ABSTRACT
With the growth of number of transistors, thermal density and market drive towards battery power, the necessity to develop low power integrated circuits is evident. There are several methodologies and techniques that help in the development of this type of SoC. There is consensus that the techniques when applied in the initial phases of the design flow, especially in design and architecture of the system have a higher impact factor in relation to those applied during the implementation or layout phase. Considering this fact, the possibility to accomplish functional verification of the system with low power design modeled in TLM and RTL is attractive. The purpose of this paper is to show a method for functional verification of power gate design in RTL. A case study is presented to demonstrate the application of power gate using new features for the OSCI SystemC simulator developed for this purpose and the accomplishment of the functional verification of the design using the VeriSC methodology.

Categories and Subject Descriptors
B.5 [REGISTER-TRANSFER-LEVEL-IMPLEMENTATION]: Miscellaneous; B.5.3 [Reliability and Testing]: Test generation.

General Terms
Verification

Keywords
Functional verification, Simulation, Power gate, SystemC, RTL.

1. INTRODUCTION
In recent years the semiconductors industry suffered modifications in its manufacture process due to the new requirements that the consumer market has been imposing. These evolutions introduced great challenges to the design with even more complex chips and high density of transistors in a tiny silicon area, motivating an inevitable increase of power and consequently heat dissipation in the chips [1]. Ahead of this fact, the industry and academic research centers are searching for new techniques to ease the power density problem and enable the development of low power ICs.

System design and architecture of a SoC can be sufficiently complex due to the numerous ways of the power plan [1]. Considering this fact, is common to use TLM (Transaction-Level Modeling) and RTL (Register Transfer-Level) to accomplish functional verification of SoCs (System on Chip).

The purpose of this work is demonstrate the functional verification of design containing the principles of power gate design implemented in RTL. Power gate consists of powering down internal modules of the SoC. New techniques will be presented in this paper, which allow the modeling and simulation of such designs through the simulator kernel enhanced by new functions to model modules blocking and resuming. This enables the dynamic behavior of the SoC to be verified before the synthesis on the target configuration (like ASIC).

To demonstrate the technique application the new VeriSC methodology [2] was adopted. It uses SystemC language and features a verification flow that does not begin by DUV (Design Under Verification) implementation. In this flow, the testbench implementation and reference model are done before DUV implementation. With the accomplishment of the design functional verification containing the implemented techniques in RTL, the probability that this design shows any fault in future steps in the SoC development flow due to bad application of some low power technique is reduced.

The remainder of this work is organized in the following way: Section 2 shows the current methodologies and techniques for low power design development; Section 3 shows a power gate overview; Section 4 shows the functions newly added in the SystemC simulation core; Section 5 shows the design description and results; Section 6 the final considerations.
2. METHODOLOGIES FOR FUNCTIONAL VERIFICATION OF LOW POWER DESIGN

Several power saving techniques can be applied during the development of a SoC, from conception of the system architectural design until prototyping. It is consensus that the techniques for low power SoC development, when applied in initial phases of the conception flow, especially in the system design and architecture phase, possess a bigger impact factor in relation to techniques applied in the implementation phase [1]. Figure 1 shows a diagram expressing the impact ratio of the decisions, where 80% of the power savings are reached due to decisions taken before of the implementation [4] [7].

The most efficient techniques for power savings are shutting down power or reducing voltage level of some regions of the device, known as power domains [1]. In the first design generation power-aware SoCs had a few power domains, but recent design have more than 20 domains, resulting in numerous power modes [8]. This leads to an exponential growth in the number of power-up and power-down transitions to be verified before the silicon prototyping. Problems related to power can be extremely critical, resulting in the need to modify the SoC.

The designers have several ways to manage power, some are implemented easily and others are complex with respect to operation frequency or area. The main techniques applied for low power SoC development are [1]:

- **Clock Gate**: Technique based on clock signal halt in determined modules that are not used during functioning of the SoC.
- **Power Gate**: Technique based on shutting down power of determined modules of the SoC.
- **Multi-VT**: Technique based on adoption of the standard cell library with different VT level.
- **Voltages Islands**: Technique based on circuit powered with different voltage levels.
- **Gate Level Power Optimization**: Technique based on remapping of logic gates for a combination of equivalents gates.
- **Dynamic Voltage Scaling**: Technique based on variation of the voltage applied.

- **Dynamic Frequency Scaling**: Technique based on variation of the operating frequency.

To determine which techniques designers most use to manage power, a research was conducted with 115 companies, demonstrating that the participant designers correspond to 31% in the wireless telecommunications sector, 21% in class of portable electronic devices and 27% of networking equipment. Figure 2 shows the preferences of the researched companies [4].

![Figure 2: Popularity of the techniques of power management](image)

Currently five technologies are applied to accomplishment the verification: PDLM (Power Definition Markup Language) specification, power-aware simulation, verification of the powered structure, relationships power assertions and formal analysis of the logic control of power. These technologies are essential components of an effective methodology for functional verification aiming at high-reliability of low power design techniques application [8].

- PDML as CPF (Common Power Format) and UPF (Unified Power Format) provides a way to specify the architecture of the powered design independently of the description in RTL. The PDML specification includes the powered connectivity, shutdown behavior control and interaction between the different power domains. Several development tools leverage the use of a PDML file, so that the planning architecture of the powered can be properly implemented and verified. All features of powered chips can be succinctly captured in PDML.

- Power-awareness simulates the inclusion of power management features, in which the RTL simulator reads and interprets the PDML, where the behavior in power-up and power-down can be modeled. A power-aware simulator can also check the responses to the events related with the power.

- The verification of the powered structures can be accomplished at the beginning of the architectural power modeling.

- Assertions, some control functions and relationships with time are specified in PDML, they can be automatically transformed into assertions using a standard format, such as the assertions in SystemVerilog or a proprietary language for specification.
• Formal analysis, being exhaustive in nature, so it can find all faults related to the assertions, whereas in simulation it is likely that a given set of tests does not exercise all important behavior with respect to power.

During implementation of RTL-to-GDSII flow, most projects of integrated circuits have been successful in incorporating power management, using techniques such as clock-gating, power-gating, Multi-VDD and etc. The implementations of these techniques are not fully automated and the evaluation of the tradeoff between the techniques is not easy. Some of these power analysis and optimization are assisted by EDA (Electronic Design Automation) tools [5] [6].

3. OVERVIEW OF POWER-GATE

Power gate strategy is based on adding mechanisms to turn off blocks within the SoC that are not being used, the act of turning off and on the block is accomplished in appropriate time to achieve power saving while minimizing performance impact [1].

When the event of turning off happens, the energy savings is not instantaneous due to thermal issues of the previous activity and the nature of technology is not ideal for power gate. In the event of turning on the block requires some time that cannot be ignored by the system designer for the block to retake the activity [1]. Figure 3 shows an example of the activity of a block with power gate implemented.

![Figure 3: Profile with Power Gating](image)

3.1 Power-Gate Design Structure

Differently of a block that is always active, a power-gate block is powered by a power-switching network that will supply VDD or VSS power gate block, the CMOS (Complementary Metal–Oxide–Semiconductor) switches are distributed within or around the block. Control of CMOS switches is accomplished by a power gating controller. In some cases it is necessary to retain the state of the block during the turned off period to restore the state when it is turned on. This restraint is implemented using special flip-flops. Figure 4 shows the diagram with the structure of the SoC with power gate.

![Figure 4: Block Diagram of a SoC with Power Gating](image)

4. NEW SYSTEMC FEATURES

In order to implement the functional verification of low power design using a functional simulator, a similar approach developed to simulate partial and dynamic reconfiguration [9, 10] was selected. This is a bottom-up approach adding functions to activate and deactivate modules by the programmer, simulating the run-time reconfiguration.

SystemC is open-source, free and enables the modeling and simulation at TLM and RTL using object-orientation concepts [3]. It does not allow deactivation of modules during simulation, but as an open-source tool, it could be modified for these purposes.

The strategy implements two new special functions for turning on and off modules during simulation named `sc_lp_turn_on` and `sc_lp_turn_off`, respectively. These functions were written modifying the SystemC kernel source-code. The routine `sc_lp_add_constraint` was also created and is used to store modules attributes about their energy consumption and the turn-on delay, always present when a module is re-activated on chip. Table 1 presents how the functions signatures in `sc_simcontext.h` SystemC kernel file.

<table>
<thead>
<tr>
<th>Table 1: Functions declarations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>extern void sc_lp_turn_on(std::string module_name);</code></td>
</tr>
<tr>
<td><code>extern void sc_lp_turn_off(std::string module_name);</code></td>
</tr>
<tr>
<td><code>extern void sc_lp_add_constraint(std::string module_name, sc_time wakedelay);</code></td>
</tr>
</tbody>
</table>

A linked list is used to store the names of the modules that must be not executed (turn-off). The routine `sc_lp_turn_off` adds the module name to the list, while `sc_lp_turn_on` removes the module from the list, allowing it to be executed (activity). Another list is kept to store the module constraints (wake delay and energy consumption). This list is required when the `sc_lp_add_constraint` function is called. In this case, constraints are added to the list and cannot be removed, just overwritten.

The extern key-word indicates that the routine can be called outside the `sc_context` class. In other words, those functions can be called by user code on regular simulations.

5. DESIGN DESCRIPTION

An example design was described in SystemC RTL language [11]. Its specification is the following, serially receive words of 4 bits in the BCD format and convert to segments 7 format, after of the data encoding are available in the output. Figure 5 shows the design block diagram.

![Figure 5: Design block diagram](image)
5.1 System behavior and implementation
The system has the objective to demonstrate the application of low power design and accomplish the functional verification of it. To achieve this objective two versions of the design were implemented in RTL using SystemC version 2.2.0, one with and another without the features for low power design.

- **Design Version 1 (DV1), without low power design:** The implementation was done in a simple way. The two Serial2Parallelo and BCD2Segment7 converters were implemented, the output of Serial2Parallelo converter is connected directly to the BCD2Segment7 converter. The operation of the circuit is follows: While the Serial2Parallelo converter will parallelizes the bits that are read the BCD2Segment7 converter encodes the values provided by the Serial2Parallelo converter, after reading complete word of 4 bits by Serial2Parallelo converter, this module provides the result in the output.

- **Design Version 2 (DV2), with low power design:** The implementation is similar to the preceding, differing in the addition of the PGC (Power Gating Controller) module. The operation of this version differs from the previous in the following point: while the Serial2Parallelo converter reads the word of 4 bits, the PGC module leaves the BCD2Segment7 module turned off with its state output retained. After reading the complete word the PGC module turns on BCD2Segment7 module, which from this point will wait a time of 150ns corresponding to the delay between wake and activity, after this time the module accomplishes its function and on the step next provides the value in the output.

5.2 Functional Verification
VeriSC methodology adopts projects with hierarchy concept, therefore a project can be divided into parts to be implemented and verified [2]. BVE-Cover library was chosen to accomplish the functional verification with coverage of the design. Testbench components were implemented following each step of the methodology. To implementation of the elements was used SystemC language (SC). Figure 6 shows the testbench structure.

![Figure 6: Testbench structure](image)

Several simulations were performed with different versions of SystemC simulator and design:

- **SystemC + DV1:** At this stage were used the original SystemC version 2.2.0 and the first implementation of the design.

- **SystemC–LP + DV1:** At this stage were used the SystemC with LP functions added and the first implementation of the design.

- **SystemC–LP + DV2:** At this stage were used the SystemC-LP version and the second implementation containing the low power design.

5.3 Results and Analysis
At the end of the functional verification some results were extracted. One of them was obtained at end of the second simulations that consists in the validation of the modified simulator. DV1 was successfully verified using the two versions of SystemC simulator, as much in the original version as in the modified version, thus demonstrating that the new functions do not interfere with the others functions.

Twenty lines of extra code were added to DV2 to implement the PGC that consists of a state machine calling the new SystemC methods. The added lines represent 12% of the total lines of DUV code. With the introduction of the PGC it was possible to simulate low power design. During the simulation we can verify the power gate principles operating. Figure 7 shows the waves forms of the designs.

![Figure 7: Waves forms](image)

Figure 7A presented the changes of state of the BCD2Segment7 module while the Serial2Parallelo module parallels the words. In Figure 7B is possible to see that while occur the parallels the word the BCD2Segment7 module maintains output state constant, due to retention that occurs prior to shutdown. At the end of the reading of the word from the BCD2Segment7 module back to the activity to accomplish its function and provide the value in the output.

Another point that can be observed is the delay between wake and activation times, Figure 8 contains a zoom of the wave forms of the Figure 7. For this design was used a delay of 150ns, due to this, the delay “B” is precisely the time for the module to return to full activity. After sending the new result to output the PGC module puts the BCD2Segment7 module in sleep state. Comparing the time when the BCD2Segment7 modules update the results can be observed that delay “A” is equal to delay “B”, showing the delay that the module suffers to return to full activity.
A negative point discovered in this work was the loss of performance of the simulator, fact occurred due to the strategy to modify the simulator scheduler. Figure 9 show graphic with the simulators performance.

6. FINAL CONSIDERATION
It is consensus the need of low power SoC development. There is a variety of development methodologies supported by a variety of EDA tools that are available to designers. This work demonstrated the use of the modified open-source SystemC simulator to meet the needs of designers to accomplish the functional verification of power gate design in RTL. The decrease in performance of the SystemC simulator is a negative point, to be studied in future work. The functional verification of power gate design in RTL provides the reduction in the risk of applied this technique in steps future within the development flow of the SoC.

7. REFERENCE