Low Power Design and Verification Techniques

Stephen Bailey
Gabriel Chidolue
Allan Crone
Mentor Graphics

Mentor Graphics®

Introduction

Both market forces and process technology have driven power to the forefront of all factors constraining electronic design.

The increasing demand for high-performance, battery-operated, system-on-chips (SoC) in communication and computing has shifted the focus from traditional constraints (such as area, performance, cost, and reliability) to power consumption. Just as important, though not so obvious, is the need to reduce power consumption for non-portable systems, such as base stations, where heat dissipation and energy consumption are critical concerns.

At process nodes below 100 nm, power consumption due to *leakage* (static energy loss) has joined switching activity (dynamic power consumption) as a primary power management concern. It has been reported that leakage constitutes over 40 percent of total power expenditures at the 65 nm technology node [J. Kao, et al. 2002]. The quadratic dependency of power leakage on total transistor count easily qualifies leakage optimization as a key design objective. Consequently, designers have redirected their efforts toward exploring various techniques that reduce leakage and, thereby, increase battery life in the final product.

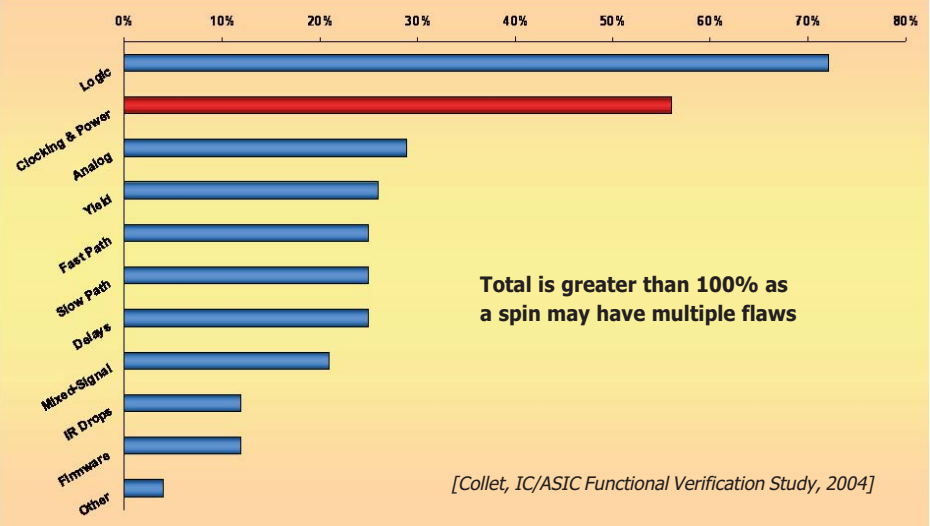

Figure 1. Type of flaw found in failed silicon spins.

There are many techniques that have been developed over the past decade to address the continuously aggressive power reduction requirements of most ASIC and SoC designs. They include clock gating, multi-switching (multi-Vt) threshold transistors, multi-supply multi voltage (MSMV), power gating with or without state retention, dynamic voltage and frequency scaling (DVFS), and substrate biasing.

The use of any of these techniques comes at a cost and their benefit varies depending on the technique used. In short, they introduce risk to the product development schedule and impact all aspects of ASIC and SoC development, including design, implementation, and verification. Each technique used must be considered with the rest of the system requirements.

For example, designers will have to choose between power gating with or without state retention. Power gating with retention achieves faster wake up times and preserves state information, but it requires more silicon real estate. As well, retention flops must be powered up when the rest of the domain is powered down, so some power is consumed. Still, of all the low power design techniques, power gating is the most effective in reducing leakage power. However it is also the technique with the most impact to current design and verification flows and methodologies.

Power gating with state retention involves switching off an area of a design when its functionality is not required, then restoring power when it is. The areas created by power gating are called *power domains* (also known as *power islands*). Power gating requires the use of *retention memory elements* to prevent data from being lost in a specific power domain during power down. Retention strategies such as retention flops and latches save state information before a power domain is switched off and restore it when the power domain is turned back on. As well, restore protocols must be used to ensure that the power domain returns to a known good state when powered up. Finally, each power domain must be isolated from the rest of the design when powered down so that it does not corrupt downstream logic.

Power gating requires early verification. However, thus far, verification of the functionality of power domains within the context of a power management scheme has been performed only at the gate level, if at all. Gate–level verification poses many issues such as:

- **Time**: Gate-level simulations are slow
- **Debug**: Debugging at gate level is difficult as the user-defined RTL specification has been transformed into implementation through synthesis
- **Problem Rectification**: It takes longer and requires more resources to resolve functional problems uncovered at the gate level, compared to RTL

For these reasons, waiting to perform power-aware design verification at the gate-level is too costly in terms of resources and design cycles.

This paper describes the basic elements of low power design and verification and discusses how the Unified Power Format (UPF) along with innovative techniques enable power-aware verification at the register transfer level, using traditional RTL design styles and reusable blocks. The result is a multi-tool solution that can be used throughout the RTL to GDSII flow, applying consistent semantics for both verification and implementation.

The Rising Cost of Leakage

At sub-100 nm feature sizes, the material properties of silicon change. At this point, features are so small that silicon is no longer a perfect insulator, allowing electrons to leak across the silicon partitions—much like pins of light pierce an aging sunshade. The smaller the geometry, the thinner the silicon partition, the greater the energy drain due to leakage.

Traditional low power design techniques, such as lowering the power supply voltage VDD and clock gating, become less effective and may even be undesirable because voltage reduction increases circuit delay. In order to maintain performance, designers compensate for this increased delay by decreasing the threshold voltage. Unfortunately, this radically increases leakage current due to the exponential nature of leakage current in the sub-threshold regime of the transistor.

However, the impact of leakage power strongly depends on whether a circuit is a continuously operated circuit or an event-driven circuit [A. Chandrakasan, et al. 1996]. In continuously operated circuits the switching component of power dissipation typically dominates that of leakage. Leakage is the main fraction of the total power for event-driven circuits. Event-driven circuits allow the switching activity to be kept null during idle time through clock gating. Dynamic power is eliminated, yet the circuit still dissipates energy due to leakage. Most handheld devices come under the category of event-driven circuits; in other words, they are characterized by intermittent operations with long periods of idle time (the familiar *standby* mode).

Power Management Techniques

Designers have developed various low power management techniques to reduce leakage power consumption. These techniques make use of some form of *sleep* operation. Placing power gating structures is a well-known technique for reducing power leakage during standby mode, while maintaining high speeds in active mode. During standby one or more portions of the circuit are switched off and the passage from the source to ground is blocked, eliminating leakage in those areas. One side effect of power gating is that data in the storage elements can be lost. As a result, designers use retention memory elements to retain key state data during the sleep mode so that it can be restored correctly upon power up. This is *retention sleep mode*.

Some designs use a conventional sleep operation in which the power supply of the entire design is cut off when the circuit is not in use. Such designs do not require data to be retained in the registers/latches used in the design. *Power down* in notebook computers is an example of such a sleep mode. However, even with the conventional sleep operation, functional verification of the design is required to ensure that the *awake* portions of the design function properly while other parts are *sleeping* and that the system will operate correctly when power is restored to the sleeping logic blocks.

With retention sleep mode, the operation of a logic circuit is stopped only if that specific circuit is not in use. Some low power devices in portable equipment use retention sleep mode during intermittent operations, such as waiting for input from a keyboard or communicating through a slow interface. These devices resume operation without restarting because the common registers and pipeline registers also preserve the data during sleep. These sequential cells are known as *retention flip-flops* (RFF) or *retention latches* (RLA). To implement sleep power management techniques, a well-defined *power management block* (PMB) must be created with specific *power control signals* and power domains (Figure 2).

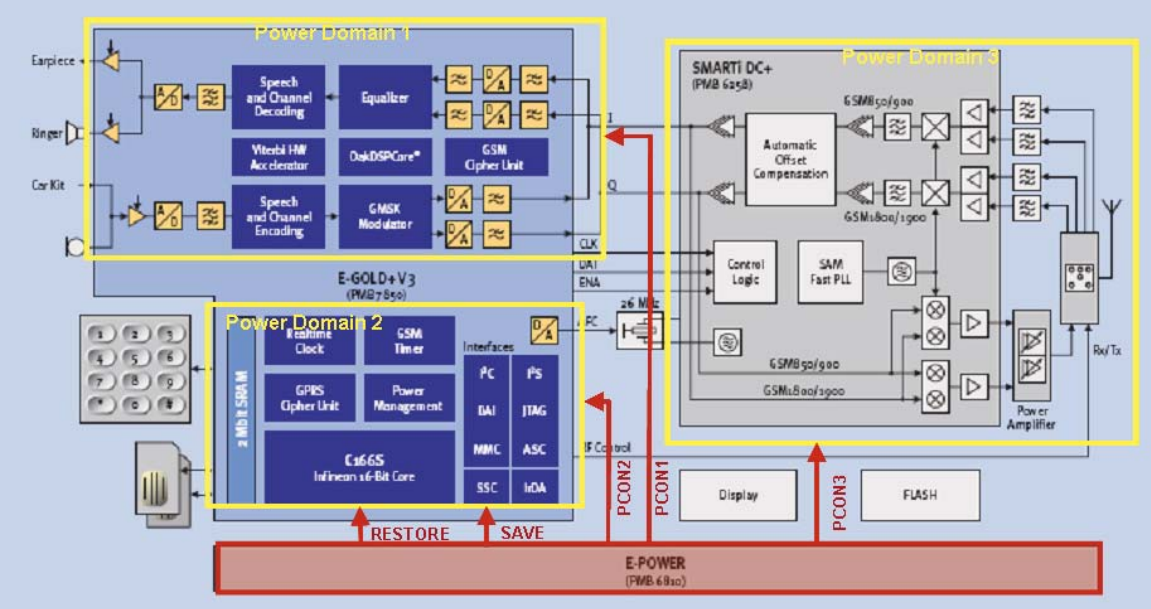

Figure 2. Power management design structure. [original Infineon diagram with added power domains]

Power Management Design Structure

As the power management techniques employ turning off and on various segments of a design, these designs are divided into different power domains, based on areas of functionality that support common operations or tasks. In other words, a power domain constitutes a collection of functionality that can be turned on or off as a whole, and it runs at the same operating voltage level, having a single set of power control signals. Power control signals are used to control sequential retention cells, isolation cells, and the switches that serve as gatekeepers to a domain's power supply.

In addition, every power aware design has at least one primary domain, which is always on. This is known as either the *wake-up* or *always-on* domain.

Retention Memory Elements

To maintain the state of registers and latches during sleep mode, retention elements are employed that can retain their data when in sleep mode. Alternatively, voltage threshold scaling may be used for retention without requiring a bubble latch to hold the retained value. RFFs and RLAs are affected by the power signals that control the RTL region to which they belong. The registers are corrupted when the power is switched off [S. Mutoh, et al. 1995]. Corruption is typically represented by *X* (unknown). The register value is restored after power up if the value was saved successfully before power down and the restore protocol executed successfully. Otherwise, the register value remains unknown until it is set, reset, or a new value latched into it. These elements will behave as normal memory elements when the power is switched on and the power control signals are not asserted.

An example of a retention flip-flop is a clock–low retention FF. These types of RFFs have one control signal, known as *RET*. A clock-low retention FF requires that the clock be gated low during the save

and restore operations and most likely during power down retention as well. When RET is asserted and the clock is low, clock–low retention FFs perform the save operation. When RET is de-asserted and the clock is low, the restore operation is performed. They behave as normal FFs when RET is low.

Isolation Cells

Isolation cells (also known as *clamps*) are logic gates that determine the values of a power domain's input or output port when the domain is powered down. (Note: A power domain's input and output ports are the ports on a logic block within the power domain that is connected to a fan-out or fan-in located in another domain.)

Isolation cells are necessary because each power domain represents a design area comprised of particular features, and each feature corresponds to an area of physical silicon. Even though they may represent different power domains that can be independently powered on and off, these areas of silicon remain physically connected; therefore, when one domain is turned off, it is still connected electrically to other domains.

For example, when the MP3 component of a cell phone is turned off, it is still connected to the touch pad circuitry, which remains on. This is a potential source of leakage, because electricity seeks a balance in voltage level.

There are other reasons why it is important to control the values of a power domain's ports when it is powered down. For example, when a wire is used as a reset in a downstream block that is powered on, it should not become active incidentally because the power is shut down in an upstream domain (or temporarily toggles to an active level during the wake-up power-on sequence). If it is active low, it may start as a 1, but if it doesn't continuously drive a 1, it could end up floating and drop below the threshold and be seen as logic 0, causing that part of the design to reset when this is not intended.

To prevent this, the output of the upstream domain can be clamped to its current value, or a specific value, when isolation is enabled just prior to power shut down. When the clamp value is a "don't care," the typical default clamp value is 0 (low), as 0 is 0 at any voltage level, eliminating the need for level shifting of the isolation value. Clamps maintain the integrity of the downstream power domain. Thus, when one domain powers off it does not corrupt elements that are still on.

Similarly, it may be necessary to isolate input ports. For example, a clock input signal to a domain's logic should be gated when the power is gated. Isolating the clock input signal is one way to implement clock gating.

Moving Low Power Specification to the RTL

As mentioned earlier, power gating requires early verification. Waiting for the gate-level netlist is too costly for a number of reasons, including slow simulation times and more difficult debugging and problem resolution. In addition, information is needed at the RTL to validate that low power techniques are implemented correctly in the early as well as the final stages of the design flow.

Power information can be input at the RTL in two ways:

- Directly specified in the RTL code
- Indirectly specified via a *side file*.

By directly integrating the power information in RTL, it is guaranteed that the corresponding power information is packaged together with the RTL. However, this methodology has a number of disadvantages. It requires that any legacy RTL must be updated to add power information. Organizations typically require full module or sub-system re-verification when the RTL code is changed — a difficult and time-consuming task. Most registers and latches in a RTL design are inferred and not explicitly coded as part of the HDL; the RTL coding style would be significantly impacted if designers were required to explicitly instantiate retention registers and latches wherever persistent state information occurs. Furthermore, when designs go through a technology spin, there is no guarantee that the retention cells will be functionally the same in the new technology library. Similarly, explicit routing of power control signals to retention cells and the instantiation of isolation cells and level shifters within the RTL code create an unnecessarily tight coupling between the design functionality and the low power design intent. Finally, as significant aspects of the low power design intent is related to the technology implementation, it is usually modified more often than the RTL functional specification.

Therefore, the two should be specified separately to facilitate maintenance, changes, and verification. For these reasons, it makes sense to provide the power specification in a side file. If the low power design intent is specified separately from the RTL code, yet is related to it, then familiar RTL coding styles can be maintained and reuse of the RTL maximized. Thus, a means of specifying the low power intent separately from the functionality in RTL is not only very useful but also essential. The low power specification side file should provide the information required to overlay the RTL functionality with the power control network (PCN) and the power aware functionality.

The power specification data provides the low power design intent. The corruption semantics (for that matter, any semantics) are implied by that intent. Thus, the power specification captures the system's power states; enough information about the power supply network to know how the power supply is distributed and controlled for each state; which registers need retention; how isolation and level shifting are handled; and how retention is performed. This information can be used to functionally verify the design at the RTL (or higher) and ensure that the low power design intent is implemented in the gate-level design through synthesis. Synthesis tools can use the same information to create an appropriate power aware gate-level netlist automatically.

UPF: A Portable Low Power Standard Format

Low power specifications are needed at each step of the design flow so that correct power management components can be implemented at the RTL, inferred correctly during synthesis, and placed-and-routed efficiently and accurately in the physical design. This requires a single power format accepted by all tools in the flow at any given abstraction level. A single power format eases

implementation and validation and helps meet design schedules. It must also address reusability, allow early and thorough validation, and have built-in extensibility.

Accellera, an organization focused on identifying and creating new standards and methodologies for the electronic design industry, recently approved a standard for low power design intent specification. This standard is called the Unified Power Format (UPF).

Written in tcl, UPF captures the low power design specification in a portable form for use in simulation, synthesis, and routing, reducing potential omissions during translation of that intent from tool to tool. Because it is separate from the HDL description and can be read by all of the tools in the flow, the UPF side file is as portable and interoperable as the logic design's HDL code.

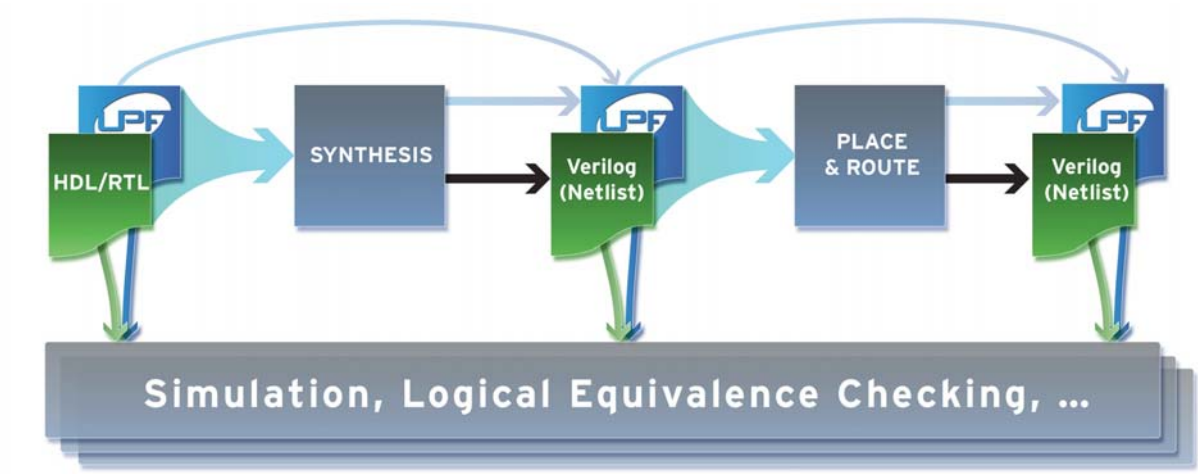

Figure 3. The UPF side file provides a consistent semantic for all tools throughout the design flow.

Defining System Power States and the Supply Network

When designing the low power aspects of an electronic system, you should start by defining the system power states. For example, a system power state may be such that the modem is in sentinel mode, waiting for an incoming call; the information management system is checking for scheduled appointments; and the rest of the system is in sleep mode to conserve power. Such a deep sleep state must be defined in terms of the functionality in the system.

UPF provides two commands for defining a power state table that captures the system power state information. The power state table defines the power states in terms of the supply net states, ensuring integration of the system power design with the low power design implementation. The UPF also allows you to specify all the supply network information needed to verify and implement the power supply distribution, and it provides the control required to realize the system power states.

The supply network consists of power switches; supply ports that are defined for power domains and power switches, supply nets that connect supply ports and logic ports and propagate the supply state; and the supply states. Each supply port has one or more supply states defined. The port may drive only

one state at any given time. That state is propagated by the supply net connected to the port. The power state table is defined in terms of these states. UPF automates the connection of the supply network to the logic elements in the design based on the semantics defined for specific types of supply nets.

		DOMAINS			
		Micro Processor	**Power Control**	**Memory**	**Objective**
STATES	**Sleep**	Off Retain	0.9 V Slow Clock	0.7 V	Leakage
	Conserve	0.9 V Slow Clock	0.9 V Slow Clock	0.9 V	Dynamic
	Overdrive	1.2 V Fast Clock	0.9 V Fast Clock	1.0 V	High Performance

Table 1. System power states.

Specifying Power Domains

Power domains enable the automation of supply network connectivity for primary supplies. A power domain is a collection of design logic elements that share a primary supply. A primary supply consists of a single power and a single ground supply net pair. This definition of a power domain enables the automatic connection of the primary power and ground nets to all logic elements within the domain.

For verification, UPF specifies the semantics of power-off as primary power and/or primary ground are in the off state. The behavioral semantics of power-down mirrors what happens in actual hardware:

- All registers are corrupted.
- All signals driven by powered-down logic are corrupted.
- All behavioral processes within the powered-down domain are deactivated.

Equally important are the semantics for when power is restored to a domain. On power-up:

- All combinatorial and latch (level sensitive) processes are evaluated, including continuous assignment statements.
- Edge-triggered processes (flops) are not evaluated until the next active edge.
- All behavioral processes are re-enabled for evaluation.

Retention, Isolation, and Level Shifting

UPF recognizes two other types of supplies — *retention* and *isolation*. Like primary supply nets, these supplies have auto-connection semantics that correspond to retention and isolation logic specified via UPF. UPF supports the specification of retention and isolation strategies. In this context, a strategy is a general rule on how to implement these low power design functions.

Although level shifters have the minimal functional semantic of a buffer, low power designs are frequently multi-voltage or employ dynamic voltage and frequency scaling and, thus, require the use of level shifters. A level shifter swings a logic value in one voltage (e.g., input) to the same logic value in a different voltage (e.g., output). As complex designs utilize multiple voltage domains (power domains operating at different voltage levels), it is necessary to insert level shifters to ensure that a logic '1' in one domain is recognized as a logic '1' in a different domain.

UPF supports the specification of level shifter strategies, including voltage tolerance threshold, whether the strategy applies to up-shift, down-shift, or both. It also supports the designation of whether a strategy applies to input or output mode ports. Implementation tools take this information and, optimizing based on the power state table specification, infer level shifters wherever they are required in the design.

There are additional key concepts and capabilities of UPF, which need to be mentioned, even if briefly.

- Anything created in a UPF specification is created within a specific scope of the logic design. This helps to provide easy mapping from the UPF to the logic design, which facilitates writing the UPF code as well as debugging and analyzing the low power design specification.
- To facilitate verification, UPF defines a support package in SystemVerilog and VHDL. This package defines routines that facilitate querying the current state of a supply net or port and setting the state of a supply port. This allows the testbench to mimic the off-chip power supplies that are connected to the chip's supply pads.
- The ability to define a UPF supply net to an HDL logic value conversion (and vice versa) is provided when a supply net is connected to a logic port in the design. This facilitates integrating the supply network defined in UPF to power-aware models in the design without requiring re-writing of the model to work with the UPF definition and representation of supply net states.

The Power Aware Simulation Flow

Power Aware Simulation (PA Simulation) solves the problem of functional verification of power-aware designs. It gives designers the ability to functionally verify their power management techniques at the RTL, reducing costs significantly both in terms of effort and time.

PA Simulation works with normal RTL coding styles. Designers do not need to hand-instantiate gate-level retention cells for state data, and the PCN does not have to be intertwined tightly with the RTL

functional specification. Thus, legacy RTL blocks are easily reused without modifying the RTL code, and new reusable blocks can be created independently of the power-aware environment they may be used within.

The simulator must be able to:

- Identify all sequential elements inferred by the RTL design (registers, latches, and memories).
- Overlay the RTL design with the PCN.
- Pull in the appropriate retention-cell model behavior.
- Dynamically modify the behavior of the design to reflect the specified low power design intent in power down and up situations.

Based on these requirements, PA Simulation is broadly categorized into four steps:

- Register/latch recognition from the RTL design.
- Identification of power elements and their power control signals.
- Elaboration of the power aware design.
- Power Aware Simulation.

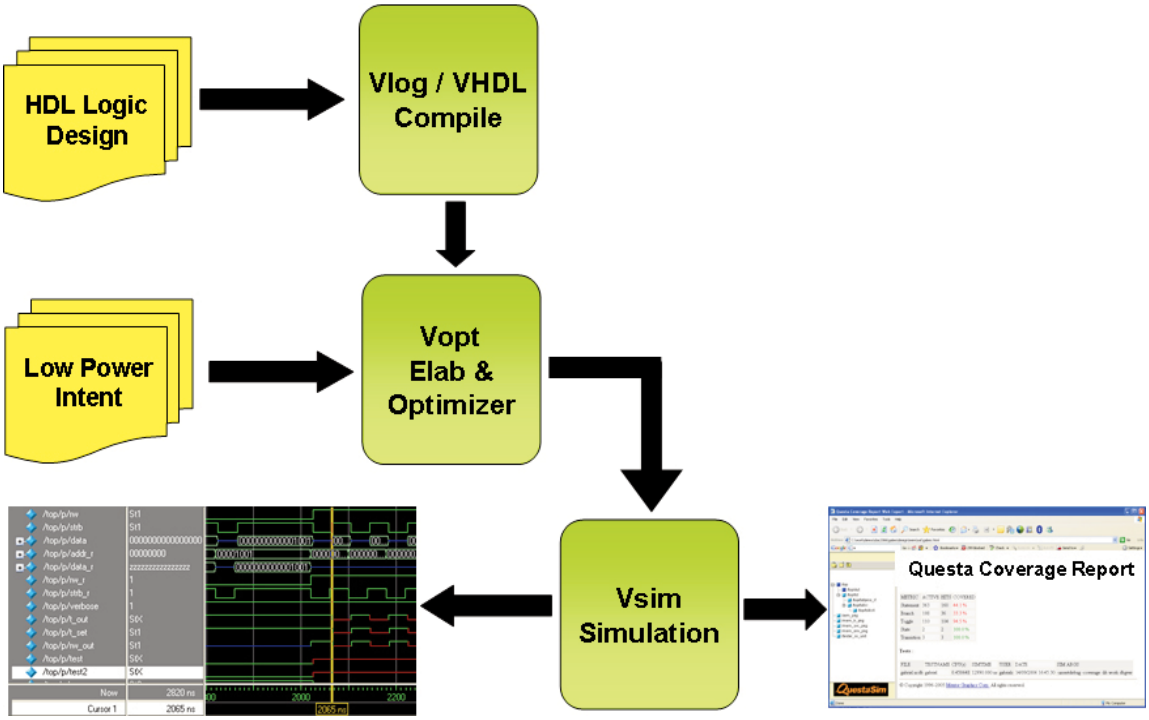

Figure 4. Questa simulation flow with Power Aware modeling.

Register, Latch, and Memory Recognition

Register, latch, and memory recognition from the HDL is normally associated with the front-end of synthesis, which transforms the HDL into a structural netlist containing flip-flops, latches, memories, and combinatorial logic. To accomplish recognition of sequential elements, Mentor Graphics enhanced its verification platform, Questa™, with the ability to infer sequential elements and recognize memory structures, providing a complete power aware verification solution.

Identification of Power Elements

The UPF contains the low power design intent and is processed for the following information:

- Power and voltage domains and power control signals.
- Mapping of sequential elements to retention models (RFFs, RLAs, memories).
- Intended corruption semantics (outputs of a power domain and sequential elements).
- Isolation and logic strategy implementation

Using the UPF and sequential element information, the simulator is able to create the PCN, overlay the functional network with the PCN, and integrate the specified power-aware behavior (retention, corruption, and isolation) with the RTL functionality.

Elaborating the Power Aware Design

Power management could be under software or hardware control; for example, a power management block (PMB) specified in RTL internal to the SoC or an external hardware PMB. Either of these power management controls drive the signals that define the PCN, based on the system's power management strategy—signaling power domains to retain state, enable isolation, power down (turn off switches), power up (turn on switches), disable isolation, and restore state. Although the signals of the PCN exist in the design, they are typically not routed through the design hierarchy in the RTL. During elaboration, the tool automatically routes the PCN based on the power specification in the UPF.

During this process, power domains specified to have state retention in the power configuration file are identified. The sequential elements within these domains are identified and the power aware retention behavior mapped to these sequential elements; for example, power down behavior-corruption, state retention, power up behavior, and state restoration. All other logic elements within a power domain will have their behavior modified such that at power down their outputs are corrupted and with power up they resume normal operation. When a power domain is turned off, it goes to an unknown value; in other words, it is corrupt. Simulation verifies whether the system works properly when the low power design intent is exercised as specified.

The power aware models are mainly abstract models of the power aware retention behaviors. These models typically contain named events recognized by Questa. The interfaces to these models are also predefined. The corruption model simply corrupts at power down and releases at power up. This model is used as a generic corruption model.

Retention registers (flip-flops and latches) can be created with Questa Power Aware modeling capabilities. Using this capability, an accurate model that encapsulates the following power aware behaviors can be developed:

- Saving register values (start of retention mode)
- Power down
- Power up
- Restoration of saved values
- Any special considerations (corner cases) on inputs during the above phases. Note that the interface/ports of these power aware models are fixed.

These models work in tandem with the RTL when simulated in Questa to mimic equivalent power aware behavior after implementation. Semiconductor companies create power aware retention models for their corresponding retention gate-level flip-flops, latches, and memories.

Power Aware Simulation in Action

After integrating the low power design specification with the RTL functional specification, the PA simulation is ready to start. The design simulates normally. Typically, the testbench, mimicking the software power management system will exercise the PMB through various system power states. The PMB implements those system states by switching power supplies, enabling or disabling isolation, gating clocks, and executing save and restore protocols.

While specific domains are powered down by the PMB, the testbench can verify that the *awake* portions of the design continue to operate properly. Assertions written using PSL or SVA can be employed to verify the correct sequence for power up/down, retention, and isolation. They can also be used to ensure proper functioning of the awake portions of the design in various system power states.

When the power management strategy (based on test stimulus) determines that power domains need to be turned on, the PMB enables power to the power domains that were previously turned off and signals restoration of retained values for the sequential elements. Verification continues to ensure that the power domains come up in good known states and that the entire system can continue operating normally.

PA simulation exposes many functional bugs including:

- Failure to retain sufficient state information to enable restoration of functionality when power is restored.
- Dependency on output values.
- Problems when interacting state machines in different power domains restore to states that create deadlock or live lock situations.
- Improper sequencing of save and restore operations by the PMB.
- Failure to reset a block upon power-on to a known good state for non-retentive blocks.

Equivalence Checking

By applying consistent semantics for both verification and implementation, the UPF supports a multi-tool verification flow that uses simulation and equivalence checking tools to fully verify the low power design intent from RTL to GDSII. At the gate level, equivalence checking accelerates the verification of the power specification. It mathematically compares the post-synthesis netlist to the RTL to ensure that the power aware functionality has been properly implemented. Because it uses statistical algorithms, equivalence checking provides 100 percent coverage of the netlist virtually instantaneously. It checks every control wire going to every retention flip-flop and every net; some of which simulation could miss because simulation tests only what the engineer tells it to test.

For example, using UPF an equivalence checker reads both the RTL and the power aware specification to find the retention flip-flops indicated in the power aware file. The checker tags these with a property that identifies them as a retention flip-flop. This information is used later during comparison of the RTL and the netlist. The tool finds the flip-flops identified at the RTL then looks at the UPF library cell that the netlist calls out to determine whether it has the retention cell property. If the RTL and gate-level netlist match, it meets the retention property requirement. If it does not, then it records that failure. Equivalence checking does the same check for isolation buffers and level shifters.

Conclusion

The advent of power aware design with its reliance upon multiple power domains introduces additional opportunities for error. Anytime this occurs, companies need to put in place tools that verify correct functionality and identify errors in the specification or implementation.

The naturally loose coupling of low power and functional design intent permits separate specification of both. Separate specification provides the freedom to define a concise, high-level mechanism for specifying low power design intent. It also enables greater reuse of IP within different low power architectures.

The Mentor Graphics low power design specification and verification solution has been developed in conjunction with partners who are successfully using it in production designs, finding low power design flaws that would have required chip respins. The recent UPF standardization of a low power design specification enables engineers to portably define power distribution architectures, create power aware strategies, and verify their low power designs throughout the RTL to GDSII flow.

Visit the Mentor Graphics web site at www.mentor.com for the latest product information.

Corporate Headquarters
Mentor Graphics Corporation
8005 S.W. Boeckman Road
Wilsonville, Oregon 97070USA
Phone: 503-685-7000
North American Support Center
Phone: 800-547-4303
Fax: 800-684-1795

Silicon Valley
Mentor Graphics Corporation
1001 Ridder Park Drive
San Jose, California 95131 USA
Phone: 408-436-1500
Fax: 408-436-1501

Europe
Mentor Graphics
Deutschland GmbH
Arnulfstrasse 201
80634 Munich
Germany
Phone: +49.89.57096.0
Fax: +49.89.57096.400

Pacific Rim
Mentor Graphics Taiwan
Room 1603, 16F,
International Trade Building
No. 333, Section 1, Keelung Road
Taipei, Taiwan, ROC
Phone: 886-2-27576020
Fax: 886-2-27576027

Japan
Mentor Graphics Japan Co., Ltd.
Gotenyama Hills
7-35, Kita-Shinagawa 4-chome
Shinagawa-Ku, Tokyo 140
Japan
Phone: 81-3-5488-3030
Fax: 81-3-5488-3031

MGC 9-07 TECH7730-w