certain component activity scheme based on a specific use case.

Power dissipation of electrical components is transformed into heat. Therefore, it is necessary to calculate the power dissipation and the resulting junction temperature of the device. If the calculated junction temperature exceeds the limits defined in the datasheets, a proper heat sink has to be designed.

The spreadsheet-based power estimation method uses simple equations to analytically calculate the power dissipation for each component of the HW architecture. Some chip vendors provide their own calculation tools or spreadsheet templates to calculate the power dissipation for a device or technology.

One of many ways for power dissipation calculation is given within the Synopsys Power Compiler Manual [6]. In the following, a method for calculating power dissipation of discrete components is presented [2]. In this calculation approach the power dissipation is defined as:

$$P_{(total)} = P_{(dynamic)} + P_{(short\text{-}circuit)} + P_{(leakage)}$$

The part of the power dissipation that is generated when a transistor changes its state is also called switching power and consists of dynamic and short-circuit power (which can often be neglected for logic design). Switching power can be calculated as:

$$P_{(switching\ power)} = V * V * f * C * 0.5$$

V ... logic high output voltage
f ... switching frequency
C ... load capacitance

Leakage power dissipation starts as soon as a device is powered. Various effects cause leakage power due to small currents flowing at almost all junctions (e.g. subthreshold leakage and gate induced drain leakage).

The calculation of the power dissipation for each component depends on available information provided in datasheets. The main obstacle with this method is the high effort for preparing the calculations for even an average HW design with hundreds of components combined with several use cases. The complexity of the calculation increases with the number of components, use cases and calculation scenarios. For this reason modeling of detailed working phases is complex and error-prone.

B. Power Analysis with Synopsys Platform Architect

A more detailed power modeling and analysis method takes static and dynamic power behavior into account. For this, models with system timing information are necessary. Monitoring and analysis of these models requires an infrastructure for power modeling. Depending on the design environment, such could require the implementation of a power simulation engine,

monitors, loggers, a database middleware, power trace analysis classes, and many more. Furthermore, such an infrastructure has to be designed carefully to allow a clear separation between functional and non-functional models, for supporting exchangeability and quick design changes. Manually implemented and verified infrastructure requires efforts, which can be saved by utilizing the infrastructure shipped with a tool.

Synopsys Platform Architect MCO (in short PA) provides a simulation environment on the Electronic System Level and the means to create power models in parallel to functional models (e.g. SystemC components). Power modules provided with the power library are platform-agnostic. So-called *Power Analysis Modules* (PAMs) allow the definition of power state machines triggered by events. The decoupling of design units and power models ensures that there is no or little impact caused by the power models on the design units, which makes them easily replaceable.

Power Modeling Flow

Power modeling with PA (Figure 2) begins with establishing fundamental information such as functional specification, temporal system behavior and power behavior models with power dissipation. Once these are available, task graphs, system architecture and power models are designed and simulated. The simulation results are evaluated in *VPExplorer* together with 3rd party spreadsheet software for CSV files.

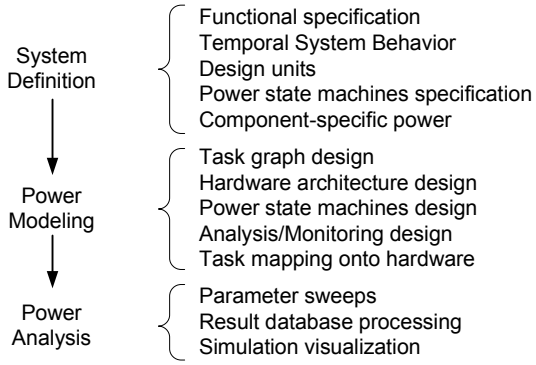

Figure 2: Power Modeling Flow with PA

In brief, the ESL design flow with PA, *Interconnect and Memory Subsystem Performance Optimization* (IMPO), spans across the requirements phase down to the validation phase of a design [3]. Concerning analysis and architecture exploration and optimization, the flow provides batch simulations over a set of simulation parameters (e.g. task durations, bus widths, etc.) that are defined comfortably in spreadsheets. Simulation results are written into a database that is visualized with VPExplorer and could also be processed and exported to spreadsheets. At this point, the effects of various parameter values on the design can be obtained.

C. DOCEA Aceplorer

In contradiction to Platform Architect, which beside the discussed usage for power modeling aims at the exploration of a wide range of system level properties, such as computing power, memory needs and communication throughput, DOCEA's Aceplorer is dedicated to modeling and analysis of power dissipation. To this end it splits the electronic system into power components, their interconnections and use cases. Power states specify the power consumption behavior of components, whereas the interconnections describe the power distribution. The temporal flow of power states is defined in scenarios (see example in Figure 3).

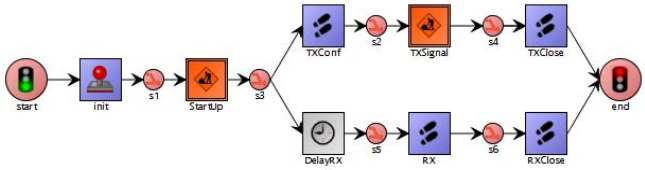

Figure 3: Example scenario in use case perspective

Scenarios consist of steps, which define how long a given component, or a set of components, associated with a task remain in a specific power state. Aceplorer embeds this methodology in a well-structured, hierarchical, object-oriented graphical representation on top of a Python scripting layer. The simulation results include average power consumption figures, as well as their distribution over time, for every hierarchy level from the top down to the leaf power components.

III. Power Analysis Use Case

The following signal processing system serves as an actual use case for the different power modeling approaches. This system (Figure 4) generates a specific signal, sends that signal via actuators through an amplifier setup, and then measures returning signals, followed by signal analysis and data processing.

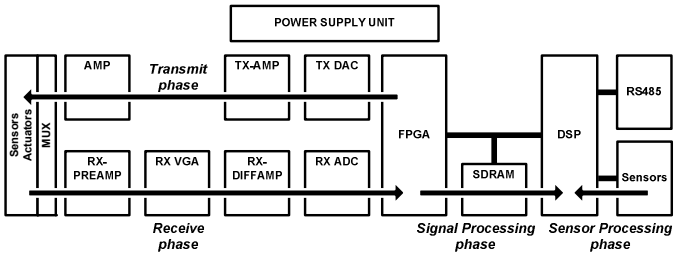

Figure 4: Signal processing use case schematics

System Behavior

The timing diagram for the signal flow is given in Figure 5. The FPGA generates the signals, which go through several amplifiers until they reach the multiplexers ("Transmit" phase). Received signals coming from the multiplexers are forwarded

into the FPGA's memory ("Receive" phase). Another round of Transmit and Receive follows, and then the DSP fetches and processes the received data from the memory ("Receive Signal Processing" and "Sensor Processing" phase).

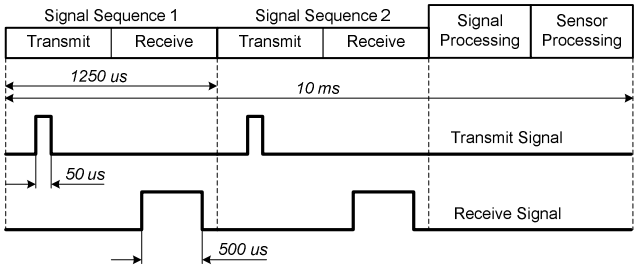

Figure 5: Use case timing

IV. Simulation Results and Comparison

This chapter describes significant power modeling steps taken with the respective modeling tool. The spreadsheet-based approach is only considered for comparison in the conclusion chapter.

A. Simulation Model with Synopsys Platform Architect

Figure 6 shows the top level simulation model in Synopsys Platform Architect MCO. Each component is modeled as Virtual Processing Unit (VPU) and changes its power dissipation value when processing data. A VPU consists of at least one processing unit that executes tasks and providing its internal processing state on an external port. This processing state is used to trigger power state changes in the simulation model. The power states of the amplifier modules and converter modules are controlled by the FPGA. In the simulation model a separate SystemC module (enable controller) generates enable signals which trigger the power states of corresponding PAMs.

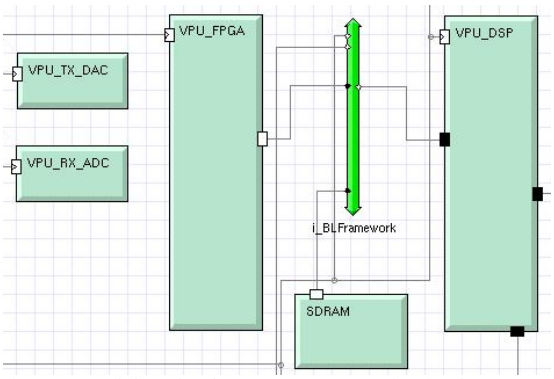

Figure 6: Partial top level PA system perspective for the use case

The whole power model simulations run for a total time of *50 ms* with an analysis interval size of *10 us* (PA calculates the power measurements statistics as sum and average values) – this includes five use case sequences. Figure 7 shows the system power behavior of one complete use case sequence (*10 ms*) including power events and power states of all components. Two bursts run through the signal chain, starting with component activations at about *10 ms* and *11 ms* respectively. The periodic bumps (e.g. *13.4 ms* and *14.4 ms* with *800 mW*) are caused by the DSP accessing the SDRAM.

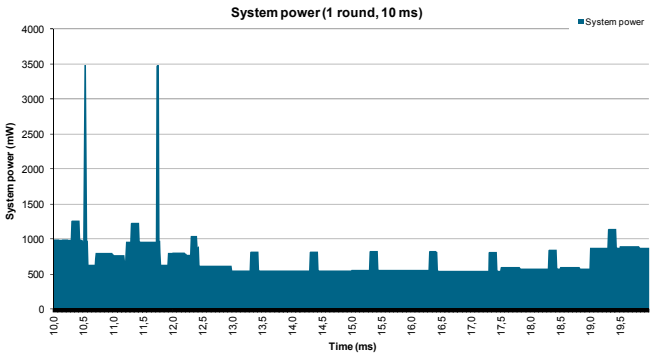

Figure 7: System power dissipation for the use case

B. Simulation Model with DOCEA Aceplorer

The top level Aceplorer representation of the introduced signal processing use case is depicted in Figure 8. Originating from the power source module all supply voltages are distributed to the components, which calculate their drawn currents on every supply interface.

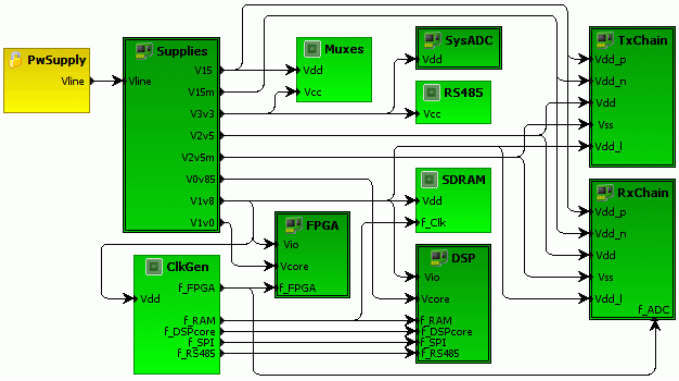

Figure 8: Top level Aceplorer power perspective for the use case

The blocks with a bold outline are so-called macro-components, which contain a collection of further macro- or leaf components (e.g. TxChain, RxChain). In addition to the supply interfaces logical interfaces are used to propagate state information, like operating frequencies or activity rates.

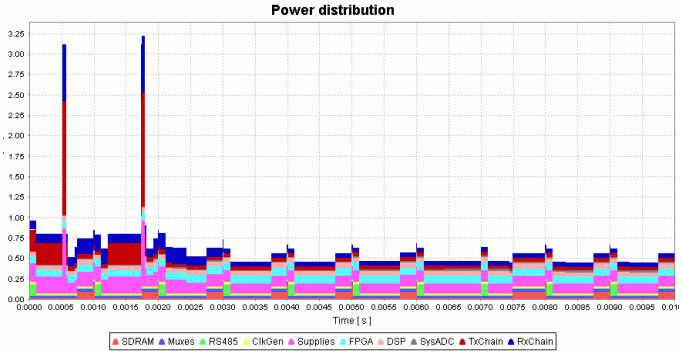

Figure 9: Aceplorer power dissipation simulation result

A comparison of Figure 7 and Figure 9 shows that both PA and Aceplorer are able to produce results of similar quality. The minor quantitative differences originate from diverging interpretations of datasheet information from the design teams rather than from inabilities of the tools.

V. CONCLUSION

Each of the compared power analysis approaches has its own home-ground, which can be seen in TABLE I. Spreadsheet-based approaches are valuable for determining quick average values, but power simulation with timing is hardly possible with justifiable effort. This is where dedicated power modeling tools have their strengths. Temporal information paired with power behavior eases identifying power peaks and their causes. With the spreadsheet-based approach such peaks might easily be missed. Specialized tools provide power modeling and simulation infrastructures such as adjustments of power analysis granularity and visual debugging capability for component power break-down. Within such tools power-hungry components can easily be re-located to more "quiet" system activity intervals.

PA offers a fully-fledged virtual prototyping environment that is enhanced with a power analysis infrastructure. If ESL models are already available the implementation effort for the additional power state machines pays off with the capability of analyzing power next to functionality (SystemC/TLM models) in a single environment. In larger systems with high concurrency the task-driven approach with PA can ease system optimization activities significantly (e.g. optimal bandwidth), since scenario sweeps over various sets of parameters is supported.

Aceplorer is primarily focused on plain power analysis. Components are designed with consideration of voltage domains and input currents (in comparison, dissipation power is directly assigned to power state machines in PA). The effort for designing power analysis models is lower than for the spreadsheet-based approach with the benefit of detailed power simulation. In comparison to PA, the effort is as well lower, but this is due to the combination of (non-)functional system level models and power state machines in PA.

At the end, the use cases decide which power analysis approach brings the most benefits, depending on analysis detail, license costs and system level model integration.

TABLE I. COMPARISON OF POWER MODELING APPROACHES BASED ON THE USE CASE

	Power modeling approaches		
	Spreadsheet-based	**Power modeling with PA**	**Power modeling with Aceplorer**
Focused design phase	Specification	Specification / early design	Specification / early design
Intended use cases	Scenarios such as min, max, average, sum	Min, max, average, sum, optimization, dynamic behavior	Min, max, average, sum, optimization, dynamic behavior
Expressiveness	Static use cases	Parametrizable use cases	Parametrizable use cases
Power modeling style	Equations based on component datasheets	Power state machines sensitive to functional model	Power states controlled by scenarios
Accuracy	Averaged values over large time intervals	Power dissipation over time	Power dissipation over time
Modeling effort	Depends on the intended accuracy	Power state machines need to be implemented next to a functional specification	Low overhead for building the power model structure
Simulation performance	Instant results	Approximately 1 minute	In the low seconds range
	(50ms of the given use case with 1 GHz Dual-Core AMD Opteron(tm) Processor 8222, 64 GB RAM)		
Additional benefits	Straight forward post-processing without additional tools	Integration of SystemC functional models, bus transaction tracing	Co-simulation with thermal analyzer
Identification of potential optimizations	Only via averaged power values (peaks might be missed)	Easy identification of optimization possibilities	Easy identification of optimization possibilities
Simulation results presentation	Power values in customized spreadsheet diagrams	Graphical power values analysis with VPExplorer	Graphical power values visualization
Cascading	Manual linking of inter-dependent component sheets	Additional manual effort to model dependencies between components	Support for cascaded power modules due to separation of voltage and current measurements

REFERENCES

[1] Synopsys Inc., www.synopsys.com

[2] Preeti Ranjan Panda, B. V. N. Silpa, Aviral Shrivastava, Krishnaiah Gummidipudi, "Power-efficient System Design", Springer US, ISBN 978-1-4419-6388-8, 2010.

[3] Synopsys Inc., "Synopsys Platform Architect Product Family: Example of Interconnect and Memory Subsystem Performance Optimization Flow", Synopsys Inc., 700 E. Middlefield Road Mountain View, CA 94043, Version G-2012.06-SP1, September 2012.

[4] Synopsys Inc., "Synopsys Platform Architect Product Family: MultiCore Optimization Example Manual", Synopsys Inc., 700 E. Middlefield Road Mountain View, CA 94043, Version G-2012.06-SP1, September 2012.

[5] Synopsys Inc., "Synopsys Platform Architect Product Family: Task Modeling and Virtual Processing Unit User's Guide", Synopsys Inc., 700 E. Middlefield Road Mountain View, CA 94043, Version G-2012.06-SP1, September 2012.

[6] Synopsys Inc., "Power Compiler™ User Guide", Synopsys Inc., 700 E. Middlefield Road Mountain View, CA 94043, Version H-2013.03, March 2013.

[7] Tcl (Tool Command Language), http://www.tcl.tk.

[8] Ridha Hamza, "Winning the power and temperature battle with ESL exploration", Docea Power, 166B, Rue du Rocher du Lorzier, Moirans 38430, France, online article, http://www.techdesignforums.com/practice/technique/winning-the-power-and-temperature-battle-with-esl-exploration/, June 2010.