

LOW-POWER SYNTHESIS OPTION

TABLE OF CONTENTS

Introduction	1
Power challenges today	1
The LPS methodology	3
LPS optimizations	5
Optimization techniques	8
LPS power analysis	11
Conclusion	13

TABLE OF FIGURES

Figure 1	The conventional flow versus the LPS flow	1
Figure 2	Various design stages	2
Figure 3	Low-power design methodology	3
Figure 4	RTL exploration	4
Figure 5	Commitment at gate level	5
Figure 6	A typical use of clock gating	6
Figure 7	Clock gating applied to register banks	6
Figure 8	The addition of logic for testability	7
Figure 9	Sleep-mode analysis	7
Figure 10	Partial sleep mode	8
Figure 11	Gate sizing to reduce power	9
Figure 12	Pin swapping to reduce power	9
Figure 13	Reducing power with gate merging	10
Figure 14	Reducing slew by introducing a buffer	10
Figure 15	Logic restructuring for lowering power	11
Figure 16	Toggle count generation	12
Figure 17	Visual power analysis	13

INTRODUCTION

Today, low power is emerging as a critical issue in ASIC design. The primary driving factors are the remarkable success and growth of personal computing devices (portable desktops, audio- and video-based multimedia products), and wireless communications systems (personal digital assistants, pagers, and cellular phones). These technologies demand high-speed computation and complex functionality, making low-power consumption a crucial design concern.

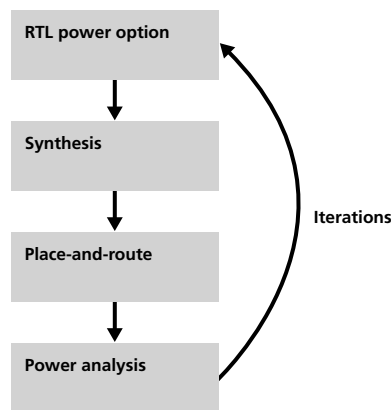
Another development that emphasizes the need for low-power solutions is the growing number of large, high-performance designs. Millions of transistors switching at high clock speeds leads to power consumption problems that can be difficult to solve late in the design cycle. The exorbitant cost associated with packaging and cooling strategies for high-performance designs is yet another factor that makes it imperative to have a seamless low-power methodology.

Because power dissipation is becoming just as important as performance and area, power analysis and optimization must be an integral part of the design methodology. The Cadence® Low-power Synthesis Option (LPS) addresses low-power issues early in the design cycle. LPS explores options for lowering power at RTL, but does not make any commitments until gate level. Also, all gate-level transformations take into account power, delay, and placement information. This white paper provides an overview of the methods employed today to address low power and outlines the solution we are putting in place to create low-power designs.

POWER CHALLENGES TODAY

The biggest challenge in low-power design is the lack of an integrated design flow that addresses power-related issues.

Conventional flow



LPS-PKS flow

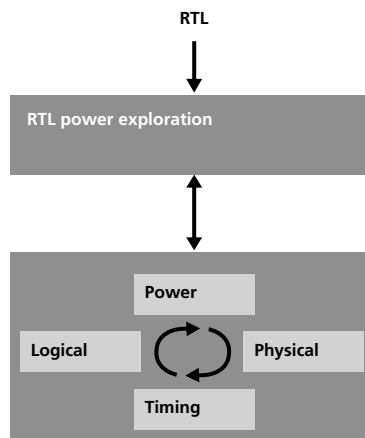


Figure 1: The conventional flow versus the LPS flow

The flow on the left in *Figure 1* shows a typical low-power design flow used today. This conventional power flow is shown in greater detail in *Figure 2*. At the RT level, some low-power optimizations might be performed (step 1 through 2). This is followed by post-RTL timing optimizations (step 2 through 3). After timing constraints are met, power optimization is performed (step 3 through 4). You then do place-and-route (P&R) and power analysis (step 4 through 5) and check that your power goals have been met. If they have not been met, you must go back to RTL (step 1) and iterate. You might then modify the RTL and follow the same flow. The problem with this method is that you are separately doing power optimizations, timing-driven synthesis, and P&R. They are not integrated, forcing time-consuming iterations.

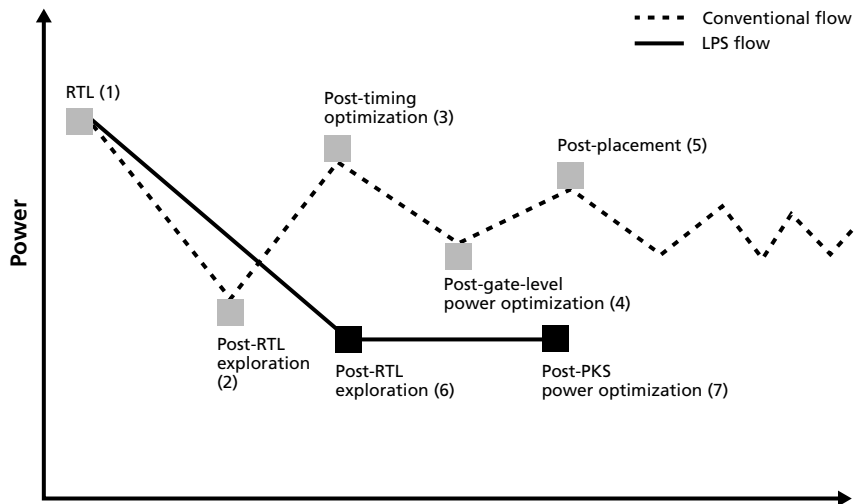


Figure 2: Various design stages

One specific example of the problem with the conventional power flow is having the tool insert clock gating for all register files greater than 8 bits wide. Once the tool inserts the clock gating as specified, you would perform power analysis and discover that either a timing violation occurred or the power consumption was higher because of all the clock-gating logic inserted by tool. The result is that you would have to go back to the RTL. You would have to tell the tool to do something else and make another guess about where power could be saved. This is a hit-and-miss approach. The basic problem of iterations remains.

In the LPS-to-Cadence® Physically Knowledgeable Synthesis (PKS) power flow, we do RTL exploration followed by physically knowledgeable gate-level optimization that includes timing, placement, and power optimization together in a completely-integrated tool. At the RTL exploration stage, we look for all possible ways to save power and pass that information forward to the gate-level optimization stage of design.

The following are the benefits of the LPS-to-PKS power flow:

- It is a completely automated solution where all possibilities to save power are explored with no user directives or intervention
- It is a one-pass solution that goes from RTL through P&R for a power-optimized design after P&R

The following sections of this paper will go into greater depth about the Cadence LPS methodology, the optimizations performed at the RTL and gate level, and our power analysis methods.

THE LPS METHODOLOGY

Cadence has developed a low-power solution that is consistent with methodologies designers use today on timing-critical designs. Low-power analysis and optimization is used throughout the design cycle starting from RTL design through timing-driven synthesis and place-and-route. *Figure 3* shows the design flow that we recommend for low-power design.

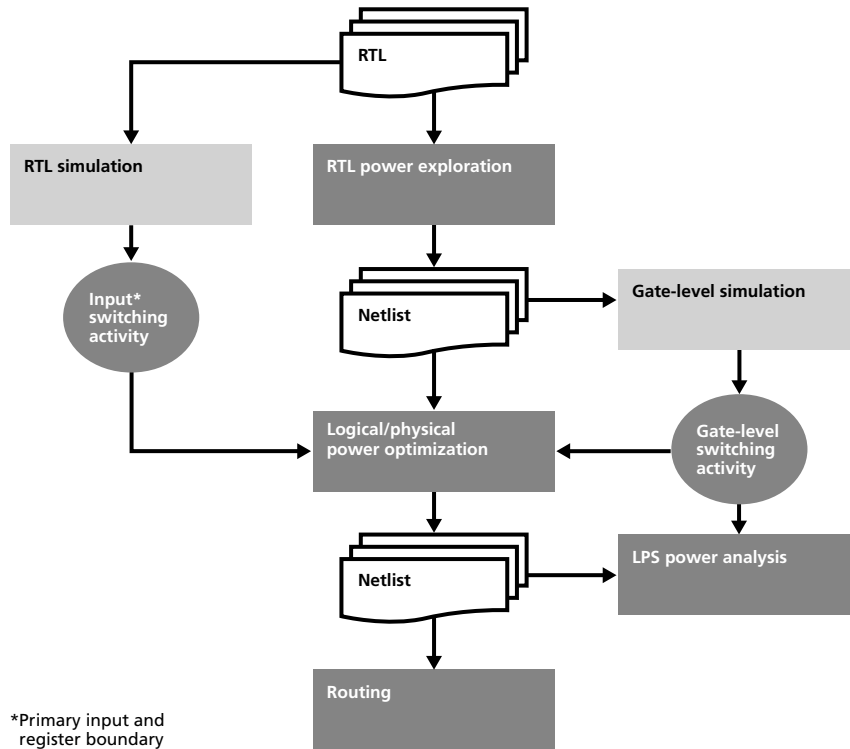


Figure 3: Low-power design methodology

The low-power design flow illustrates the various points in the design flow where we do power analysis and optimization. When the RTL is ready for synthesis, low-power synthesis is performed as part of the normal synthesis process. The design is then analyzed to determine where power savings can be made. Clock-gating and sleep-mode logic are inserted accordingly. After the netlist is generated, a more detailed power analysis is done after simulating the gate-level netlist. Gate-level switching activity helps drive the gate-level power optimization where techniques—such as gate sizing, logic restructuring, and buffer removal—are analyzed for potential power savings that preserve timing. The final step of the gate-level optimization is committing all inserted RTL and gate-level transformation logic.

This solution addresses the lack of an integrated timing and power solution, which is the root cause of the iterations between the RTL and power estimation shown in *Figure 1*. We address this by performing RTL power analysis, exploring all possibilities for reducing power, and then selectively choosing the transformation that reduces power and improves timing during gate-level optimizations. This kind of integrated solution from the RTL to gate-level enables our flow to perform exploration at RTL and commitment at the gate level—the key factor differentiating our solution from other low-power solutions.

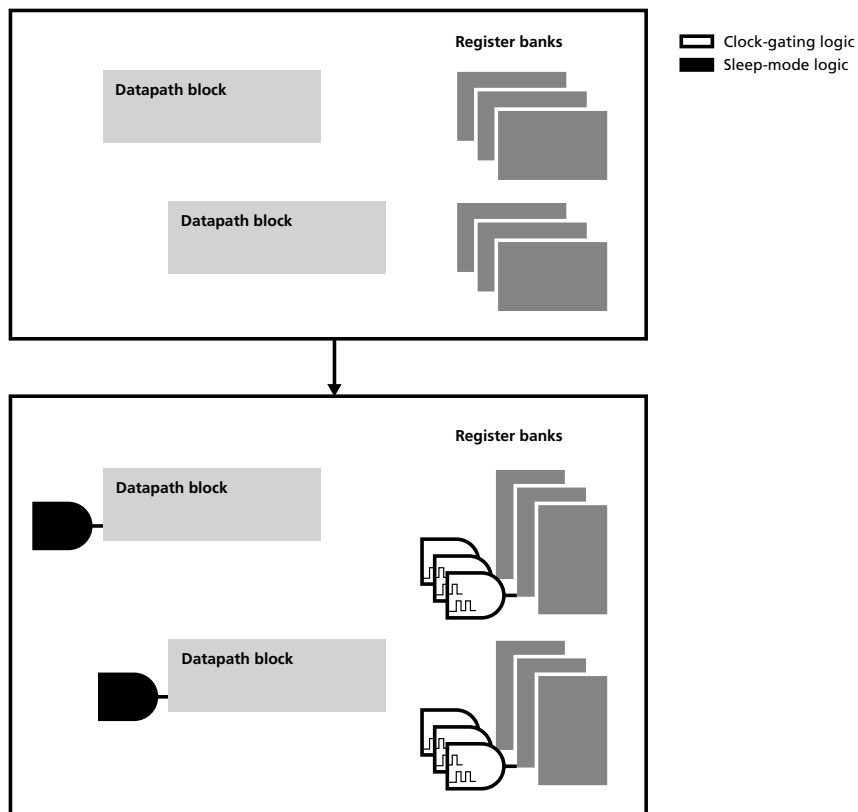


Figure 4: RTL exploration

EXPLORATION AT RTL AND COMMITMENT AT GATE LEVEL

LPS begins looking at the design at the RTL and exploring all possible ways to save power. For register banks, clock-gating logic is created and inserted. For functional blocks, sleep-mode logic is created and inserted. Note that at this point none of the clock-gating or sleep-mode logic is committed to the rest of the logic. We simply “mark” all the registers and functional blocks where the logic could be used to save power. This information is then taken forward to the gate-level optimization design stage. *Figure 4* shows this exploration process at the RTL.

When the design is at the gate level, detailed timing information and toggle rates and activity on every net from simulation is available. This is when we look at all explored power savings possibilities and narrow them down based on timing and overall power savings. For example, in *Figure 5* you can see that the inserted clock-gating or sleep-mode logic is going to impact timing on your critical path, so the clock-gating or sleep-mode logic will be removed along this path.

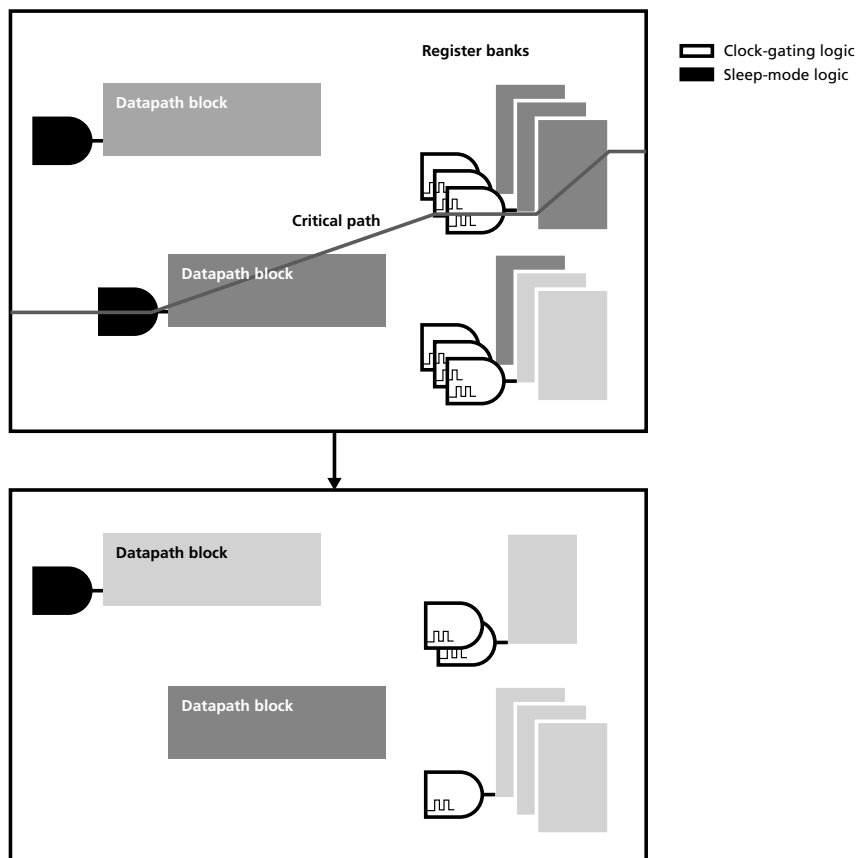


Figure 5: Commitment at gate level

LPS also looks at off-critical paths and decides whether to keep the clock-gating or sleep-mode logic based on power savings. Whenever clock-gating or sleep-mode logic is added to the design, the logic is also going to consume power, so we ensure that only the logic resulting in net power savings is kept. The switching activity obtained from simulation is used to calculate net power savings in each case.

LPS OPTIMIZATIONS

At the end of gate-level optimization we have narrowed down the choices from RTL exploration and have made final decisions about which clock-gating and sleep-mode logic to keep based on timing and power savings. For real designs, at the RT exploration level, you might have thousands of clock-gating and sleep-mode candidates marked, but it may make sense to keep only 30% when you're at the gate level. This means you are saving design time; you don't spend time guessing what will save the most power without affecting timing. You also don't have to go back and undo unnecessary optimizations. The tool automatically performs the process of exploring at RTL and committing at the gate level either with or without user directives.

CLOCK GATING

In many design situations, data is infrequently loaded into registers, but the clock signal continues to switch at every clock cycle, driving a large capacitive load. This makes the clock signal a major source of dynamic power dissipation. Significant amounts of power can be saved, by identifying periods of inactivity in the registers and disabling the clock during those periods.

Our clock-gating technique involves the following three steps:

1. Exploration—Identifying periods of inactivity in registers, as well as the set of registers to clock gates
2. Insertion—Putting in gating logic for clocks contained in the inactive periods identified in the first step
3. Commitment—Committing gating logic based on power savings, performing timing checks for the setup and hold times of the gated clocks, and removing gating logic where timing constraints are violated or if there is no possible power savings

Figure 6 shows a typical scenario where clock gating has been applied to a circuit where the enable signal determines the loading condition of the data into the register. In this example, the clock is disabled whenever the data is not changing, which is determined by the gating function. The glitches in the gating function are filtered through a latch so that the gated clock is a hazard-free signal. The enable signal is always set to logic one.

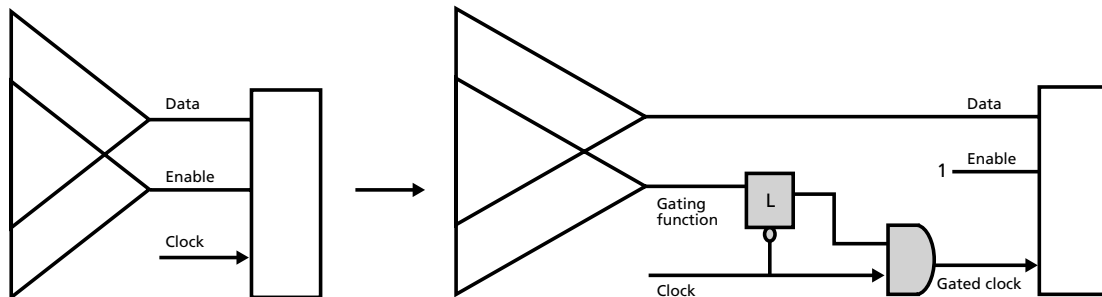


Figure 6: A typical use of clock gating

Register power consumption is lowered by using clock gating because:

- Power is not consumed during the idle periods since the register is shut off by the register during these times
- Power is saved in the clock circuitry
- The logic on the enable circuitry in the original design is removed

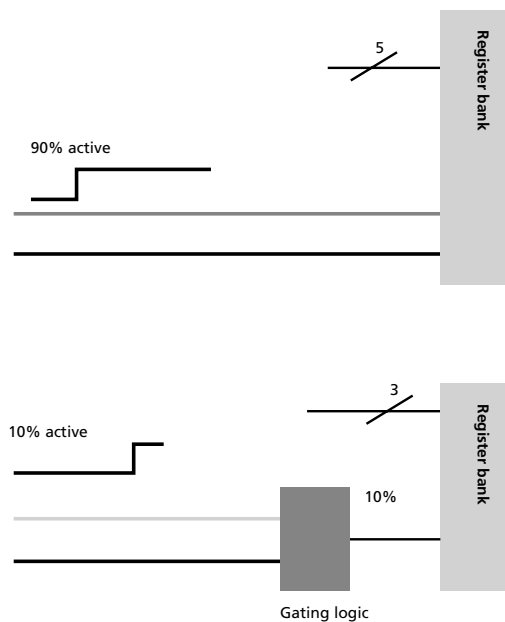


Figure 7: Clock gating applied to register banks

Clock-gating logic is added to register banks based on timing and power savings. We look at the enables of the registers to determine the activity and then decide if we should add clock-gating logic. Figure 7 shows that, if the activity is low (such as 10%), it may make sense to insert clock-gating logic on a 3-bit register. So, it might also make sense to create clock-gating logic for smaller register banks, where activity is low, rather than for larger register banks, where activity is high. In other solutions being offered in the market today, you must direct the tool to insert clock-gating logic. This method can cause problems if, for example, you chose to insert clock-gating logic for all register banks wider than 4 bits. As Figure 7 shows, this may not work.

Clock gating can also impact testability of the design. LPS can increase controllability and observability of a given design by adding control logic for the test mode, as shown in *Figure 8*. During the test mode, the gating function is made observable by tapping the output of the latch and feeding it to the observability detection circuitry. The controllability of the gated clock signal is enforced by the chip test signal, *Test_Enable*, during the test mode.

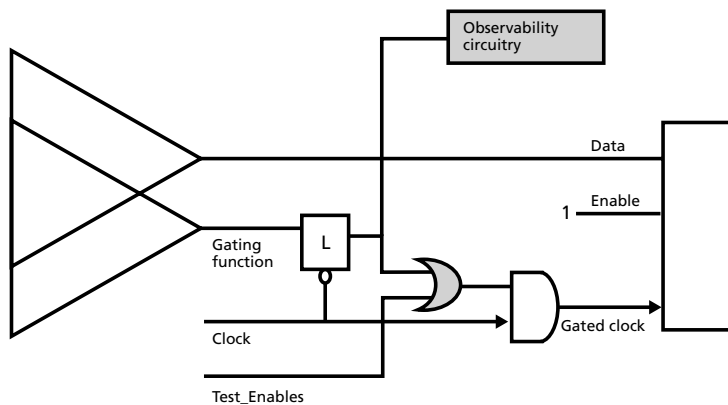


Figure 8: The addition of logic for testability

Our clock-gating technique is more advanced than the competitive solutions because:

- During optimization, we selectively gate the clock of inactive registers. Because clock-gating logic consumes power, carelessly inserting gated clocks will likely increase the power of the modified design
- We perform timing checks for the setup and hold times of the gated clock and remove the logic if timing is violated. Because timing optimization is tightly linked with the power optimization engine, we can make intelligent decisions such as not inserting clock-gating logic in critical paths

SLEEP MODE

Although clock gating is very effective in reducing the dynamic power dissipation of a digital circuit, it is restricted to saving power only on sequential elements. *Figure 9* shows a design where a register takes the result of the comparator only as its input. This means that the power dissipated by the multiplier is wasted because it continues running useless computations when only the output register needs the result of the comparator. Also, power lost is significant because the multiplier is one of the most power-consuming function units. Clock gating cannot be used for this problem because either the multiplier or comparator always uses the outputs of the two input registers. Instead the solution is to shut down the function unit(s) when their results are not used. To do this, LPS performs sleep-mode analysis.

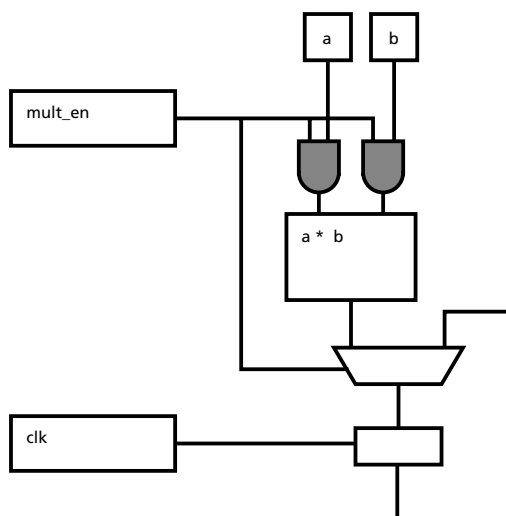


Figure 9: Sleep-mode analysis

The biggest challenge of RTL optimization is that the transformations are best performed at the RTL, when accurate power and timing information is not yet available. Because of the inaccuracy of power estimation at RTL, sleep-mode transformations performed at RTL can actually cause power increases of the whole chip. More importantly, doing sleep-mode transformations at RTL can result in a final gate-level implementation that violates timing constraints, even though the design meets timing without the transformations. Although the tool could undo the sleep-mode logic, LPS would then be forced to work unnecessarily on the “wrong” critical paths and thus make an extra effort to optimize “real” critical paths after undoing the unneeded transformations.

LPS addresses this issue using a two-phase approach:

1. An exploration phase at RTL
2. A commitment phase at the gate level

During the exploration phase, a functional and structural analysis of the control data flow graph (CDFG) is performed to identify potential sleep-mode transformation candidates, including arithmetic units, functional blocks, and even entire logical hierarchical blocks. Control logic necessary for shutting down these candidates is also identified. Candidates and control logic are then “marked” on the gate-level netlist. No actual implementation is performed in this phase because inserting the logic at this stage could increase the timing or power. The commitment of these transformations is delayed until gate-level optimization has been done and accurate power and timing information is available.

In the commitment phase, a decision of whether to commit each individual transformation is made based on if the power is reduced without violating the timing constraints. If a sleep-mode candidate violates the timing constraints, LPS can modify the original control logic to partially commit the transformation to meet the timing constraints with some power savings, instead of blindly removing the gating logic without any power savings. This technique also saves runtime.

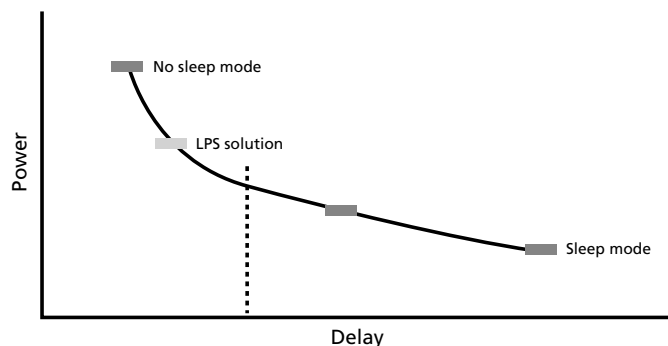


Figure 10: Partial sleep mode

All other solutions in the market today put a module in sleep mode, but, if it violates timing, they go back and completely remove the sleep-mode logic. Figure 10 shows a simple example of how we look at intermediate solutions and find a compromise that meets timing as well as reduces power by introducing partial conditions to shut off logic.

For example, assume that the condition for gating off a functional block is $x + y + z$ where x , y , and z are signals in the design. If signal x is on the critical path, then we use the condition $y + z$ to gate off the functional block, even though this condition will leave the functional block on when x is 1 and y and z are 0. We can perform a much more detailed analysis and explore all possible options, choosing the one that meets timing with minimum power consumption.

OPTIMIZATION TECHNIQUES

After RTL synthesis, we perform various optimization techniques, such as logic restructuring and gate sizing, to minimize power and meet timing constraints. All of these optimization techniques interact with both the timing analysis and LPS engines to identify tradeoff points in the delay power curve. LPS has a number of effort levels that call various transformations. The effort levels are employed to obtain the best quality of result and runtime tradeoff.

Power optimization techniques predominantly operate on the combinational logic blocks, with some techniques embedded to reduce the power of the sequential elements. If switching activities are not provided for the primary inputs and register boundaries, default values are assumed. Though not required, switching activities of internal nodes (if present) also aid in optimization. Internal node-switching activities can be calculated using a probabilistic technique with LPS.

The power savings due to the optimizations depend largely on the switching activities of the internal nodes. Thus, accurate switching activity numbers produce power savings with a higher degree of confidence. The greater the coverage on the circuit nodes with switching activity information, the more likely the power optimization of the circuit will be optimal.

GATE SIZING

Gate sizing is the process of changing the CMOS gate size, smaller or larger, so that the total power is minimized without violating timing constraints. The goal is to find the gate size that consumes the least power. In general, reducing the size of a gate leads to a decrease in the power and an increase in the delay.

The overall internal cell power of a gate consists of:

- Leakage power
- Short-circuit power
- Power consumed by the parasitic capacitance

Sizing the gate reduces the short-circuit power and power consumed by the parasitic capacitance. Gate sizing could, however, increase the short-circuit power of the fanout logic because the slew of the output signal could increase when this transformation is performed. LPS makes tradeoffs between the gate sizing and the increase of power due to slew in order to optimally reduce the power of the design. *Figure 11* demonstrates this process.

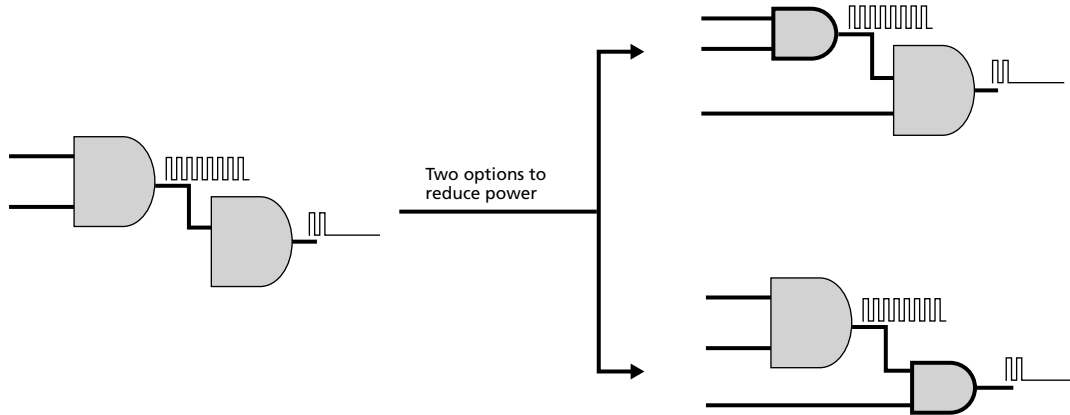


Figure 11: Gate sizing to reduce power

PIN SWAPPING

Pin swapping matches nets connected to the symmetric pins so that power dissipation can be minimized. Pins with higher capacitance are matched to nets with lower switching activity. If not done carefully, pin swapping can actually increase the delay of the design. LPS works to find the optimal configuration that reduces power without affecting timing. *Figure 12* shows how pin swapping works.

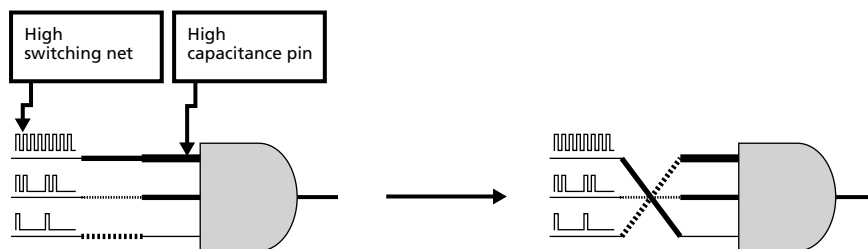


Figure 12: Pin swapping to reduce power

BUFFER REMOVAL

Sometimes timing optimization adds buffers to shield the critical path from a high capacitive load. During timing optimization, the critical path itself could shift. The result is that there are sometimes unnecessary buffers on non-critical paths.

Although these unnecessary buffers are removed in a post-processing step, only a limited number can be removed without violating timing. So, in order to maximize savings for power, LPS removes unnecessary buffers driving highly active nets. It does so by selectively removing these power-driven buffers to minimize power without violating timing constraints.

GATE MERGING

A majority of the dynamic power is consumed in driving the capacitive load of a net. A significant amount of power can be saved by dissolving the net because the dynamic power of a net is proportional to the switching activity of the net. This savings is achieved by merging the gates on either side of the net. Although gate merging eliminates the power loss on the net, it could increase the internal power of the new gate. LPS makes the correct tradeoffs.

Figure 13 shows how the net has been dissolved inside the AOI2 gate.

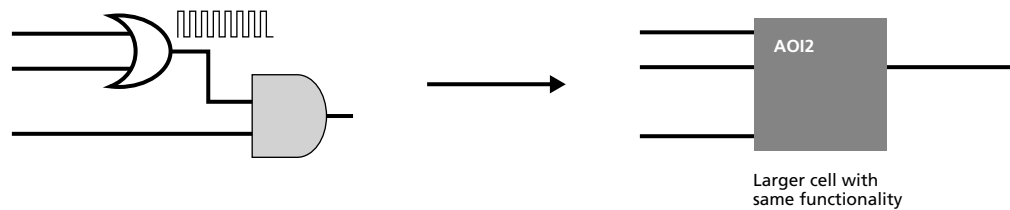


Figure 13: Reducing power with gate merging

SLEW OPTIMIZATION

The internal power of a cell depends largely on the slew of the input signals. During that time, power is drained from the design power rails to ground. If a signal has a large slew, driving the signal through a buffer improves the signal's slew and reduces the overall power of the cells driven by the signal.

Figure 14 illustrates how slew optimization works during timing-constrained power optimization. LPS has the capability of reducing slew. The challenge is in choosing the set of gates to be buffered. LPS performs a detailed analysis to identify the optimal partition that will reduce power as much as possible without violating timing.

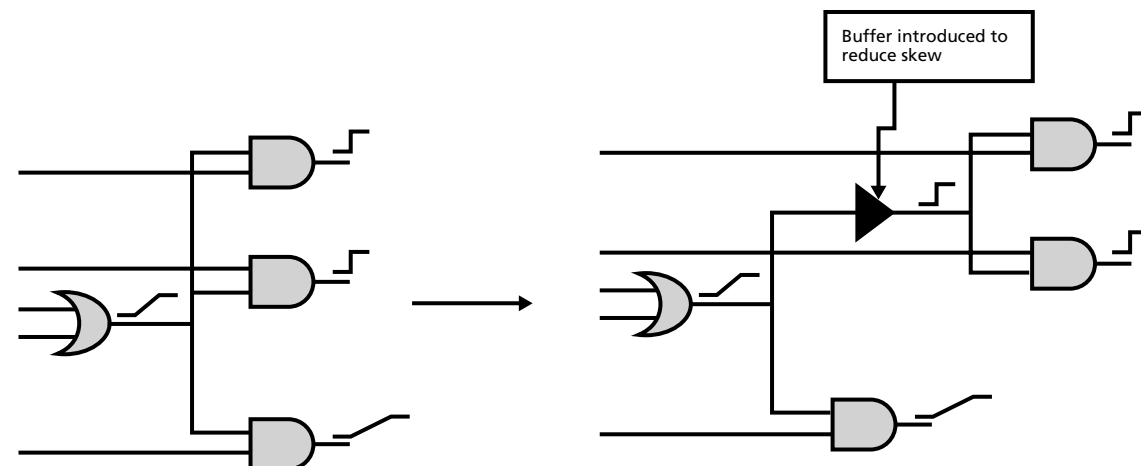


Figure 14: Reducing slew by introducing a buffer

LOGIC RESTRUCTURING

Logic restructuring is different from other transformations performed during power optimization. Unlike gate merging, buffer removal, and other small local transformations that do not make significant changes to the circuit structure, logic restructuring actually changes the topological structure and internal functions of a circuit without compromising the circuit's functional integrity.

Figure 15 shows a circuit before and after logic restructuring. Signal A has the lowest switching activity among all inputs. Although the final output of the circuit has a low switching activity, the two internal nodes are highly active, causing the bulk of the switching power dissipation. The logic restructuring finds a more efficient circuit implementation that cannot be done by other power optimization transformations. The new circuit dissipates much less power than the original circuit because the logic functions at the two internal nodes are now less active.

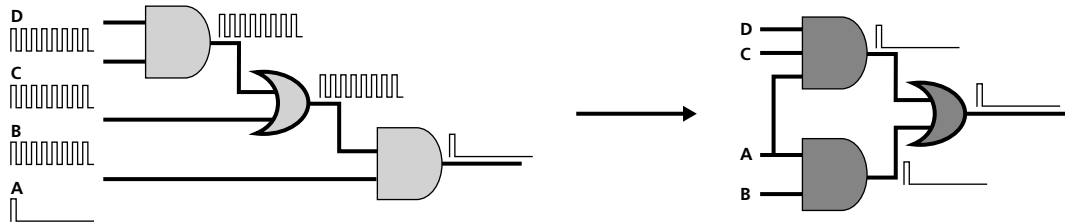


Figure 15: Logic restructuring for lowering power

LPS POWER ANALYSIS

LPS estimates power at the gate level in your design, giving you the ability to characterize and evaluate various design alternatives. This means you can make design tradeoffs between performance and power.

The following inputs are taken into account when calculating power estimates:

- A timing library format (TLF) or algebraic logic functional (ALF) file
- Power models in the technology file (TLF or ALF)
- Slew from the timing analysis engine
- Net capacitance estimates from the Cadence BuildGates® Synthesis electrical system
- Switching activity on each net

INCREMENTAL POWER ANALYSIS

Our incremental power analysis is a unique solution in low-power design. First, incremental power analysis is fast. It computes power only on the design area that has changed. Second, incremental power analysis is accurate. The incremental mechanism ensures this.

Before performing power estimation calculations, LPS reads in the net-switching activities and stores them as assertions on nets. Assertions are toggle count values and probabilities taken from the toggle count format (TCF) file. The switching activities of the new nets created during optimization are incrementally computed whenever the power of that net is required.

LPS monitors changes in the circuit. Whenever there is a query on a net, LPS checks for an assertion and returns that value if the assertion exists. If the assertion does not exist for that net, LPS looks for a previously-calculated value. If there is no value, LPS recalculates the value using the probabilistic technique. If there is an existing value, but the fanin cone has changed and caused an invalidation of the value, LPS recalculates the value.

THE PROBABILISTIC TECHNIQUE FOR COMPUTING SWITCHING ACTIVITY

To compute the switching activity of an internal node, LPS uses the probabilistic technique for propagating the switching activities from the nodes containing the asserted values. If none of the logic nodes in the fanin cone contains assertion values, LPS assumes default values on the primary inputs and outputs of sequential elements. The default value of signal probability is 0.5, and the default value of the toggle count is half the clock frequency driving the node. If the node is not driven by a clock, the toggle count is assumed to be zero. The probabilistic technique takes spacial correlation into consideration; but because LPS does not propagate across sequential elements, temporal correlations are not considered.

Figure 16 shows the process of obtaining the TCF file for input to the power analysis engine using either a Verilog® or VHDL simulator. The cell-specific power values are obtained from the technology library. The net capacitance is estimated from the electrical system in the synthesis tool and the slew on a pin is derived from the timing analysis engine.

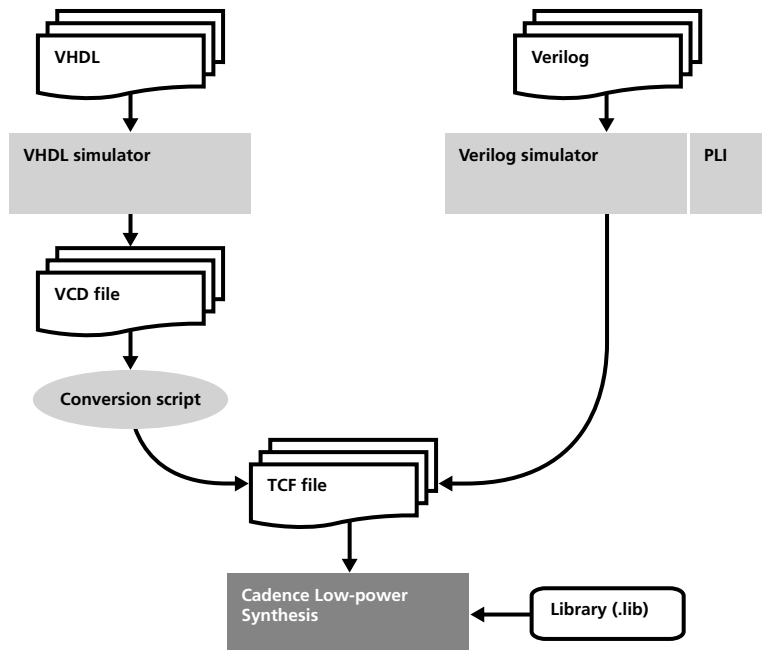


Figure 16: Toggle count generation

POWER CONSUMPTION

Power consumption for a circuit includes two general types of power: static and dynamic power.

Static power is generally dependent on the state of a cell. Static power dissipation can be defined as power that is lost while a circuit's signals are not actively switching. Static power dissipation includes:

- Leakage power due to subthreshold leakage current in CMOS circuits
- Standby power dissipation caused by the direct current (DC) being continuously drawn from power to ground

The overall dynamic power of a cell is based on the effects of voltage and temperature, load, input slew rate, as well as the cell's state. Dynamic power dissipation can be defined as power lost while a circuit is actively switching at a given frequency. Dynamic power dissipation includes short-circuit power and switching or capacitive load power.

LPS uses the following power types to model power dissipations:

- Leakage power
- Internal power (short-circuit power and switching power of the internal cell)
- Capacitive load (net) power, which is calculated using capacitance, supply voltage, and toggle rate

VISUAL POWER ANALYSIS

Power analysis is also available in the visual version of Cadence BuildGates Synthesis. Some of the features included are:

- A colorized module browser and schematic
- Pie charts
- Physical power maps
- Power reports

Figure 17 shows how you can use visual power analysis to get a better idea of power consumption in your design.

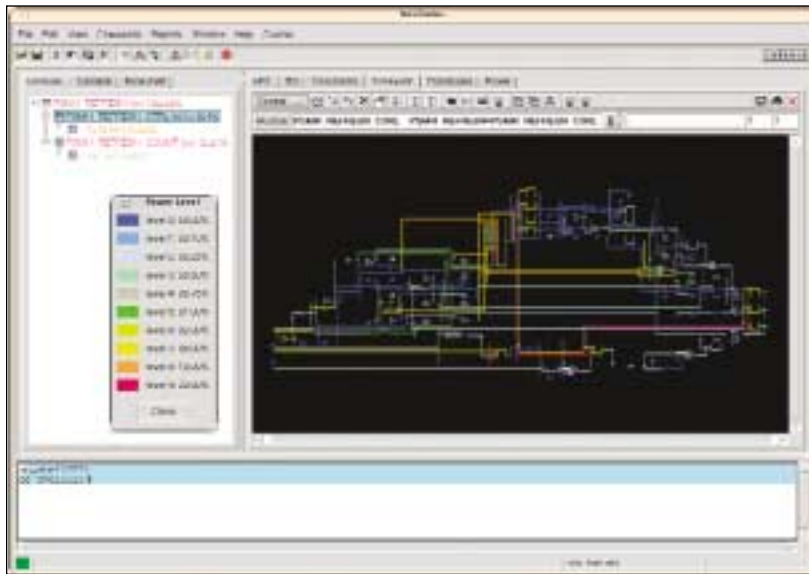


Figure 17: Visual power analysis

CONCLUSION

The Cadence Low-power Synthesis Option solution provides a completely automated solution for the designer. It takes both the guesswork and iterations out of low-power design. The solution is complete, taking the designer from RTL to GDSII. The completeness and automation is due to the integration of timing, power, and physical views of the design into one tool. Another level of integration is between the RTL exploration and gate-level optimization stages of design, enabling LPS to produce optimal low-power designs.



Cadence Design Systems, Inc.

Corporate Headquarters

2655 Seely Avenue

San Jose, CA 95134

800.746.6223

408.943.1234

www.cadence.com

© 2001 Cadence Design Systems, Inc. All rights reserved. Cadence, the Cadence logo, BuildGates, and Verilog are registered trademarks, and "how big can you dream?" is a trademark of Cadence Design Systems, Inc. All others are properties of their respective holders.

Stock #3285 10/01