Understanding the Low Power Abstraction

Dr. Gary Delp
Silver Loon Systems LLC
911 23rd Ave SW
Rochester, MN, USA
+1 507 289 7276
gary.delp@silverloonsystems.com

Erich Marschner
Mentor Graphics Corporation
3919 River Walk
Ellicott City, MD, USA
+1 410 750 6995
Erich_Marschner@mentor.com

Kenneth Bakalar
Mentor Graphics Corporation
5950 Symphony Woods Road
Columbia, MD, USA
+1 443 367 2504
Kenneth_Bakalar@mentor.com

ABSTRACT

We define four abstract models in common use today for electronic design—electrical, digital gate, digital RTL, and transactional—and discuss the relationships among them. The new low-power model described by IEEE Std 1801-2009 UPF is introduced, and its relationship to the other signal-level models for digital and analog design is defined. We then discuss the connections between the low-power model and the underlying physical implementation of a chip, elucidating some of the concepts in the low-power model that are missing from the commonly used abstract models. We define extensions to the electrical/RTL boundary model to support application of the low power model in mixed-model simulation. Finally, we recommend extensions to the existing transaction model to reflect the concepts of the low-power model.

Categories and Subject Descriptors

B.5.2 [Register-Transfer-Level Implementation]: Design Aids --- Automatic synthesis, Hardware description languages, Optimization, Simulation, Verification, B.6.3 [Logic Design]: Design Aids, B.7.2 [INTEGRATED CIRCUITS]: Design Aids --- Simulation, Verification, D.3.0: [Programming Languages]: General --- Standards, D.3.3 [Programming Languages]: Language Constructs and Features --- Constraints, Modules, Packages

General Terms

Algorithms, Documentation, Design, Standardization, Languages, Verification

Keywords

Design Abstraction, Low Power, Power-aware, 1801-2009, VHDL, Verilog, SPICE, VHDL-AMS, Verilog-AMS, UPF

1. INTRODUCTION

Electronic systems are inherently complex, and increasingly so with greater integration. What has allowed us to design and verify such systems at all has been the adoption of abstract models that deliberately exclude certain details in order to focus on the characteristics that are essential to the current stage of the design flow. The most abstract models tend to focus on function; less abstract models consider both function and timing. The least abstract models encompass function, timing, and also physical effects.

Various abstract models are in use in electronic system design, such as the model for digital design that has been incorporated into digital hardware description languages Verilog and VHDL, and the model for analog design that has been incorporated into analog hardware description languages SPICE, VHDL AMS, and Verilog-AMS.

The increasing importance of power considerations in electronic system design has highlighted the need for a model of active power control and its effects. This model may be referred to as the “low power abstraction”, since it is used primarily to reduce power consumption through active power management. It introduces new concepts such as “corruption” and its implications. Similarly, but orthogonally, the increasing importance of system level design has led to a transaction-level model that excludes some of the signal-level details of the digital model in order to focus on the system-level behavior that emerges from the integration of subcomponents.

As the number of abstract models in use increases, the relationships among them become more complex. To ensure that all of these abstract models can be used together, each at the appropriate stage of the design process, we need to examine those relationships and confirm that the models are semantically consistent with each other. Understanding these relationships will also help guide the implementation and integration of tools based on different abstract models. In particular, it is important to understand how the new low-power model relates to the other existing models used in electronic design today.

2. Classic Design Flow

The digital design flow is based on a progression of abstract representations describing incremental refinements of the design that lead to a physical implementation. For decades designers have worked with the digital abstractions. These abstractions have allowed for the separation of functional specification and verification from the eventual physical implementation. An implementation problem that would be unmanageable due to its complexity is made dramatically simpler by treating the design first as a digital abstraction, and then manipulating real interconnected shapes into an implementation of that digital circuit.

The analog block designer uses these digital abstractions only to create place-holder models with idealized function to complete the digital design hieararchy. Detailed design is performed using an electrical circuit abstraction. The completed analog block is substituted for the placeholder late in the design flow, in some cases not until layout.

The Register Transfer Logic (RTL) of the verified digital design is treated as a “golden” description of the design intent. This collection of files that make up the design can be logically verified for correct functionality with many optimizations made possible by working with the abstract description.

The same design is presented to the synthesis and back end tools. The various physical design tools, such as physical synthesis, place-and-route, detailed routing, and optical phase correction, progressively transform the digital design into a physical artifact that implements the digital design.

During the back-end transformations, a number of tools are used to verify the assumptions of the digital abstractions: clocked-based designs, independence of the logical function of the elements, long term operation, and stability of the circuit are all checked. Tools to validate the digital design to physical implementation equivalence include: static timing analysis (STA), voltage droop, cross-talk, and electro-migration checks, a variety of logical equivalence (LEC) checks, and design rule checks.

3. Design Abstractions

An electronic system, and in particular, a system on chip (SoC), is a physical construct consisting of various materials that, when provided with a power source, interact based on their physical properties. The behavior of such a system is the result of these interactions.

Designing an electronic system requires creation of a model of the system that reflects its structure and behavior. However, the structure and behavior of a physical system are far too complex to model in full detail for large systems. Instead, we define abstract models that allow us to focus on those aspects that are most relevant for a given design task.

Many abstract models have been defined over the years for modeling electronic systems. With the physical realization of a system as the basis, these models can be thought of as successive layers of abstraction, each building on a previous layer. Following is a short summary of the models typically used for electronic system design. For each one, the key characteristics of the model are identified, with emphasis on what detail each model abstracts away from the previous layer. Each model maintains connectivity information from the previous layer, while mapping more complex structures to simpler ones. This layering of successively more abstract models enables iterative refinement and allows us to make decisions early, while preserving multiple implementation paths.

3.1 Electrical

The lumped-element electrical model provides a mathematical model of physical structure and behavior at frequencies below those at which wave propagation effects become important. It represents the physical characteristics of the implementation with interconnected, lumped elements, and represents behavior with equations that characterize voltage and current flows among these elements. In this model, no qualitative distinction is made between power sources and control/data signals.

The lumped-element electrical model describes a circuit as a graph whose vertices represent nodes of the electrical network. The model associates two signals with each edge of the graph, current and voltage, and a differential equation describing the time-varying relationship of the two signals. These so-called “branch constitutive equations” are augmented with equations derived from Kirchhoff’s Laws, which add the constraints of the conservation of charge and energy. The solution of the resulting algebraic-differential system is a vector-valued function demonstrating the evolution of the currents and voltages over time. In practice, the system of equations is solved numerically rather than analytically.

In the general form of the electrical model, a procedural process examines the values of the function at the time of interesting events and may modify the branch constitutive equations, but not the topology, of the graph. The modified equations are then solved in the subsequent interval. One example of an interesting event is the crossing of a fixed threshold by a voltage.

Models of electrical components described by separate graphs are connected to each other to form a composite by adding arcs between selected vertices of one graph with vertices of the other. The constitutive equation associated with each new edge is “V=0”; that is, the pair of vertices are shorted together. The individual models may represent devices of a wide range of complexity, from a simple resistor through a single CMOS transistor to a complete base-band subsystem for a receiver.

3.2 Digital Gate

The digital gate model focuses on the logical structure and behavior of the design. This model distinguishes between connections that convey data and control signals among elements and connections that provide power for those elements. In many applications, the power connections are abstracted away, with the assumption that power is always on, sufficient, and stable. This model also abstracts away the continuous nature of time-varying voltage and current, and instead uses discrete logic values for control and data signals.

Combinations of electrical structures such as transistors are abstracted and represented as simple logic gates, and electrical capacitance is represented as signal delay. The connections among logic gates at this level correspond to unabstracted connections among electrical elements in the electrical model.

The behavior of a digital gate model is defined in terms of a discrete event model, in which value changes on inputs trigger computations that result in new values being assigned to state bits and outputs at some later time. The delay between the input changes and the output effects represents the ramp time for the output signal based on its capacitance and other electrical characteristics.

3.3 Digital RTL

The digital RTL model presents a finite-state machine view of a design. This model distinguishes between the asynchronous evaluation of combinational logic and the synchronous update of state elements. As a result, it abstracts away detailed timing information, under the assumption that the clock cycle length is sufficient for each combinational logic block to settle by the time its output is sampled.

In this model, certain logic structures are abstracted as state elements, so the design logic is partitioned into state elements and combinational logic blocks. This model also tends to involve more abstract data types, in which combinations of signals (bit vectors) are interpreted as integer or enumeration values or composites (for example, arrays) thereof.

The digital RTL model involves a cycle-based behavioral model. At each successive clock tick, each state element controlled by that clock loads the output value from the combinational block driving its input, and then updates its output. Changes on the output of a state element then trigger evaluation of the combinational logic blocks that they drive.

3.4 Transaction

The transaction-level model focuses on the operations performed by a design. It uses the notion of a “transaction” to model both units of communication among major system components and units of computation within system components. Transaction-level models tend to be untimed or exhibit approximate timing in terms of clock cycles.

The transaction-level model distinguishes signal interconnections that represent buses and abstracts away the protocols used for communicating over such buses. As a result, bi-directional signal-level communication in the RTL model is often replaced with uni-directional transaction flows representing the resulting data transfers.

Behavior in the transaction-level model is typically represented by algorithms invoked as functions. The interaction of multiple independent behavioral elements is evaluated using a computational model similar to Communicating Sequential Processes (CSPs) [8], in which multiple processes communicate via function calls, some of which block to allow for synchronization or resource arbitration.

4. Model Interactions

These models can coexist, either at the same time, in a mixed-abstraction model of a system, or distributed in time, as successive representations of the same system, or a combination of both. To coexist effectively (that is, in a manner that accurately represents the physical implementation that is the eventual goal), any two models that interact must have a well-defined interface that correctly maintains structural connectively while at the same time enabling interchange of information necessary for both sides of the interface to correctly model system behavior at the appropriate level of abstraction.

The primary challenge is to enable the more abstract model to provide the more detailed model all the necessary information that model needs for its behavioral evaluations, even though the more abstract model has, by definition, abstracted away information that was present in the more detailed model. Nonetheless, techniques have been developed to enable these interactions to occur. The interface needs to synthesize the additional information the detailed model needs by making reasonable default assumptions, augmented by additional information provided by the designer. The following sections describe some of these techniques.

4.1 Current Electrical / Digital Gate

New issues arise when a composite hierarchy combines and connects elements modeled using the digital and electrical abstractions that are not salient in homogeneous digital or electrical hierarchies.
The analog elements may have ports that in a pure analog hierarchy would be connected to electrical power supplies. These electrical power ports either have no counterpart on the digital side, or at best are represented crudely as a single on/off bit or a real number. The electrical model is fundamentally dependent on appropriate power for correct operation. It follows that a supply must be created and connected to each electrical power port or power/ground port pair during the process of assembling the composite hierarchy. The supply is made dynamically dependent on the digital side, if it exists.

Signal ports of digital elements may be connected to the signal ports of electrical elements in a composite hierarchy. The flow of information may be digital to electrical, electrical to digital, or bidirectional. In the general case the connection may be many-to-many; a set of digital ports and a set of electrical ports may all be connected together. We consider here only the simplest, one-for-one case, which generalizes to the many-to-many case with a few manageable complications.

If the flow of information is from the digital side to the electrical side, then when the mixed hierarchy is assembled, an extra power supply element must be interposed between analog and digital ports. The new power supply is selected from a pre-defined set and parameterized by the digital source. The power presented at the analog side varies dynamically with the changes in the digital source.

If the flow of information is from the analog side to the digital side then a sensor element is interposed between analog and digital ports. The sensor measures some characteristic of the analog side (for example, the voltage at a particular node) and interprets the measured value as a logic value (‘0’, ‘1’, or ‘X’ to represent ‘unknown’), which it then transmits to the digital side.

If the flow of information is bidirectional, then both a supply and a sensor must be added.

In the Verilog-AMS modeling language, the supplies and sensors are realized as “connect modules”. In other contexts they are called “hypermodels” ,“boundary elements” or “converters”. Here we refer to these aspects abstractly as the “boundary model”.

The mixed hierarchy by itself does not provide sufficient information to select and calibrate these supplies and sensors. The missing information may simply be assumed (for example, all power supplies are ideal 2.5 volt sources). It may be obtained from some external repository defining which supply to use in a given sub-hierarchy or for a given set of elements. The selection of a supply or sensor may be narrowed to a particular element by using additional information attached to the ports as static types or attributes. It is fortunate that modern mixed-signal simulators provide considerable automated support for building the composite hierarchy.

4.2 Current Digital Gate / Digital RTL

The digital RTL model abstracts away power connections and detailed timing information that is often present at the digital gate level, and introduces new abstract data types. Fortunately, the mapping from the RTL model to the gate model is relatively simple. Abstract data types such as integers, reals, arrays, and records are mapped to aggregations of single bits. Delta delays used as abstractions of combinational logic delays at the RTL level work equally well at the gate level, provided that timing checks are not required at the interface. Clock signals in RTL are modeled with the same detailed timing as that eventually used in gate level models.

RTL models typically assume that power is always on, from the initial reset of the system and throughout its execution. This assumption also applies in gate level models built from cells that do not have power pins. For gate level models built from cells that do have power pins, the assumption that power is always on need not be made; some of the cells in the netlist may be switches or regulators that can control the characteristics of the power being supplied to other cells. For interactions between digital RTL models and gate level models with power connections, it becomes necessary to consider what happens when a digital gate model driving an input to a digital RTL model is powered down.

The evaluation of digital RTL models, although basically a finite state machine model, is typically accomplished using the same event-driven simulation techniques as those used for digital gate models, so behavioral interactions between the two models are well-defined.

4.3 Current Digital RTL / Transaction

Transaction level models abstract away much of the detailed signal-level activity occurring concurrently in the digital RTL model. Interactions between transaction level and RTL models must map between abstract transaction activity in the transaction model and the detailed signal activity in the RTL model. This typically involves transactors that implement signal-level handshaking corresponding to a transaction. Such transactors can also inject signal-level time delays appropriate for the bus protocol, even though the transaction-level model has no timing or only approximate timing.

5. The Low Power Abstraction

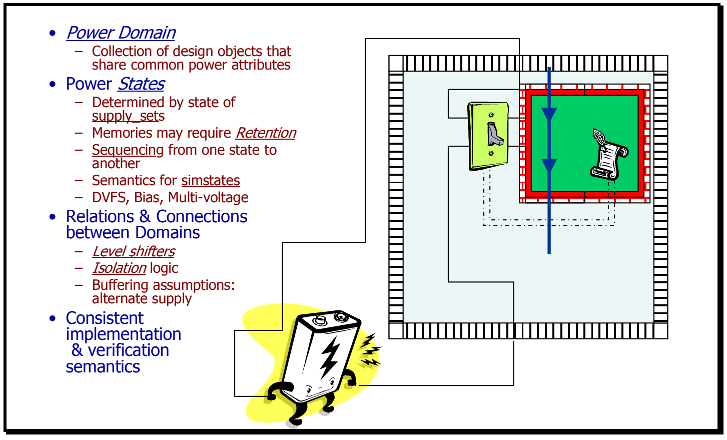
Several of the more abstract models above (those that distinguish power from other connections) were developed with the fundamental assumption that power is always on, always at a consistent voltage, and always sufficient for the system to operate correctly. Some were refined to assume that at time=0 nothing is known about initial conditions, so initial values are unknown, but for the most part, that was the extent of consideration of power as an input to the design modeling process. That assumption is no longer sufficient to produce all of the information that we need to ensure correct operation of the physical artifacts that we produce. As process technology advances continue to deliver the ability to manufacture chips with ever smaller feature sizes, the power required to operate those chips is becoming exponentially greater. Even with techniques such as clock gating to minimize dynamic power losses, the power requirements of today's chips are so large that active power management is necessary to minimize static leakage. This means that the models that assume power is on must be adapted to address the fact that the power provided to any portion of a system may vary significantly over time, from fully on to completely off, and various points in between. Similarly the voltage component of the power supply may vary depending upon the performance requirements.

Fortunately, the consideration of varying power can be abstracted to a few elements and a few new rules. Initially, these additional abstractions can be that power issues can be ignored and the resulting model will still tell us useful things about the system. We still think so; we code the RTL model without reference to power, so it must be that it still useful. We are adding a refinement step, RTL plus power intent superimposed. Or, we are doing both RTL and power at once, but separating concerns to cut the problem to manageable size.

To reflect the change in this fundamental assumption, the Digital Gate, Digital RTL, and Transaction level models have power aware semantics superimposed to represent the addition of those power-related characteristics that are often or always ignored in those models. These semantics include both structural elements and behavioral aspects.

The structural elements represent the power distribution and control network, the elements required to manage the interactions between portions of the design that have different power characteristics at times, and the elements required to handle restoration and/or re-initialization of state information in response to power being turned off and on again. The behavioral aspects represent the effect of power variations on the underlying computational model used to model behavior at each level of abstraction (as described below: conditional corruption based on power states). For the purposes of this paper, we can think of this superimposition as the *Low Power Abstraction*, which, when applied to an existing model, produces a new model augmented with the ability to represent systems in which active power management is involved.

The low power abstraction is based on the concepts illustrated in Figure 1.



- *Power Domain*
  - Collection of design objects that share common power attributes
- Power *States*
  - Determined by state of supply_sets
  - Memories may require *Retention*
  - Sequencing from one state to another
  - Semantics for simstates
  - DVFS, Bias, Multi-voltage
- Relations & Connections between Domains
  - *Level shifters*
  - *Isolation* logic
  - Buffering assumptions: alternate supply
- Consistent implementation & verification semantics

Figure 1 The Structure and Elements of a Power-Controlled Electronic Design

The IEEE Standard for Design and Verification of Low Power Integrated Circuits (IEEE Std 1801-2009) [2] defines the Unified Power Format (UPF), a notation for specifying how power is to be supplied, distributed, and actively managed in a low power digital system. UPF provides a means of specifying the missing "power intent", or power distribution and control information that is necessary to adapt the Digital RTL and Digital Gate models to represent the structure and behavior of designs with active power management.

5.1 Structural Elements

The structural elements of the low power abstraction include

- Power domains
- Power switches and power supply nets/sets
- Isolation and level-shifting logic
- State retention logic
- Power control signals and control logic

Power is supplied to *design elements* by *supply sets*. A supply set can be thought of as including –at least– the *functions* of *power* and *ground*, and may also include various bias and secondary functions. Within a supply set, all of the functions are in relation to each other. There may, in a single supply set be multiple functions with power-like or ground-like uses; the interconnection of these functions to a specific component is specified in implicit and/or explicit mappings. The supply set can be an abstract object early in the design process, but by the end, all of the functions within the supply set must resolve to *supply nets,* each of which must be driven by a *root power source*.

A supply set may be used as a power input to a *switch* or *regulator*, a different supply net emerges from the output of the switch or regulator and will be incorporated into a different, switched supply set. (A switch is binary; a regulator may be modeled as a variable voltage supply.) It is useful to consider the electronic system to be decomposed into *power domains*. Within each domain, the *primary supply set* is the default supply for all of the *elements* in the domain. There may be additional supply sets within the domain, for example, a *retention supply* to provide for preserving the state of *storage elements* under reduced primary power, or an *isolation supply set* to drive some of the *isolation logic*. The primary supply set is the default, and in many designs and domains provides complete coverage of power connection. The primary supply set is considered by many to be the *rail supply* on the region in which the logic is placed in an ASIC, and although this is not specified in the standard, it is an excellent rule of thumb to follow.

Before the advent of UPF, the design assumption was that the elements of the design are always powered on. Static Timing analysis was used in combination with timing constraints to validate the assumption that signals arrive at their endpoints before they are latched. This is still the case, but a design with UPF superimposed may have several additional named modes under which it must be validated.

The Interfaces between design elements supplied by different supply sets need to be examined. There are four major issues:

- The first issue is to reliably transport digital information across boundaries between design elements which are powered by two or more supply sets that are being driven to a NORMAL (see 5.2) level of operation. This becomes an issue when the source and sink supplies have different voltage levels. The concept of level-shifters applies here. The level shifter has a range of input and output operation over which it ensures the digital abstraction of on and off remains consistent from input to output.
- The second issue is to reliably handle the case of a partially powered (or unpowered) domain feeding signals to a fully powered domain. Because the driver may be providing unreliable voltage levels, it is wise to clamp known and safe values onto the signals so that the active domain is not corrupted. Determining what the clamp level needs to be, what signal or signals activate the clamp, and what supplies power both the clamp and the control signal (and any buffering thereof) is part of the topic of isolation. Input isolation provides for *logical safety*.
- If a powered net is driving an unpowered domain, unless the value is clamped, there may be unwanted current flow. Output isolation provides for *electrical safety*.
- The appropriate sequencing of the power, resets, saves, restores, isolation controls, and clocks is critical to the correct and safe operation of a multi-domain system. The supply that is used for the isolation control must be active when needed as well.

5.2 Behavioral Aspect

The behavioral aspect of the low power abstraction is built on the concept of a "simstate". A simstate characterizes the behavior of design elements within a power domain, for a given power state of that domain. UPF defines the following levels of operation of the elements in a design as simstates:

NORMAL: Operations proceed normally and are not influenced by a degraded power state. This simstate represents the situation in which, in the physical realization of the system, the supplied power is sufficient to support full functionality with characterized timing.

CORRUPT_STATE_ON_CHANGE: Combinational logic operations proceed normally, without being affected by the power state. However, state elements have their values corrupted if any attempt is made to load a new value that is different from the previous value. This simstate represents the situation in which, in the physical realization, the supplied power is sufficient to support combinational logic, but not sufficient to support reliable switching activity in state elements

CORRUPT_STATE_ON_ACTIVITY: Combinational logic operations proceed normally, without being affected by the power state. However, state elements have their values corrupted if any attempt is made to load any value, regardless of whether the value to be loaded is the same or different from the previous value. This simstate represents the situation in which, in the physical realization, the supplied power is sufficient to support combinational logic, but not sufficient to support reliable activity of any kind in state elements.

CORRUPT_ON_ACTIVITY: No operations proceed normally. The state of state elements is maintained, provided that no activity occurs. This simstate represents the situation in which, in the physical realization, the supplied power is sufficient to maintain the state of state elements, but not sufficient to support reliable switching activity.

CORRUPT: The entire power domain is corrupted. This simstate represents the situation in which, in the physical realization, the supplied power is insufficient to maintain state or support reliable switching activity.

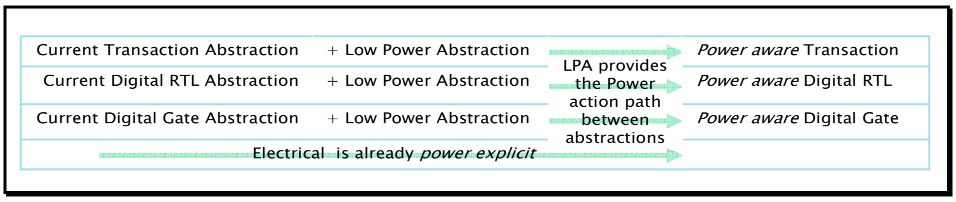
These five abstract simstates represent functionally distinct points on the continuum of power supply levels; each simstate can be useful for modeling certain power states of power domains in a power aware simulation. During power aware simulation, each simstate effectively modifies the behavior of any design element to which it applies by causing its state elements and possibly the drivers of its combinational logic to be corrupted when certain activity occurs on its inputs.

Application of a simstate to a design element is determined by the "power state" of the power supply that provides power to that element. UPF is used to specify that a given combination of voltage levels on the various functions of a supply set have a name, the power state. The power state of a supply set can have an associated simstate as part of the specification. Rather than derive the detailed behavior of the circuit through electrical modeling, the UPF annotated design asserts the class of behavior. The assertion is made and used in simulation, then the implementation must match that assertion, and detailed analysis must validate that the assertion is accurate.

The enumeration of named states with the voltage values identified with that state, and the simstate which is asserted when those voltages are present provides an efficient abstraction. When validating, the collection of named states provide the range over which the validation tools must prove the operation of the particular implementation.

A simstate can be associated with the power state of a power supply. When the power supply is in a given power state, the associated simstate applies to the design elements powered by that power supply. In particular, if the power supply is the primary supply of a power domain, then it applies to all design elements within that power domain.

Preserving memory during times of power shutdown can be accomplished by a variety of means. In UPF, these means fall under the category of retention, and usually involve specially designed elements with an additional supply or a non-volatile element.



Current Transaction Abstraction + Low Power Abstraction → *Power aware* Transaction

Current Digital RTL Abstraction + Low Power Abstraction → *Power aware* Digital RTL

Current Digital Gate Abstraction + Low Power Abstraction → *Power aware* Digital Gate

LPA provides the Power action path between abstractions

Electrical is already *power explicit*

Figure 2 The Addition of the Low Power Abstraction

The essence of the Low Power Abstraction is this:

During a power aware simulation,

a) The structure of the design determines which power states apply to a given design element – that is, the power states of any supply set connected to the design element

b) The applicable power states for that design element determine the applicable simstate for that element

c) If there is more than one supply set providing power to an element that does not have specific behavioral code to determine its corruption levels, then the most corrupting simstate determines how that element behaves

d) Signals that cross power domain boundaries are evaluated for supply difference corruption.

e) The propagation of corruption follows the already established techniques of X propagation at the digital level.

6. Power-Aware Additions to the Design Contract

The simple 1/0 abstraction of a single unified design that we have used for so long breaks when presented with a design in which parts will be "turned off" while other parts are "turned on". For example, in formal verification, constraints on the primary inputs influence correctness of the whole design – until you carve it up into power domains. Multiple power domain designs may require constraints on each power domain reflecting the isolation conditions and clamp values in each state to be evaluated.

IEEE Standard 1801-2009 / UPF 2.0 provides crisp syntax and semantics to represent:

- A collection of elements that share common power characteristics: a *power domain*
- An abstract and refinable way to represent the delivery of power: *supply sets*
- The relationships, boundaries, activities, and side effects of multiple power domains
- Corruption of unpowered or under-powered logic corresponding to simstate

This implementation semantic “contract” is that what is asserted by UPF as the behavior of the circuit must be validated in the implementation.

It is also the case that:

- Logical *Reset* is required not only at time zero (initial), nor just for the entire chip at the same time. There can be multiple reset operations: a global one for the whole system, and potentially several local reset sequences for each power domain.
- IEEE Standard 1801-2009 / UPF 2.0 supports IP block and subsystem reuse and incrementally-refined design. Objects can be defined, then updated. As long as the updates are consistent with previously specified information, it is the same design; incompatible “-updates” are errors and require the consideration of the design produced as a new design, and hence this requires the rebuilding of the base of verification equity.[1]

Just as the traditional digital abstraction of design/verification was based on ones and zeros, the new Low Power Abstraction for power aware design/verification uses the reliable and consistent semantics of the languages to separate the work required on each side of the abstraction. The combination of HDL/UPF is the handoff across the boundary between asserted behavior and implementation directives.

7. Application of the Low Power Abstraction

As is illustrated in Figure 2, adding the Low Power Abstraction to the three of the four abstractions that were missing power semantics allows the full stack to consider variable power supplies at the appropriate level for each abstraction. The Digital RTL model + Low Power Abstraction correspond to “Verification semantics” in IEEE 1801, while the Digital Gate model + Low Power Abstraction correspond to “Implementation semantics” in IEEE 1801.

7.1 Power Aware Digital RTL/Gate models interact with the Electrical model

We mentioned in section 4.1 that the mixed hierarchy by itself does not provide sufficient information to select and calibrate the supplies and sensors that must be inserted at the boundary between digital and electrical elements. It is now easy to see that the low power abstraction created by UPF can be used to supply the missing information.

Every element in the design hierarchy is associated with (at least) a primary supply set, which contains primary power and ground functions. The state and voltage value of the appropriate functions will parameterize the power supply connected to the power pins of the analog element..

The power supply and sensor of the boundary model will also be parameterized by information superimposed by the low power abstraction. The sensor must interpret the voltage level of the analog side in the voltage context provided by the primary supply net of the digital side. As an example, in the simplest case, any voltage rising above half the primary supply net of the target would be interpreted as a ‘1’ by the sensor, and ‘0’ otherwise.

In a parallel fashion, the power supply that drives the analog side of a digital to analog connection will be calibrated to match the power domain of digital side of the connection.

[1] Verification equity can be considered both in sense of capital expenditures (building the test bench), and operational expense (running the testbench through coverage points). When the design is made new with a "small" change, the capital changes are often small, but the operational expense may be large; many of the coverage points must now be recovered. When equivalence can be demonstrated, verification equity can be preserved.

In all three cases, the dynamic changes in power will be accurately reflected at the analog/digital boundary.

7.2 The Low Power Abstraction applied to transaction level models

Models at the transaction level communicate via messages (transactions). In the context of variable power, these transactions must be augmented to ensure three key goals, safety, progress, and accuracy. This need implies that the underpinnings of the transaction communication must be aware of the power state of the initiator and the responder for each transaction. Transactions for power control events must be added, and when an object is in a non-normal *simstate*, it may neither initiate nor respond to a transaction. All transactions in the queue of a corrupt agent are either discarded or answered with a power down response.

At the functional level, abstractly, every operation must potentially wake up the resources it needs, and should then release them afterwards. This is not required for the model, but is an addition to the functional model of use of resources at any level in a power variable environment.

7.3 Power Aware Digital RTL/Gate models interact with Power Aware Transaction models

RTL and Gate level (electrical signal interconnected models) depend upon the 4+ state signal models to communicate power corruption. Transaction Monitors should be extended to be power aware of the systems they are monitoring, either by determining the supply set and registering for simstates changes, or by exporting supply information of their own, and then reading and writing 4+ state signals on the signal interface.

8. Conclusions

The Low Power Abstraction extends our current set of digital design abstractions to enable accurate modeling within a variable power environment. Low Power Abstraction also enables improvements in the interaction between the digital models and the electrical model, by providing some of the detail required for accurate modeling at the interface. Finally, the Low Power Abstraction suggests ways in which the transaction level model should be extended to support very abstract modeling and analysis of active power management.

9. ACKNOWLEDGMENTS

We recognize the IEEE P1801 working group for the development of the Low Power Abstraction.

10. REFERENCES

[1] UPF 1.0, Unified Power Format, Accellera Technical Working Group, see: http://www.accellera.org/apps/group_public/documents.php?wg_abbrev=p1801

[2] UPF 2.0, Also "IEEE Standard for Design and Verification of Low Power Integrated Circuits," IEEE Std 1801-2009 , vol., no., pp.C1-218, March 27 2009 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4809845&isnumber=4809844

[3] “Verification Methodology Manual for Low Power”, Srikanth Jadcherla, Janick Bergeron, Yoshio Ionue, David Flynn, Synopsys, Inc and ARM LTD 2009, ISBN 978-1-60743-413-9

[4] OVM, Open Verification Methodology, http://ovmworld.org/docs_2.0.2/html/index.html

[5] “Open Verification Methodology Cookbook”, Glasser, Mark, 2009, ISBN: 978-1-4419-0967-1, http://www.amazon.com/gp/product/1441909672

[6] "IEEE Std 1666 - 2005 IEEE Standard SystemC Language Reference Manual," IEEE Std 1666-2005 , vol., no., pp. 0_1-423, 2006 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1617814&isnumber=33906

[7] *Low Power Methodology Manual: For System-on-Chip Design,* Michael Keating, David Flynn, Rob Aitken, Alan Gibbons, Kaijian Shi, Springer, 2007

[8] Hoare, C. A. R. (1978). "Communicating sequential processes". Communications of the ACM 21 (8): 666–677. doi:10.1145/359576.359