A Policy-Based Approach to Guiding Low Power Design Methodology

Bhanu Kapoor, Debabrata Bagchi, Satrajit Pal, Sanjay Churiwala
Atrenta Inc.
2001 Gateway Pl. 440W
San Jose, CA 95110

ABSTRACT

Every low power design is different and tends to have its own interesting set of issues to solve. You have to think through various elements of chip design if you want to operate at very low power levels and you have to introduce appropriate design techniques early in the design cycle. This article discusses a policy-based approach to help guide the construction of a design description such that it incorporates low power design techniques as well as ensures adherence to new low power design methodologies arising from advances in technology.

The first part of the paper describes the model for a policy management system. A policy-based system that will allow designer to define, in a succinct and organized fashion, design policies that automatically point out low power design related issues during the RTL code development process. The second part of the paper presents a number of techniques that enable a typical low power design methodology. We present some examples of low power SoC design issues such as clock gating and voltage management to address both dynamic and leakage power consumption and how these issues can be addressed early in the design cycle using a framework of policy-based design methodology.

1. INTRODUCTION

Silicon technology now allows us to build chips consisting of tens of millions of transistors. While this technology promises new levels of system integration onto a single chip, it also presents significant challenges in terms of figuring out a design methodology for effective use of available transistors on a single chip.

Battery life is often a key concern (and product differentiator) in applications such as digital cellular telephones, personal digital assistants, MP-3 players, and laptop computers. There is a great deal of interest in reducing power consumption [3] of chips that make up such systems. And even for the applications that are line powered, such as cable modems and set-top-boxes, lower power designs are a must to ensure lower cost and improved reliability.

Processor and system designers have a number of ways to decrease power consumption. These include using lower supply voltages and taking advantage of power management features where available, as well as some programming techniques for reducing power consumption.

The challenge in designing for low power is that, until recently, designers had no visibility into the power ramifications at the design stage. The only tools that existed worked at later phases of design cycle - much too late to make fundamental design changes. Now, there finally are some techniques that can be employed early in the design cycle, as the design is being coded, that can make a big difference.

Every low power design is different and tends to have its own interesting set of issues to solve. You have to think through various elements of chip design if you want to operate at very low power levels and you have to introduce appropriate design techniques early in the design cycle. Power consumption is proportional to the square of supply voltage, which means that reducing a processor's supply voltage can result in dramatic savings in power. For example, reducing the supply voltage from 3.3 to 1.0 volts reduces power consumption by a factor of 10. Similarly, there is a direct dependence of power consumption on capacitance and activity of nets. Being the most active net in the design, clock nets tend to account for a large proportion of dynamic power consumption.

As a result, design methodologies now being employed to reduce power consumption include techniques such as partitioning designs into multiple voltage domains to reduce switching power, powering down certain portions of the chip to reduce leakage of power, and gating the clock with appropriate conditions when a part of the circuit is either holding the state or inactive. A big part of the problem is in just ensuring that these new design methodologies are being adhered to during the creation of complex system-on-a-chip (SoC) designs.

Although very large (multi-million-gates) ASIC and SoC designs are routinely manufactured, designing them correctly and producing them on time, and in volume, with adequate quality, all involve methodology of design. ASIC/SoC methodologies are needed that offer designers integration of complete systems with reusable blocks on a single chip.

We present a new methodology to accelerate the design of complex ASICs and SoCs through predictive analysis and policy-based RTL code development. This approach performs detailed structural analysis on RTL in order to check coding styles, RTL-handoff, design re-use, clock/reset requirements, verification, timing, design for testability, low-power guidelines and much more. Our focus in this paper is on issues related to

the low power design methodology involving SoCs. It requires a "look-ahead" engine that is based on fast-synthesis technology and a fast in-built cycle-based simulator to carry out such analysis during the RTL code development process.

The rest of the paper is organized as follows: Section 2 describes the basic model for policy-based RTL design, elements of policy enforcing engine, and the policy management process. Several examples of low power design issues and their application the context of policy-based approach is described in section 3. We then conclude the paper with a summary and a list of references.

2. POLICY-BASED APPROACH

What is needed is a comprehensive, policy-based system that will allow designer to define, in a succinct and organized fashion, design policies that automatically point out time consuming downstream issues during the RTL code development process. The end result is that time-to-market goals are met and predictable schedules become a reality.

The goals of policy-based RTL design, outlined in this paper, include creation of a system that addresses RTL design using policies which guide the design process efficiently towards design goals under given design constraints. Specifically, the system performs many functions including:

- Policy Application
- Policy Creation
- Result analysis and reports

The main goals of analyzing such issues using policy-based analysis include creation of a system that addresses RTL design using policies that guide the design process efficiently towards design goals under given design constraints. A policy is a collection of rules for specific purpose such as rules associated with certain standard or certain design tool. Policies can be extended thus allowing one to develop and manage customized groupings of rules more easily.

This is important in the context of low power in that every low power design is different and tends to have its own interesting set of issues to solve. Specifically, the system performs many functions including: policy application, policy creation, and policy analysis and reporting. Policy enforcement is enabled through an engine consisting of fast synthesis, cycle-based built-in simulation and switching analysis engine, and a fast design database traversal working at a higher-level of abstraction.

Basic elements constituting policy-based RTL design are now explained a predictive analysis software that enables a policy-based design methodology.

2.1 Policy

A policy is a collection of rules for a specific purpose, such as rules associated with a standard, a silicon vendor, or a specific design tool. Policies enhance user extensibility, allowing you to develop and manage customized groupings of rules more easily. The design methodology includes a set of policies that you can select during the process of RTL code development. Examples of such policies include lint, reuse [1, 2, 4, 5], verification, timing, testability, specific design practice, a silicon vendor's requirements, and techniques for efficient usage of design tools.

2.2 Rule Groups

A collection of rules within a policy is termed a group. Typically, groups consist of rules addressing a particular area of interest in the RTL code. Groups are hierarchical, meaning that a group can contain other lower-level rule groups as well as individual rules. A group provides an additional level of modularity in applying policies to a given RTL design.

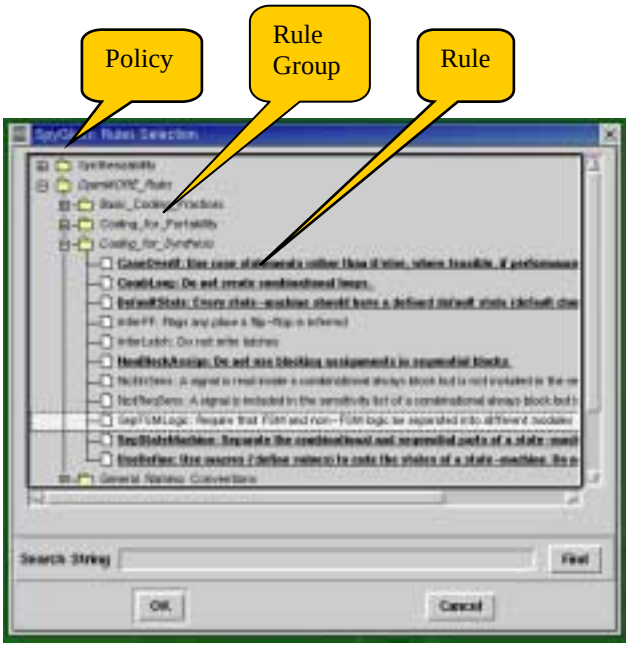

Figure 1: Rule and Group Selection

2.3 Rules

A rule is the most fundamental element in the policy-based management system. It describes a set of conditions that – when checked by the policy engine – result in an indication of a specific problem with the RTL code. Rules allow standard analysis of the RTL code. An example of a policy, rule groups, and rules in the SpyGlass system is shown in Figure 1. With this system, you can selectively turn on specific groups or specific rules within a group.

The result of applying policies to an RTL design is s set of rule violations with different severity levels, cross-linking of these violations with RTL code and schematic, and a process of suggested correction of the issue in the RTL code. Each policy can be tailored for application with a list of parameters. Figure 2 shows an example of parameters associated with a design policy such as levels of logic, fanout, and the allowed size for multiplexers.

Policy management involves the selection of rules and policies, application of policy parameters, creation of new rules and policies, policy result analysis, and report generation.

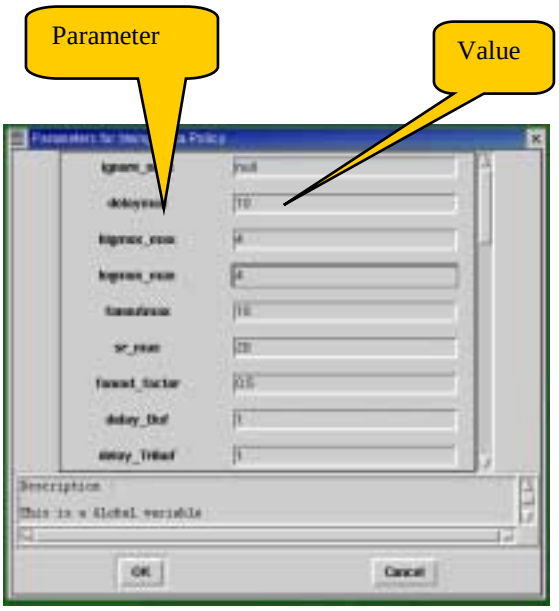

Figure 2: Parameters associated with a policy

2.4 Policy Engine

Policy enforcement is enabled through an engine consisting of fast synthesis, fast cycle-based simulation, and a fast design database traversal working at a higher-level of abstraction.

Policy engine accelerates electronic product development by enabling development teams to capture, aggregate, distribute and apply rules and requirements early in the development cycle. The fast synthesis engine internally creates the design structure and foresees downstream issues early in the development cycle, thereby eliminating errors at the earliest possible stage. Such an engine ensures that designs are always compliant with design methodology by identifying, advising on how to correct violations as the design progresses towards completion. The engine preserves the correlation of RTL to the fast-synthesis netlist so that any errors detected there can be traced back to the origin for quick problem identification and correction.

Although you can check many complex rules statically on the internal synthesized structural view, other rules require some understanding of the logic function of the design. This is particularly true for testability-related checks. In order to perform a testability check, you must use an evaluator. The evaluator in the policy engine is a cycle-based simulator you can use to resolve functional design constraints, as well as carry out a simulation required to set up the design for testability analysis.

Policy implementation requires a traversal engine that works on the RTL netlist produced by fast synthesis. Policy engine provides a rich set of functions for carrying out design traversal. The connectivity information, coupled with the traversal primitives, enable you to create rules that look for violations across the design hierarchy.

3. LOW POWER DESIGN METHODOLOGY MANAGEMENT

The ability to address potential low power design issues early in the design cycle is critical to achieving high productivity in the design process. Not only do we get more optimized designs as we address these issues during the RTL code development but also achieve an improved efficiency for the rest of the tool flow being used in the design process. In addition, there is a lot to be gained from having a truly golden RTL code for the current as well as future implementations of these systems.

Most chips are fabricated using CMOS technology. To a first order, the dynamic power consumption of CMOS circuitry is given by the formula:

$$P = 1/2 * a * C * V^2 * f$$

where P is the power in Watts, a is the activity factor, C is the load capacitance in Farads, V is the supply voltage in Volts, and f is the clock frequency in Hertz. This equation suggests that there are essentially three ways to reduce power:

- Reducing the supply voltage, V
- Decreasing the capacitive load, C
- Slowing down clock, f
- Reducing the net activity factor, a

A major issue in 0.13-micron and more advanced processes is leakage power. Since the leakage current increases by approximately factor of 5 for every generation of new process technology, this is likely to be a dominant source of power consumption in near future. One of the best ways to handle leakage dissipation is to turn off portions of the design completely during certain modes of system operation. Clearly, such an approach has important design methodology implications that need to be taken care of during the design process. We will now discuss some examples of issues related to the low power design methodology that can be addressed through a policy-based infrastructure.

3.1 Elements of Low Power Design Methodology

The design methodologies now being employed to reduce power consumption include techniques such as partitioning designs into multiple voltage domains to reduce switching power, powering down certain portions of the chip to reduce leakage of power, and gating the clock with appropriate conditions when a part of the circuit is either holding the state or inactive. Each of these design techniques can introduce significant complexity in the design process. A big part of the problem is in just ensuring that these new design methodologies

are being adhered to correctly during the creation of complex SoC designs.

3.1.1 Clock Gating

Clock nets account for a large proportion of dynamic power consumption for two reasons: clocks are the most active nets in the design, and clocks nets account for a large portion of capacitive load associated with the design. While clock gating is seen as a useful technique for reduce clock power consumption, careful management of manually added gated clocks and those that can possibly be introduced automatically by a tool becomes a challenging problem if one has to obtain maximize the benefits of gated clocks while keeping track of testability issues that may arise as a result of adding gated clocks in the design.

It is important that a tool helping designers in the gating process is able to understand the existing gated clocks in the design in order to effectively guide the designer to introduce additional gated clocks in the design. It is important that the designer is able to analyze the impact of gated clocks in the design by inspect the regions in the design being impacted by a given gated clock.

In addition, the tool has to be able to analyze each flop in the design to come up with a good set of candidates for clock gating for a given design. In doing so, designer chosen heuristics should guide these choices towards the ones that are going to have the maximum impact on power consumption of the design.

A set of policies that can guide the clock gating process can then include aspects such as:

- Identify and analyze existing gate clocks in the design
- Find candidates for clock gating driven by designer guidance such as only look for candidates where a minimum number of specified registers can be gated together
- Using the knowledge of existing gated clocks in the design, flag conditions where further gating may result in back-to-back gating conditions
- Order candidates for clock gating based on likely impact in reducing power consumption
- Ensure gated clocks are bypassed in the test mode such that testability issues are taken care of during the creation of gated clocks.

Figure 3 shows an example of execution of one such policy related to finding a set of registers as candidate for clock gating. Set of enabled registers in the design that share clocks and enable can be a good candidate for clock gating for set sizes

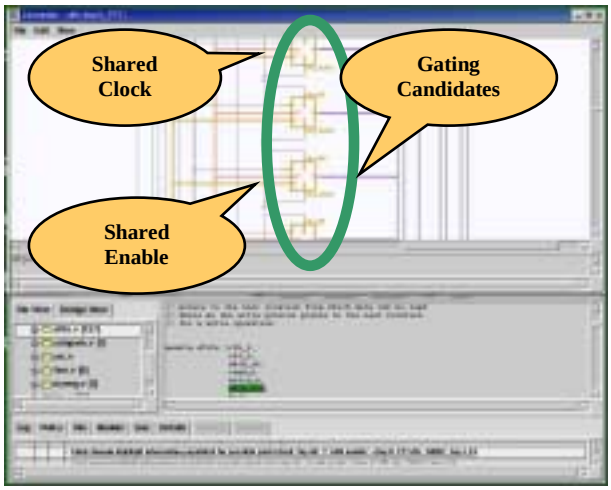

Figure 3: Finding candidates for clock gating through early analysis of the design

larger than a user specified limit. Such flops can be spread across the design unit boundaries. In addition, flops with a combinational feedback loop around them, indicating that data is being held by that flop, become good candidates for clock gating.

As the gated clocks are introduced in the design to achieve low power goals, there are testability implications of having gated clocks [6] in the design. One must ensure are all internally generated and derived clocks by-passed in the test mode as the design is being readied for insertion of scan chains. To get the design ready for testability, the testability policy has rules which address topology related issues as well as functionality dependent issues. The topology related rules depend only on part type, pins, and interconnections and functionality dependent rules involve test mode, test clock pins, parameters and circuit reset checks. The topology category includes rules such as combinational feedback detection and port-to-port path connection whereas functionality dependent rules deal with issues such as tri-state contention and propagation of test clock under test mode assertion.

Figure 4 illustrates a portion of a large chip where the RTL designer provided a by-pass for both a derived clock and a PLL generated clock so that both U1 and U2 were to be clocked by the pin clk when the test mode pin, tm, was held at 1. The RTL equation defining Mux 2 interchanged the connections from the PLL clock and the direct clock so the test clock was not controlling U2 in test mode. Values held constant in test mode are colored green and lines receiving test clock pulses are colored pink. The connection from Mux 1 to U1 is pink indicating that a test clock pulse can reach U1. The line from Mux 2 to U2 is not colored indicating that the by-pass clock, clk, is not reaching U2 in test mode. The Mux 2 selector input has the correct color for test mode so the problem of the interchanged connections to the I0 pin and I1 pin is easily diagnosed.

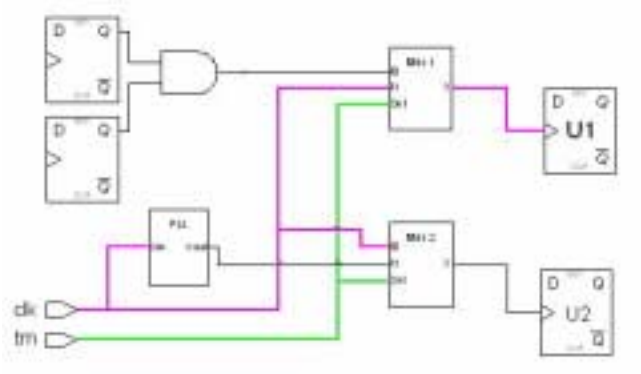

Figure 4: Clock by-pass error and display of the clock distribution in Test Mode

3.1.2 SoC Voltage Management Issues

Power consumption is proportional to the square of supply voltage, which means that reducing a processor's supply voltage can result in dramatic savings in power. For example, reducing the supply voltage from 3.3 to 1.0 volts reduces power consumption by a factor of 10. This allows for saving in power consumption when scaling voltage based on throughput needs.

Operating only the most critical portions of the design at a higher voltage allows for tremendous savings in power consumption. For example, if a design only needs VDD for the 25% of the design and the rest of the design can be operated at 60% of the VDD without any impact on the throughput, this simple arrangement of two voltage domains can cut the power requirements into approximately half of the original requirements if the full chip were to operate at VDD. Process technologies now allow for creations of multiple voltage domains in the design. Even though the idea of having voltage domains is a relatively simple one, there are significant methodology issues related to using the idea on a real chip design. One such challenge is ensuring level conversion of the signals that cross voltage domain boundaries. When partitioning the design into multiple voltage domains, appropriate level shifting elements are needed on signal crossing possible pairs of voltage domains. Whether the insertion happens at RTL or at the netlist-level, depending upon the design methodology in place, RTL code can be used to guide the insertion and provide a check against the final design netlist. Appropriate methodology checks may need to be enforced at RTL, gate, and layout levels of abstraction.

Figure 5 shows an instance of a signal crossing voltage domain boundary on which a proper level shifter has not been placed. A policy-based approach finds each crossing between all pairs of specified voltage domain to find if there is a missing level shifter on those signals.

While switching power is the most important contributor to the total power consumption now, leakage power is playing an increasing important role in determining the overall power consumption of a given design. Sleep or idle modes typically

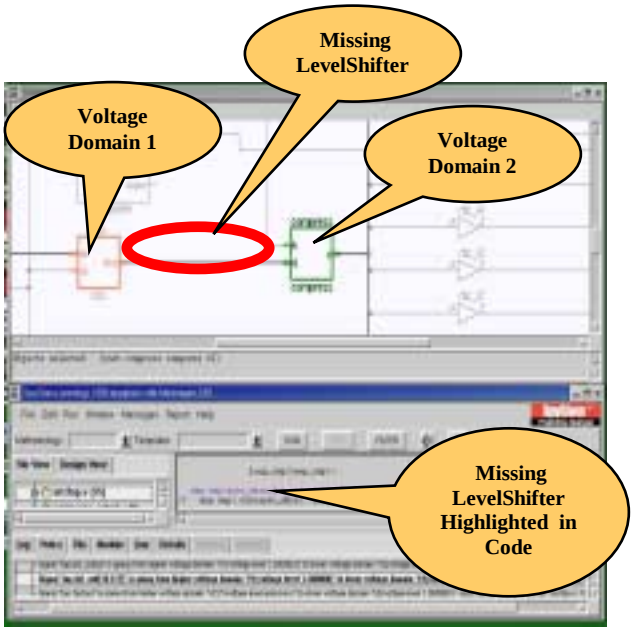

Figure 5: Missing level-shifters as a signal crosses voltage domain boundary is cross-probed into code and schematic

turn off the clock to all but certain sections of the processor to reduce power consumption. Because CMOS power consumption is proportional to signal toggling frequency, turning off the clock to the processor can greatly reduce power consumption. But the system may continue to dissipate leakage power even under such modes of operation.

One of the best ways to handle leakage dissipation is to turn off portions of the design completely during certain modes of system operation. Clearly, such an approach has important design methodology implications that need to be well taken care of during the design process. In this scenario, signals to and from domains that may be switched on and off require special attention as they may introduce “floating net” issues. Proper isolation methodology must be in place in order to avoid such design issues and checked accordingly. A policy that allows specification of power domains that can be shut down during portions of design operation, understands isolation mode conditions, and is able quickly synthesize and simulate this scenario can be effectively used here. This is also a voltage management issue that requires attention early in this design cycle as either missing an isolation cell or putting an incorrect isolation cell late in the design cycle is likely to result in a possible costly re-spin of the design.

4. SUMMARY

Although very large (multi-million-gates) ASIC and SoC designs are routinely manufactured, designing them correctly and producing them on time, and in volume, with adequate quality, all involve methodology of design. We have described a design methodology based on policy-based RTL design with the focus on low power issues. The elements of policy-based RTL design have been outlined. Low power design involves successfully placing a design methodology that may involves

multiple voltage domains, special techniques to deal with leakage power, and a host of techniques that can be applied to the construction of RTL code. In this regards, we have described a policy-based approach to help guide the construction of a design description such that it incorporates low power design techniques as well as ensures adherence to new low power design methodologies arising from advances in technology.

5. ACKNOWLEDGMENTS

The author will like to thank Martin Baynes, Tom Carlstedt-Duke, Aloke Das, Phil George, Sushil Gupta, Al Joseph, Sam Lay, Ralph Marlett, Mo Movahed, Bernard Murphy, and Ghulam Nurie, and Mona Singh for their valuable contributions towards the preparation of this paper.

6. REFERENCES

[1] M. Keating and P. Bricaud, “Reuse Methodology Manual: For System-on-a-Chip Designs”, Kluwer Academic Publishers, 2001.
[2] L. Bening and H. Foster, “Principles of Verifiable RTL: A functional coding style supporting verification processes in Verilog”, Kluwer Academic Publishers, 2001.
[3] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, “Low Power CMOS Digial Design”, IEEE Journal of Solid State Circuits, pp. 473-484, April 1992.
[4] Semiconductor Technology Academic Research Center, “STARC open Design Style Guide”, www.starc.or.jp/.
[5] OpenMORE, “The Industry Reference for IP Measure of Reuse Excellence” www.openmore.com/
[6] R. Marlett, “Putting the D back into Design for Test”, EDA Vision Magazine May 2002, www.atrenta.com